ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра системного аналізу

Пояснювальна записка

до кваліфікаційної роботи
на ступінь вищої освіти бакалавр

на тему: «**Розробка інформаційно-аналітичної системи телекомунікаційної компанії «X-Byte»**

Виконав: студент 4 курсу, групи САД-41
спеціальності
124 Системний аналіз
(шифр і назва спеціальності)

Горобець Вячеслав Юрійович
(прізвище та ініціали)
Керівник Ярцев Володимир Петрович
(прізвище та ініціали)
Нормоконтроль_____
(прізвище та ініціали)
Рецензент_____
(прізвище та ініціали)

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Системного аналізу _____
Ступінь вищої освіти - «Бакалавр»_____
Напрям підготовки – 124 «Системний аналіз»_____

ЗАТВЕРДЖУЮ
Завідувач кафедри
Системного аналізу
Золотухіна О.А.

"____" _____2020року

З А В Д А Н Н Я
НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ

Горобець Вячеслава Юрійовича
(прізвище, ім'я, по батькові)

1. Тема роботи: «Розробка інформаційно-аналітичної системи телекомунікаційної компанії «X-Byte»

Керівник роботи: Ярцев Володимир Петрович, к. т. н. _____,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від_____

2. Строк подання студентом роботи _____

3. Вихідні дані до роботи:

- концептуальна модель БД;
- база даних у СКБД MySQL;
- додаток що працює з базою даних.

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

4.1 Аналіз технологій та засобів проектування сучасних інформаційних систем.
4.2 Аналіз предметної області та бізнес - процесів IT-відділу.
4.3 Розробка та побудова бази даних інформаційної системи.
4.4 Розробка та побудова екранних форм додатку, що працює з базою даних.
4.5 Розрахунок економічної ефективності проекту інформаційної системи.

5. Перелік графічного матеріалу:
5.1 Інформаційно-логічна модель БД.

5.2 Схеми бізнес-процесів IT-відділу.
5.3 Проект екранних форм додатку, що працює з базами даних.
5.4 Графічні представлення результатів дослідження.

6. Дата видачі завдання _____12.05.2020_____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
	Підбір науково-технічної літератури	12.05-14.05.20	Викон.
	Аналіз предметної області та бізнес процесів підприємства	16.05-17.05.20	Викон.
	Розробка концептуальної моделі БД	16.05-20.05.20	Викон.
	Створення БД у СКБД MySQL	21.05-25.05.20	Викон.
	Розробка архітектури додатка, для роботи з БД	26.05-28.05.20	Викон.
	Розробка програмної складової додатка	29.06-01.06.20	Викон.
	Розрахунок економічних показників	02.06-03.06.20	Викон.
	Розробка презентації до доповіді	03.06-04.06.20	Викон.

Студент _____ Горобець В.Ю.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Ярцев В.П.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Текстова частина кваліфікаційної роботи 49с., 32 рис., 16 джерел.

Мета роботи – розробка інформаційно - аналітичній системи для автоматизації роботи відділу інформаційних технологій інтернет-провайдера X-Byte.

Об`єкт дослідження – розробка концептуальної моделі зберігання, обробки та візуалізація даних для автоматизації роботи IT відділу інтернет-провайдера X-Byte.

Предмет дослідження – база даних з архітектурою «клієнт-сервер», додатки, щодо роботи базою, для супроводження інформаційної системи.

Методи дослідження – системний аналіз предметної області, способів проектування інформаційних систем, комп'ютерне моделювання об'єктів бази даних, створення екранних форм у системі об'єктно-орієнтованого програмування, статистичній аналіз.

У роботі проведено аналіз предметної області відділу IT телекомунікаційної компанії «X-Byte», розроблена концептуальна модель БД системи підтримки прийняття рішень по роботі з користувачами. На основі інформаційно - логічної моделі спроектована база даних у СКБД MySQL. Розроблені екранні форми та запити для роботи з базою даних. Створена форма дозволяє автоматизувати процес додавання запитів клієнтів, надання послуг з підключення, та дозволяє краще, розпоряджатись діями співробітників, облегшує створення форм звітності та дозволяє вести пошук по їхнім полям та найменуванням. Реалізована функція формування нового замовлення. У додатку передбачена можливість розрахунку сумарної вартості по приєднанню користувача, до мережі Виконана робота дозволяє значно скоротити обсяг ручної роботи, полегшити процес співпраці з клієнтами, формування звітності.

Викладені основні принципи побудови систем підтримки прийняття рішень, розглянуто предметну область надання послуг з електронних платежів, етапи проектування БД з використанням СКБД, ООП.

Розроблений додаток, може бути використаний в будь-якій телекомунікаційній компанії.

Апробація кваліфікаційної роботи**:**

1) Розробка інформаційно-аналітичної системи телекомунікаційної компанії «X-Byte». «Дорожня карта інформаційно-телекомунікаційної галузі» . 25 квітня 2020р.
2) Розробка інформаційно-аналітичної системи телекомунікаційної компанії «X-Byte». "Сучасні інфокомунікаційні технології". 25 травня 2020р.

СПИСОК УМОВНИХ СКОРОЧЕНЬ

БД – База даних
ІС– Інформаційна система
ІТ – Інформаційні технології
ОС – Операційна система
СКБД – Система керування базами даних
СППР – Система підтримки прийняття рішень
ADO – Active Data Objects
SQL – Structured Query Language
MS – Microsoft
TDM – Toad Data Modeller
.NET – NET Framework
ІПС – Інформаційно-пошукова система
САПР – Система автоматизованого прийняття рішень
ООБД – Об’єкто-орієнтована база даних
ACID – Atomicity, Consistency, Isolation, Durability
ПЗ – Програмне забезпечення
DML – Data Manipulation Language
DDL – Data Definition Language
РБД – Реляційна база даних
CLR – Common Language Runtime
GUI – Graphical User Interface

ЗМІСТ

Сторінка

Вступ

Зараз в Україні зростає величезна кількість користувачів інформаційних послуг, що викликає масові появи компаній, що займаються послугами в сфері телекомунікацій. Послугами таких компаній є надання високошвидкісного доступу до мережі Інтернет, кабельного телебачення, телефонного зв'язку.

У таких підприємств виникає потреба в створенні інформаційних систем, які будуть виконувати автоматизовані завдання з контролю та обліку даних клієнтів, з управління взаємовідносинами з клієнтами, аналізу можливості підключення, обліку даних користувачів, надання інформації про підключення та розрахунку вартості підключення з урахуванням тарифів.

Для таких підприємств важливо знати потреби клієнтів не тільки в послугах, але і в якості обслуговування, швидкості і коректності реагування системи на звернення клієнтів в відділи компанії. Основною задачею відділу IT в телекомунікаційній компанії «X-Byte», є оцінка можливості підключення, формування ціни на підключення з урахуванням тарифів і послуг, також ведення бази даних клієнтів та формування звітів та надання довідки, стосовно послуг, тарифів, співробітників, клієнтів.

Для ефективного керування діями, які відбуваються в IT-відділі, телекомунікаційної компанії необхідно проаналізувати бізнес-процеси, взаємодію співробітників та користувачів, та створити додаток, який буде працювати з базою даних, для того, щоб облегшити роботу, оптимізувати час на виконання замовлення та час співробітників, надавати актуальну інформацію про можливість підключення, користувачів, тарифів, послуг, та формувати звіти, для аналітики, розбору і оптимізації бізнес-процесів в телекомунікаційній компанії.

1. СУЧАСНІ ПРИНЦИПИ ПОБУДОВИ ІНФОРМАЦІЙНИХ СИСТЕМ

1.1 Аналіз сучасних методів проектування інформаційної системи

Побудування ІС охоплює три основні області:

1. створеня об'єктів даних, які будуть застосовуватися в базі даних;
2. розробка програм, екранних форм і звітів для підтримки запитів даних;
3. топологія мережі, конфігурація обладнання, використовувана архітектура «файл-сервер або клієнт-сервер», паралельна обробка, обробка даних і т.д. наприклад, розгляд конкретного середовища або технології.

Проектування інформаційних систем мети проекту завжди починається з визначення. Як правило, мета проекту може бути визначена наступним чином, рішенням ряду взаємопов'язаних завдань, в тому числі забезпечення, коли система запускається і весь час експлуатується[7].

Необхідний рівень функціональності та відповідності системи:

- зміна умов праці;
- необхідна ефективність системи;
- час відгуку, необхідний для запиту системи;
- час роботи системи;
- необхідний рівень безпеки;
- простота використання і підтримка системи[8].

1.1.1 Класифікація сучасних інформаційних систем

Інформаційні системи можна класифікувати по ряду різних ознак.

Ця класифікація заснована на найбільш важливих характеристиках, які визначають функціональні можливості та особливості побудови сучасних систем. Пов'язаний залежно від обсягу розв'язуваних завдань і використовуваних технічних засобів,функціонуючі організації діляться на ряд

груп (класів).

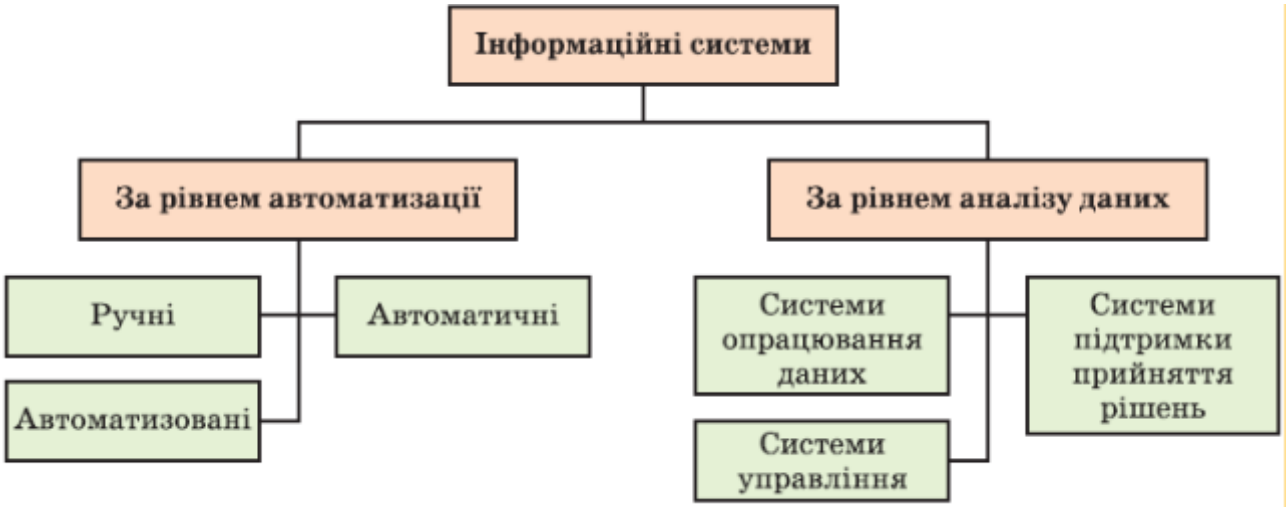


Рисунок 1.1 – Класифікація інформаційних систем

1. Залежно від типу збережених даних ІС є фактичні і документальні. Фактографічні системи, призначені для зберігання та обробки інформації структуровані дані у вигляді цифр і текстів. Документарні системи інформація подається у вигляді документів, що складаються з найменувань,пояснення.

2. Ступінь автоматизації інформаційних процесів система керування підприємством, розділена на інформаційні системи вручну,автоматичний і автоматизовані. Ручні ІС характеризуються відсутністю сучасних технічних засобів обробка інформації та виконання всіх операцій однією людиною.Всі обчислювальні операції виконуються в автоматизованому ІС без втручання людини. Автоматизовані інформаційні системи припускають участь в процесі обробки інформації і людини, і технічних засобів, причому основна роль у виконанні однотипних завдань операцій обробки даних використовується комп’ютер[7,8].

Залежно від характеру обробки даних, інформаційні системи ділиться на отримання (посилання) і декодування інформації.

В даний час було створено і успішно експлуатовано велику кількість інформаційних систем для різних цілей; запити інформації про Користувача. Характерна особливість таких систем – Інформація, що міститься відповідно до його запиту, не використовується безпосередньо у тій же системі, заданій

користувачем інформація, отримана для будь-яких цілей, яких вона потребує. Пошук є одинм з основних операціямй в таких системах, отже, системи інформаційнопошуковими (ІПС).

Введення інформації в пошукових системах, здійснює систематизацію декомпозиції,зберігання та доставка за запитом користувача без складнихперетворень даних.

В силу характеру використання інформації про вихід такі системи часто діляться на управлінські та консультаційні. Отримана інформація менеджерів ІС перетворюється безпосередньо в людські рішення. Вона характеризується обчислювальними завданнями для цих систем і задається обробкою великих обсягів даних.

Управлінська система генерує інформацію, яка замість того, щоб ініціювати певні дії, залучається до уваги людини і враховується при формуванні управлінських рішень. Ці системи імітують інтелектуальні процеси обробки інформації, а не дані[8].

Залежно від області застосування розрізняють наступні класи ІС. Інформаційні системи корпоративного керуванняпромислові підприємства і непромислові об'єкти призначена для автоматизації функцій такого управлінського персоналу. До основних функцій таких систем відносяться: оперативний контроль і регулювання, оперативний облік і аналіз прогнозної звітності, оперативне планування, бухгалтерський облік, керуванняпродажами, постачання та іншіорганізаційні завдання. ІС керуваннятехнологічними процесами - слугують для автоматизації функцій персоналу з контролю та керуваннявиробничими операціями. В таких системах він часто використовується для визначення параметрів технологічних процесів дає можливість існування сучасних засобів вимірювальних процедур для перевірки прийнятності значень параметрів і регулювання технологічних процесів. Автоматизоване проектування ІС (САПР) - призначена для автоматизації функцій інженерів-конструкторів, проектувальників, архітекторів, проектувальників при створенні нової

машини або технології. Основними функціями таких систем є: інженерні розрахунки, створення графічних документів створення документації, моделювання об'єктів для проекту.

Подляють, ІС однокористувацький, корпоративні (інтегровані) і він належить компанії[15].

Інтегровані (корпоративні) ІС використовуються для автоматизації всіх функцій фірми і охоплюють весь бізнес-цикл, починаючи від планування заходів і закінчуючи продажем продукції. Вони включають в себе сукупність модулів (підсистем), що функціонують в єдиній інформаційній області і виконують функції підтримки суміжних областей діяльності.

Аналіз сучасного стану ринку ІС показує попит до збільшення послуги корпоративного керування. А попит на повністю інтегровані системи керуванняпродовжує зростати. Автоматизація окремої функції, такої як бухгалтерський облік або облік продажів готової продукції, вже вважається давно минулим.

Існує класифікація ІС в залежності від того, який рівень керуваннясистемою використовувати.

Інформаційна система операційного рівня-підтримує продуктивність, обробка даних про транзакції та події (рахунки-фактури, рахунки-фактури, зарплата, кредити, потік). Інформаційна система операційного рівня-це з'єднання зв'язок між фірмою і зовнішнім середовищем. Експертні інформаційні системи-підтримка роботи з даними та інформація, підвищення ефективності та ефективності виробництва інженерів і дизайнерів. Завдання таких систем, їх нова інформація, організація і допомога в обробці паперових документів. Інформаційні системи на рівні управління-використовується співробітники середнього керуваннядля моніторингу, контролю, прийняття рішень та управління. Основні функції цих інформаційних систем система[15]:

- порівняння поточних і минулих показників;
- створення періодичних звітів протягом певного періоду часу, а не звітність по ньому;

- поточні події, як на оперативному рівні;
- надання доступу до архівної інформації тощо.

Стратегічна інформаційна система-комп'ютерна інформаційна система, це система, яка підтримує прийняття рішень про реалізацію стратегічних цілей довгострокові цілі розвитку організації. Інформаційні системи на стратегічному рівні допомагають на вищій ланці управлінців вирішувати неструктуровані завдання, здійснювати довгострокове планування.

Структура файл-сервер - перша Інформаційна система. Як модулі реалізації, так і дані розміщуються в окремих модулях. Файли ОС отримують доступ до даних по шляху і використовують файлові процеси (відкриття, читання, запис). Для зберігання даних використовується призначений для користувача сервер(окремий комп'ютер) - файловий сервер. Збережені організаційні одиниці, або на робочих станціях, або на файловому сервері. В останньому випадку процедури керуваннябули спрощені, але вимоги до надійності мережі зростали.

Структура клієнт-сервера - це нова архітектура. Модель замінює застарілі поняття і його задача полягає в тому, що клієнт (організаційний підрозділ) запитує певні послуги відповідно до протоколу обміну даними. Однак, на відміну від ситуації з файлами сервера, немає необхідності використовувати прямі шляхи операційної системи: клієнт «не знає» їх, але він «знає» ім'я джерела даних та іншу особисту інформацію, використовувану для авторизації клієнта на сервері. Сервер, який може фізично перебувати на тому ж комп'ютері або на іншому кінці світу, обробляє запит клієнта, після виконання відповідних маніпуляцій з даними передає запитану частину даних клієнту.

У напрямку клієнт-сервер є два основних "акценти": "тонкий" і " товстий". Системи на базі тонкого клієнта засновані на використанні потужного сервера баз даних, високопродуктивного комп'ютера і так званих процедур, що дозволяють проводити облік і реалізацію продукту базової обробки даних безпосередньо на сервері. Відповідно, клієнтська програма знижує вимоги до обладнання робочих станцій. Головною перевагою цих

систем є відносна наявність ліцензій у абонентських станцій. Навпаки, клієнт-інтенсивні системи реалізують основну логіку обробки даних.

На клієнті і сервері це чисто сервер баз даних, який забезпечує виконання тільки стандартизованих запитів на маніпулювання даними (як правило, читання, запис або зміна даних в таблицях реляційних баз даних). У системах цієї категорії вимоги до робочої станції вище, а до сервера нижче. Перевага архітектури полягає в тому, що сервер можна переміщати.

Компоненти серверів різних виробників: промислові сервери, реляційні бази даних підтримують стандартну мову обробки даних SQL, але кожен сервер має свою внутрішню вбудовану обробку лінгвістичних даних, необхідних для реалізації логіки обробки на сервері.

