**АНОТАЦІЯ**

У даному документі представлено етапи розробки програми анатомічного атласа-посібника, котра написана на мові програмування С#. Поставлено задачу на розробку цієї системи і визначено технології для її створення. Також представлено детальний опис роботи основних модулів програми.

**Ключові слова:** ІНФОРМАЦІЙНО-ДОВІДКОВА СИСТЕМА, АНАТОМІЧНИЙ АТЛАС-ПОСІБНИК.

**ANNOTATION**

This document presents the stages of the information system Aircraft Company, which is written in the programming language С#. Tasked to develop this system and identified technologies for its creation. Also, for a more detailed description of the main modules of the program.

**Keywords:** REFERENCE INFORMATION SYSTEM, AIRCRAFT COMPANY.

ЗМІСТ

[ВСТУП 5](#_Toc437816281)

[1 ОГЛЯД ТА АНАЛІЗ ЗАCОБІВ ПРОЕКТУВАННЯ ТА РОЗРОБКИ ІНФОРМАЦІЙНОЇ СИСТЕМИ АВІАБУДІВНОГО ПІДПРИЄМСТВА 8](#_Toc437816282)

[1.1 Призначення та область застосування об’єкта проектування 8](#_Toc437816283)

[1.2 Огляд способів і засобів вирішення поставлених задач 8](#_Toc437816284)

[1.3 Аналіз переваг і недоліків існуючих методів (аналогів) 9](#_Toc437816285)

[1.4 Постановка задачі на розробку 9](#_Toc437816286)

[1.5 Висновки до першого розділу 10](#_Toc437816287)

[2 ФОРМУЛЮВАННЯ ТА АНАЛІЗ ВИМОГ ДО ІНФОРМАЦІЙНОЇ 11](#_Toc437816288)

[СИСТЕМИ АВІАБУДІВНОГО ПІДПРИЄМСТВА 11](#_Toc437816289)

[2.1 Вибір технології проектування 11](#_Toc437816290)

[2.2 Формулювання вимог 13](#_Toc437816291)

[2.2.1 Первинні вимоги 14](#_Toc437816292)

[2.2.2 UML – діаграма прецедентів 14](#_Toc437816293)

[2.3 Визначення поведінки об’єктів 19](#_Toc437816294)

[2.4 Висновки до другого розділу 20](#_Toc437816295)

[3 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ АВІАБУДІВНОГО ПІДПРИЄМСТВА 21](#_Toc437816296)

[3.1 Архітектурне проектування 21](#_Toc437816297)

[3.2 Проектування баз даних 23](#_Toc437816298)

[3.3 Детальне проектування 26](#_Toc437816299)

[3.4 Висновки до третього розділу 27](#_Toc437816300)

[4 РОЗРОБКА І ТЕСТУВАННЯ СИСТЕМИ 28](#_Toc437816301)

[4.1 Опис логічної структури програми 28](#_Toc437816302)

[4.2 Опис і обґрунтування методу організації вхідних і вихідних даних 28](#_Toc437816303)

[4.3 Опис і обґрунтування вибору апаратних і програмних засобів 29](#_Toc437816304)

[4.4 Вибір мови програмування і СУБД 29](#_Toc437816305)

[4.4 Тестування програмного продукту 30](#_Toc437816306)

[4.5 Висновок до четвертого розділу 32](#_Toc437816307)

[5 КЕРІВНИЦТВО КОРИСТУВАЧА 33](#_Toc437816308)

[ВИСНОВКИ 44](#_Toc437816309)

[СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ 45](#_Toc437816310)

[Додаток А 46](#_Toc437816311)

[Додаток Б 47](#_Toc437816312)

[Додаток В 48](#_Toc437816314)

# ВСТУП

Ознакою теперішнього часу, є лавинне нагромадження інформації та бурхливий розвиток мікроелектронної техніки. Наша цивілізація нестримно прямує до комп’ютерної ери. Комп’ютери вже увійшли в різноманітні сфери повсякденної діяльності суспільства, відбувається перехід до інформаційних технологій в економіці, політиці, школах, медичних закладах.

В цей час часто виникає потреба працювати з даними з різних джерел, кожне з яких пов’язане з певним видом діяльності. Для координації всіх цих даних необхідні певні знання й організаційні навички. У сучасному часі будь-яка інформаційна система повинна базуватись на використанні баз даних.

Якість продукції формується на стадії розробки продукції і супроводжується нормативно-технічною документацією. Якість продукції повинна забезпечуватися на всіх стадіях виробництва і підтримуватися на стадіях зберігання, транспортування та реалізації.

Заплановане забезпечення рівні якості залежить від багатьох факторів:

* від чіткого формулювання в нормативно-технічної документації вимог до якості продукції;
* від якості вихідної сировини або напівфабрикатів;
* досконалості рецептури та технології;
* дотримання технологічних процесів;
* рівня технічного оснащення виробництва;
* рівня кваліфікації кадрів;
* організації виробництва та обслуговування;
* дотримання контролю якості продукції на всіх стадіях її виробництва;
* зацікавленості у випуску високоякісної продукції.

Темою дипломної роботи є розгляд питань щодо розробки інформаційної системи ательє авіабудівного підприємства.

**Актуальність теми** полягає у тому, що із розвитком новітніх технологій більшість посібників стало електронними, та створюються програмні продукти для вивчення різних предметів, в тому числі вивчення анатомії людини, тобто для користувачів, які хочуть вивчити системи організму буде полегшено сприйняття інформації та запам’ятовування розташування органів та функцій організму. Це може бути забезпечено засобами сучасних інформаційних технологій.

**Об’єкт дослідження** – методи та засоби створення інформаційної системи авіабудівного підприємства.

**Предмет дослідження** – це програмні засоби для адміністрування та автоматизації відповідних процесів.

**Область застосування:** допомога користувачеві інформаційно-довідкової системи з роботою над замовленнями, постачаннями та роботою з клієнтами та працівниками.

**Мета даної роботи** є розробка інформаційної системи, що служитиме для автоматизації внутрішньої роботи авіабудівного підприємства. В інформаційній системі будуть оброблятися замовлення, постачання, працівники та їх задачі.

Майбутня система повинна виконувати такі завдання:

1. Додавання замовника.
2. Додавання постачальника.
3. Додавання відділу.
4. Створення формули виготовлення деталей.
5. Управління робітниками, що виготовлюють деталі.
6. Створення деталей.
7. Пошук за будь-яким атрибутом по замовленнях.
8. Створення замовлення на деталі.

Система призначена для комп’ютеризованого управління виробництвом.

**Практичне значення** одержаних результатів полягає в розробці програмного забезпечення для автоматизації процесів допомоги користувачам інформаційно-довідкової системи з адмініструванням авіабудівного підприємства з урахуванням потреб користувача. Ця програмна розробка може знайти своє застосування в будь-яких авіабудівних підприємствах.

Дипломна робота складається із вступу, п'яти розділів, висновків, списку використаної літератури, трьох додатків. Робота містить одну таблицю, вісімнадцять рисунків. Роботу викладено на сорока восьми сторінках друкованого тексту [15].

**1 ОГЛЯД ТА АНАЛІЗ ЗАCОБІВ ПРОЕКТУВАННЯ ТА РОЗРОБКИ ІНФОРМАЦІЙНОЇ СИСТЕМИ АВІАБУДІВНОГО ПІДПРИЄМСТВА**

**1.1 Призначення та область застосування об’єкта проектування**

Авіаційна промисловість — це сукупність підприємств, зайнятих конструюванням, виробництвом і випробуваннями літаків, космічних апаратів і кораблів, агротехніки, а також їхніх двигунів і бортового обладнання (електричної та електронної апаратури). Ці підприємства належать державі або приватним власникам. Авіаційна промисловість має важливе політичне й економічне значення. Нею значною мірою визначаються промисловий потенціал і престиж держави: її підприємства постачають свою продукцію на внутрішній та зовнішні ринки, забезпечують замовленнями інші галузі господарства, надають велику кількість робочих місць.

