Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра програмного забезпечення



**Звіт**

**Про виконання проекту**

**з дисципліни «Основи командної розробки програмного забезпечення»**

**Виконав:**

студ. групи ПІ -35

Черешня Д.С.

**Прийняв:**

Квятковський Б.Л.

«\_\_\_» \_\_\_\_\_\_ 2015 р.

∑ = \_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_

Львів – 2017

Зміст

[1. Вступ 3](#_Toc500613101)

[2. Призначення, мета 3](#_Toc500613102)

[3. Загальний опис 3](#_Toc500613103)

[3.1. Обґрунтування теми. 3](#_Toc500613104)

[3.2. Короткий аналіз існуючих продуктів 3](#_Toc500613105)

[3.3. Функції продукту 3](#_Toc500613106)

[3.4. Середовище функціонування 3](#_Toc500613107)

[4. Характеристика системи 3](#_Toc500613108)

[4.1. Початок роботи 3](#_Toc500613109)

[4.2. Робота програми. 4](#_Toc500613110)

[5. Розподіл задач 4](#_Toc500613111)

# Вступ

Даний звіт описує проект командної розробки та документацію програмного продукту.

# Призначення, мета

Найменування системи – «Логування дій у базі даних».

Призначення системи – забезпечити можливість роботи з базами даних. Передбачити автоматичне додавання до бази даних тригерів та таблиці логування, де будуть відображатись усі дії в базі даних.

# Загальний опис

## Обґрунтування теми.

При роботах з базами даних є важливою функція логування, тобто запису усіх внесених у базу змін із зазначенням часу та користувача, що їх вносив. Це дозволяє відстежувати як діяльність окремих користувачів так і події, що відбувались у базі даних. В результаті стає можливим формування різноманітних статистичних даних, а також полегшується відстеження можливих помилок та неточностей.

## Короткий аналіз існуючих продуктів

Фактично кожна СУБД має вбудовані системи відстеження змін, що вносяться у БД, але далеко не завжди можливість перевіряти ці логи є доступною користувачам бази. Зазвичай лише адміністратори системи мають доступ до подібної інформації.

## Функції продукту

Користувачам програмного продукту доступні такі основні функції:

* Підключення до БД
* Основні функції роботи з БД: переглянути, додати, змінити, видалити запис із БД
* Перегляд таблиці внесених змін

## Середовище функціонування

Програмний продукт являтиме з себе програму Windows та працюватиме з MS SQLServer. Тому для його роботи необхідно Microsoft .NET Framework 3.5, а також Microsoft SQL Server 2012 і вище для організації бази даних, а також роботи з системою керування базою.

# Характеристика системи

## Початок роботи

* + 1. Підключення до бази даних.
    2. Автоматичне створення тригерів та таблиці логів.

## Робота програми.

* + 1. Вивід даних з бази даних.
    2. Функції додавання, редагування та видалення записів БД.
    3. Відстеження виклику тригерів та запис в таблицю логів.

# Розподіл задач

|  |  |
| --- | --- |
| **Задача** | **Виконавець** |
| Проектування системи | DahakaCher  RoseThorn95 |
| Перевірка коректності роботи та тестування |
| Розподіл та планування задач |
| Розробка системи | DahakaCher |
| Розробка інтерфейсу користувача | RoseThorn95 |
| Оформлення документації |

# Хід роботи

## Створення віддаленого репозіторію

Для забезпечення коректної командної роботи було обрано систему контроля версій Git та створено репозиторій на сайті github.com.

Репозиторій доступний за посиланням: <https://github.com/DahakaCher/OKR_TRIGGER_CREATOR_LOG.git>

## Встановлення та налаштування Git

Для роботи з СКВ Git необхідно його завантажити з офіційного сайту та встановити. Після встановлення необхідно задати основні налаштування конфігурації.