1.1.2 Методика проектування сучасних інформаційних систем

Всі методи проектування ІС, можна класифікувати на:

За виконанням тех.процесу проектування, аналіз, синтез, декомпозиція, моделювання та фаомалізаційні моделі.

По ступеню автоматизації проектної роботи (Самостійне (оригінальне), типове (більш індивідуальний підхід), автоматизоване).

По рівню організації процесів проектування – мають різні методи організації.

Метод виконання тех.процесів проектування використоувує, аналіз, ситез, та метод декомпозиції, який в свою чергу поділяється на два напрямки. Декомпозиція даних, розділення даних на прості складові з виявленням взаємодій між ними. Декомпозиція процесів, допомогає створити «профіль транзакцій», графічне зображення процесів обробки певного скупчення даних, транзакція починає формуватись на етапі проектування системи[7,8].

Метод, який характерезується автоматизацією проектної роботи, його проектування передбачує, що всі проектні роботи орієнтуються на створення індевідуального проекту для замовника (підприємства) , де

будуть включені і враховані всі специфічні особливості замовлення. Типова частина проектування, залежить від рівня декомпозиції яка була здійснена на рівні задач та інших проектних рішень, на базі математичного, технічного, інформаційного та програмного забезбечення, для кожного компоненту. Автоматизоване проектування в свою чергу створює ІС на основі САПР, які грунтуються на глобальній інформаційні моделі. Така модель містить в собі формалізований опис компонетів ІС та їх відношення один до оного, включаючи їх агоритми і зв'язки. Для такого проектування ІС, використовують стандарті засоби ОС, САПР, пов'язані інструментні засоби проектування та засоби для модернізації існуючої ІС. Організаційний метод проектування, охоплює ланку послідовного створення і проектування, набору спеціалістів для певного етапу, забезпечує якісну документації по кожному етапу проектування, контролює етапи проектування, забезпечує виконання інтересів інформаційних так і програмних. Для такого моделювання належить метод «зверху вниз», яка характерезується поетапним виконанням процесу проектування, яке схоже на «дерево», проеткування тут можна розпочати з любої задачі, або взагалі робити їх паралельно.

Модульний метод, зв'язаний з програмним і інформаційним забезпеченням з купою незалежних модулів. Такий метод надає змогу проектувати оптимального синтезу функціонально незалежних частин так званих модулів, які разом формують та виконують задані функції системи з необхідною ефективністю.

Структурний метод, передбачає наяність програми , що буде денамічно підстроюватись на структурні масиви інформаційного фонду системи. Але масив слід формалізувати, а збереження і підтримка повинні бути організовані в систему інформаційного фонду.

Метод який базується на «математичній моделі» передбачений для рішення задач, вибір та розробку економічно-математичної моделі, яка являє собою алгоритм розв'язання та складування прикладних програм.

Метод безпереривного розвитку системи полягає в тому, що після створення ІС в процесі її функціонування , зявляються нові, міняються діючі постановлені задачі[8].

Методи проектування ІС дозволяють підвищити якість створюваних проектів, підвищення продуктивності праці спеціалістів, спеціалістів - розробників проекту, зниження вартості та витрат при проектуванні, зменшення термінів на проектування та спрощення впровадження, модернізацію та супроводу функціонуючої ІС.

1.2 Аналіз сучасних систем керування і засобів розробки БД

1.2.1 Основні моделі зберігання данних

Модель даних - це інструмент для визначення логічного представлення фізичних даних, пов'язаних з додатком. У процесі розробки основ створення баз даних Вибір моделі в основному залежить від об'єкта впровадження, а також від стадії впровадження. Записи, що зберігаються в базі даних, мають логічну структуру-модель представлення даних. Раніше найбільш поширеними моделями були ієрархічні, мережеві та реляційні. В даний час розробляються нові моделі: постріляційні, багатовимірні і цілеспрямовані. Зібрані моделі будуються на основі цих моделей: реляційної, дедуктивно-об'єктивної, індикативної та концептуальної. Деякі підтримують моделі з декількома даними (кеш) одночасно. Перш ніж приступити до вивчення реляційних систем баз даних, необхідно ознайомитися з реляційними базами даних, де внутрішня організація реляційних систем багато в чому заснована на використанні методів і систем заздалегідь, це буде корисно для розуміння шляхів розвитку постріляційних баз даних. Розглядом загальніпідходи до регулювання трьох типів ранніх систем[8]:

Реляційних (SQL)

NoSQL (NOT ONLY SQL)

Об’єктно орієнтовані (Object-oriented Database)

Ці системи активно використовуються в різних компаніях за різним

призначенням. Інтерактивний доступ до бази даних підтримувався тільки розробкою відповідних додатків з локальним інтерфейсом. Навігаційний характер цих систем і доступ до даних на рівні записів користувачів змусили максимально використовувати базу даних без будь-якої підтримки з боку систем. Після введення цих систем більшість з них обладнуються користувацькими (графічними) інтерфейсами. Однак у більшості випадків це насправді не робить їх комфортними для роботи з ними, оскільки даними все ще можна керувати в їх перивиному стані.

1.2.2 Реляційна модель

Реляційна база даних-це сукупність взаємопов'язаних таблиць, кожна з яких містить інформацію про конкретний тип об'єкта. Рядок таблиці містить дані про один об'єкт (наприклад, продукт або клієнт), а стовпці таблиці описують різні характеристики цих об'єктів атрибутів (наприклад, ім'я, код продукту та інформацію про клієнта). Записи, тобто рядки таблиці, мають однакову структуру: вони складаються з полів, в яких зберігаються атрибути об'єкта. Кожне поле, тобто стовпець, описує тільки одну характеристику об'єкта і має строго певний тип даних.

Всі записи мають однакові поля, але вони відображають різні інформаційні властивості об'єкта.

Для роботи з даними використовуються системи керуваннябазами даних. Основні функції :

- Визначення даних (опис структури бази даних);
- Обробка отриманих даних .
- Керуванняданими.

Структура бази даних є найбільш важливим завданням, яке необхідно вирішити при проектуванні бази даних. Структура бази даних (таблиця, формат і відносини) є одним з конструктивних рішень при створенні додатків з базою даних. Структура БД, створена розробником, описується мовою визначення даних . Це дозволяє виконувати наступні операції з даними :

- Додавання записів в програму;
- Видалення записів з таблиці;
- Оновлення значень деяких полів в одній або декількох записах таблиць бази даних;
- Пошук однієї або декількох записів, що задовольняють вказану вимогу.
- Для виконання цих операцій використовується механізм запиту.

Результати проведених розслідувань перетворюються в набір записів або таблиць, відібраних на основі конкретних критеріїв. Запит бази даних, створеної на спеціально створеній мові, називається "Structured Query Language" (SQL).

Керуванняданими, часто як захист даних від несанкціонованого доступу, підтримує кілька користувачів для керуванняданими і забезпечує цілісність і узгодженість даних. Приклади реляційних баз даних,стандартні реляційні бази даних дозволяють користувачам керувати заздалегідь визначеними відносинами даних між кількома базами даних. Популярні приклади стандартних реляційних баз даних включають Microsoft SQL Server, Oracle Database, MySQL і IBM DB2[7,8].

1.2.3 NoSQL модель даних

NoSQL (Not only SQL) - СКБД не використовують реляційну модель структуризації даних. На відміну від традиційних СКБД, деякі бази даних NoSQL, наприклад, MongoDB, дозволяють групувати колекції даних з іншими базами даних. Такі СКБД зберігають дані як одне ціле. Ці дані можуть являти собою одиночний об'єкт на зразок JSON і разом з тим коректно відповідати на запити до полів. NoSQL бази даних не використовують загальний формат запиту (як SQL в реляційних базах даних). Кожне рішення використовує власну систему запитів.

NoSQL не означає відсутність SQL; замість цього це означає не тільки SQL. Іншими словами, ми не обмежені одним варіантом збереження та вилучення даних. Як випливає з його назви, не існує фіксованого визначення того, що

NoSQL є або робить, але наступні поняття можуть допомогають зрозуміти, що таке NoSQL: 1)Немає реляційної моделі; 2)Дані можуть бути кластеризовані; 3)Більшість рішень з відкритим вихідним кодом; 4)Створено для Інтернету; 5)Відсутність схеми. Як і в будь-якії іншої системи керування базами даних, в NoSQL, маються свої переваги та недоліки[8]:

Переваги:

- Можливість зберігання великих обсягів неструктурованої інформації. У NoSQL немає обмежень на типи збережених даних, а при необхідності можна додавати нові типи даних.
- NoSQL-бази краще піддаються масштабуванню. Хоча масштабування підтримується і в SQL-базах, це вимагає набагато більших витрат людських і апаратних ресурсів. Використання хмарних обчислень і сховищ.
- Використання для тестування і розробки локального обладнання, а потім перенесення системи в хмару.
- Швидка розробка. NoSQL бази даних не вимагають великого обсягу підготовчих дій, який потрібен для реляційних баз.

Недоліки:

- Труднощі швидкого переходу з однієї нереляційної бази даних на іншу.
- Доводиться розробляти власні інструменти для роботи з БД.
- Додаток сильно прив'язується до конкретної СУБД. Мова SQL універсальна для всіх реляційних сховищ і тому в разі зміни СУБД не доведеться переписувати весь код.

1.2.4 Обєкто-орієнтована модель зберігання даних

Об'єктно-орієнтовані бази даних-бази даних, в яких інформація представлена у вигляді об'єктів, як в об'єктно-орієнтованих мовах програмування. Об'єктно-орієнтовані бази даних зазвичай рекомендовані для тих випадків, коли потрібна високопродуктивна обробка даних, що мають складну структуру[6]. Основні труднощі об'єктно-орієнтованого моделювання

даних виникають з того, що такого розвиненого математичного апарату, на який могла б спиратися загальна об'єктно-орієнтована модель даних, не існує. У великій мірі тому досі немає базової об'єктно-орієнтованої моделі.

Об'єктно-орієнтований підхід базується на концепціях:

- об'єкта та ідентифікатора об'єкта;
- атрибутів і методів;
- клас;
- ієрархії та успадкування класів.

Об'єктно-орієнтована модель бази даних є альтернативною реляційній моделі. Об'єктно-орієнтована база даних чимось схожа на об'єктно-орієнтовану мову програмування. Система управління об'єктно-орієнтованою базою даних являє собою гібридний додаток, який використовує комбінацію принципів об'єктно-орієнтованої і реляційної бази даних для обробки даних[6]. Його соновною перевагою є, інтегроване сховище інформації, яке використовується декількома користувачами, декількома продуктами, декількома додатками на декількох платформах.Це також вирішує наступні проблеми:

1. Реальна модель, майже не відрізняється від концептуальної моделі.

2. Мови програмування та системи баз даних повинні взаємодіяти для вирішення проблем програми. Але стиль мови, структури даних мови програмування (наприклад, C#) і СКБД (наприклад, MySQL) різні. ООБД підтримує програмування загального призначення в рамках OODB.

3. Нові вимоги до додатків: особливо в OA, CAD, CAM, CASE, об'єктна орієнтація є найбільш природною і зручною.

А недоліками такої системи є:

1.Обмежена підтримка обмежень цілісності.Відсутні механізми оголошення ключових властивостей атрибутів (наприклад, атрибут класу не може бути оголошений первинним ключем класу), або обмежень унікальності, явних обмежень цілісності, а також перед - і постумов методів.

2.Обмежені можливості налаштування продуктивності.У більшості

ООБД є лише обмежені засоби параметризованої настройки продуктивності. У РБД інсталяторам надається можливість налаштовувати продуктивність системи шляхом завдання великого числа параметрів, що встановлюються системним адміністратором.

3.Недостатня підтримка складних об'єктів.Повна функціональність складних об'єктів все ще не підтримується. Можна здійснювати навігацію по посиланнях і кодувати операції із застосуванням цих посилань, але відсутні зумовлені родові операції, в яких використовуються різні види семантики посилань.

4.Обмежена інтеграція з об'єктно-орієнтованими системами програмування. Конфлікти по іменах; необхідність переробляти ієрархії класів; схильність ООБД до перевантаження системних операцій[6,11].

1.2.5 Вибір архітектури БД

Існують дві архітектури, які створені на системі бази даних, технології обробки даних: Архітектури файл-сервер та клієнт-сервер. Клієнт ,це програма що дає змогу користувачеві отримати доступ до ресурсів на сервері. Детально розберемо кожну з них:

Архітектура файл-сервер, являє собою допоміжну ланку. Вся обробка інформації, обчислення, цілісність БД і тд, залежить від додатку, що запущений на робочій станції. Завдяки тому, що, обробка даних здійснюється робочою станцією, по мережі протікає необхідна інформація для цієї обробки даних.

Був досить довгий час, основним методом, але через великий обсяг даних, які передаються по мережі, швидко її перевантажує, навіть при малій кількості користувачів, тим самим зменшуючи зростання такої БД, до того ж має не високі показники надійності збереження та обробки даних.

Архітектура клієнт-сервер, використовує спеціальне ядро-сервер, для обробки даних, воно бере на себе обробку запитів користувача (клієнта). Користувач посилає запит, на вибірку даних, наприклад на оновлення

,видалення, вставку. Сервер в свою чергу, аналізу, відбирає і вже відправляє тільки необхідну користувачеві вибірку даних.

Клієнт-сервер - обчислювальна або мережева архітектура, в якій завдання або мережеве навантаження розподілені між постачальниками послуг (сервісів), так званими серверами, і замовниками послуг, тобто клієнтами. Часто клієнти і сервери взаємодіють через комп'ютерну мережу і можуть бути як різними фізичними пристроями, так і ПЗ.

Клієнт-серверна архітектура, зберігає логічну цілісність бази даних, що означає, що ця система є більш стійкою. Це можливо завдяки здатності передачі турботи за цілісністю бази, серверу.

Для захисту системи від невірних дій користувача, сервер має велику кількість наборів і вбудованих в нього захисних механізмів. З цих механізмів, можна виділити такі, як віртуальна таблиця, авторизація користувача, обмеження цілісності, декларування цілісності посилань, та інші.

Клієнт-сервер, є основною архітектурою для багатьох теперішніх систем, які працюють з колективним користуванням даних[7].

До переваг клієнт-серверної архітектури:

- Дозволяє, розподіляти обчислювати функції системи, на декількох незалежних комп’ютерах в мережі. Це спрощує обслуговування системи, спрощує ремнот, заміну чи модернізацію, при цьому ніяк не впливаючи на роботу клієнта.
- Дані які зберігаються на сервері, загалом є більш захищеними, ніж більшість клієнтів. Права доступу, контроль повноважень, легше контролювати на сервері, щоб доступ мали тільки вповноважені клієнти.
- Дає змогу об’єднати різні клієнти. Дозволяє використовувати ресурси сервера з різних платформ, та з різним ПО.

Недоліками ж є:

- Непрацездатність сервера, може «положити» всю обчислювальну мережу.
- Для підтримки безперебійної роботи, потрібен спеціальний майстер,

системний адміністратор.

- Висока вартість обладнання.

«Товстий» клієнт – це більш поширений варіант яким реалізують архітектуру «клієнт-сервер», в впроваджених і жваво використовууваних системах. Ця модель, являє собою поєднання в додатку клієнта, презентаційної-логіки, та бізнес логіки. Серверна ланка, при такому підході є сервером БД, який дає змогу реалізувати логіку, щодо дотупо до ресурсів.

Переваги:

- Має більш широкий функціонал, ніж у «тонкого».
- Мультикористувацький режим.
- Дає змогу продовжити роботу, навіть при обриві зв’язку з сервером.
- Підключення до БД ,без доступу до інтернет мережі.
- Висока швидкість.

Недоліки:

- Дистрибутив має великий розмір.
- Залежність від платформи розробки (в роботі клієнта)
- Тяжко реалізувати віддалений доступ до інформації.
- Складний процес монтування так і налаштування.
- Неактуальність даних, через складність оновленя.

«Тонкий» клієнт – комп’ютер-клієнт мережа, яка базується на архітектурі «клієнт-сервер», який перекладає більшу частину задач по обробці даних на сервер. Така модель, популярна в корпоративному середовищі, та активно використовується через доступність і поширеність інтерне-технологій, головним чином, веб-браузерів. В такій архітектурі додаток клієнта реалізує презентаційну логіку, а сервер вже єднає як бізнес-логіку так і логіку доступу до ресурсів.

Термін «тонкий клієнт» являє собою, з точки зору архітектури, широкий ряд пристроїв і програм, які об’єднують їх властивості: здатність працювати в термінальному режимі. Тим самим для роботи з «тонким» клієнтом, потрібен термінальний сервер. Це головна відмінність тонкого клієнта від товстого,

який обробляє дані незалежно від серверу, та використовує його лише для зберігання інформації. Для розширення його функціоналу, додають можливості автономної роботи, але зберігають його відмінність- сесійну роботу з терміналом серверу.

Коли до клієнта додається жорсткий диск, з'являється змога автономної роботи і він стає універсальним клієнтом, тим самим перестає бути «тонким» клієнтом у чистому його виді. Трьохрівнева архітектура, в комп'ютерних технологіях, передбачає наявність наступних компонентів: Клієнтський додаток (тонкий клієнт чи термінал), під'єднаний до серверу додатків, який вже приєднаний до серверу БД[8].

На першому рівні, виноситься проста бізнес-логіка, інтерфейси для авторизації, шифрування, перевірка на вірність формату, задачами які вже завантажені в термінал, наприклад, сортування, групування і тд.

Другий рівень, представляє собою сервер додатків. Вже на ньому, зосереджена більша частина бізнес-логіки, сюди вже не потрапляють фрагменти експортовані в термінал, так і процедури які відбуваються на третьому рівні.

Третій рівень, це сервер БД, який забезпечує зберігання даних. В основному це реляційна або об'єкто-орієнтована .

Порівняно з клієнт-сервер та файл-сервер, архітекторами, трирівнева архітектура має такі переваги:

- Масштабування.
- Висока надійність та безпека.
- Низька планка для швидкості між терміналом і сервером.
- Ізоляція рівнів один від одного.
- Низька потреба в продуктивності апаратних характеристик терміналу, це дає змогу зробити терміналом навіть смартфон.