Основним призначенням об’єкта проектування є – адміністрування роботи авіабудівного підприємства.

Основною областю застосування даної предметної області є – авіабудівні підприємства.

## 1.2 Огляд способів і засобів вирішення поставлених задач

До засобів для вирішення задачі є вибір правильного програмного середовища розробки.

Для адміністрування виробництва, що є основною функцією програмного додатку, крім всіх інших додаткових, без яких даний програмний продукт не мав би значної популярності серед авіабудівних підприємств, було вирішено:

* створити редактор формул деталей;
* створити систему додавання деталей;
* додавати деталі через редактор формул деталей;
* створити моделі додавання працівників, відділів, постачальників і замовників.

## 1.3 Аналіз переваг і недоліків існуючих методів (аналогів)

У час новітніх технологій та розвитку комп’ютерної техніки, кількість схожих програм, які використовують користувачі для задоволення власних потреб, різко зростає. При розробці власної системи, необхідно знайти та дослідити схожі програми, які були розроблені раніше іншими людьми. Це пов’язано із тим, що необхідно з’ясувати, які є плюси та мінуси у досліджуваних програмах, та використовуючи отриману інформацію – розробити власну систему, яка буде без тих мінусів, які присутні в інших системах.

Програмних продуктів, які допомагають здійснювати автоматизоване управління авіабудівної промисловості не було знайдено.

## 1.4 Постановка задачі на розробку

Інформаційна система авіабудівного підприємства включає клієнтів, замовлення, робітників, відділи, деталі, формули виготовлення деталей, постачальників, поставки.

Спроектувати базу даних, призначену для зберігання інформації про підприємство, яке виробляє різні деталі для авіабудівного підприємства. Кожна деталь має свій унікальний номер, за яким визначається відпускна ціна. Для виготовлення кожної деталі використовується формула-шаблон, яка містить деталі-компоненти, їх кількість та послідовність операцій над ними, деякі деталі не мають формули виробництва. На деталі-компоненти відомі фіксовані закупівельні ціни за одиницю ваги. Деталі-компоненти купуються у постачальників. База містить інформацію про клієнтів та постачальників, робітників та відділи.

## 1.5 Висновки до першого розділу

В даному розділі було проаналізовано методи обробки та адміністрування даних авіабудівного підприємства та їх замовлення. До засобів випробувань можуть відноситися технічні пристрої, матеріали для їх проведення, які потребують значних фінансових витрат.

У даному розділі наведено результати теоретично-аналітичного інформаційного пошуку на задану тему дипломної роботи «Інформаційна система авіабудівного підприємства». В ньому аналізується фактичний стан задачі, що розв’язується, на основі матеріалів, які характеризують об’єкт проектування (зібраних під час проходження переддипломної практики); містить теоретичний виклад важливих аспектів проблеми, що досліджується, та критичний огляд наукової літератури і періодичних джерел з визначеної тематики.

Поставлена задача на створення інформаційної системи авіабудівного підприємства.

# 2 ФОРМУЛЮВАННЯ ТА АНАЛІЗ ВИМОГ ДО ІНФОРМАЦІЙНОЇ

# СИСТЕМИ АВІАБУДІВНОГО ПІДПРИЄМСТВА

На етапі аналізу вимог проходить структуризація вже зібраних раніше вимог. Мета етапу – надати чіткий перелік не дубльованих вимог до системи, які повинні бути виділені з надлишкових і частково дублюються сценаріїв і користувальницьких історій, які були отриманих на попередньому етапі.

Правильно згруповані вимоги допоможуть обійтися мінімальним кількістю функціоналу для задоволення максимально більшої кількості цілей, а це, в свою чергу, допоможе заощадити бюджет і не дасть розповзтися рамкам проекту.

## 2.1 Вибір технології проектування

Для найбільш ефективного виконання поставленого завдання потрібно вибрати необхідну технологію проектування, вибір варіювався між структурно-орієнтованою та об’єктно-орієнтованою технологіями проектування інформаційної системи (ІС) [8].

Зараз у програмній інженерії є два основних підходи до розробки ПЗ ІС, принципова різниця між якими зумовлена різними способами декомпозиції систем: функціонально-модульний (структурний) підхід, в основу якого покладений принцип функціональної декомпозиції, при якій структура системи описується в термінах ієрархії її функцій і передачі інформації між окремими функціональними елементами, та об’єктно-орієнтований підхід, що використовує об’єктну декомпозицію, описує структуру ІС у термінах об’єктів і зв’язків між ними, а поведінку системи ‒ в термінах обміну повідомленнями між об’єктами.

Отже, сутність структурного підходу до розроблення ПЗ ІС полягає в його декомпозиції на автоматизовані функції: система розбивається на функціональні підсистеми, що у свою чергу поділяються на підфункції, вони ‒ на задачі і так до конкретних процедур. При цьому ІС зберігає цілісність подання, де всі складові є взаємопов’язаними. При розробці системи «знизу нагору», від окремих задач до всієї системи, цілісність втрачається, виникають проблеми при описі інформаційної взаємодії окремих компонентів.

Базовими принципами структурного підходу є:

1. Принцип «поділяй і пануй».

2. Принцип ієрархічного упорядкування ‒ принцип організації складових системи в ієрархічні деревоподібні структури з додаванням нових деталей на кожному рівні. Виділення двох базових принципів не означає, що інші принципи другорядні, оскільки ігнорування кожного з них може призвести до непередбачених наслідків.

У структурному аналізі використовуються в основному дві групи засобів, що ілюструють функції, виконувані системою і відносини між даними. Кожній групі засобів відповідають певні види моделей (діаграм), найбільш поширеними серед яких є наступні:

* SADT (Structured Analysis and Design Technique) моделі і відповідні функціональні діаграми;
* DFD (Data Flow Diagrams) діаграми потоків даних;
* ERD (Entity–Relationship Diagrams) діаграми «сутність-зв’язок».

Перераховані моделі в сукупності дають повний опис ІС незалежно від того, чи є вона існуючою чи знову розробляється. Склад діаграм в кожному конкретному випадку залежить від необхідної повноти опису системи.

Об’єктно-орієнтоване проектування програмного забезпечення зводиться насамперед до розробки моделі реальної системи, для вирішення завдання якої вона створюється.

Метод об’єктно-орієнтованого проектування ґрунтується на: моделі побудови системи як сукупності об’єктів абстрактного типу даних, модульній структурі програм, спадному проектуванні, використовуваному при виділенні об’єктів [11].

При використанні об’єктно-орієнтованого підходу проектування в якості будівельних блоків використовуються об’єкти, що містять свої власні коди і дані. На відміну від структурного проектування, де структура програм представляється деревом ієрархій, структура програм у об’єктно-орієнтованому підході представляється графом взаємодії об’єктів.

Основними переваги об’єктно-орієнтованого проектування є:

* активне використання абстракції предметної області;
* підвищення стабільності в представленні змін;
* існування різнопланових модельних поглядів;
* на базі модельних поглядів покращуються можливості для повторного використання;
* з’являється можливість управління масштабованістю;
* покращується підтримка аспектів надійності та безпеки;
* успадковується підтримка паралелізму.

Спираючись на перелічені вище факти, для проектування розроблюваної системи було обрано об’єктно-орієнтовану технологію проектування. Вона дозволить краще і простіше спроектувати інформаційну систему підтримки здорового способу життя.