* Вказати ім’я користувача командою

git config --global user.name "ваше ім’я"

* Вказати електрону пошту командою

git config --global user.email "ваша пошта"

* А також рекомендовано додатково конфігурувати функцію закінчення рядків для операційної системи Windows.

git config --global core.autocrlf true  
git config --global core.safecrlf true

## Строрення локального репозиторію та синхронізація

Для початку роботи необхідно визначити робоче місце. Для цього, у вибраній папці слід виконати команду git clone “адреса репозиторію”

Після цього у вибраній папці з’являться усі дані, що уже містяться на репозиторію і ви зможете працювати з ними.

У випадку якщо вам необхідні внесені оновлення до проекту, необхідно скористатись командами git fetch або git pull. Різниця між ними полягає в тому, що у першому випадку ви отримаєте копію відсутніх у вас файлів, а в другому відбудеться злиття вашої гілки з гілки, з якої ви затягуєте дані.

## Проектування системи

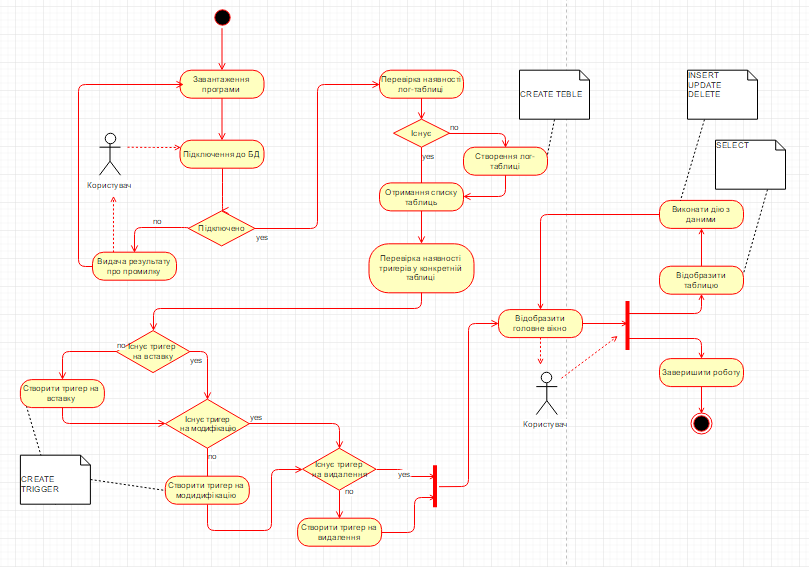
Розробка основної логіки роботи програми була представлена схематично у вигляді діаграми діяльності.

Рис. 1: Діаграма діяльності

Одним з ключових пунктів проектування було визначення структури таблиці логування. Вона мусить містити такі поля:

* індекс;
* виконана дія;
* таблиця, в якій виконували дію;
* ідентифікатор рядку, що опрацювали;
* дата та час виконання;
* ім’я користувача, що виконав дію.

Таким чином логіка роботи тригерів була визначена так: після виконання операції додати запис до таблиці логування.

## Проектування інтерфейсу

Було визначено, що інтерфейс має бути максимально простим і зрозумілим користувачеві. Вибір було зроблено на графічно інтерфейсі за допомогою Windows Form. Як основні було встановлено такі критерії:

* дані з бази даних мають виводитись таблично у DataGridView;
* для вибору таблиці забезпечити випадаючий список ComboBox;
* основні функції реалізувати за допомогою кнопок Button;
* створити окрему форму для підключення до БД;
* створити окрему форму для редагування та додавання запису (поелементної роботи).

Додаткові критерії інтерфейсу:

* створення головного меню;
* рядок стану, в якому вказані дані щодо БД до якої здійснене підключення.