Недоліки випливають з переваг:

- висока складність створення додатків;
- Складне розгортанні та адміністрування;

- високі вимоги до продуктивності серверів додатків і сервера бази даних як слідство і висока вартість серверного обладнання;
- високі вимоги до швидкості каналу (мережі) між сервером бази даних і серверами додатків.

1.3 **Вибір системи автоматизованого проектування БД**

Система керування базами даних-це програма, яка дозволяє адміністраторам отримувати доступ до бази даних, змінювати її та аналізувати.

Існує кілька адміністративних завдань, необхідних у базі даних, таких як відстеження змін або резервне копіювання та відновлення даних, і правильне програмне забезпечення дозволяє вам легше вносити зміни на рівні системи, визначаючи при цьому будь-які проблеми. За допомогою адміністратори можуть керувати даними, ядром бази даних і схемою бази даних, встановлюючи логічну структуру. Тим не менш, дуже важливо переконатися, що ми отримуємо максимальну віддачу від , це дає здатність контролювати бази даних і налаштовувати продуктивність бази даних, щоб вона залишалася ефективною[13].

Є кілька різних схем баз даних, кожна з яких має свої переваги і недоліки. Вибір типу бази даних може бути справою балансування таких факторів, як організація, безпека, обсяг сховища та ефективність. Для початку вибір буде залежати від того, який з основних типів баз даних використовується. Система керуванняреляційними базами даних, це свого роду , заснована на реляційній моделі даних. Реляційні бази даних які ще називають базами даних SQL, (Structured Query Language), що використовується для програмування більшості програм керуванняреляційними базами даних. Реляційна база даних по суті малює зв'язки між таблицями або електронними таблицями всередині бази даних. Коли додаються дані до бази даних, просте додавання стовпців до тієї ж таблиці створить статичний і негнучкий набір даних; з часом ваша база даних стане заплутаною та марною. Але в реляційній базі даних таблиці повинні мати стовпець первинного ключа, який однозначно ідентифікує всі

дані в рядку. Потім таблиці можуть використовувати зовнішній ключ для зв'язку з первинним ключем іншої таблиці. Однією з великих переваг реляційних баз даних є узгодженість, ізольованість і відповідність вимогам довговічності (ACID) [16].

1.3.1 Сервер БД MySQL

Є багато причин для популярності MySQL у всьому світі, але одна з головних причин-це його архітектура, в той час як є багато великих гравців, таких як Oracle, Microsoft SQL і DB2, архітектура MySQL робить його унікальним і кращим вибором для більшості розробників. Обговоримо внутрішню архітектуру системи керуванняреляційними базами даних MySQL[16].

Основних компонентів:

Архітектура MySQL описує, як різні компоненти системи MySQL співвідносяться один з одним. Архітектура MySQL - це в основному клієнт-серверна система. Сервер баз даних MySQL-це сервер, а додатки, що підключаються до сервера баз даних MySQL, - це клієнти. Архітектура MySQL містить наступні основні компоненти.

прикладний рівень:

Цей шар є самим верхнім шаром в архітектурі MySQL; ви можете бачити цей же шар у багатьох архітектурах клієнт-сервер. Цей рівень включає в себе деякі служби, які є загальними для більшості клієнт-серверних додатків, деякі з них наведені нижче:

- Обробка З'єднань.
- Ідентифікація.
- Безпека.

Обробка з'єднань:

Коли клієнт підключається до сервера, він отримує свій власний потік для підключення. Всі запити від цього клієнта виконуються в межах зазначеного потоку. Потік кешується сервером, тому вони не повинні

створюватися і знищуватися для кожного нового з'єднання.

Ідентифікація:

Всякий раз, коли клієнт підключається до сервера MySQL, сервер виконує аутентифікацію на стороні сервера. Аутентифікація заснована на імені користувача, хості клієнта і паролі користувача клієнта.

Рівень сервера MySQL:

Цей рівень піклується про всі логічні функції системи керуванняреляційними базами даних MySQL. Мозок сервера MySQL знаходиться в цьому шарі. Логічний шар MySQL розділений на різні субкомпоненти, які наведені нижче:

- Сервіси та утиліти MySQL.
- Інтерфейс SQL.
- Парсер SQL.
- Оптимізатор.
- Схованки і буфери.

Сервіси та утиліти MySQL:

MySQL порівняно надає широкий спектр послуг і утиліт. Це одна з головних причин популярності MySQL. Цей рівень надає послуги та утиліти для адміністрування та обслуговування системи MySQL, деякі з них згадуються нижче:

- Резервне копіювання та відновлення.
- Безпека.
- Копіювання.
- Скупчення.
- Розбиття.
- Верстак.

Інтерфейс SQL:

Structured Query Language (SQL) - це мова запитів, що використовується для запитів до сервера MySQL. Це інструмент для взаємодії між Користувачем клієнта MySQL і сервером. Деякі компоненти інтерфейсу SQL наведені нижче.

- Мова обробки даних (DML).
- Мова визначення даних (DDL).
- збережені процедури.
- Перегляд.
- Подразник.

Синтаксичний аналізатор:

MySQL аналізує запити, щоб створити внутрішню структуру (дерево синтаксичного аналізу). Парсер MySQL поводиться як компілятор з одним проходом. Згідно з внутрішніми даними MySQL, структура парсера наведена нижче.

Лексичний аналіз (складання слів або лексем з символьного потоку) реалізується на першому етапі, при розборі регулярних виразів.

Синтаксичний аналіз(складання "пропозицій"), семантичний аналіз (перевірка того, що ці пропозиції дійсно мають сенс) і генерація коду (для компіляторів) – всі вони виконуються одночасно, на етапі коду[13].

Оптимізатор:

Після створення внутрішнього дерева синтаксичного аналізу MySQL застосовує різні методи оптимізації. Ці методи можуть включати в себе Переписування запиту, порядок сканування таблиць і вибір правильних індексів для використання. Насправді ви можете попросити сервер пояснити різні аспекти оптимізації.

У кеші MySQL (query cache) зберігаються повні результуючі набори для операторів SELECT. Навіть перед розбором запиту сервер MySQL консультується з кешем запитів. Якщо будь-який клієнт видає запит, ідентичний тому, який вже знаходиться в кеші, сервер просто пропускає синтаксичний аналіз, оптимізацію і навіть виконання, він просто відображає вихідні дані з кешу.

Движок зберігання:

Функція Pluggable storage engine робить MySQL унікальним і кращим вибором для більшості розробників. Це та особливість, яка дозволяє MySQL

досягти переваги над великим гравцем. MySQL дозволяє нам вибирати різні механізми зберігання даних для різних ситуацій і вимог. Ми збираємося обговорити особливості кожного механізму зберігання даних в наступній статті, просто список підтримуваних механізмів зберігання даних наведено нижче:

- Движок MyISAM.
- Движок InnoDB.
- Mrg_MyISAM.
- CSV.
- Пам'ять.
- Архів.
- Performance_schema.

MySQL надає це в якості підключаються механізмів зберігання, різні механізми зберігання можуть бути використані на рівні таблиці. База даних може містити таблиці з декількома механізмами зберігання.

mysql > Show Engine;

Команда Show Engine виведе список всіх движків зберігання, підтримуваних вашим сервером. Я сподіваюся, що цей пост дасть вам загальне уявлення про архітектуру MySQL. У наступному пості давайте обговоримо різні механізми зберігання даних і їх особливості.

1.3.2 Система автоматизованого програмування БД SQL Toad Data Modeler

Сучасні автоматизовані системи обробки інформації є складними і надскладними системами, що враховують як елементи і зв'язки окремих найпростіших процесів (інформаційних і матеріальних), так і узагальнені структури і потоки між ними. Проектування подібних систем в даний час пов'язано з графічнимпредставленням моделі даних для реляційних систем керуваннябазами даних.

Найбільш широко використовуваною для такого представлення моделлю

є модель «сутність-зв'язок, що дозволяє визначити основні інформаційні об'єкти «сутності» і відносини між ними «зв'язки», необхідні для роботи проектованої інформаційної системи. У класичних підручниках з проектування баз даних наводяться наступні визначення: "сутність - це клас однотипних об'єктів, інформація про які повинна бути врахована в моделі. Екземпляр сутності – це конкретний представник даної сутності. Атрибут сутності – це іменована характеристика, що є деяким властивістю сутності. Зв'язок-це деяка асоціація між двома сутність. Одна сутність може бути пов'язана з іншою сутністю або сама з собою»[1, 2].

Посилення інформатизації суспільства призводить до ускладнення проектованих систем і вимагає все більш потужних програмних засобів моделювання та проектування. Одним з таких засобів, що підтримують моделювання системи в термінах ER-діаграм, є Toad Data Modeler [3] . Дана програма являє собою " багатофункціональний інструмент розробки баз даних і додатків, який об'єднує в одному інтегрованому середовищі об'єктно-орієнтовані і концептуальні можливості моделювання фізичних даних».Володіти зрозумілим для розробників баз даних графічним інтерфейсом, підтримує велику кількість популярних (рис. 1.2.), що дозволяє його рекомендувати для проектувальників інформаційних систем.

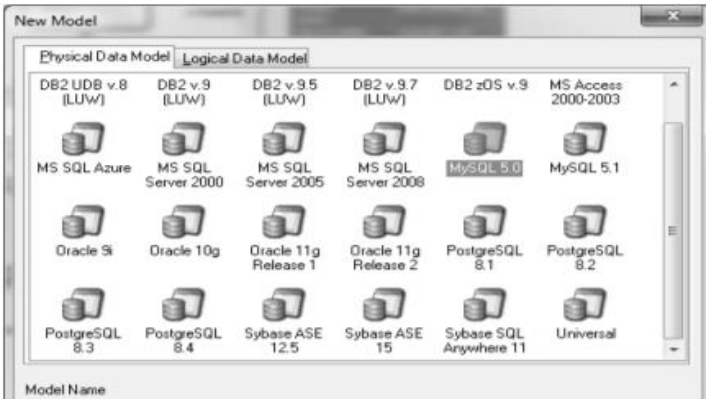


Рис. 1.2 – СКБД, з якими працює Toad Data Modeler

Методологія IDEF1 при створенні моделі сутність-Зв'язок пропонує єдині вимоги графічного позначення елементів діаграм: сутність позначається прямокутником, зв'язки – лініями між сутностями. Для кожної сутності

необхідно визначення її імені та атрибутів, що характеризують інформаційні потреби роботи системи. Інструменти по створенню моделі в термінах діаграм сутність–зв'язок в програмі Toad Data Modeler досить прості і зрозумілі розробнику, знайомому з методологією IDEF1X [3]. Вікно завдання властивостей сутності містить велику кількість вкладок, що дозволяють визначити не тільки основні властивості інформації - об'єктів (імена сутностей, бази даних, типи підтримуваних таблиць, імена та типи даних для атрибутів,первинні ключі), а й додаткові, що дозволяють сформувати бізнес-логіку обробки даних клієнт-серверних архітектур (обмеження полів, таблиць, тригери тощо).

У процесі інформаційного моделювання будується логічна інформаційна модель, що представляє собою сукупність взаємопов'язаних сутностей і їх атрибутів, що не залежить від конкретної системи керуваннябазами даних,але відображає специфіку предметної області. Будь-яка передача інформації в працюючій системі призводить до необхідності її зберігання або у вигляді сутності, або у вигляді її атрибутів. А потім формується фізична модель, відповідна обраної. Програма Toad Data Modeler встановлює спосіб перетворення логічної моделі в фізичну, відповідно до правил перетворення діаграм «сутність-зв'язок» в реляційну модель даних, а також відповідно до підтримуваних типів даних і особливостями зберігання інформації в . Ліва частина робочого вікна проектованої моделі містить ієрархічне дерево об'єктів, що входять до складу моделі, причому основні компоненти моделі змінюються в залежності від її виду (логічна або фізична). Права частина являє собою робоче поле з визначеними вмоделі об'єктами і зв'язками. Таким чином, використання даного програмного продукту істотно полегшує розробку автоматизованих інформаційних систем, дозволяючи визначити структури даних, що зберігаються, що забезпечуютьосновні інформаційні потреби користувачів

1.4 Засоби розробки додатків які працюють з БД

1.4.1 Система об’єктно - орієнтованого програмування C#

Об'єктно-орієнтоване програмування є результатом розробки процедурного програмування.Нижче наведено два різних варіанти. Однак необхідність в об'єктно-орієнтованому програмуванні полягає в розробці дещо іншого підходу до програми.

Об'єктно-орієнтоване програмування, дані і методи додаються в класи (формуються відносини між ними). Об'єкти створюються на основі класів-основних елементів програми. Під час виконання програм об'єкти взаємодіють один з одним (обмінюються інформацією). В об'єктно-орієнтованому програмуванні будь яка програма являє собою набір об'єктів, що взаємодіють один з одним.Об'єктно-орієнтовані програми називаються класами в шаблонах побудованих об'єктів. Об'єктно-орієнтоване програмування дозволяє долати труднощі при створенні складних програм. Принципи об'єктно-орієнтованого програмування:

- Обсяг;
- Успадкування;
- Поліморфізм.

Клас обсягу визначає дані та дії, які виконують ці дані. Обсяг встановлюється, таким чином, зв'язок між ними, що дозволяє поміщати дані і методи в один клас.

Принцип "успадкування" дозволяє створити базовий клас, який може бути успадкований іншими більш спеціальними (похідними) класами, які визначаються тільки тим, що надають таким елементам специфічність.

Принцип "поліморфізму" означає, що методи з однаковою назвою містять різні коди. Надає змогу працювати з об'єктом, який не знає, свою належність до лінії або класу, дозволить йому працювати[11].

Мови програмування C# - це один з найбільш просунутих об'єктно-орієнтованих мов програмування, що належать до знаменитого сімейства C-family. Мова програмування C # також підтримує компоненти програмування.

Розробка сучасних додатків все більше схиляється до створення програмних компонентів в формі автономних та самоописувальних пакетів, які реалізують окремі функціональні можливості. Спеціальна функціональність реалізована в автономних і самоописывающихся пакетах у вигляді програмних компонентів. Це одна з головних особливостей, такі компоненти являють собою атрибути з програмною моделлю, патернами і подіями. Це частина інформації про відомості, які вони надають атрибутам. C# - мовна структура, яка підтримує цю концепцію безпосередньо в роботі. Це робить C# придатним для створення і використання програмних компонентів. Функції C# для забезпечення надійності та стабільності додатків, такі як[12]:

- Освобождение в пам'яті процесів, які є невикористовуваним доступні об'єкти зайняті;
- Обробка винятків-це структурований і розширюваний метод виявлення і відновлення помилок;
- Тип мови робить неможливим читання з неможливих змінних в безпечну структуру, індексні масиви поза їх межами або індексні змінні.

1.4 .2 Опис додатку MS Visual Studio

Microsoft Visual Studio - повнофункціональна інтегроване середовище розробки (IDE) з підтримкою популярних мов програмування, серед яких C, C, C #,

Функціональність Visual Studio охоплює всі етапи розробки програмного забезпечення, надаючи сучасні інструменти для написання коду, проектування графічних інтерфейсів, збірки, налагодження і тестування програм. Можливості Visual Studio можуть бути доповнені шляхом підключення необхідних розширень[4].

Редактор коду Visual Studio підтримує підсвічування синтаксису, вставку фрагментів коду, відображення структури і пов'язаних функцій. Істотно прискорити роботу допомагає технологія IntelliSense - автозавершення коду під час введення.

Вбудований відладчик Visual Studio використовується для пошуку і виправлення помилок у вихідному коді, в тому числі на низькому апаратному рівні. Інструменти діагностики дозволяють оцінити якість коду з точки зору продуктивності і використання пам'яті.

Дизайнер форм Visual Studio незамінний при розробці програм з графічним інтерфейсом, допомагаючи спроектувати зовнішній вигляд майбутнього програми і роботу кожного елемента інтерфейсу.

Нарешті, Visual Studio надає комплекс інструментів для автоматизації тестування додатків в частині перевірки роботи інтерфейсів, модульного і навантажувального тестування.

Для командних проектів Visual Studio пропонує підтримку групової роботи, дозволяючи виконувати спільне редагування і налагодження будь-якій частині коду в реальному часі, а в якості системи керуванняверсіями використовувати Team Foundation або Git[4].

В робочому просторі MS Visual Studio, з правої сторони відображаються актуальні новини для розробників, корисна інформація щодо оновлень, та допомога в різних питаннях. Зверху зліва, відображена робоча середа, яка дозволяє створювати проекти, редагувати, відкривати, оновлювати їх. Також там можна налаштувати вид додадку, вид проекту та запустити процес відладки. Також в вкладкі «Средства», є змога під’єднатись до бази даних чи серверу, що сутєво спрощує роботу з базами даних з цього додатку, та надає змогу, працювати зними, прямо під час проектування програми у Windows Forms, що дає наглядне розуміння того, як програма буде виглядати в кінцевому результаті, що буде бачити користувач та, що саме за дані з БД, беруться для кожного пункту в додатку.

1.4.3 Модель програмування Windows Forms

У Windows Forms термін "форма" - синонім вікна верхнього рівня. Головне вікно програми - форма. Будь-які інші вікна верхнього рівня, які має додаток - також форми. Вікна діалогу також вважаються формами. Незважаючи на назву, програму для використання Windows Forms, не виглядають як форми. Подібно традиційним Windows-додатків додатки здійснюють повний контроль над подіями у власних вікнах.

Програми, що використовують Windows Forms використовують класи System.WinForms. Цей розділ включає такі класи, як Form, який моделює поведінку вікон або форм; Menu, який представляє меню; Clipboard, який дає можливість додаткам Windows Forms використовувати буфер обміну. Він також містить численні класи, які надають кошти управління, наприклад: Button, TextBox, ListView, MonthCalendar і т.д. Ці класи можуть бути включені в додаток або з використанням тільки імені класу, або з використанням повного імені, наприклад: System.WinForms.Button.

В основі майже кожної програми, написаного із застосуванням Windows Forms, - похідний клас від System.WinForms.Form. Зразок цього класу представляє головне вікно програми. System.WinForms.Form має безліч властивостей і методів, які мають багатий програмний інтерфейс до форм.

Додатки, засновані на Windows Forms, які використовують кнопки, списки та інші типи компонентів Windows, використовують класи керуванняSystem.WinForms, значно спрощують програмування управління.