## **2.2 Формулювання вимог**

Аналіз вимог полягає у визначенні потреб та умов, які висуваються щодо обробки даних враховуючи можливо конфліктні вимоги різних замовників, таких як користувачі.

Аналіз вимог є критичним для успішної розробки проекту. Вимоги мають бути задокументованими, вимірними, протестованими, пов’язаними з бізнес-потребами, і описаними з рівнем деталізації достатнім для конструювання системи. Вимоги можуть бути архітектурними, структурними, поведінковими, функціональними та не функціональними [15].

### 2.2.1 Первинні вимоги

Первинні вимоги документують бажання і потреби вимог замовника і відображаються діаграмою прецедентів (варіантів використання, Use Case). Перед побудовою діаграми вони описуються мовою, що є зрозумілою замовнику.

Основними вимогами для створення інформаційної системи є:

* реєстрація клієнта в системі;
* перегляд інформації про замовлення;
* перегляд інформації про відділи;
* перегляд інформації про постачальників;
* перегляд інформації про замовників;
* перегляд інформації про працівників;
* зручний пошук;
* каталог формул;
* каталог деталей.

### 2.2.2 UML – діаграма прецедентів

Для відображення вимог щодо взаємодії із зовнішнім оточення була розроблена діаграма прецедентів. Діаграма прецедентів є графом, що складається з множини акторів, прецедентів (варіантів використання) обмежених границею системи (прямокутник), асоціацій між акторами та прецедентами, відношень серед прецедентів, та відношень узагальнення між акторами. Діаграми прецедентів відображають елементи моделі варіантів використання. Суть даної діаграми полягає в наступному: проектована система представляється у вигляді множини сутностей чи акторів, взаємодіючих із системою за допомогою так званих варіантів використання. Варіант використання (use case) служить для опису сервісів, що система надає актору. Іншими словами, кожен варіант використання визначає деякий набір дій, чинений системою при діалозі з актором. При цьому нічого не говориться про те, яким чином буде реалізована взаємодія акторів із системою.

Для відображення засобів взаємодії системи із зовнішнім середовищем, була використана діаграма прецедентів. Діаграма прецедентів описує функціональність, яка буде надаватись користувачам системи, котра проектується. Представляється шляхом використання прецедентів та акторів, а також відношень між ними. Набір усіх прецедентів діаграми фактично визначає функціональні вимоги.

Діаграма прецедентів зображена на рисунку 2.1, на якій представлені актор, а точніше, адміністратор, він може виконувати такі дії над системою: створення нових моделей, реєстрація користувачів, додавання тканин.

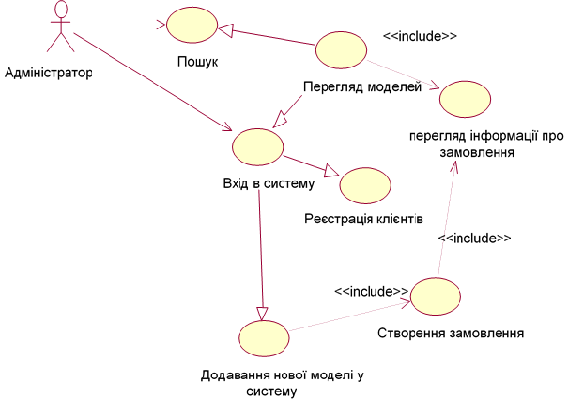


Рисунок 2.1 – Діаграма прецедентів

#### 2.2.3 Детальні вимоги

Детальні вимоги документують вимоги спеціально структурованою мовою за допомогою діаграм діяльності та діаграм взаємодій. Вони деталізують первинні вимоги.

На основі первинних вимог були виділені такі актори: Адміністратор, зареєстрований користувач та не зареєстрований користувач. Відповідно до цього були сформовані такі детальні вимоги:

1. Реалізувати тип користувача «Адміністратор».
2. Реалізувати тип користувача «Зареєстрований користувач».
3. Реалізувати тип користувача «Не зареєстрований користувач».
4. Надати можливість адміністратору додавати нову модель до бази даних.
5. Надати можливість адміністратору формувати каталог замовлень.
6. Надати можливість адміністратору додавати замовлення до бази даних.
7. Надати можливість адміністратору редагувати замовлення.
8. Надати можливість адміністратору видаляти моделі.
9. Створити пошук для будь-якого користувача системи.
10. Надати можливість адміністратору керувати персоналом.

#### 2.2.4 Формулювання та аналіз вимог за допомогою діаграми діяльності

Діаграма діяльності – в UML, візуальне представлення графу діяльностей. Граф діяльностей є різновидом графу станів скінченного автомату, вершинами якого є певні дії, а переходи відбуваються по завершенню дій.

Дія є фундаментальною одиницею визначення поведінки в специфікації. Дія отримує множину вхідних сигналів, та перетворює їх на множину вихідних сигналів. Одна із цих множин, або обидві водночас, можуть бути порожніми. Виконання дії відповідає виконанню окремої дії. Подібно до цього, виконання діяльності є виконанням окремої діяльності, буквально, включно із виконанням тих дій, що містяться в діяльності. Кожна дія в діяльності може виконуватись один, два, або більше разів під час одного виконання діяльності. Щонайменше, дії мають отримувати дані, перетворювати їх та тестувати, деякі дії можуть вимагати певної послідовності. Специфікація діяльності (на вищих рівнях сумісності) може дозволяти виконання декількох (логічних) потоків, та існування механізмів синхронізації для гарантування виконання дій у правильному порядку.

Будується діаграма діяльності для створення замовлень. Ця діаграма працює тільки з адміністратором системи. З початку треба заповнити форму замовлення, передати дані на базу даних. На рисунку 2.2 зображено діаграму діяльності для аналізу складу продукту.



Рисунок 2.2 – Діаграма діяльності створення замовлення на модель

Після діаграми діяльності будуємо діаграму послідовності. Діаграма послідовності відображає взаємодії об’єктів впорядкованих за часом. Зокрема такі діаграми відображають задіяні об’єкти та послідовність відправлених повідомлень. На діаграмі послідовностей показано у вигляді вертикальних ліній різні процеси або об’єкти, що існують водночас. Надіслані повідомлення зображуються у вигляді горизонтальних ліній, в порядку відправлення.

#### 2.2.5 Аналіз вимог за допомогою діаграми послідовності

Побудуємо діаграму послідовності для додавання нової моделі в систему. Щоб додати нову деталь в систему потрібні такі об’єкти «Адміністратор», «Формули», «База даних». Спочатку адміністратор вибирає формулу створення, або додає нову формулу в редакторі формул. Після обробки дані зберігаються в базі даних. На рисунку 2.3 зображено діаграму послідовності варіанту використання «Додавання нової деталі до бази даних» для розроблюваної системи [15].

 Рисунок 2.3 – Діаграма послідовності варіанту використання «Додавання нової деталі до бази даних»

## 2.3 Визначення поведінки об’єктів

Об’єкти не існують ізольовано, а піддаються впливу або самі впливає на інші об’єкти. Поведінка характеризує те, як об’єкт впливає або піддається впливу інших об’єктів з точки зору зміни стану цих об’єктів та передачею повідомлень. Іншими словами, поведінка об’єкта повністю визначається його діями.

Як правило, в об’єктних і об’єктно-орієнтованих мовах (ООП) операції, що виконуються над даним об’єктом, називаються методами (методичною частиною об’єкта) і входять складовою частиною в визначення класу. З практики відомо п’ять основних видів операцій над об’єктами:

* модифікатор – операція, яка змінює стан об’єкта шляхом запису або доступу;
* селектор – операція, яка дає доступ для визначення стану об’єкта без його зміни (операція читання);
* ітератор – операція доступу до змісту об’єкта частинами (в певній послідовності);
* конструктор – операція створення і (або) ініціалізація об’єкта;
* деструктор – операція руйнування об’єкта та (або) звільнення займаної ним пам’яті.

