МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК
СЕКЦІЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ПРОЕКТУВАННЯ

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: «Програмний додаток обчислень параметрів гідравлічних теплових мереж»

за напрямом підготовки 122 «Комп'ютерні науки»

Виконавець роботи: студент групи ІТ-61-8 Ковальчук Владислав Андрійович

Кваліфікаційна робота бакалавра
захищена на засіданні ЕК
з оцінкою _____«__» _____2020 р.

Науковий керівник _____ к.т.н., доц., Неня В.Г.
(підпис) (науковий ступінь, вчене звання, прізвище та ініціали)

Голова комісії _____ Шифрін Д. М.
(підпис) (науковий ступінь, вчене звання, прізвище та ініціали)

Засвідчую, що у цій дипломній роботі немає
запозичень з праць інших авторів
без відповідних посилань.

Студент _____
(підпис)

Суми-2020

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук
Секція інформаційних технологій проектування
Спеціальність – 122 «Комп'ютерні науки»
Освітньо-професійна програма «Інформаційні технології проектування»

ЗАТВЕРДЖУЮ
Зав. секцією ІТП

_____ В. В. Шендрик
«__» _____ 2020 р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА СТУДЕНТУ

Ковальчук Владислав Андрійович

1 Тема роботи *Програмний додаток розподілених обчислень параметрів гідравлічних теплових мереж*

керівник роботи *Неня Віктор Григорович, к.т.н., доцент*,

затверджені наказом по університету від « _» травня 2020 р. № 0834-III

2 Строк подання студентом роботи « » травня 2020 р.

3 Вхідні дані до роботи технічне завдання на розробку програмного додатку

4 Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) аналіз предметної області, постановка задачі та методи дослідження, проектування програмного додатку, розробка програмного додатку розподілених обчислень

5 Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) постановка задачі, аналіз аналогів, моделювання програмного додатку, аналіз технологій, етапи розробки програмного додатку «Gidravlika», висновки

6. Консультанти розділів роботи:

Розділ	**Консультант**	**Підпис, дата**	
		Завдання видав	**Завдання прийняв**

7.Дата видачі завдання_____

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз предметної області	02.03.20 – 05.03.20	
2	Пошук аналогів та обгрунтування потреби	06.03.20 – 10.03.20	
3	Визначення вимог	11.03.20 – 11.03.20	
4	Визначення інструментарію	12.03.20 – 12.03.20	
5	Планування wbs та obs	13.03.20 – 13.03.20	
6	Складання календарного плану	16.03.20 – 17.03.20	
7	Визначення ризиків	18.03.20 – 19.03.20	
8	Розробка інтерфейсу користувача	20.03.20 – 26.03.20	
9	Проектування бази даних	27.03.20 – 03.04.20	
10	Розробка програмних модулів	06.04.20 – 30.04.20	
11	Компілювання програмного коду	01.05.20 – 11.05.20	
12	Тестування розробником	12.05.20 – 14.05.20	
13	Тестування незалежною особою	15.05.20 – 19.05.20	
14	Оформлення документації	11.03.20 – 21.05.20	
15	Архівація проекту	22.05.20 – 25.05.20	
16	Введення в експлуатацію	26.05.20 – 31.05.20	

Студент _____ Ковальчук В.А.
(підпис)

Керівник роботи _____ к.т.н., доц. Неня В.Г.
(підпис)

РЕФЕРАТ

Тема дипломної роботи «програмний додаток розподілених обчислень параметрів гідравлічних теплових мереж».

Дипломна робота складається зі вступу, трьох розділів, висновка, списку використаної літератури та додатків.

Пояснювальна записка містить 84 с., 53 рис., 5 табл., 4 додатки, 18 джерел.

У першому розділі досліджується актуальність проблеми, виконується пошук та аналіз існуючих аналогів, формується мета дипломної роботи та задачі проекту, виконується аналіз технологій для реалізації проекту.

Другий розділ присвячений проектуванню програмного додатку, де наведені діаграми у нотації IDEF0, Use Case, ER-діаграму бази даних та математичну модель.

В останньому розділі виконується детальний опис практичної реалізації програмного додатку: створення прототипу діалогового вікна, розробка бази даних, розробка головного функціоналу та допоміжних модулів.