Інша важлива грань моделі програмування Windows Forms - механізм, який форми використовують для відповіді на введення в меню, засобів керуванняі інших елементів GUI додатки. Традиційні Windows-додатки обробляють повідомлення WM_COMMAND і WM_NOTIFY використовуючи події процесу Windows Forms. У C # і на інших мовах, які підтримують .NET Common Language Runtime (CLR), події - члени типу першого класу нарівні з методами, полями і властивостями. Фактично всі керуючі класи (control classes) Windows Forms (а також і багато некеровані класи) створюють події.

Наприклад, кнопка (екземпляр System.WinForms.Button) після натискання створює подія Click. Форма, яка повинна відповісти на натискання кнопки може використовувати наступний код, щоб з'єднати кнопку на обробником події Click:

```
MyButton.Click + = new EventHandler (OnButtonClicked);
private void OnButtonClicked (object sender, EventArgs e)
{ MessageBox.Show ( "Click!");
}
```

EventHandler - спеціальний обробник подій, який виконує метод OnButtonClicked коли MyButton створює подія Click. Перший параметр OnButtonClicked ідентифікує об'єкт, який викликав подія. Другий параметр в основному має сенсу для події Click, але використовується деякими інші типами подій, щоб передати додаткову інформацію[5].

2. ПРОЕКТУВАННЯ ЕЛЕМЕНТІВ УПРАВЛІННЯ ВІДДІЛУ ІТ

2.1 Визначення завдань бази даних

Перш ніж приступити до побудови додатки, необхідно мати чітке подання, що потрібно зробити. Необхідно створити докладний списоск,всіх основних,завдань якінеобхідно вирішуватиздопомого ю додатки, включаючи завдання, які не стоять прямо зараз, але можуть виникнути в майбутньому. В як найбільш важливих завдань бази даних «інтернет – провайдера X-Byte» можна виділити наступні:
Відображення основної інформації про клієнтів, працівників, замовлення, тарифах, послуги, оплаті;

- Можливість додати нового співробітника, нове замовлення, новий тариф, нового клієнта;
- Відображення днів народжень співробітників і клієнтів;
- Отримання інформації про клієнтів\співробітників;
- Здійснення різного роду пошуку по заданих критерієм (вартість тарифу, за прізвищем співробітника або клієнта та інше.)

Для більш детального вивчення предметної області, отже, більш якісного проектування БД і програми, виділимо основні бізнес процеси, побудуємо діаграми варіантів використання і діаграми активності.

2.2 Бізнес-процеси відділу ІТ

Для початку виділимо основні типи користувачів системи та їх основні взаємодії з іншими елементами системи:

Клієнти- бажають отримати певний перелік послуг, здійснюють оплату за надані послуги відповідно з тарифами і тарифікацією, звертаються за наданням інформації чи звітів.

Відділ ІТ – приймає замовлення від відділу по роботі з клієнтами, реєструють замовлення в «картотеці» і формують робочі задачі для співробітників.

Системні адміністратори – працюють з серверним обладнанням, та працюють з іншою технікою (маршрутизатори, роутери, свічі і інше).

Керівники компанії – запитують звітність.

Відповідно з представленої структурою можна уявити бізнес-процеси, протікають в предметній області, в наступному вигляді (див. рисунки 2.1-2.3).

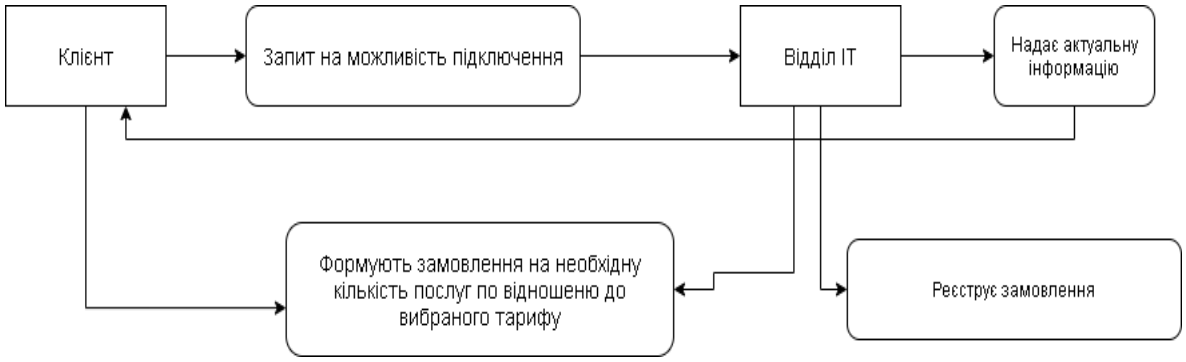


Рисунок 2.1 – Створення замовлення

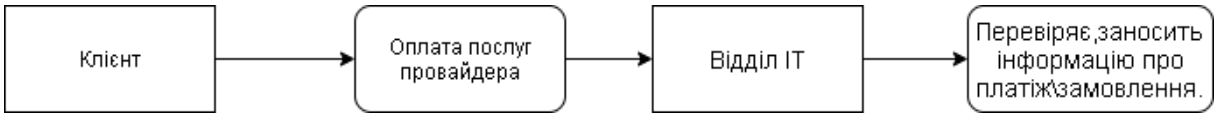


Рисунок 2.2 – Інформація про оплату послуг

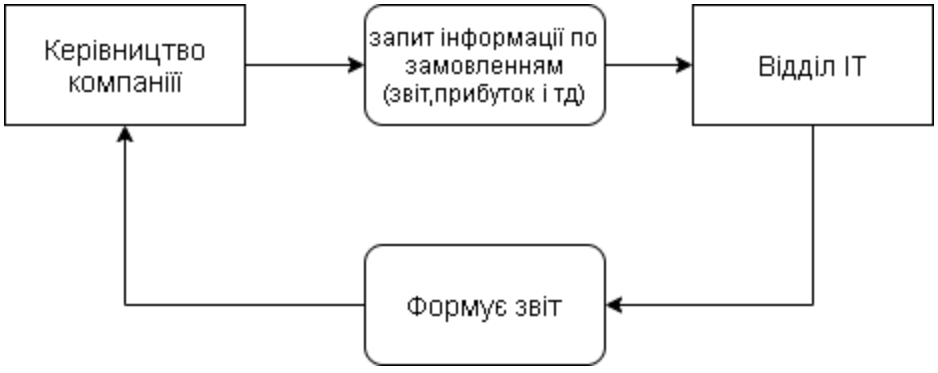


Рисунок 2.3 – Взаємодія керівництва з відділом IT.

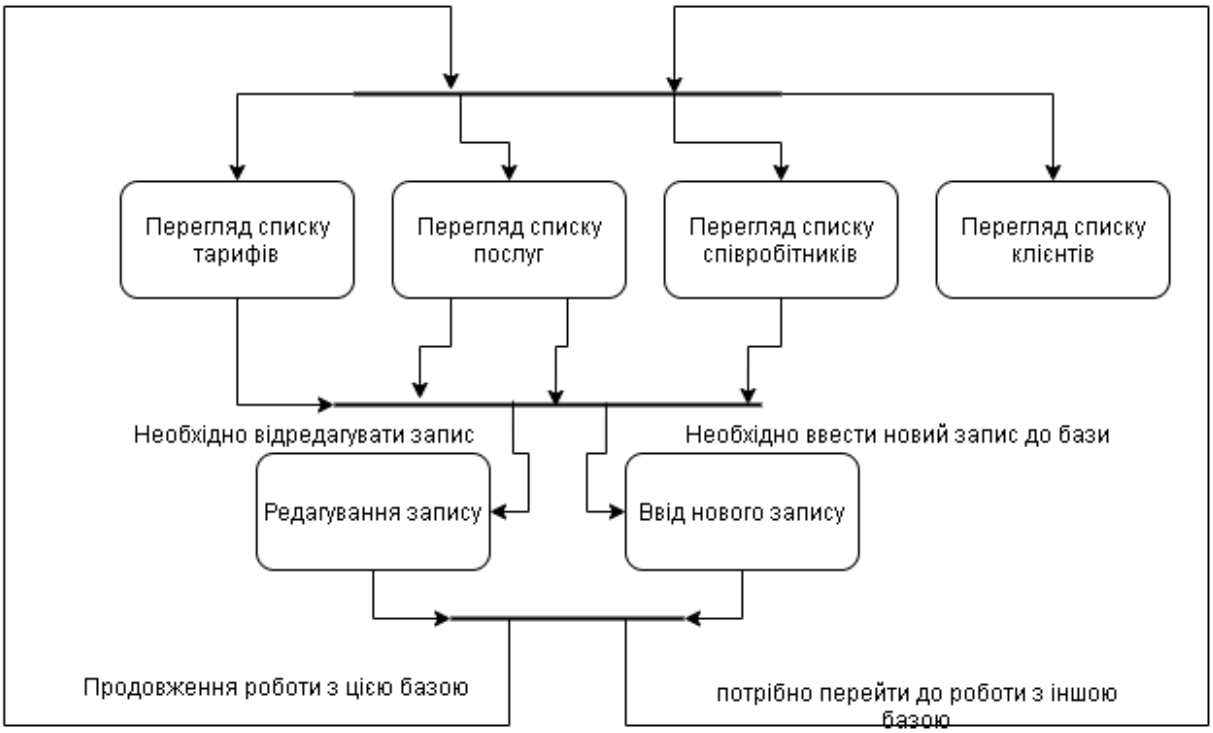


Рисунок 2.4 – Довідкова система

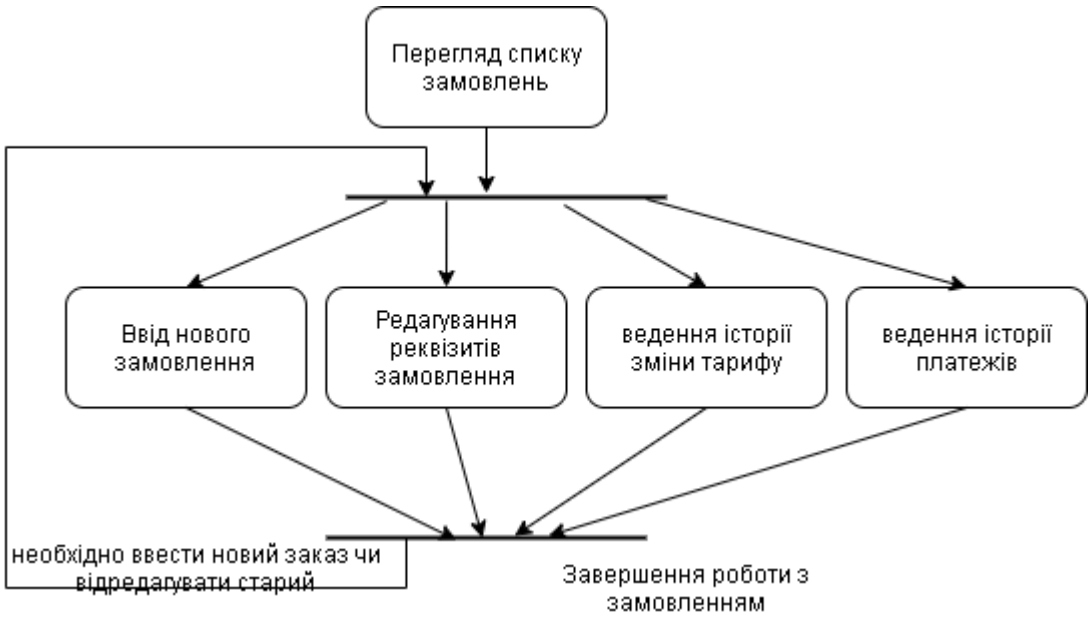


Рисунок 2.5 – Ведення замовлень

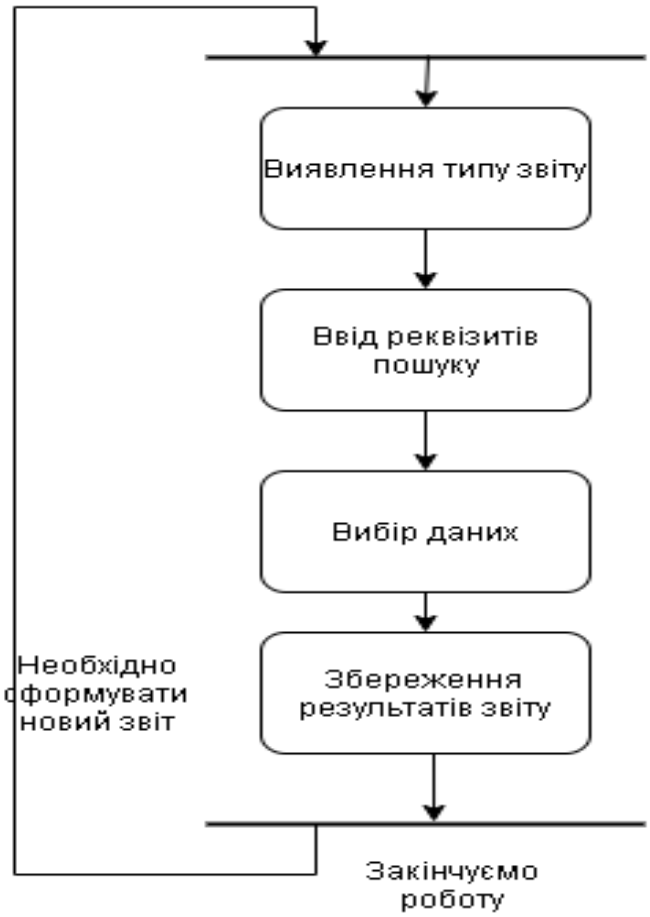


Рисунок 2.6 – Формуванні звітності

2.3 Аналіз предметної області.

Телекомунікаційна компанія, що надає послуги доступу до мережі інтернету та інші послуги пов'язані з інтернетом. За організацію цього процесу під’єднання клієнтів до мережі, відповідає ІТ-відділ, а користувач, в свою чергу складає договір з компанією на надання будь-яких послуг. Користувач може укласти тільки один договір. Термін дії договору обговорюється, після закінчення терміну дії, він автоматично продовжується. При укладенні договору користувач вибирає бажаний тарифний план(сукупність цін і послуг, що надаються). В користувача є можливість вибрати наступні тарифні плани:

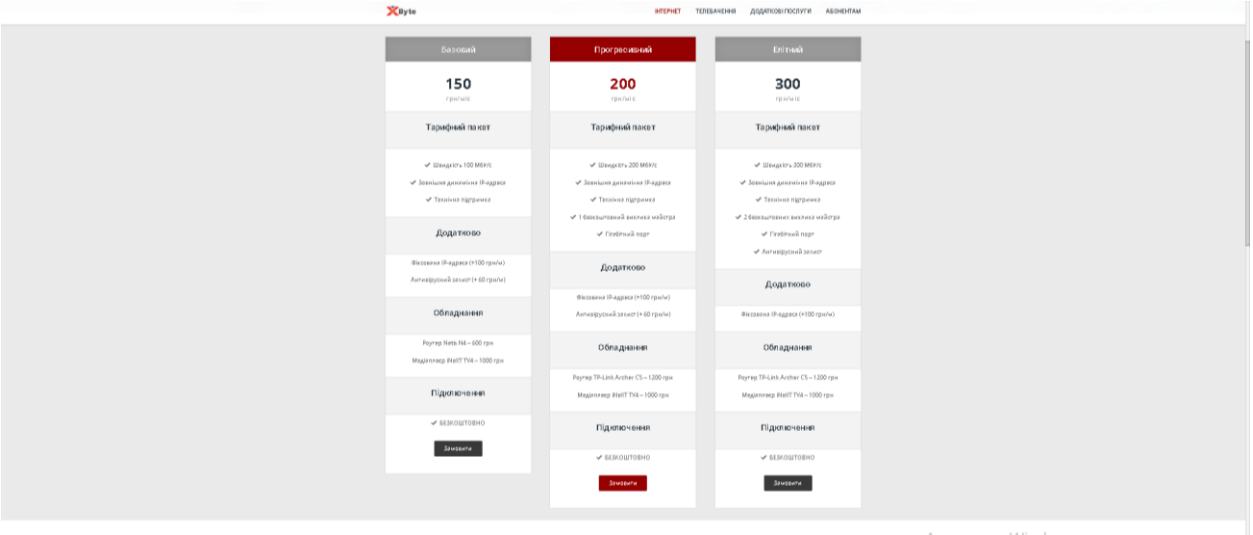


Рисунок 2.7 – Види тарифних планів

Провайдер залишає за собою право перевести користувача на інший тарифний план в новому місяці, попередньо повідомивши його. Списання грошових коштів за договором відбувається в кінці місяця. Трафіком вважається обсяг переданої інформації. Трафік ділитися на:

- Зовнішній (вхідний, вихідний)
- Локальний (вхідний, вихідний)

Локальним трафіком вважається весь трафік проходить в діапазоні IP адрес провайдера. Локальний трафік не обмежується в швидкості. А так само не тарифікується (безкоштовний). Зовнішнім трафіком вважається весь трафік крім локального. Тарифікується тільки вхідний трафік. Вихідний є безкоштовним. Поповнення балансу виконується за допомогою банківських карт чи через термінали оплати. Для оплати потрібно зазначити свій унікальний номер який формується в базі даних провайдера, його прив’язують до клієнта, за допомогою номеру телефона, адреси, та інших особистих даних.

Для активації послуг слід авторизуватись і ввести номер і пароль, після чого оплатити послуги, після того, як оплата буде виконана, буде «дозволено» клієнту, користуватись бажаною послугою. Функціональна структура повинна мати можливості:

- надання актуальної інформації по можливості підключення;
- перегляд користувачів;

- поповнення балансу;
- редагування інформації користувачів;
- статистика оплат;
- активність портів;
- корисна інформація (можливість підключення, обладнання і тд.).