## 2.4 Висновки до другого розділу

Аналіз вимог дає можливість правильно зрозуміти поставлене завдання. Визначивши поставлені вимоги можна перейти до проектування системи. Спочатку аналізують первинні вимоги, щоб потім розробити детальні вимоги. Детальні вимоги потрібні програмістам, щоб перейти до проектування системи. Вимоги розробника документують спеціалізованою мовою за допомогою діаграм взаємодій та діаграм діяльності.

Далі відбувається перехід до об’єктного аналізу інформації. На цьому етапі визначають перелік об’єктів на основі діаграм прецедентів. Таким чином модель прецедентів керує об’єктним аналізом. Результат проектування на цьому етапі має вигляд таблиць в яких відображено перелік об’єктів та їх функції.

Після об’єктного аналізу визначають поведінку ідентифікованих об’єктів. Ці об’єкти можуть мати просту поведінку, поведінку яка визначається станами та поведінку з неперервним станом. Діаграми послідовностей, діаграми станів та часові діаграми дозволяють описати сценарії поведінки розробленої системи.

## 3 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ АВІАБУДІВНОГО ПІДПРИЄМСТВА

Якщо аналіз об’єкта проектування – це розробка відповідної логічної моделі, яка описує усі можливі та прийнятні варіанти розв’язання задач, які вказують, що повинна робити система, то проектування – це вироблення рішення відповідно до моделі аналізу, яке оптимізує набір критеріїв проектування, які показують, як ця поведінка (робота) системи може бути реалізована.

Оскільки, у даній роботі застосовується об’єктно-орієнтована технологія проектування, тому проектна частина повинна складатися із таких етапів:

1. Архітектурне проектування – описує логічну структуру ІС (програмні класи, підсистеми, пакети) та їх зв’язки.
2. Детальне проектування – описує структури даних та алгоритмів всередині окремих класів [15].

## 3.1 Архітектурне проектування

Один зі шляхів архітектурного проектування – традиційний неформальний підхід до визначення архітектури системи, її компонентів, способів їхнього подання й об’єднання в систему, який можна назвати загальносистемним. Фактично архітектура, що створюється згідно з таким підходом, є чотирерівневою і містить у собі:

1. Системні компоненти. Вони здійснюють взаємодію з периферійними пристроями комп’ютерів (принтери, клавіатура, сканери, маніпулятори і т.п.), використовуються при побудові операційних систем.
2. Загальносистемні компоненти. Вони забезпечують взаємодію з універсальними сервісними системами середовища роботи прикладної системи, такими як операційні системи, СКБД, системи баз знань, системи керування мережами і т.п.
3. Специфічні компоненти певної прикладної області, що входять до складу компонентів програмної системи і призначені для розв’язання задач в межах означеної області (наприклад, бізнес-задачі).
4. Прикладні програмні системи, що призначені для виконання завдань з обробки інформації, які постають перед окремими групами споживачів інформації з різних предметних областей (офісні системи, системи бухгалтерського обліку й ін.) і можуть використовувати компоненти нижчих рівнів.

Компоненти кожного з виділених рівнів використовуються, як правило, на своєму або вищому рівні. Кожен рівень відбиває відповідний набір знань, умінь і навичок фахівців, що створюють або використовують компоненти. Цей набір визначає відповідний розподіл фахівців програмної інженерії на аналітиків, прикладників, програмістів й ін.

При проектуванні архітектури програмна система розглядається як композиція компонентів третього рівня з доступом до компонентів першого і другого рівнів. Тобто архітектурне проектування – це розроблення компонентів третього рівня, визначення вхідних і вихідних даних рівнів ієрархії компонентів і їхніх зв’язків.

Результат проектування – архітектура й інфраструктура, що містять у собі набір об’єктів, з яких можна формувати деякий конкретний вид архітектурної схеми для конкретного середовища виконання системи, а також набір елементів керування і контролю. Проектування архітектури системи завершується створенням опису, в якому відображені зафіксовані проектні рішення, логічна і фізична структура системи, а також способи взаємодії об’єктів.

Розглянемо кожен з етапів проектування програмного забезпечення системи ательє верхнього одягу.

## 3.2 Проектування баз даних

Проаналізувавши предметну область, було виділено наступні сутності [1]:

1. Клієнти.
2. Замовлення.
3. Робітники.
4. Деталі.
5. Формули.
6. Операції.
7. Постачання.
8. Поставки.
9. Деталі постачань.
10. Деталі замовлень.
11. Операції формули
12. Деталі формули
13. Виробництво
14. Відділи

Сутності «Деталі формули», «Деталі замовлень», «Деталі постачань», «Деталі формули», «Операції формули» є слабкими сутностями.

Створення таких сутностей обумовлений діяльністю заданої предметної області – «Інформаційна система авіабудівного підприємства», для комфортної роботи із програмою.

Побудова концептуальної моделі здійснювалась за допомогою програми Sybase PowerDesighner.

Виробництво має багато співробітників. Інформацію про співробітників потрібно зберігати в базі даних. Для цього необхідно використати сутність «Робітники» з атрибутами, що містять детальну інформацію про співробітника та «Відділи». Дана сутність зображена на рисунку 1.



**Рис. 3.1** Сутності «Відділи» та «Робітники», зі зв’язком «Відносяться»

Виробництво має багато клієнтів. Клієнти замовляють замовлення, замовлення розширюється деталями. Зображено на рисунку 2.

**Рис. 3.2** Сутності «Клієнт» та «Замовлення», зі зв’язком «Замовляє», «Деталі замовлення» та «Замовлення», зі зв’язком «Деталізує».



**Рис. 3.3** Сутності «Постачальник» та «Постачання», зі зв’язком «Замовляє», «Деталі постачання» та «Постачання», зі зв’язком «Деталізує».



**Рис. 3.4** Сутність деталі

Головне створення деталі, операції, формули зображено на рисунку 4.



**Рис. 3.5** Сутності «Деталі» ,«Формули» та «Деталі формули», зі зв’язком «уточнює» , «Операції» та «Операції формули», зі зв’язком «Включає»

## 3.3 Детальне проектування

Детальне проектування – це технічна діяльність, яка слідує за вибором архітектури. Основною метою цієї діяльності є якомога повна підготовка проекту до його реалізації. Іншими словами, програмісти повинні мати можливість реалізувати детальний проект, сконцентрувавшись тільки на проблемах, пов’язаних із створенням програмного коду.

Розглянемо кожний клас окремо:

1. DB\_con – Клас в якому зорганізується робота з базою.
2. AddDetail – Модуль, де реалізовано додавання деталі.
3. AddCust – Модуль, де реалізовано додавання клієнта.
4. AddOrder – Модуль, де реалізовано додавання замовлення.
5. AddOrderDetails – Модуль, де реалізовано додавання замовлення.
6. AddWorker – Модуль, де реалізовано додавання робітника.
7. AddDel – модуль, де реалізовано додавання постачання.
8. AddSupp - модуль, де реалізовано додавання постачальника
9. HOME – Головна форма.
10. AllDetails – Модуль, де відображаються усі деталі
11. AllCustomers – Модуль, де відображаються усі клієнти
12. AllFormulas – Модуль, де відображаються усі формули
13. AllOffices – Модуль, де відображаються усі відділи
14. AllOperations – Модуль, де відображаються усі операції
15. AllOrders – Модуль, де відображаються усі замовлення
16. AllWorkers – Модуль, де відображаються усі робітники
17. AllDeliv – Модуль, де відображаються усі постачання
18. AllSuppliers– Модуль, де відображаються усі постачальники

## 3.4 Висновки до третього розділу

В цьому розділі було проведено проектування системи відповідно до моделі аналізу, що показало як саме поведінка системи може бути реалізована.