Результатом виконаної роботи є розроблений програмний додаток, який виконує розподілені обчислення гідравлічних параметрів тепломереж та збурігає отримані дані у базі даних та у вигляді звіту за бажанням користувача.

Ключові слова: PYTHON, ПРОГРАМНИЙ ДОДАТОК, ГІДРАВЛІЧНІ ПАРАМЕТРИ, ТЕПЛОМЕРЕЖА

ЗМІСТ

ВСТУП

Теплові мережі є основою сучасної системи централізованого теплопостачання у містах, що допомагають транспортувати енергію тепла від джерел до споживачів. Якісне опалення вже давно стало нагальною потребою населення. Для споживача найбільш важливими показниками ефективності функціонування тепломережі є оптимальна температура, вартість постачання тепла, а також безперервна його подача. У свою чергу, організації постачання теплоенергії звертають увагу на рентабельність та кількість аварій систем теплопостачання[1].

Актуальність розвитку систем централізованого теплопостачання, а також підвищення ефективності їх використання, набирає все більших обертів у зв'язку з проблемами екології та енергозберігання [2].

Сучасні проблеми потребують сучасних рішень із застосуванням комп'ютерних інформаційних систем, що використовуються для управління та контролю. Реалії сьогодення роблять необхідним збір, структуризацію та аналіз наявної інформації про структуру мереж теплопостачання в одній інформаційній системі[1]. Упровадження таких систем підвищує ефективність управління роботою тепломереж завдяки автоматизації збору інформації[3].

Зазвичай для визначення ефективності роботи тепломереж відповідно до нормативних документів використовується розрахунок оптимальних режимів роботи тепломереж. Такі розрахунки часто застосовуються при стратегічному плануванні. На практиці при експлуатації систем тепломереж всі необхідні розрахунки виконуються спеціальним програмним забезпеченням [2].

Метою даної роботи – є створення програмного продукту для розподілених обчислень параметрів гідравлічних теплових мереж.

Для досягнення поставленої мети необхідно виконати відповідні задачі:

- Проаналізувати предметну область;
- Провести пошук аналогів та визначити необхідні функції програмного додатку;

- Розробити інтерфейс програмного додатку;
- Розробити функціонал базових операцій;
- Провести тестування розробником програмного додатку.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Актуальність проблеми

За останні роки було проведено безліч досліджень у цілому галузі теплопостачання, прийнято програми щодо ефективного і раціонального використання енергоресурсів. Нажаль не було досягнуто суттєвих успіхів. За останніми показниками більш ніж 10 000 котелень в Україні мають знос понад 80% [4].

Протяжність тепломереж в Україні, відповідно до двотрубного обчислення, становить 35834,2 км, з яких в аварійному стані – 5620,7 км, а це близько 15,7% від загальної протяжності тепломереж. Так до п’ятірки областей з найбільш аварійним станом тепломереж відносять місто Київ, Донецьку, Одеську, Сумську та Львівську області. Зазвичай зношеність та поганий технічний стан тепломереж призводить до значних теплових витрат [5].

Вдосконалення технологічного стану теплоенергостанцій є найбільш важливою проблемою для подолання з метою підвищення надійності, економічності та екологічності виробництва теплоенергії [6]. Оскільки неможливо водночас покращити технологічних стан тепломережі, постає питання раціонального та щонайефективнішого використання ресурсів для подолання наявної проблеми.

Впровадження комп’ютерних інформаційних систем несе в собі також економічну ефективність. Як правило економічний ефект від застосування відповідних програмних продуктів має неявний характер, хоча й може бути дуже значним. Згідно з досвідом підприємств інженерних комунікацій економічний ефект ховається в багатьох підсистемах:

1. Основою окупної інформаційної системи служать перевірені дані про предмет експлуатації. Тобто в першу чергу проводиться паспортизація устаткування, тепломережі та абонентів зі збереженням отриманих даних до бази даних.

2. Автоматизоване ведення журналів диспетчерів. Співробітники можуть приймати, відстежувати різнопланові та оперативні заявки в автоматизованому режимі, якщо мають достовірну базу даних паспортизації та зручне графічне представлення мережі на екрані.
3. Автоматизоване ведення журналів диспетчерів передбачає реєстрацію всіх аварійних ситуацій і виконуваних робіт в базу даних. Таким чином можливо отримати статистику пошкоджуваності, аналіз якої дає змогу з високою вірогідністю прогнозувати можливі аварійні ситуації, а також їх причини.
4. Базуючись на достовірних даних, інформаційна система вирішує задачі моделювання мережі та режимів її роботи. Можна провести гідравлічні розрахунки, що пов’язані з моделюванням перемикань, для вибору оптимального режиму [7].