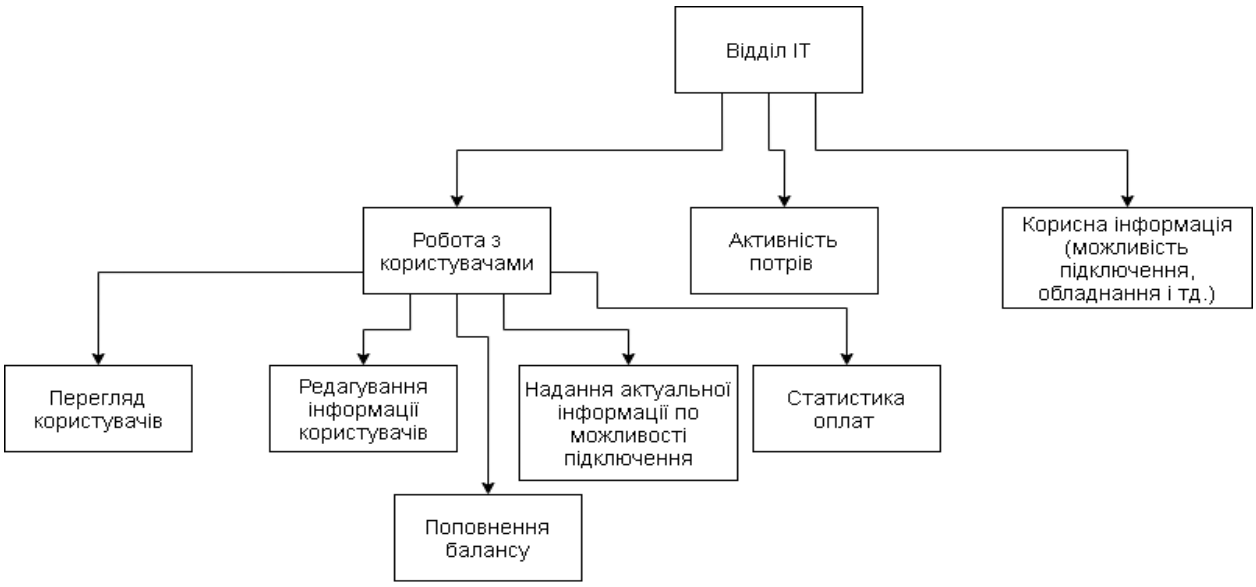


Рисунок 2.8 – Функціональна структура відділу ІТ

Функціональна структура системи являє собою набір діаграм потоків даних, які описують сенс операцій і обмежень. Діаграми потоків даних відображають функціональні залежності значень, що обчислюються в системі, включаючи вхідні значення, вихідні значення і внутрішні сховища даних. Діаграми потоків даних-це граф, на якому показано рух значень даних від їх джерел через перетворюючі їх процеси до їх споживачів в інших об'єктах. Діаграми потоків даних містить процеси, які перетворюють дані. Потоки даних, які переносять дані. Активні об'єкти, які виробляють і споживають дані, і сховища даних, які пасивно зберігають дані. Процес перетворює значення даних. Процеси самого нижнього рівня являють собою функції без побічних ефектів. Процес може мати побічні ефекти, якщо він містить нефункціональні компоненти, такі як сховища даних або зовнішні об'єкти. Потік даних з'єднує вихід об'єкта зі входом іншого об'єкта, він представляє проміжні дані обчислень. Потік даних зображується у вигляді стрілки між виробником і

споживачем даних, позначеної іменами відповідних даних; приклади стрілок, що зображують потоки даних, представлені на рисунку:

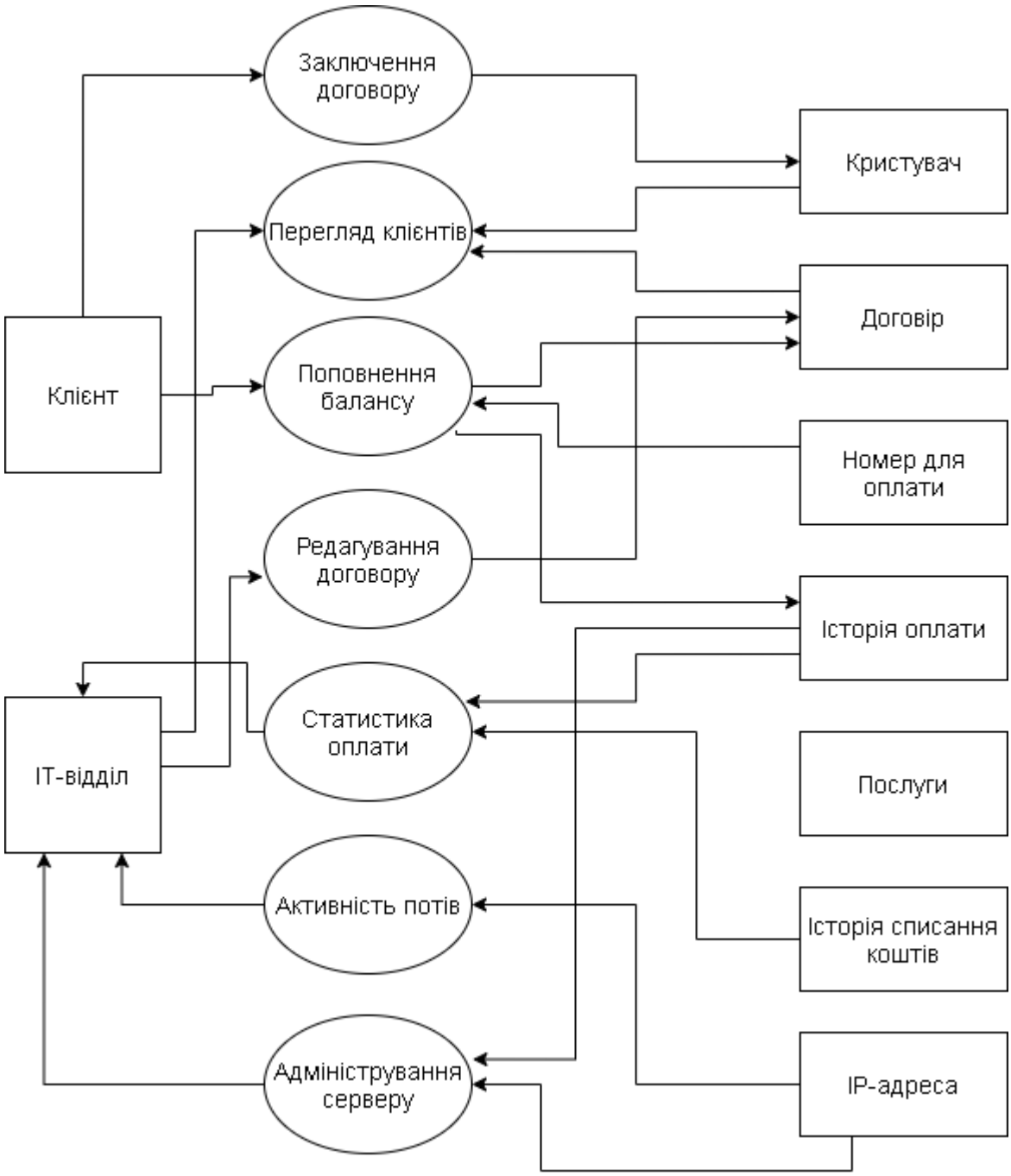


Рисунок 2.9 – Діаграма потоків даних.

Сховище даних – це пасивний об'єкт у у складі діаграми потоків даних, в якому дані зберігаються для подальшого доступу. Сховище даних допускає доступ до збережених в ньому даних в порядку, відмінному від того, в якому вони були туди поміщені. Агрегатні сховища даних, як наприклад, списки і таблиці, забезпечують доступ до даних в порядку їх надходження, або по

ключах. Діаграма потоків даних проекту представлена на (рис 2.9). На функціональній діаграмі листовими будуть функції:

- Укладення договору.
- Перегляд клієнтів.
- Поповнення балансу
- Редагування договору..
- Статистика оплат (виводить дані про оплати, здійснені Користувачем).
- Активність портів (виводить сумарний обсяг інформації прийнятий на кожен порт).

Після того, як розроблений список завдань, черговим важливим кроком проектування є створення зав’язків в базі даних, необхідних для виконання кожній задачі, а також визначення змін, яким піддаються ці дані. Відділ ІТ, має слідкувати за основною задачею, яка представляє собою базу даних компанії-провайдера «X-Byte», та наданням послуг в сфері телекомунікації і інформаційних технологій. Основними об'єктами бази є:

- Користувачі
- Співробітники
- Клієнти
- Замовлення
- Тарифи
- Послуги
- Оплата (платежі по замовленням)

Визначимо основні поля і зв'язки між таблицями. Для цього побудуємо об'єктні та фізичну (реляційну) моделі БД (див. Малюнки 2.10 – 2.13).

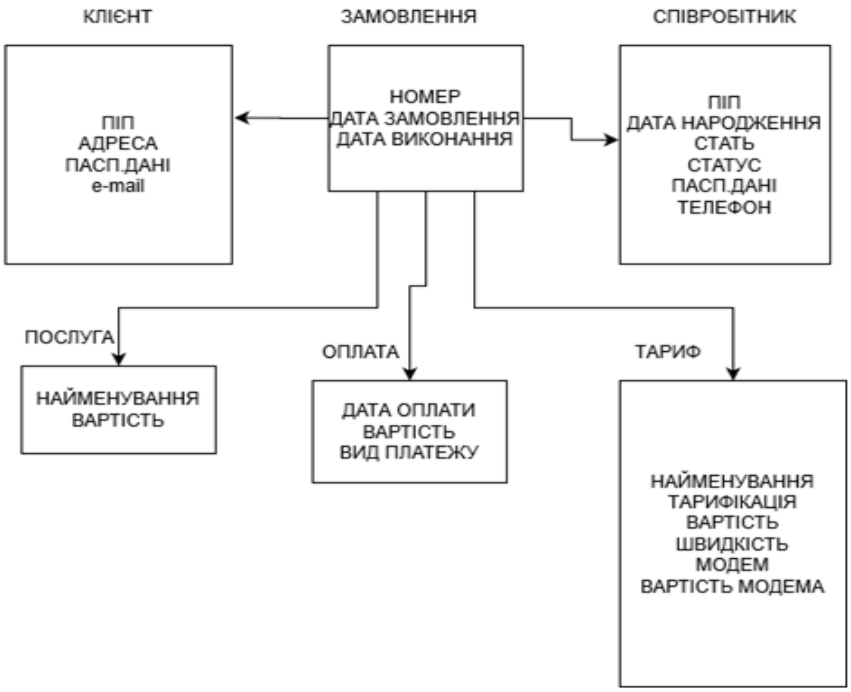


Рисунок 2.10 – Об'єктна модель

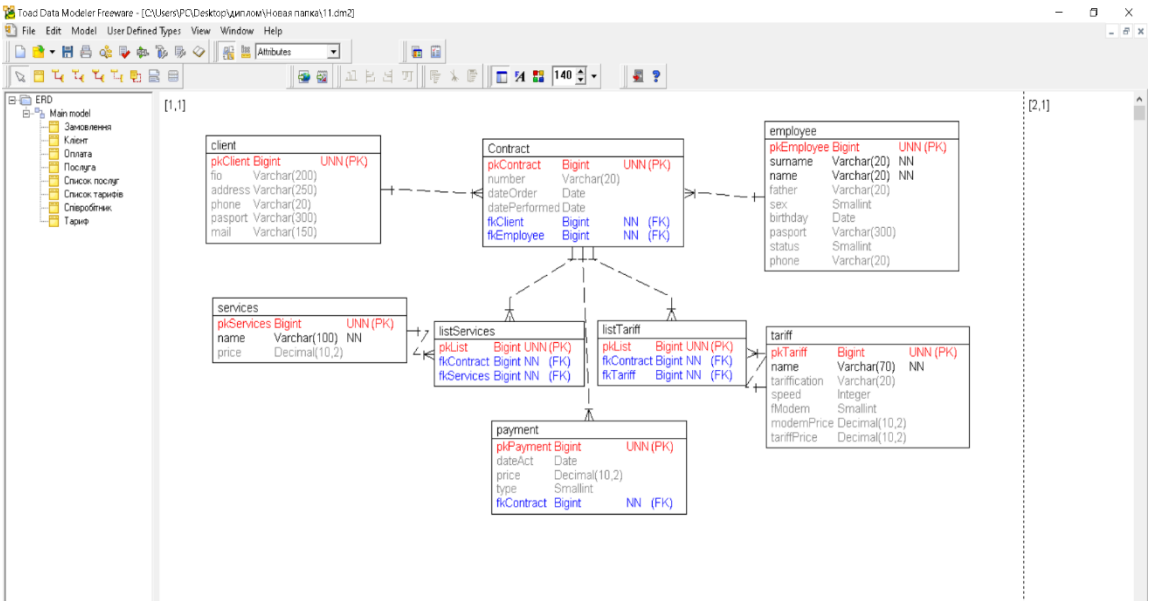


Рисунок 2.11 – Реляційна модель бази даних IT відділу.

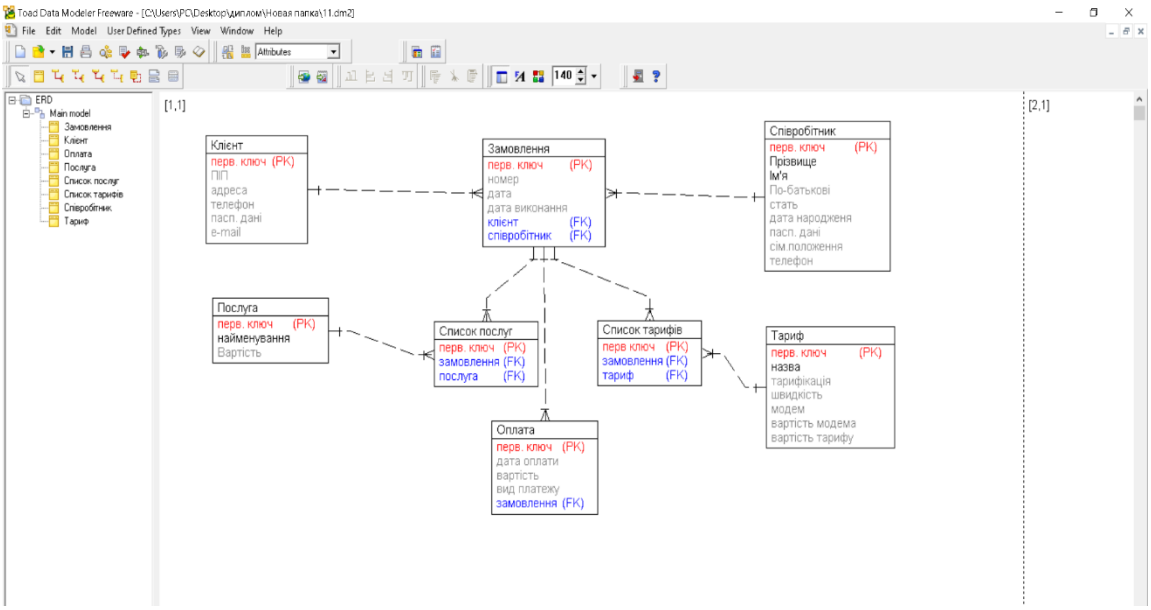


Рисунок 2.12 – Найменування атрибутів в базі даних

Після того, як була спроектована реляційна модель, в TDM був згенерований код для створення БД на сервері MySQL.

Після генерування коду він був виконаний в Query Browser для створення серверу MySQL в результаті чого, буле створена база даних для відділу IT, для роботи з даними користувачів (рисунок номер), яку потім можна проаналізувати, зберегти, обробити та використовувати в подальшому.

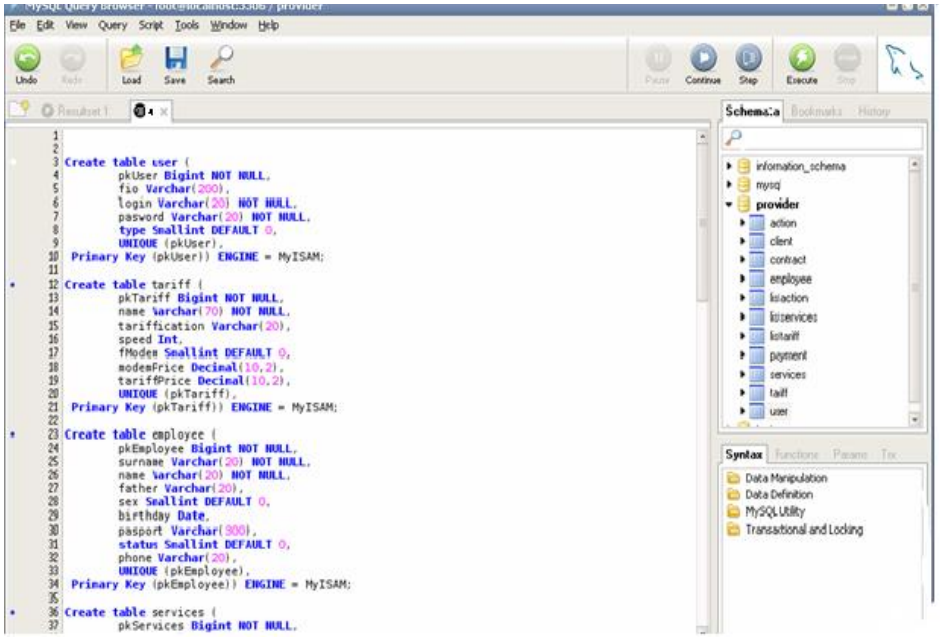


Рисунок 2.13 – Вікно в Query Browser MySQL

2.4 Доступ до БД MySQL з додатку Visual Studio C#

Спочатку, при інсталюванні програми Visual Studio, ми не маємо змоги приєднатись до будь-якої бази даних, окрім тих, які передбачені розробниками ПО. Але, проаналізувавши систему, можна зрозуміти, що програма однаково добре працює з усіма відомими БД. Щоб отримати змогу додатку приєднатись до БД MySQL, необхідно встановити спеціальний конектор (Connector/Net чи MySQL Connector), який надасть змогу програмі Visual Studio через прямий API працювати з додатком MySQL, без нього ж, там будуть відображатись лише підключення до БД від Microsoft та Oracle. Необхідні конеткори можна завантажити на офіційному сайті MySQL, перелік ти опис конекторів зображено на рисунку 2.4:

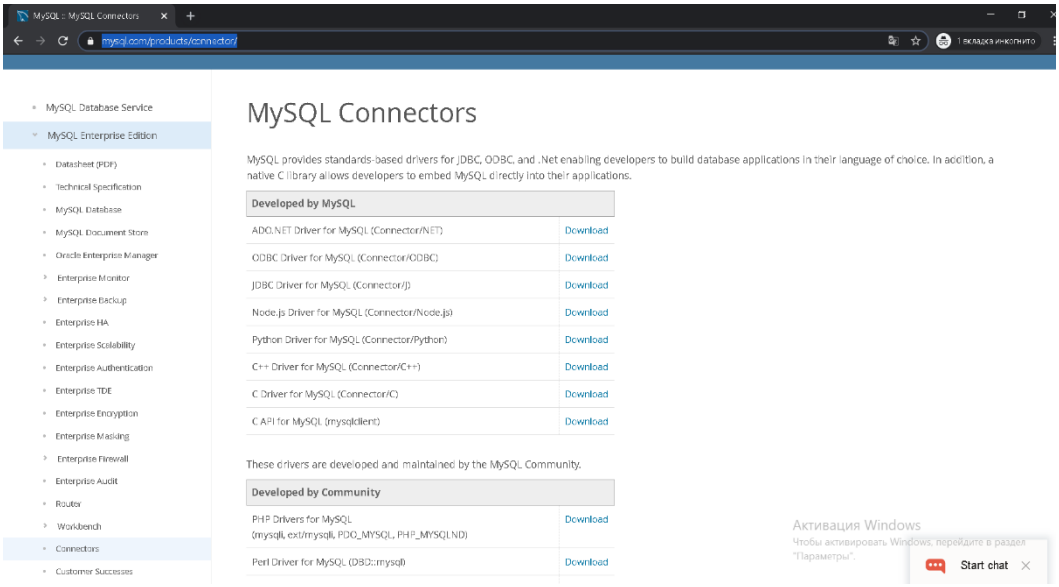


Рисунок 2.14 – Перелік MySQL конекторів

Необхідно вибрати потрібний конектор, під своє середовище розробки (в моєму випадку це, середа розробки (Visual Studio) тому необхідний конектор .NET). Після того, як інсталювання конектору буде виконано, в програмі Visual Studio, з’явиться змога приєднатись до бази даних MySQL напряму, та в real-time режимі, брати необхідні дані для створюваного додатку, що в свою чергу дозволить наглядно розуміти, як буде виглядати екранна форма додатку, і що потрібно додати чи відкоригувати, для більш кращого ефекту.