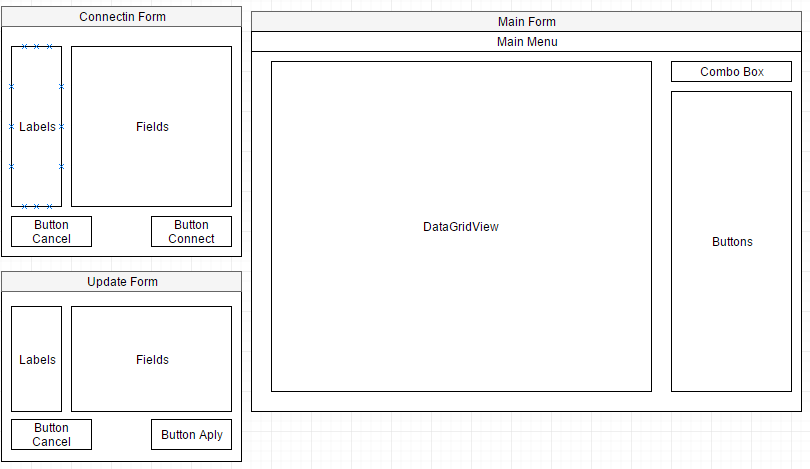


Рис. 2. Схематичний проект інтерфейсу

## Розробка прототипу

Першим основним робочим етапом була розробка базового прототипу. Наявність примітивної основи дозволить усім членам команди в подальшому вносити все більш складні зміні. Таким чином, було створено прототип, що складався із базового інтерфейсу та функціями відкриття інших форм.

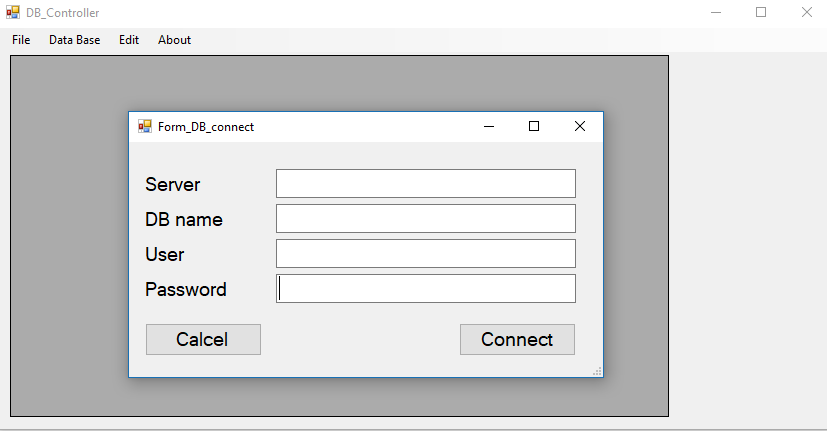


Рис. 3. Вікно прототипу

Створений прототип був доданий на головну гілку репозиторію як наступна версія програми командою git commit –m “Created prototype”.

## Подальші етапи та версії

Продовжуючи роботу поступово проект розвивався. З кожним новим комітом додавався якийсь функціонал, додавались нові функції чи виправлялись помилки.

Першою працюючою версією програми була версія 5-го коміту.

## Розгалуження на модифікацію

На певному моменті роботи, прийшло переосмислення проекту і було вирішено внести зміни в структуру системи, а саме розширити інформацію, що зберігається в логуючій таблиці. Подальша робота була виведена у окрему гілку, аби не внести серйозних помилок у проект в цілому.

За допомогою команди git brunch Upgr\_log\_sys була створена нова гілка розробки, а за допомогою команди git checkout Upgr\_log\_sys вона була встановлена як поточна робоча.

В подальшій роботі стало зрозуміло, що проект потребує глобальних змін в базовій структурі. Така робота потребує часу та ресурсів, якими наша команда не володіє, і спроба виконати таке завдання може призвести до порушень сроків виконання. Тому вирішили зафіксувати стан проекту на додатковій гілці, та зробити з неї відгалуження на ще одну підгілку. На цій підгілці буде виконана лише деяка незначна модифікація, що покращить проект, але не критично вплине на загальний робочий план.

Після того як було модифіковано та частково переписано систему логування та створення тригерів, робоча гілка була злита з основною за допомогою команд git checkout master (переключились на основну гілку в яку зливатимемо) git merge Only\_ID\_log.

Таким чином була отримана нова версія програми, що зафіксувала зміни вненсені у додатковому розгалужені.

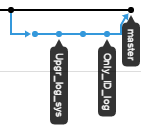


Рис. 4. Схема розгалуження