У зв’язку з описаними проблемами виникає необхідність створити програмний додаток з можливістю обчислення гідравлічних параметрів тепломереж, беручи до уваги досвід підприємств з використання схожих додатків.

Розроблений програмний додаток повинен бути розроблений в адаптивній формі, зрозумілій для диспетчерів, що обслуговують теплові мережі.

Даний проект буде мати попит серед ТЄС, а також інших компаній обслуговування інженерних комунікацій у сфері енергопостачання.

1.2 Аналіз існуючих аналогів

Для розробки функціоналу програмного продукту, що буде відповідати поставленим вимогам, потрібно провести аналіз аналогів: різноманітні програмні продукти для розрахунку гідравлічних мереж.

Перший аналог – «Korf Hydraulics». Даний програмний продукт застосовується для розрахунків витрат і тиску в трубах трубопровідних мережах.

Програмний продукт має гнучкий інтерфейс, а також дозволяє вирішувати різні задачі з визначення величин невідомих витрат, швидкостей, тисків, а також діаметрів труб гідравлічних мереж [8]. «Korf Hydraulics» може застосовуватись для розрахунків як одного трубопроводу, так і складних гідравлічних мереж, де невідома швидкість потоку у всіх відгалуженнях. На рис.1.1-1.2 представлено головне вікно програми з побудованою мережею і діалогове вікно для введення даних відповідно.

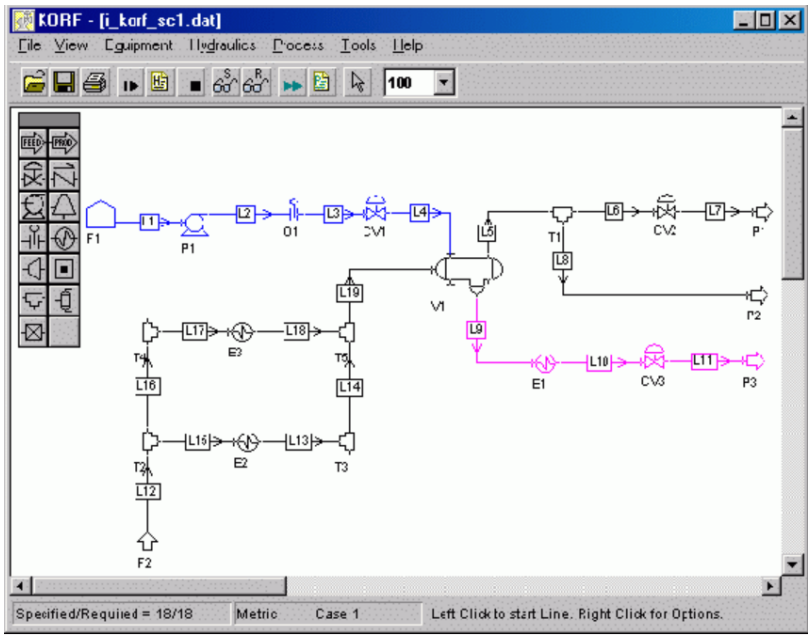


Рисунок 1.1 – Головне вікно програми

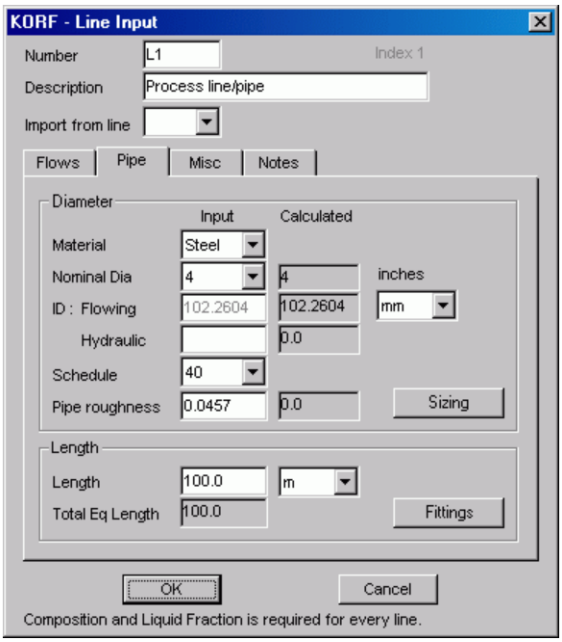


Рисунок 1.2 – Діалогове вікно для введення даних

Даний програмний продукт надає користувачу можливість задання необхідної шорсткості труб, використання різних видів рідин, розрахунку теплових витрат та ін. Результати розрахунків представляються в діалоговому вікні і рисунку та можуть бути збережені у файл звіту.