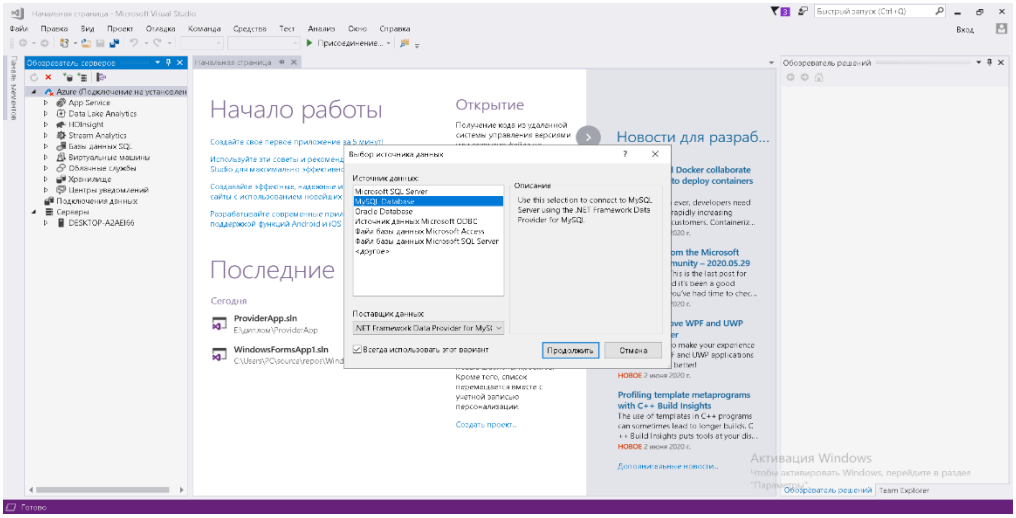


Рисунок 2.15 – Зображення підключення до БД MySQL

Після створення зв’язків, між базою і додатком і якщо база не потребує оновлення даних, а лише перегляд та корегування уже існуючих даних, Visual Studio, дозволяє запускати збережену інформацію з БД, з бібліотек “.DLL.”, що в свою чергу, суттєво зменшує навантаження на комп’ютер для налаштування\внесення даних, та запобігає випадкових змін даних в самій базі даних. Серверний додаток дозволяє користувачам використовувати тільки запит SELECT і викликати збережені процедури, що не вносять змін в БД. Адміністратори відділу IT мають повний доступ до всіх елементів бази данх. Клієнтська програма в залежності від ролі користувача дозволяє йому редагувати і додавати записи, переглядати звіти, виконувати запити і використовувати збережені процедури. Клієнтський додаток керує через екранні форми взаємодією користувача з таблицями даних і сервером за допомогою технології ADO.NET, який входить до .NET фреймворку який автоматично вмонтовується в середовище програми при побудуванні її в Visual Studio, також отримує доступ до серверу, завдяки конектору MySQL CONNECTOR/.NET, який було інстальовано раніше.

3 ВІЗУАЛІЗАЦІЯ ІНТЕРФЕЙСУ ДОДАТКУ ТА СТАТИСТИЧНИЙ АНАЛІЗ

3.1 Розробка користувальницького інтерфейсу для додатку

Програма складається з багатьох форм, що дозволяють редагувати та переглядати інформацію про замовлення для надання постачальнику послуг. Основні програмні модулі та їх взаємодія представлені на рисунку 3.1:

Рисунок 3.1 – Форма проекту

На рисунках (3.2 – 3.14) зображені екрані форми програмного забезпечення створеного для відділу IT за допомогою Visual Studio та Windows Forms.

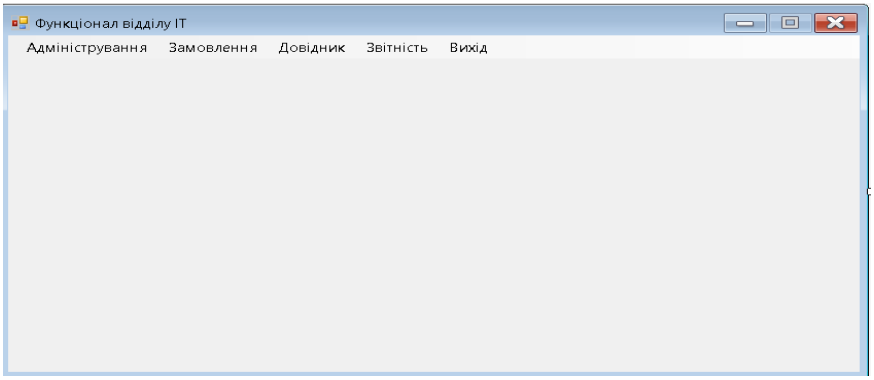


Рисунок 3.2 – Головна форма

На даному рисунку (3.2), зображена головна форма додатку, завдяки якій, можна потрапити в різні його розділи, такі як «Адміністрування», «Замовлення», «Довідник», «Звітність». Кожен з розділів, містить в собі підрозділи, які дозволять вибрати конкретну задачу, наприклад зразу додати замовлення через вкладнику «Замовлення».

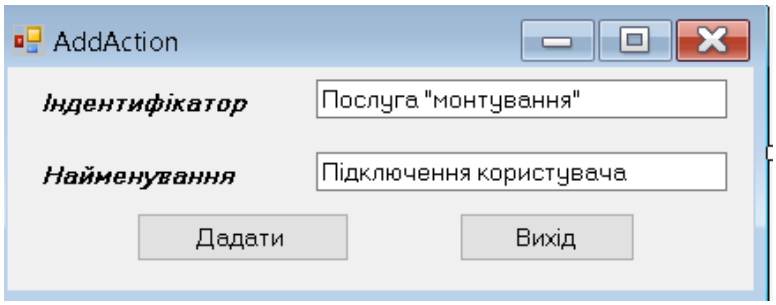


Рисунок 3.3 – Форма підключення послуги

На рисунку (3.3) зображена форма підключення нової послуги, при описі послуги і натискання «додати», програма автоматично додає нову інформацію до бази даних в колонку «список послуг» та «замовлення».

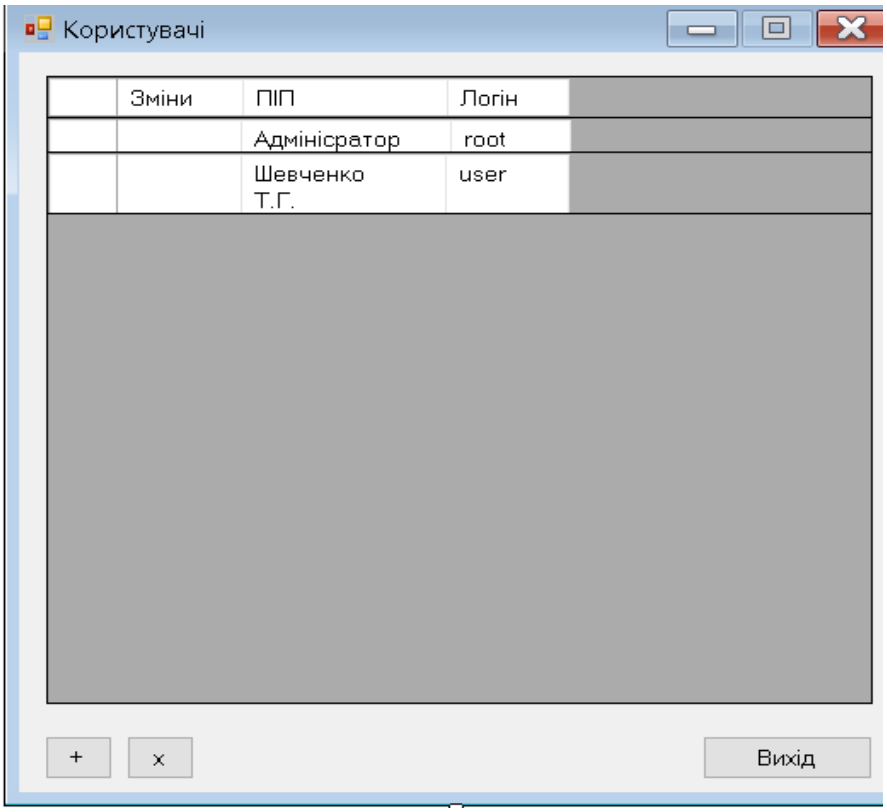


Рисунок 3.4 – Перегляд і редагування списку користувачів

На даному рисунку(3.4), приведено список авторизованих в системі користувачів, тут видно статус користувача (root-користувач, тобто адміністратор чи простий користувач), і їх ПІП, на кнопку «+», адміністратор може додати нового користувача, чи видалити, за допомогою кнопки «х». Вся інформація з цеї екранної форми, зберігається в таблицях бази даних: «Співробітник» та «Клієнт». Адміністратор, може змінювати\видаляти їх обидві, а користувач може редагувати, лише таблицю «Клієнт».

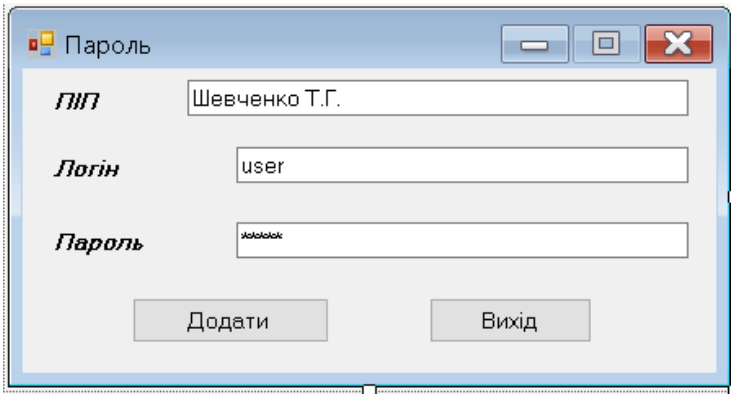


Рисунок 3.5 – Форма авторизації користувача

Рисунок (3.5), зображує форму авторизації користувача, тут можна встановити логін та пароль для подальшої авторизації користувача, вести його ПІП. Всі користувачі окрім root-користувачів, вносяться до таблиці «Клієнт».

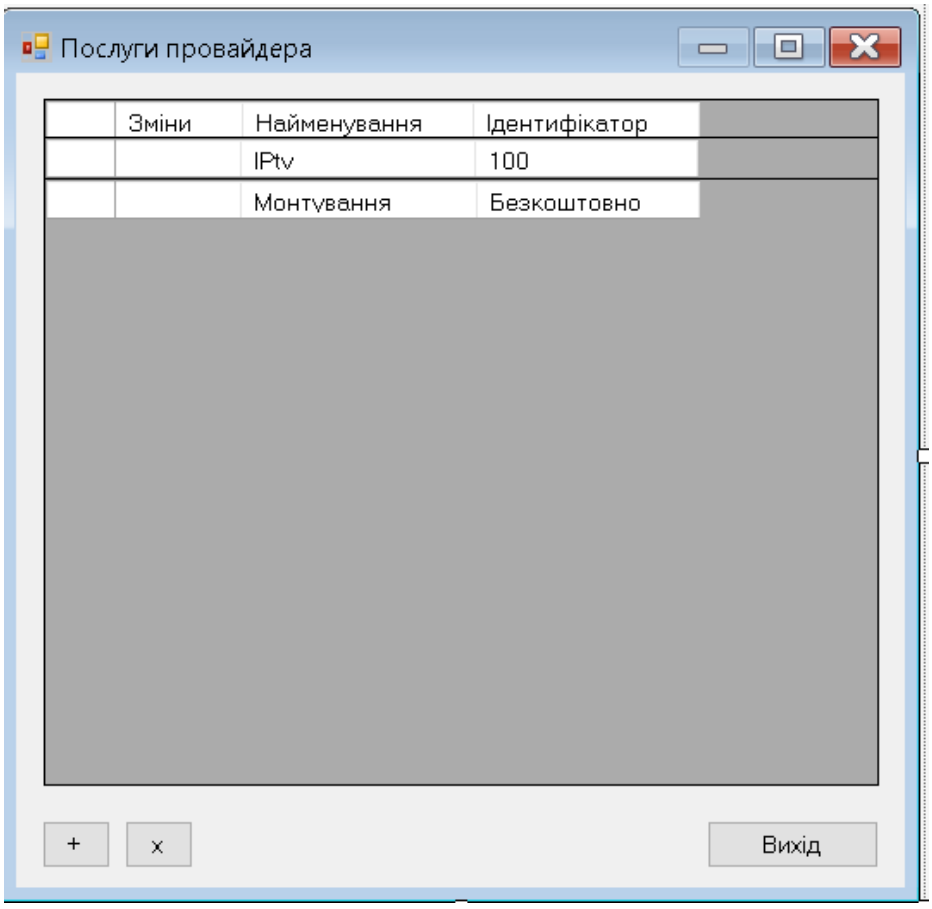


Рисунок 3.6 – Довідкова форма послуг і вартості

На рисунку (3.6) зображена довідкова форма послуг і їх вартості, тут можна побачити назву послуги і її вартість, редагувати їх чи взагалі видалити, вся інформація зберігається в таблиці БД, під назвою «Список послуг».

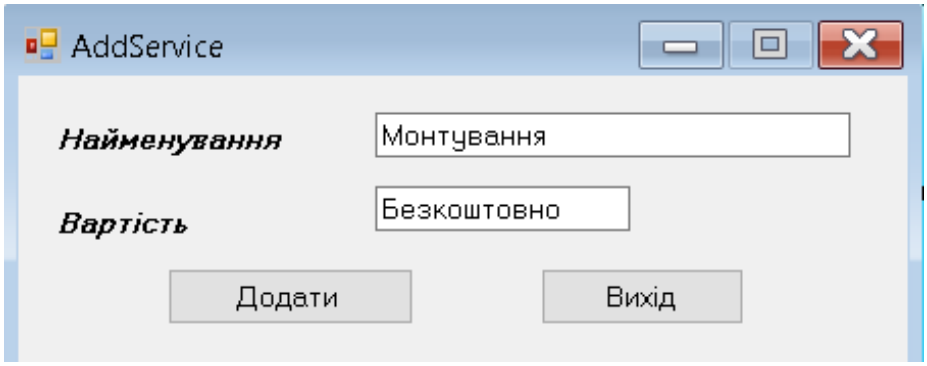


Рисунок 3.7 – Ввід нового запису послуги

На рисунку (3.7) зображено додавання нової послуги до довідкової форми, можна вести назву послуги і її вартість, дані будуть збережені в таблиці «Список послуг» та «Послуги».

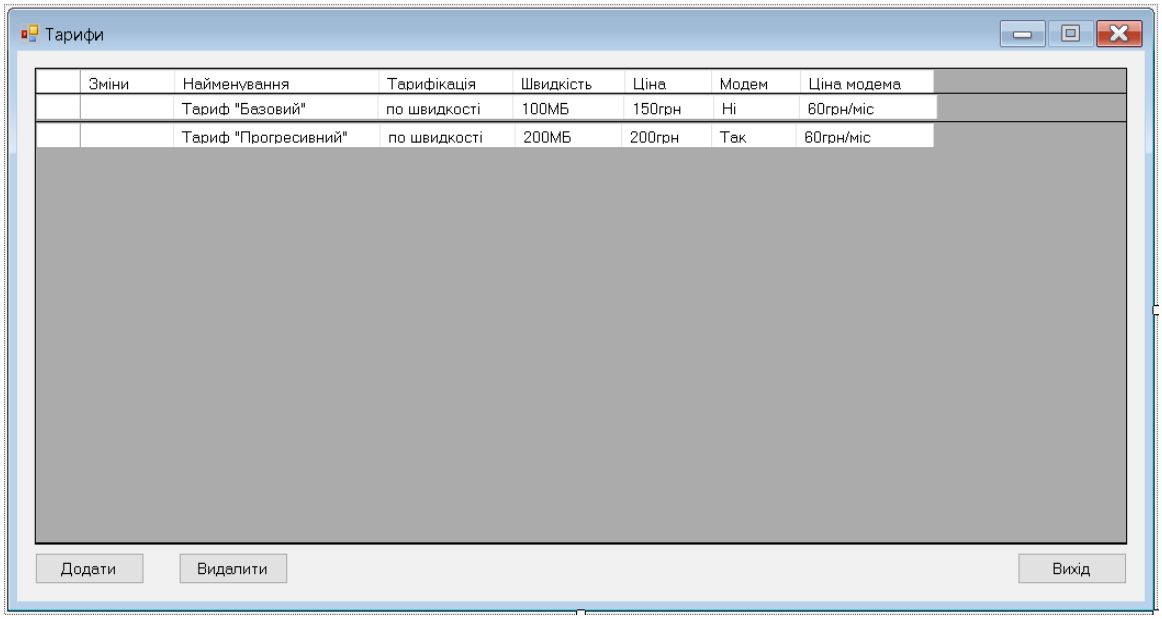


Рисунок 3.8 – Довідкова інформація про тарифи

На рисунку (3.8) зображена довідкова форма тарифів, тут можна побачити назву тарифу, швидкість, його вартість та чи входить до нього модем і за яку ціну. Тут можна редагувати інформації про тарифікацію, додати нові тарифи чи видалити застарілі. Вся інформація заносяться в таблицю БД, під назвою «Список тарифів».