Проектна частина пояснювальної записки містить опис процесів проектування інформаційної системи відповідно до її призначення. Оскільки застосовується технологія об’єктно-орієнтованого проектування, тому процес проектування складається із таких етапів: архітектурне і детальне проектування.

Спочатку було створено архітектуру на найвищому логічному рівні. Етап архітектурного проектування повинен бути відображений шляхом опису процесу об’єктно-орієнтованої декомпозиції системи до рівня переліку підсистем та їх зв’язків, опису її логічної структури у вигляді пакетів.

Після архітектурного проектування починається детальне проектування. Основним будівельним блоком детального проектування є класи. Результат етапу детального проектування відображений діаграмами класів, які призначені для статичного моделювання об’єктів.

Діаграми класів вважають основним засобом для представлення структури систем в термінах базових будівельних блоків і відносин між ними.

При детальному проектуванні були описані структури даних програми, а також більшість алгоритмів необхідних у програмному забезпеченні.

## 4 РОЗРОБКА І ТЕСТУВАННЯ СИСТЕМИ

## 4.1 Опис логічної структури програми

При розробці програми використовуємо фізичну модель бази даних, яка знаходиться у Додатку Г.

Прикладна програма буде складатися із декількох модулів.

Головне вікно

Клієнти

Замовлення

Додати

Редагувати

Видалити

Всі замовлення

Постачальники

Поставки

Додати

Видалити

Видалити

Всі поставки

Деталі

Формули

Додати

Видалити

Видалити

Операції

Відділи

Робітники

Додати

Видалити

Видалити

Всі робітники

**Рис 4.1** Функціональна схема програми

## 4.2 Опис і обґрунтування методу організації вхідних і вихідних даних

Вхідними даними у програму буде:

* дані про клієнта;
* зображення моделі;
* інформація при створенні замовлення.

Вихідні дані – це оформлений короткий звіт, що складається із 10 останніх замовлень та база даних клієнтів.

## 4.3 Опис і обґрунтування вибору апаратних і програмних засобів

Для комфортного використання даної програми рекомендується використовувати систему середньої потужності.

Приблизні характеристики:

* + - * Процесор с тактовою частотою 1 ГГц (рекомендується 1.5 ГГц або вище).
* Microsoft Windows 2008 Server, Microsoft Windows Vista, Microsoft Windows 7, Microsoft Windows 8/8.1, Microsoft Windows 10.
* 300 Мб RAM (рекомендується 512 Мб або більше).
* 700 Мб вільного місця для установки програми на жорсткому диску.
* Відео-карта (рекомендується nVidia GeForce 7800 GTX з розширенням 1366x768 або більше).
* Сумісна з Windows мишка.
* Сервер Microsoft SQL Server.
* Можливість з’єднання з локальним або віддаленим сервером MS SQL.

## 4.4 Вибір мови програмування і СУБД

Для виконання даної роботи використовувалось програмне забезпечення Sybase PowerDesigner, Microsoft SQL Server 2012 Express, Microsof Visual Studio 2012 Express for Desktop.

Концептуальна та фізична моделі реалізовані у програмі Sybase PowerDesigner, клієнт-серверна архітектура програми реалізована завдяки використанню Microsoft SQL Server, прикладна програма розроблена і протестована у середовищі Microsof Visual Studio. Для написання прикладної програми використовувалась мова C#.

Такий вибір обґрунтовано тим, що середовище розробки бази даних вільно підходить до середовища розробки програмного забезпечення.

## 4.4 Тестування програмного продукту

Для тестування даного програмного продукту була обрана методика «Білого ящика». Результати тестування наведені у таблицях 4.1-4.2.

Таблиця 4.1 – Тестування ПЗ

|  |  |  |
| --- | --- | --- |
| **Дія** | **Очікуваний результат** | **Отриманий результат** |
| Додати Клієнта | У базу даних внесеться новий запис | У базу даних був внесений новий запис |
| Видалити Клієнта | З бази видаляється запис | З бази даних був видалений запис |
| Додати замовлення | Вибір клієнта. Заповнення необхідних полів. В базу додається запис. | В базу був внесений новий запис. |
| Додати Постачальника | У базу даних внесеться новий запис | У базу даних був внесений новий запис |
| Додати постачання | Вибір постачальника. Заповнення необхідних полів. В базу додається запис. | В базу був внесений новий запис. |
| Додати робітника | Заповнення всіх необхідних полів. В базу даних додається новий запис. | Запис був доданий в базу даних. |
| Додати деталь | Назвати її. Заповнити потрібні поля Додавання деталі в базу. | В базу був доданий запис. |
| Видалення клієнта з замовленням | Видалення клієнта з бази. | Виведення помилки про порушення цілісності. |
| Додати операцію | Назвати її. Заповнити потрібні поля Додавання операції в базу. | В базу був доданий запис. |
| Таблиця 4.2 – Тестування ПЗ | | |
| **Дія** | **Очікуваний результат** | **Отриманий результат** |
| Додати формулу | Назвати її. Заповнити потрібні поля Додавання в базу. | В базу був доданий запис. |
| Додати відділ | Заповнити потрібні поля Додавання деталі в базу. | В базу був доданий запис. |
| Редагування | Зміна вибраних полей | Запис був оновлений в базі |
| Видалення відділу | Вибір рядка і натискання кнопки видалення | Відділ був видалений з бази разом з робітниками |
| Видалення формули | Вибір рядка і натискання кнопки видалення | Формула була видалена з бази |

## 4.5 Висновок до четвертого розділу

У цьому розділі було проведено кодування та тестування системи. За мову програмування було обрано мову C#. В якості СУБД використано середовище Microsoft SQL Server. Для системи було розроблено інтерфейс адміністратора, який забезпечує взаємодію користувача та програмно-технічного забезпечення інформаційної системи.

Після того як додатки було написано, їх було протестовано за допомогою тестування на функціональність та коректність режиму роботи користувача. Система пройшла тести, тим самим підтвердила коректність виконання своїх завдань її програмним забезпеченням.

## 5 КЕРІВНИЦТВО КОРИСТУВАЧА

Проведемо тестування програми. Припустимо, потрібно замовити деталь. При першому запуску програми з’являється заставка рис.6 а потім головне вікно рис.7