Наступний аналог – програмний продукт для проектування систем внутрішнього водопроводу та каналізації будівель «Умная вода». Даний програмний подукт розроблено для виконання всіх необхідних розрахунків водопостачання, каналізації, систем пожежогасіння та виконання підбору обладнання відповідно до нормативно-технічної документації. На рис.1.3 зображено розрахунок споживання води за допомогою програмного продукту «Умная вода».

«Умная вода» використовується для розрахунку різних внутрішніх систем, як протипожежний водопровід або дождьова каналізація, виконує підбір необхідного устаткування та матеріалів, має можливість збереження звітів у різних форматах, та інше [9].

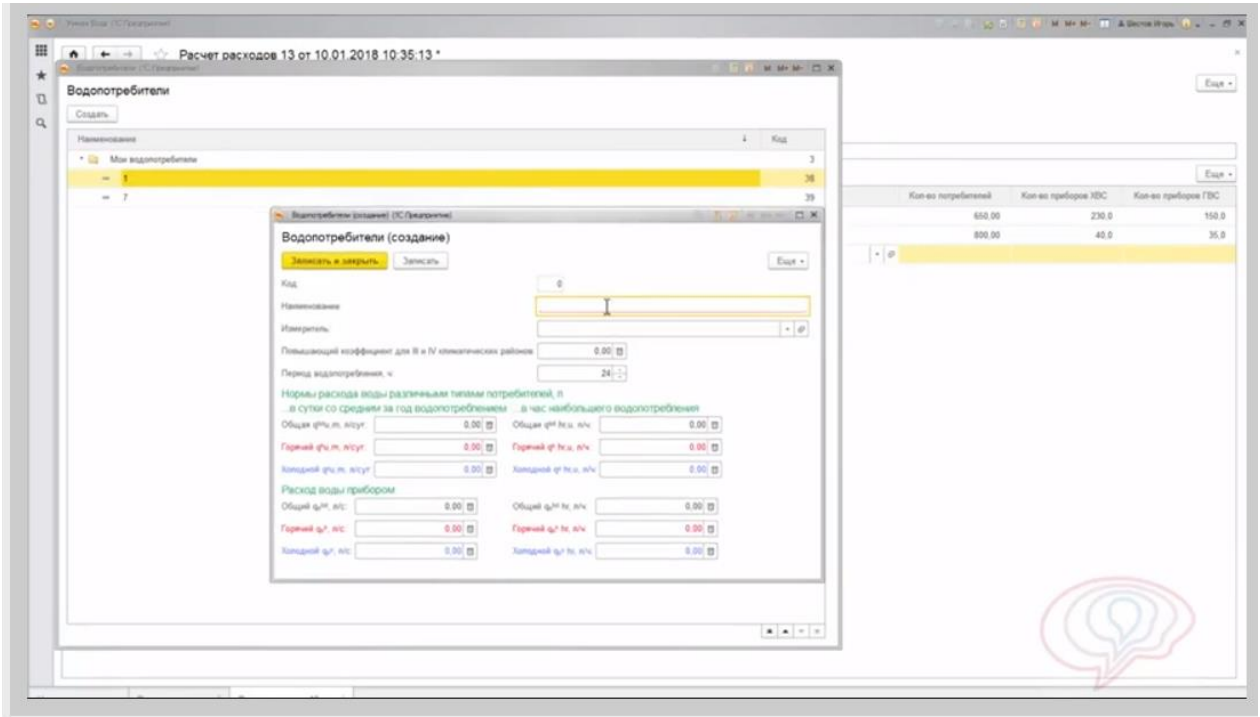


Рисунок 1.3 – Приклад роботи програмного продукту «Умная вода»

Останній аналог – група програмних продуктів з інженерної сантехніки «VALTEC». Перший програмний продукт – «VALTEC.PRG» що дає можливість користувачу розрахувати водяне, радіаторне, підлогове і настінне опалення, визначити теплонеобхідність приміщень, необхідні витрати води та інше. Крім того, в розпорядженні користувача – зручно скомпонована добірка довідкових матеріалів. На рис.1.4 представлено приклад теплотехнічного розрахунку.