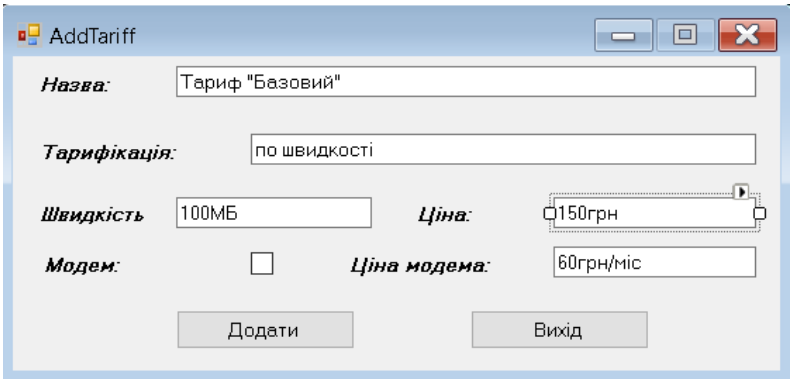


Рисунок 3.9 – Робота з існуючим тарифом

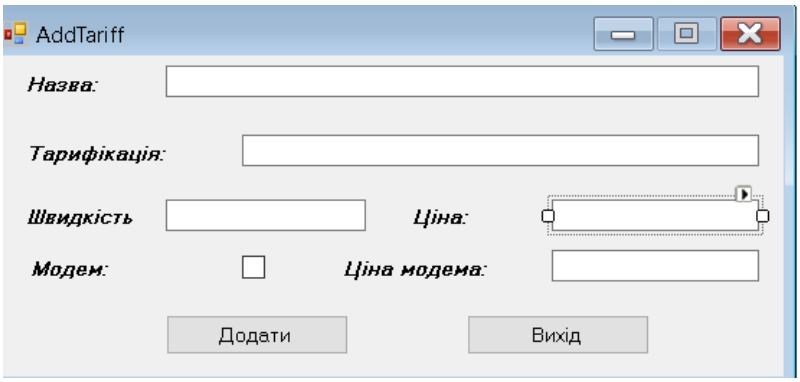


Рисунок 3.10 – Введення нового тарифу

Рисунок (3.9 – 3.10) демонструє додавання нового тарифу, тут можна вести його назву, спосіб тарифікації, швидкість та ціну, також чи входить модем в тариф і його вартість. Нова інформація про тариф заноситься в таблицю бази даних «Тариф» та «Список тарифів»

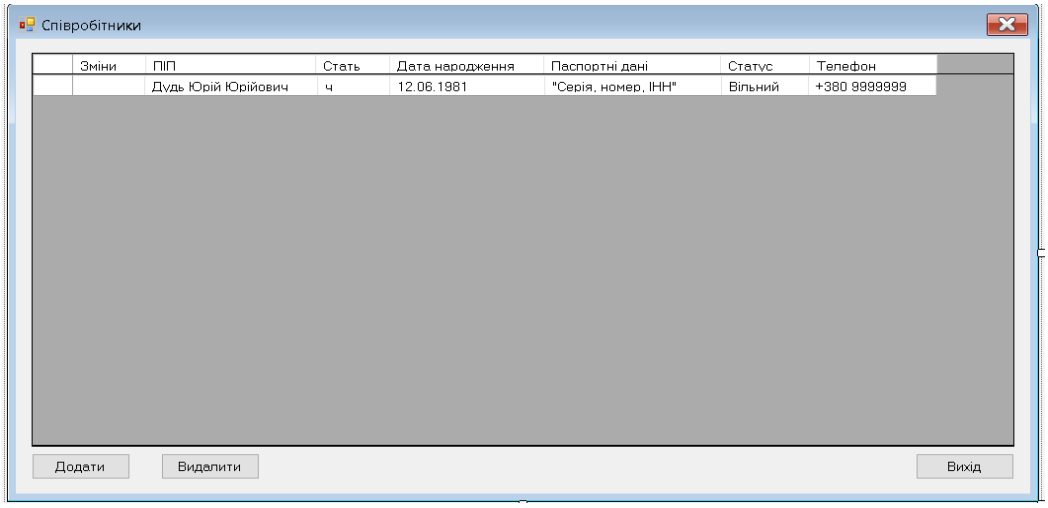


Рисунок 3.11 – Робота з даними співробітника

На рисунку (3.11) зображена форма з даними співробітників, їх ПІП, стать, дата народження, паспортні дані, статус їх зайнятості на даний момент та номер телефону. Тут можна додати нового працівника, змінити дані існуючого чи видалити старого, змінити статус його зайнятості «Вільний\Зайняти», дані вносяться в таблицю «Співробітник».

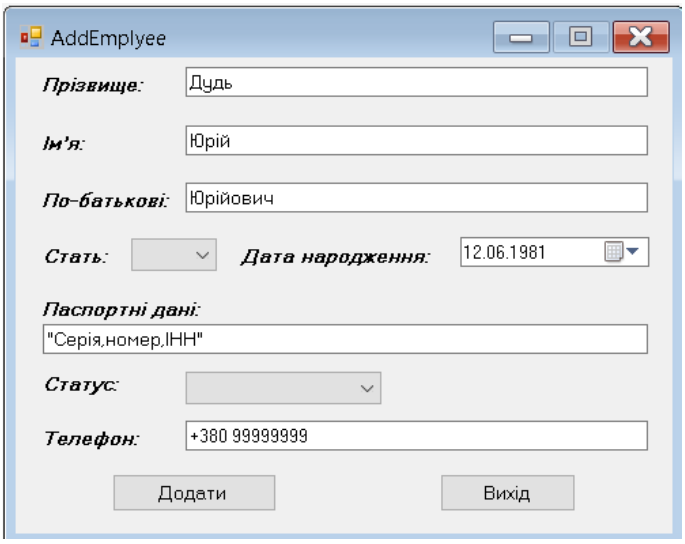


Рисунок 3.12 – Додання співробітника до таблиць

На рисунку (3.12), зображена форма додавання нового співробітника до екранної «Співробітники», тут можна вести ПІП співробітника, стать, дату народження, паспортні дані та телефон. Всі зміни з цієї форми зберігаються в таблиці БД «Співробітники».

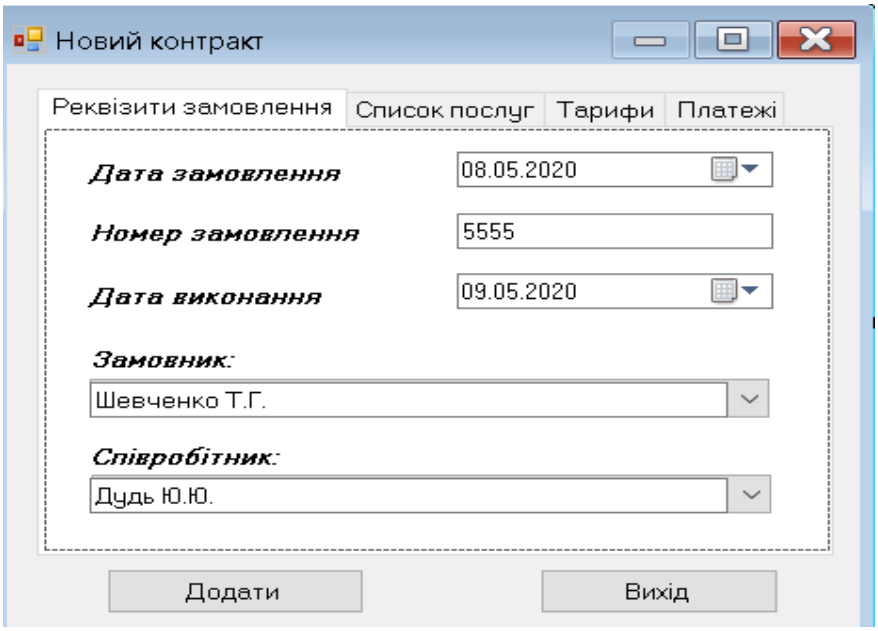


Рисунок 3.13 – Введення нового завдання

На рисунку (3.13), зображує додавання нового замовлення, тут формується дата замовлення, номер, та дата виконання замовлення. Також тут вибирається користувач-замовник, та співробітник, який буде виконувати

завдання. Дані з цієї форми пов’язані з таблицями БД: «Послуги», «Тариф», «Співробітник», «Клієнт», «Оплата» та «Замовлення».

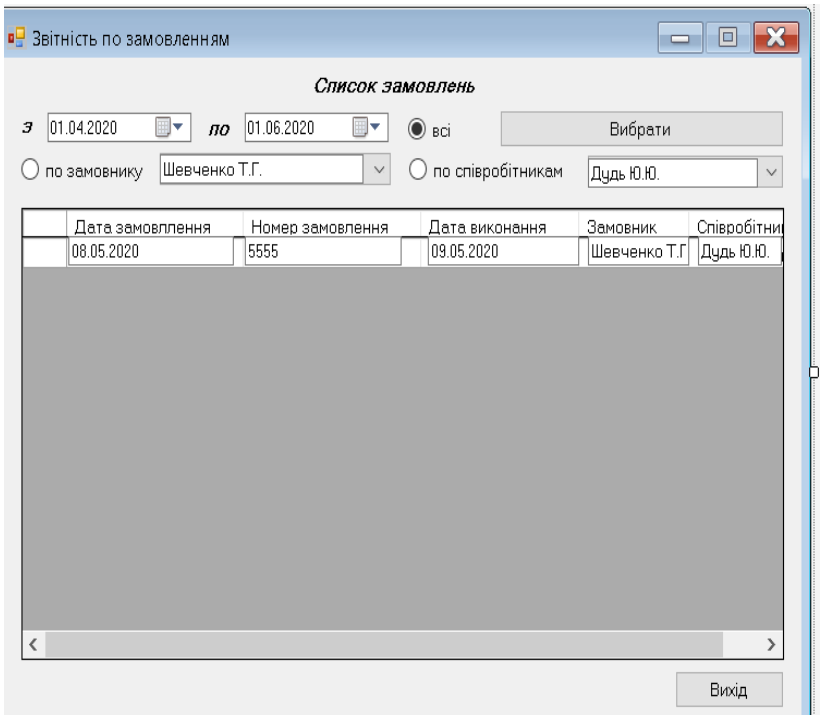


Рисунок 3.14 – Формування звіту по замовленням

На рисунку (3.14) зображено форму звітності по замовленню, тут можна переглянути замовлення в період «З\ПО» но конкретному користувачеві чи по співробітнику, його номер, коли та ким було виконано. Данна форма, під’єднана до таблиць в БД: «Клієнт», «Співробітник», «Замовлення».

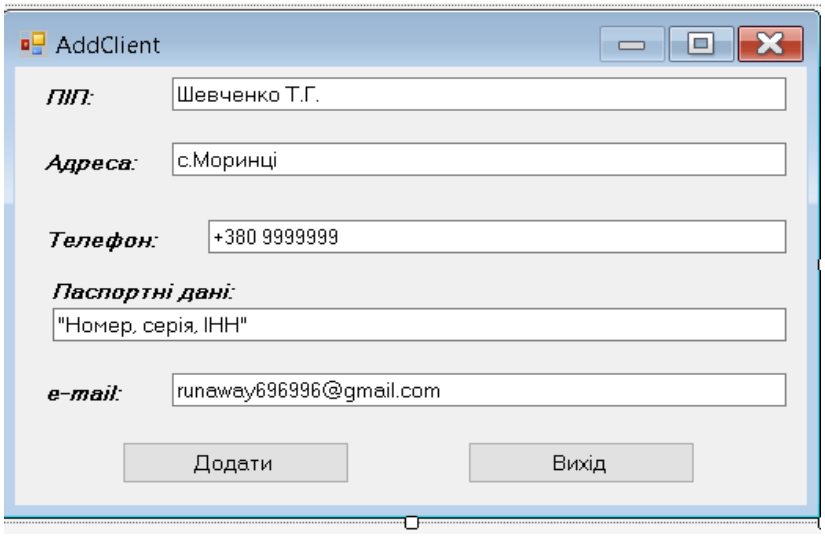


Рисунок 3.15 – Форма додання користувача

На рисунку (3.15) зображена форма додавання клієнта до бази даних, тут потрібно вести його ПІП, адресу підключення, номер телефону, паспортні дані та пошту. Ця форма, взаємодіє з таблицями в БД: «Клієнт».

3.1 Вплив впровадження додатку на економічну складову

Економічний ефект від інтегрування розробки за рік використання, можна визначити за формулою[9]:

$E_Г = E_У - E_Н \cdot З_К$, де $З_К$ – калькуляція витрат на розробку програмного продукту, $E_н$ – нормальний коефіцієнт капітальних вкладень (0.2), $E_у$– економія затрат на ручну обробку інформації, від заміни на автоматизовану обробку. Економія коштів при інтегруванні продукту можна розрахувати за формулою: $E_у = З_р - З_а$. Тут $З_р$ – затрати на ручну обробку інформації, грн, ,

$З_р = О_н * Ц * Г_д / Н_р$, $О_н$, – Обсяг інформації оброблений в ручну, Ц – вартість одної години роботи, грн/год, $Г_д$ - коефіцієнт, який враховує додаткові затрати часу на логічні операції при ручній обробці інформації, $Н_р$ – норма опрацювання даних, Мбайт/год. $З_а$ – Затрати на автоматизовану обробку, грн, $t_а$– час автоматичної обробки (год), $Ц_м$– вартість одної робочої години машинного часу, грн/год; t_0 – час роботи відділу, год; $Ц_0$ - вартість одної робочої години в відділі ІТ, грн/год.

$E_у = О_н * Ц * Г_д / Н_р – t_а * Ц_м * t_0 * Ц_0 = 200*100*1.5*2 – 50*1*5*1*200$

$= 60000 – 50000 = 10000$

Ефективність розробки($E_р$):

$E_р = 10000 – 0.2*10000 = 8000$

Тоді ефективність розробки можна визначити по формулі:

$E_р = (E_г * 0,4) / З_к = 8000*0.4/10000 = 0.32$

Використовування в відділі, створеного програмного забезпечення економічно вигідно $E_р \geq 0,2$[9,10].

Для роботи додатку потрібно наступне програмне забезпечення: операційній системі Windows XP / 7 / 10 з встановленим .Net Framwork.

Апаратні вимоги даного продукту збігаються з апаратними вимогами операційної системи, для яких він призначений. Обсяг ОЗУ для ефективної роботи програми визначається за формулою: $V = V_1 + V_2 + V_3$,

де V_1 – мінімальні вимоги з боку операційної системи, V_2 – мінімальний розмір планок ОЗУ, V_3 – Додатковий простір ОЗУ.

Отже, маємо:

V= 1024+512+512=2048 мегабайт.

Вільний простір на жорсткому диску визначається рівністю:

$W = W_1 + W_2 + W_3$,

де W_1 – обсяг інсталяційного пакету програми; W_2 – обсяг файлів бази даних; W_3 – обсяг тимчасових файлів, що створюються програмою в ході запуску.

Отримуємо:

W=3+5+2=10 мегабайт.

Виклик даної програми не відрізняється від запуску будь-якої іншої програми з середовища операційної систем: Користувачеві необхідно скопіювати файли програми на жорсткий диск і запустити додаток з розширенням .exe в кореневому каталозі програми. Цей додаток для роботи вимагає наявність встановленого. NET framework 2.0 версії і вище. Ніяких додаткових дій від користувача не потрібно. Після запуску програми користувачеві необхідно ввести логін і пароль. Виклик збереженої процедури і зміна даних в таблицях дозволені, тільки якщо вхід виконаний адміністратором (root-користувач). Головне меню містить 4 підменю «Адміністрування», «Замовлення», «Довідник», "Звітність". У першому містяться команди для виконання довільного запиту; у другому-команди, що відкривають вікна для перегляду і редагування таблиць з замовленнями; в третьому – для перегляду послуг, тарифів, співробітників, клієнтів; в четвертому – для виклику списку замовлень, зміни замовлень/тарифів.

ВИСНОВКИ

Результати роботи, проведеної в бакалаврській роботі, що впроваджу інформаційно-аналітичні системи, в телекомунікаційну компанію, вчасності в ІТ відділ. Було проведено аналіз бізнес-процесів, які відбуваються у відділі, також проаналізовано предметну область, завдяки цьому, стало зрозуміло, який функціонал потрібно реалізувати в додатку для відділу ІТ. Виконання задач кваліфікаційної роботи, привело до формулювання цілі, для створення та автоматизування інформаційно-аналітичної системи. Додаток, дозволить мінімізувати затрати в часі, та більш доцільніше використовувати ресурси відділу. Що в свою чергу призведе до:

- прискореної роботи з замовленнями клієнтів та їх виконання;
- зменшити вірогідність помилки завдяки відсутності роботи з паперовою документацією;
- здійснювати керування, облік, звітність, актуальну інформацію в одному місці і доступною в пару кліків;
- вести моніторинг зайнятості всіх співробітників чи конкретної особи;
- створювати звіти буде на порядок легше, всі необхідні дані завжди доступні в додатку;
- дасть змогу легко створювати формування ціни для клієнта, розуміючи вартість підключення послуги;
- дозволить бачити важливі дати для користувача, щоб пропонувати йому акційні тарифи, знижки тощо;
- дасть змогу економити час співробітників та оптимізувати їх дії;
- забезпечить інформацією про найближчого користувача і про його обладнання;
- відобразить дату підключення та кінцеву дату тарифу, щоб своєчасно від’єднувати чи під’єднати користувача;
- буде містити в собі актуальну інформацію про дату замовлення та дату його виконання, що допоможе слідкувати та корегувати роботу з ними.