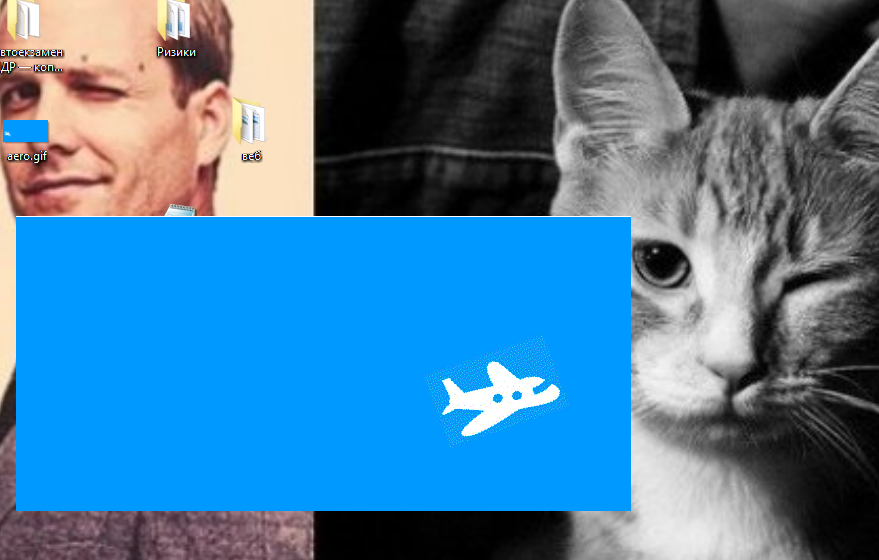


Рис. 6. Заставка програми

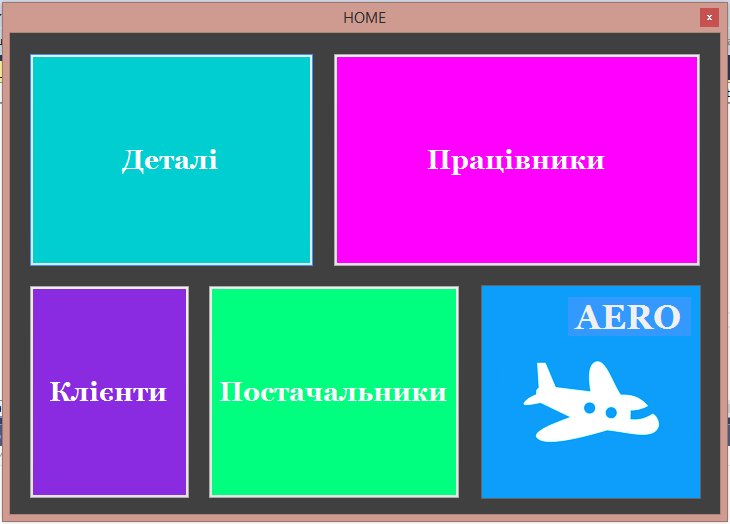


Рис. 7. Головне вікно програми

Спочатку необхідно додати клієнта у базу даних. Для цього, натискаємо на вкладку рис.8 «Клієнти», далі рис.9 «Додати клієнта»

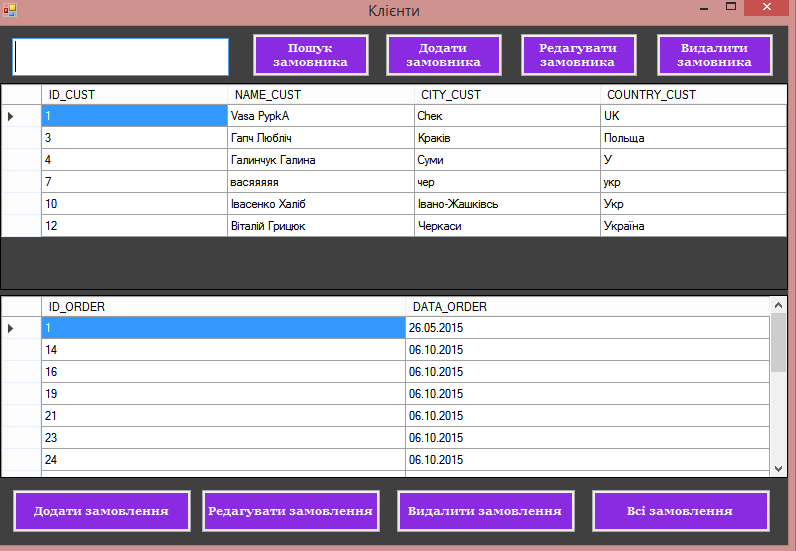


Рис.8 Вікно клієнти

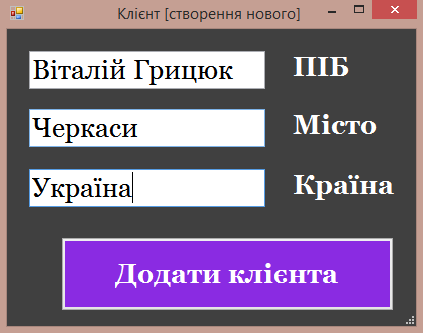


Рис. 9. Форма додавання клієнта

Далі потрібно заповнити відповідні поля. І після цих дій, отримаємо повідомлення про успішне додавання клієнту в базу.

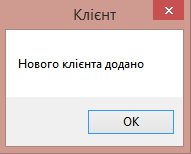


Рис. 10. Повідомлення про додання клієнта

Далі створимо замовлення

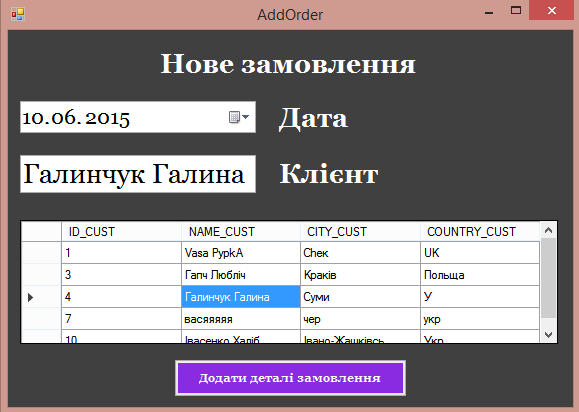
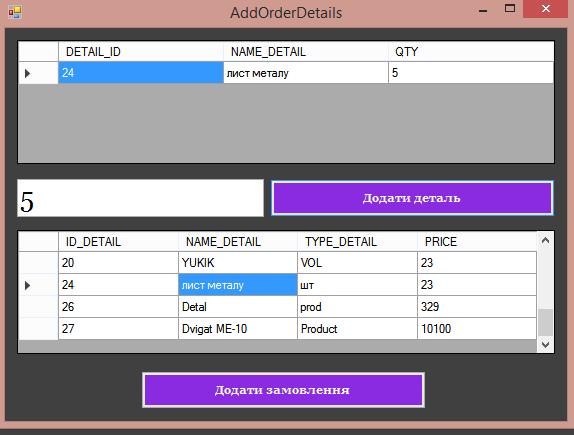
.

Рис. 11. Форма при створенні замовлення

Вибираємо із відомих вже моделей потрібну і натискаємо «Додати». В результаті цього, буде додане до замовлення. Якщо такої немає, то можна створити нову, яка після цих дій буде занесена в базу.

Рис. 11. Форма при створенні деталей замовлення

Для знаходження клієнта в текстовому полі пишемо значення і натискаємо «Пошук замовника»

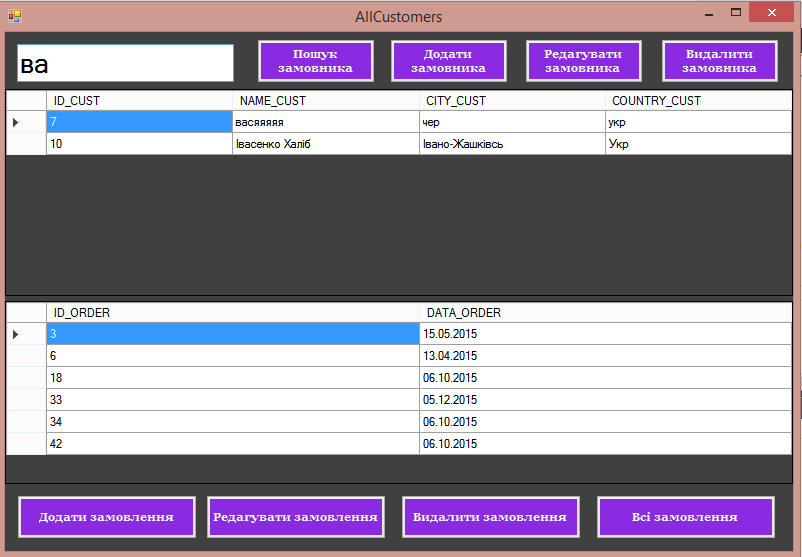


Рис.8 Вікно клієнти (Пошук по «ва»)

Кнопка «Відділи»

Далі додаємо відділ та працівника. Для цього натиснути на головній формі «Відділи»