«VALTEC C.O.» – наступний програмний продукт системи «VALTEC», що представляє собою розрахунково-графічний додаток для проектування систем радіаторного і підлогового опалення. Додаток дозволяє конструювати і регулювати системи опалення, проводити повний комплекс гідравлічних і теплових розрахунків.

«VALTEC H2O» – останній програмний продукт системи «VALTEC», що використовується для проектування систем холодного і гарячого водопостачання. Даний додаток надає можливість провести повний розрахунок і конструювання гідравлічно збалансованої системи водопостачання [10].

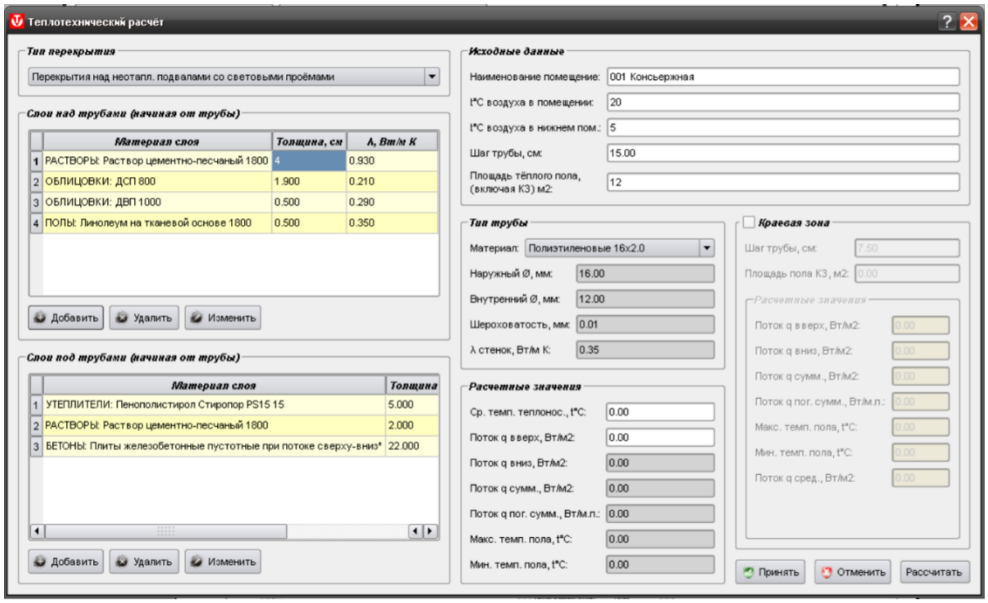


Рисунок 1.4 – Теплотехнчний рознахунок за допомогою програмного продукту «VALTEC.PRG»

Після проведення пошуку аналогів програмного додатку у сфері тепло- та водопостачання, було проведено порівняльний аналіз функцій та характеристик

кожного аналогу з розробленим програмним додатком «Gidravlika», що представлено в таблиці 1.1

Таблиця 1.1 – Порівняльна характеристика аналогів

Назва критерію	Назва ресурсу			
	«Gidravlika»	«Korf Hydraulics»	«Умная вода»	«VALTEC»
Визначення витрат тису	+	+	+	+
Можливість розрахунку тепломереж для різних режимів	+	+	+	-
Можливість збереження результатів у вигляді звітів	+	+	+	+
Наявність додаткового інструментарію для полегшення обчислень	+	+	-	+

1.3. Мета та задачі програмного додатку «Gidravlika»

Метою даного дипломного проекту є створення програмного додатку «Gidravlika» для виконання розподілених розрахунків параметрів тепломереж. Даний продукт матиме попит серед диспетчерів та інших співробітників, що обслуговують тепломережі та теплопостачання. Користувачу також буде надана можливість зберігати отримані результати та, ґрунтуючись на статистичних даних, робити висновки щодо планування і стану мережі.