Модель даної інформаційно-аналітичної системи, можна розвивати в бік досягнення більшого функціоналу, ергономічності, додавання нових можливостей, що розширюватимуть стандартні засоби. Включаючи всі фактори описані вище, впровадження такої програми, допоможе сформувати гнучкий функціональний механізм, який здатен допомогти в роботі відділу IT, в роботі з бізнес-процесами, та допоможе більш краще, працювати з користувачами, та приманювати нових клієнтів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Конноллі Т. Бази даних: проектування, реалізація та супровід. Теорія і практика / Т. Конноллі, К.Бегг. - М.: Вільямс, 2017. – 1439 с.

2. Хетагуров Я. А. Проектування автоматизованих систем обробки інформації та керування. - М.: Вища школа, 2006. - 348 с.

3. Опис середовища додатку Toad Data Modeller [Електронний доступ] // Режим доступу: http://www.itshop.ru/Quest-Software-Toad-Data-Modeler

4. Опис роботи додатку Visual Studio з базами даних[Електронний доступ] // Режим доступу: https://docs.microsoft.com/en-us/visualstudio/data-tools/installing-database-systems-tools-and-samples?view=vs-2019

5. Оформлення, візуальні ефекти, робота з графікою Автор: Jeff Prosise, Microsoft Corp.

6. С.Д. Кузнєцов. Об'єктно-орієнтовані бази даних - основні концепції, організація та управління: короткий огляд.

7. Хомоненко А.Д., Циганков В. М., Мальцев М. Г. Бази даних: Підручник для вищих навчальних закладів / За ред. проф. А. Д. Хомоненко. - Видання друге, доповнене і перероблене. - СПб.: Корона принт, 2002.- 672 с.

8. Глушаков С. В., Ломотько Д. В. Бази даних: Навчальний курс. - Харків: Фоліо, 2000. - 504 с.

9. Мішенін А.І. Теорія економічних інформаційних систем. – М.: Фінанси і статистика, 1999. - 168 с.

10. Смирнова Г. Н., Сорокін А. А., Тельнов Ю. Ф. Проектування економічних інформаційних систем: підручник для вищих навчальних закладів. – М.: МЕСІ, 2004. – 435 с.

11. Фролов А.В., Фролов Г. В. Мова C#. Самовчитель. – М.: ДІАЛОГ-МІФІ, 2003. – 560 с.

12. Шилдт Г. C#: навчальний курс. – Спб.: Пітер, 2003. – 235 с.

13. М. Кузнєцов, І.Симдянов. MySQL 5. - СПб.: БХВ-Петербург, 2010.- 1007с.

14. Д. Скіт. Програмування на C# для професіоналів. – М.: Діалектика, 2020. – 600 с.

15. Малюх В. Введення в сучасні САПР. – М.: ДМК-Прес, 2016. – 192 с.

16. Шварц Б., Зайцев П., Такченко В. MySQL по максимуму. – Спб.: Пітер, 2018. – 864 с.

КОПІЇ СЛАЙДІВ ПРЕЗЕНТАЦІЇ

Слайд 1

Міністерство освіти і науки України
Державний університет телекомунікацій
Навчально-науковий інститут інформаційних технологій
Кафедра системного аналізу

«Розробка інформаційно-аналітичної системи телекомунікаційної компанії «X-Byte»»

Виконав : студент групи САД-41
Горобець В.Ю.
Науковий керівник:
доцент кафедри Системного аналізу, Ярцев В.П.

Київ-2020

1

Слайд 2

Об’єкт, предмет, мета та медоти дослідження в бакалаврській роботі

- ***Мета роботи*** – розробка інформаційно - аналітичній системи для автоматизації роботи відділу інформаційних технологій інтернет-провайдера X-Byte.
- ***Об`єкт дослідження*** – розробка концептуальної моделі зберігання, обробки та візуалізація даних для автоматизації роботи IT відділу інтернет-провайдера X-Byte.
- ***Предмет дослідження*** – база даних з архітектурою «клієнт-сервер», додатки, щодо роботи базою, для супроводження інформаційної системи.
- ***Методи дослідження*** – системний аналіз предметної області, способів проектування інформаційних систем, комп’ютерне моделювання об’єктів бази даних, створення екранних форм у системі об’єктно-орієнтованого програмування, статистичній аналіз.

2

Слайд 3

Завдання бакалаврської роботи

- Проаналізувати технологій та засоби проектування сучасних інформаційних систем.
- Проаналізувати предметну область та бізнес - процеси IT-відділу.
- Розробити та побудувати базу даних інформаційної системи.
- Розробити та побудувати екранні форми додатку, що працює з базою даних.
- Розрахувати економічну ефективність проекту інформаційної системи.

3

Слайд 4

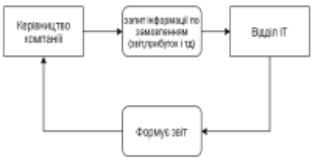
Завдання сформовані для бази даних щодо роботи з додатком:

- Відображення основної інформації про клієнтів, працівників, замовлення, тарифах, послуги, оплаті;
- Можливість додати нового співробітника, нове замовлення, новий тариф, нового клієнта;
- Відображення днів народжень співробітників і клієнтів;
- Отримання інформації про клієнтів\співробітників;
- Здійснення різного роду пошуку по заданих критерієм (вартість тарифу, за прізвищем співробітника або клієнта та інше.)

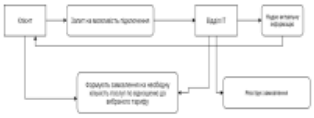
4

Слайд 5

Алгоритми бізнес-процесів IT-відділу



Звіти IT відділу керівництву



Створення замовлення

Інформація про оплату

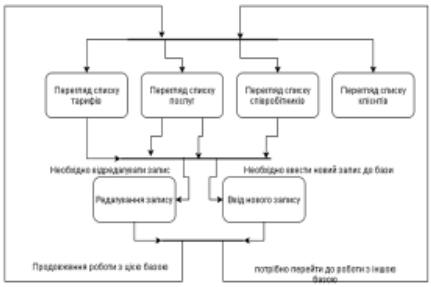
- Клієнти- бажають отримати певний перелік послуг, здійснюють оплату за надані послуги відповідно з тарифами і тарифікацією, звертаються за наданням інформації чи звітів.
- Відділ IT – приймає замовлення від відділу по роботі з клієнтами, реєструють замовлення в «картотеці» і формують робочі задачі для співробітників.
- Системні адміністратори – працюють з серверним обладнанням, та працюють з іншою технікою (маршрутизатори, роутери, свічі і інше).
- Керівники компанії – запитують звітність.

5

Слайд 6

Алгоритми бізнес-процесів IT-відділу

Довідкова система



Система формування звітності

Система введення замовлення

6

Слайд 7

Аналіз предметної область відділу IT

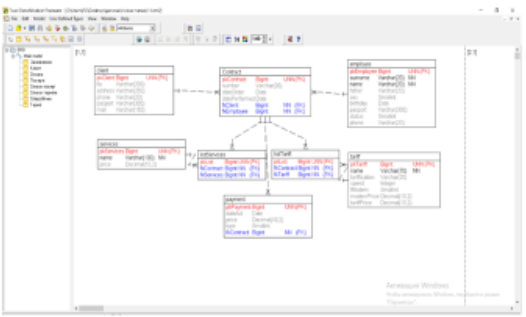
• Відділ IT, має слідкувати за основною задачею, яка представляє собою базу даних компанії-провайдера «X-Byte», та наданням послуг в сфері телекомунікації і інформаційних технологій.
Основними об'єктами бази є:

- Користувачі
- Співробітники
- Клієнти
- Замовлення
- Тарифи
- Послуги
- Оплата (платежі по замовленням)

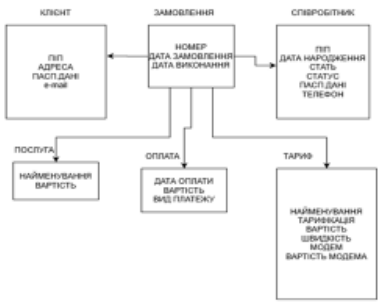
Діаграма потоків даних

7

Слайд 8



Реляційна модель бази даних IT відділу



Об'єктна модель

Об’єктна та реляційна модель БД IT-відділу

Слайд 9

Форма для створення додатку

- Програма складається з багатьох форм, що дозволяють редагувати та переглядати інформацію про замовлення для надання постачальнику послуг

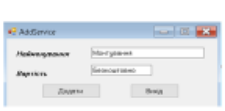
Головна форма проетку, по якій буде розроблено додаток

9

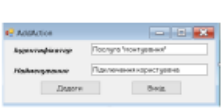
Слайд 10

Екранні форми додатку

- На даному слайді, зображена головна форма (Головне меню) та дія з додавання до системи БД, послуги та її вартості.

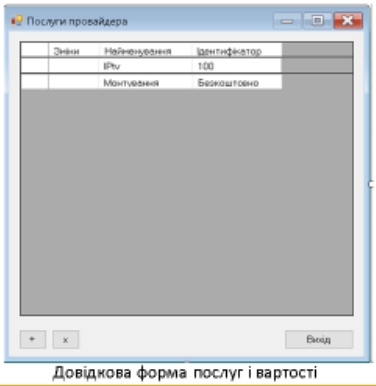


Ввід нового запису послуги



Форма підключення послуги

Головна форма



Довідкова форма послуг і вартості

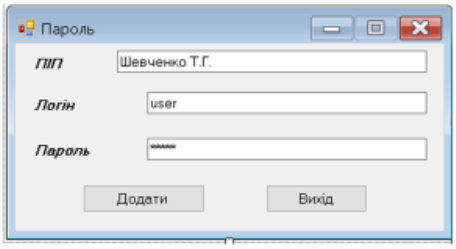
10

Слайд 11

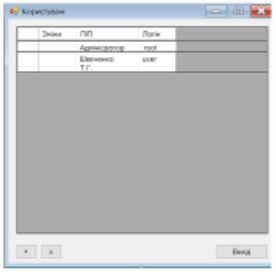
Екранні форми додатку

- На даному слайді зображені екранні форми додатку, які авторизують користувача у системі. Тут може бути створений як user-користувач, з обмеженими правами, так і root-користувач, який має повний доступ в системі.

Форма авторизації користувача



Перегляд і редагування списку користувачів



11

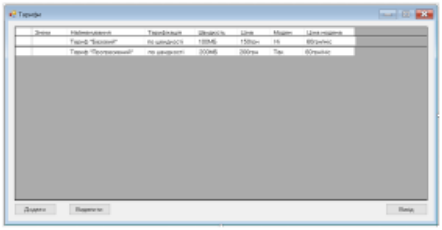
Слайд 12

Екранні форми додатку

- На цьому слайді, зображені екрани форми, які працюють з тарифікацією, ведення, зміною та додаванням тарифів. Тут можна вести його назву, спосіб тарифікації, швидкість та ціну, також чи входить модем в тариф і його вартість.

Введення нового тарифу

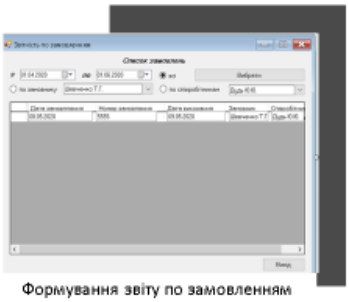
Робота з існуючим тарифом



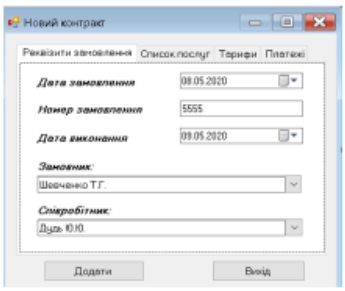
Довідкова інформація про тарифи

12

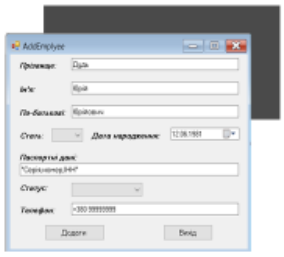
Слайд 13



Формування звіту по замовленням



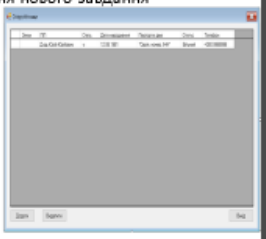
Введення нового завдання



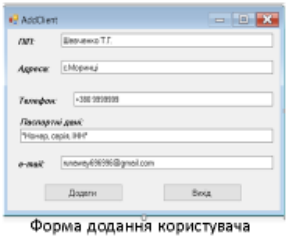
Додання співробітника до таблиць

Екранні форми додатку

- На цьому слайді, зображені екранні формі, по роботі з даними співробітника, його «зайнятістю», формування нового замовлення, додавання користувача до замовлення та співробітника, формування звіту про роботу, ким та коли була виконана, ким замовлена, за якою адресою, яку послугу\тариф, стан оплати, данні для аналізу можливості підключення, побажань клієнтів і тп.



Робота з даними співробітника



Форма додання користувача

13

Слайд 14

Вплив впровадження додатку на економічну складову

- Економічний ефект від інтегрування розробки за рік використання, можна визначити за формулою:

$E_Г = E_У - E_Н \cdot З_К$, де $З_К$ – калькуляція витрат на розробку програмного продукту, $E_н$ – нормальний коефіцієнт капітальних вкладень (0.2), $E_у$ – економія затрат на ручну обробку інформації, від заміни на автоматизовану обробку. Економія коштів при інтегруванні продукту можна розрахувати за формулою: $E_У = З_Р - З_а$. Тут $З_р$ – затрати на ручну обробку інформації, грн, ,

$З_р = О_н * Ц * Г_д / Н_р$, $О_н$ – Обсяг інформації оброблений в ручну, Ц – вартість одної години роботи, грн/год, $Г_д$ - коефіцієнт, який враховує додаткові затрати часу на логічні операції при ручній обробці інформації, $Н_р$ – норма опрацювання даних, Мбайт/год. $З_а$ – Затрати на автоматизовану обробку, грн, $t_а$ – час автоматичної обробки (год), $Ц_м$ – вартість одної робочої години машинного часу, грн/год; t_0 – час роботи відділу, год; $Ц_0$ - вартість одної робочої години в відділі ІТ, грн/год.

$E_У = О_н * Ц * Г_д / Н_р - t_а * Ц_м * t_0 * Ц_0 = 200*100*1.5*2 - 50*1*5*1*200 = 60000 - 50000 = 10000$ грн/год.

Ефективність розробки($E_р$):

$E_р = 10000 - 0.2*10000 = 8000$ грн/год

Тоді ефективність розробки можна визначити по формулі:

$E_р = (E_г + 0,4) / З_к = 8000*0.4/10000 = 0.32$

Використовування в відділі, створеного програмного забезпечення економічно вигідно $E_р \geq 0,2$

14

Слайд 15

ВИСНОВКИ

Результати роботи, проведеної в бакалаврській роботі, що впроваджу інформаційно-аналітичні системи, в телекомунікаційну компанію, вчасності в IT відділ. Було проведено аналіз бізнес-процесів, які відбуваються у відділі, також проаналізовано предметну область, завдяки цьому, стало зрозуміло, який функціонал потрібно реалізувати в додатку для відділу IT. Виконання задач кваліфікаційної роботи, привело до формулювання цілі, для створення та автоматизування інформаційно-аналітичної системи. Додаток, дозволить мінімізувати затрати в часі, та більш доцільніше використовувати ресурси відділу. Що в свою чергу призведе до:

- прискореної роботи з замовленнями клієнтів та їх виконання;
- зменшити вірогідність помилки завдяки відсутності роботи з паперовою документацією;
- здійснювати керування, облік, звітність, актуальну інформацію в одному місці і доступною в пару кліків;
- вести моніторинг зайнятості всіх співробітників чи конкретної особи;
- створювати звіти буде на порядок легше, всі необхідні дані завжди доступні в додатку;
- дасть змогу легко створювати формування ціни для клієнта, розуміючи вартість підключення послуги;
- дозволить бачити важливі дати для користувача, щоб пропонувати йому акційні тарифи, знижки тощо;
- дасть змогу економити час співробітників та оптимізувати їх дії;
- забезпечить інформацією про найближчого користувача і про його обладнання;
- відобразить дату підключення та кінцеву дату тарифу, щоб своєчасно від'єднувати чи під'єднати користувача;
- буде містити в собі актуальну інформацію про дату замовлення та дату його виконання, що допоможе слідкувати та корегувати роботу з ними.

Модель даної інформаційно-аналітичної системи, можна розвивати в бік досягнення більшого функціоналу, ергономічності, додавання нових можливостей, що розширюватимуть стандартні засоби. Включаючи всі фактори описані вище, впровадження такої програми, допоможе сформувати гнучкий функціональний механізм, який здатен допомогти в роботі відділу IT, в роботі з бізнес-процесами, та допоможе більш краще, працювати з користувачами, та приманювати нових клієнтів.

15

Слайд 16

Апробація роботи

Горобець В.Ю. Розробка інформаційно-аналітичної системи телекомунікаційної компанії «X-Byte»».//В.П.Ярцев, Горобець В.Ю.// «Дорожня карта інформаційно-телекомунікаційної галузі» .Тези доповідей. 25 квітня 2020р., Київ, Україна.

Горобець В.Ю. Розробка інформаційно-аналітичної системи телекомунікаційної компанії «X-Byte»».//В.П.Ярцев, Горобець В.Ю.// "Сучасні інфокомунікаційні технології".Тези доповідей. 25 травня 2020р., Київ, Україна.

16

ДОДАТОК А

ПЕРЕЛІК ЗАУВАЖЕНЬ НОРМОКОНТРОЛЕРА

Позначення	Сторінка	Зміст зауваження