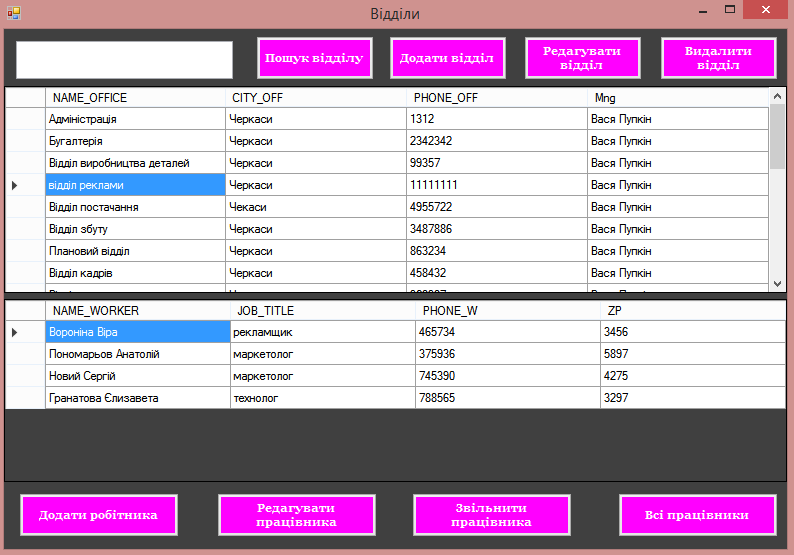


Рис. 12. Форма відділи

Натискаємо «Додати відділ»

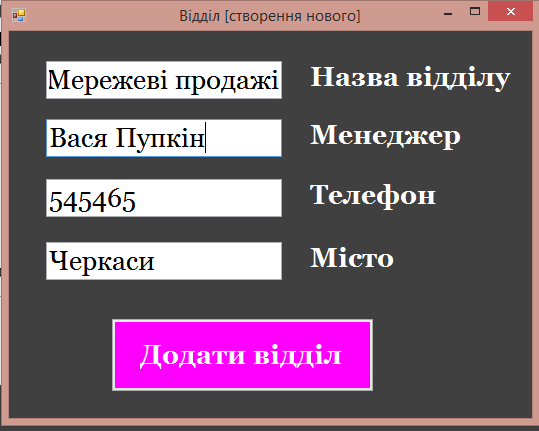


Рис. 13. Форма додавання відділу

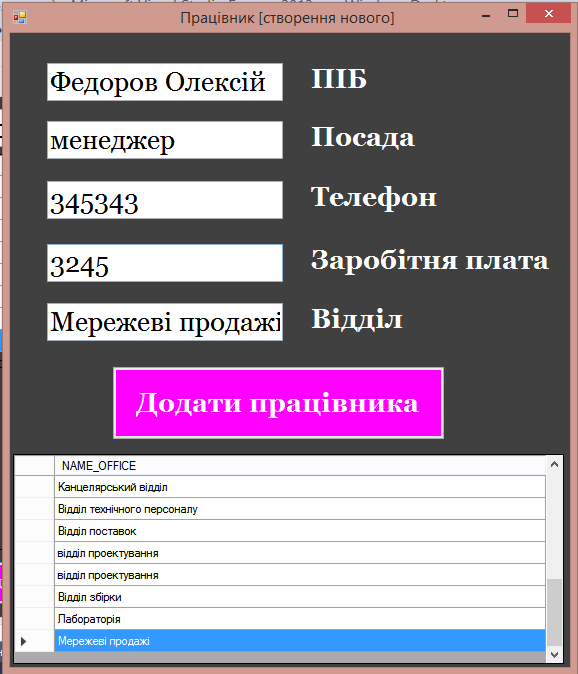


Рис. 13. Форма додавання працівника відділу

Перевіряємо додавання відділу рис 13.З рисунка видно що відділ і робітник доданий.

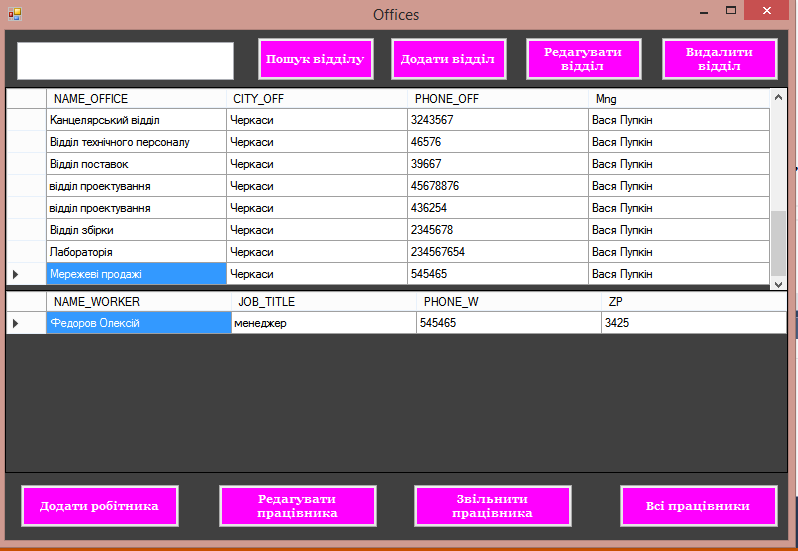


Рис. 12. Форма відділи

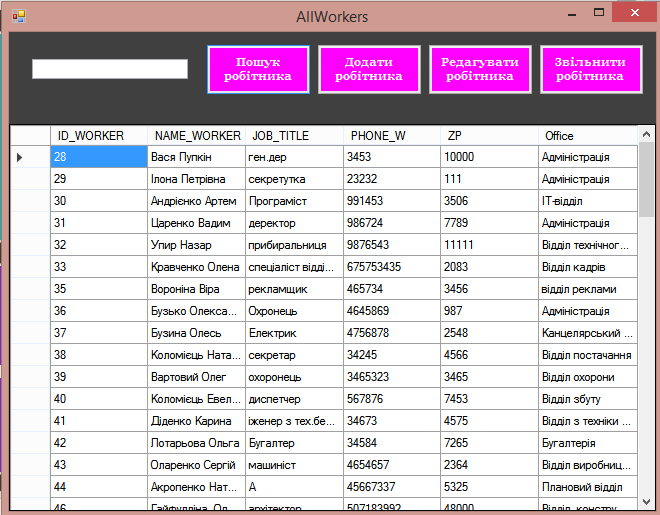


Рис. 13. Форма усі працівники

Кнопка «Деталі»

Наступним перевіримо деталі



Рис. 14. Деталі

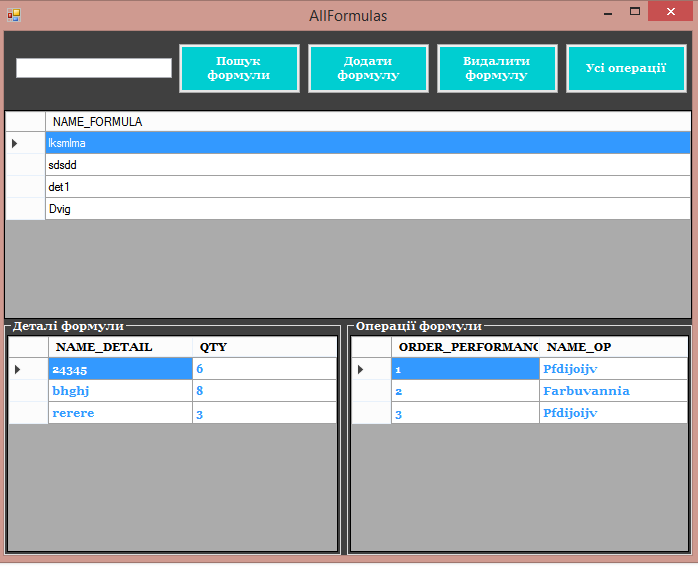


Рис. 15. Перегляд формули

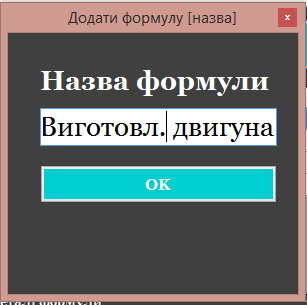


Рис. 16. Додавання нової формули

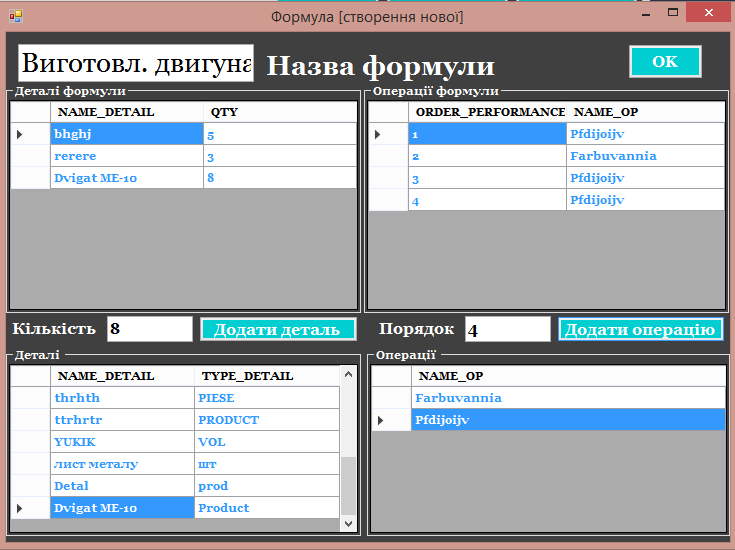


Рис. 17. Додавання деталей та операцій формули

Кнопка «Постачальники»

Натискаємо в головному вікні кнопку «Постачальники», відкривається форма додання постачальника

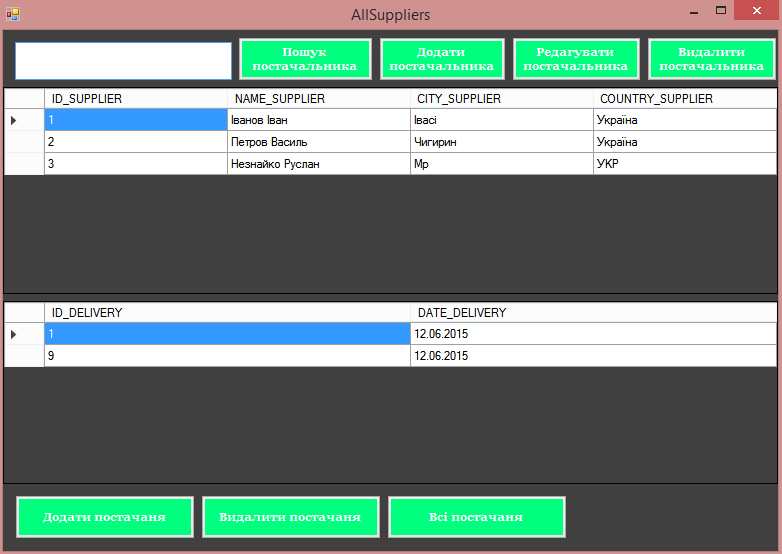


Рис. 18. Форма постачальники

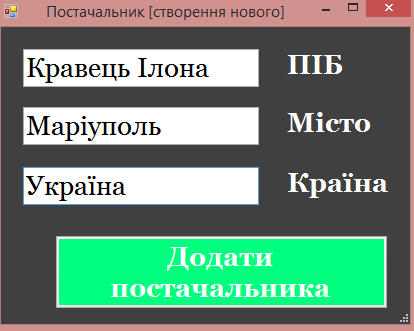


Рис. 18. Форма додання нового постачальника

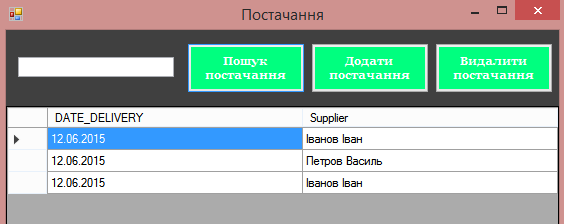


Рис. 18. Форма перегляду усіх постачальників

## ВИСНОВКИ

Завданням дипломної роботи було створення проектування та реалізація інформаційної системи авіабудівного підприємства. При виконанні завдання було досліджено предметну галузь та виявлено ті аспекти, які мають враховуватись в розробці бази даних. Було розроблено відповідно до вимог підприємства концептуальну та фізичну моделі. Також реалізовано прикладну програму для роботи з базою даних. Як сервер баз даних було використано Microsoft SQL Sever.

При створенні моделі бази даних найважливішим завданням є відображення в базі даних правил бізнес-логіки конкретного підприємства, а при реалізації важливо враховувати особливості конкретної СУБД.

Результатом виконання дипломної роботи є база даних та прикладна програма, що працює з цією базою даних, прикладна програма написана на мові програмування C#. Вона дозволяє виконувати необхідні для підприємства завдання. Цей програмний продукт має широке поле для удосконалення і введення нових ідей та розширення його функціональності.

Якщо оцінювати роботу в цілому, то реалізована програма дозволяє виконати основні задачі, необхідні для нормального функціонування.

## СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Використання об’єктів, які використовуються при роботі з SQL: [Електронний ресурс]: <http://msdn.microsoft.com/en-us/library/system.data.sqlclient.aspx>. Перевірено: 7.11.2015.
2. Архітектура програмного забезпечення [Електронний ресурс]. – Режим доступу: [http://uk.wikipedia.org/wiki/](http://uk.wikipedia.org/wiki/%D0%90%D1%80%D1%85%D1%96%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F)Архітектура програмного забезпечення. Перевірено: 7.07.2015.
3. Проектування баз даних [Електронний ресурс]. – Режим доступу: <http://pidruchniki.com/11570718/bankivska_sprava/proektuvannya_baz_danih>. Перевірено: 7.07.2015.
4. Детальное проектирование [Електронний ресурс]. – Режим доступу: <http://project.dovidnyk.info/index.php/programnye-roekty/tehnologiyarazrabotkiprogrammnogoobespecheniya/24-detal_noe_proektirovanie>. Перевірено: 7.07.2015.
5. Дейт. К. Дж., Введение в системы баз данных, 7-е издание.: Пер. с. Англ. – М.: Издательский дом «Вильяме», 2001. – 1072 с.: ил.- Парал. тит. Англ.
6. Герасиа-Молина Гектор, Джеффри Д. Ульман, Уидом Дженнифер, Системы баз данных. Полный курс: Пер. с англ. – М.: Издательский дом «Вильямс», 2003. – 1088 с.: ил. – Парал. тит. англ.
7. Кузнецов С. Д., «Основы баз данных»,– М.: Москва, 2007. – 250 с.
8. Крёнке Д. «Теория и практика построения баз данных.» 9-е изд. – СПб.: Питер, 2005. – 370 с.
9. Вендров А.М. Case-технологии. Современные методы и средства проектирования ИС, 1998. – 340 с.
10. Авраменко В.С. Проектування інформаційних систем: навчальний посібник / В.С. Авраменко, С.В. Голуб, В.І. Салапатов. – Черкаси: Черкаський національний університет ім. Б. Хмельницького, 2015. – 501 с.

## Додаток А

**ER-модель бази даних**



## Додаток Б

## Логічна модель бази даних



## Додаток В

## Фізична модель бази даних

****