Розроблений програмний додаток повинен підтримувати такі функції експлуатації, як:

- визначення діаметрів теплопроводів;
- розрахунок тиску в різних точках тепломережі;
- визначення витрат тиску на різних ділянках;
- наявність допоміжного інструментарію для обчислень;
- можливість збереження результату у вигляді звіту в форматі .xlsx;

Створений програмний додаток має бути інтуїтивно зрозумілим для користувача, містити зручний інтерфейс та навігацію.

Для досягнення поставленої мети дипломного проекту необхідно вирішити відповідний перелік задач:

- провести аналіз предметної області;
- визначити інструменти для програмування і відповідно технологію для створення аналогічних програмних додатків;
- розробити математичну модель;
- розробити прототип інтерфейсу програмного додатку;
- розробити функціональні модулі програмного додатку;
- провести тестування програмного додатку.

Для коректної та ефективної роботи програмного додатку необхідною умовою є створення та ухвалення технічного завдання, що представлено у Додатку А.

1.4. Вибір засобів реалізації

Для розробки десктопних програмних продуктів спеціалісти застосовують різні мови програмування в залежності від поставленої задачі та функцій, які повинні бути реалізовані в програмному продукті. Якщо мова йде про необхідність виконання складних математичних обчислень або застосування машинного навчання, то при таких умовах краще використовувати мову програмування Python.

Python – високорівнева мова програмування загального призначення. Незважаючи на те, що синтаксис досить мінімалістичний, бібліотеки Python включають в себе великий об'єм корисних функцій, тому використовується як основна мова програмування або для створення різних надбудов та інтеграції додатків, найчастіше з метою обробки даних та наукових задач [11].

Для створення графічного інтерфейсу під операційну систему Windows при використання мови програмування Python 3, краще за все застосовувати Qt GUI фреймворк, так як це динамічна мова, що добре підходить для швидкого прототипування. PyQt – це бібліотека, що надає можливості використовувати фреймворк Qt GUI із Python. Сам Qt було розроблено за допомогою мови програмування C++, та використовуючи його саме з Python, створбвати програмні додатки стає швидше, не жертвуючи при цьому швидкістю C++.Qt GUI створює файли на освнові XML спеціального формату .ui, в якому всі віджети зберігаються у вигляді дерева [12].

При роботі в великою кількістю даних для майбутнього аналізу потрібно зберігати отримані результати, для чого краще за все підходять бази даних. Найбільш популярною системою управління баз даних в світі є MySQL. Дана СУБД володіє такими характеристиками, як надійність, перевірена продуктивність і простота використання. Як стандартизований спосіб вилучення та керування даними, було використано мову програмування SQL [13].

2 ПРОЕКТУВАННЯ ПРОГРАМНОГО ДОДАТКУ ДЛЯ ВИКОНАННЯ РОЗПОДІЛЕНИХ ОБЧИСЛЕНЬ

Після проведення детального аналізу предметної області, визначення актуальності даної роботи, детального вивчення та аналізу існуючих аналогів, визначення мети та задач дипломного проекту, а також вибору засобів реалізації, виконується наступний етап – проектування програмного додатку «Gidravlika». Необхідно розробити функціональні моделі за методологією sadt для демонстрації роботи програмного додатку, а також сформувати діаграму use case для демонстрації взаємодії акторів та варіантів використання програмного додатку. Процес планування проекту представлено в Додатку Б.

2.1Діаграми нотації IDEF0

Функціональна модель будь-якої предметної області за методологією SADT демонструє функціональну структуру об’єкта, його дії та взаємозв’язки. Для побудови структурно-функціональної моделі було застосовано програмний продукт Allfusion Process Modeler R7. Контекстну діаграму процесу роботи програмного додатку «Gidravlika» представлено на рис 2.1.

Контекстна діаграма нульового рівня складається з наступних елементів:

- Вхідні дані;
- Вихідні дані;
- Керуюча інформація;
- Механізм.

Вхідними даними до функції «виконання розподілених розрахунків параметрів тепломереж» є:

- Технічні характеристики мережі;

– Варіанти вхідних даних .

На виході ми отримуємо обраховані гідравлічні параметри.

Виконання розподілених розрахунків параметрів тепломереж керується вимогами користувача, математичною моделлю, санітарними нормами для тепломереж, а також документацією phpmyadmin.

Механізмами є інтернет, база даних та програмний додаток.

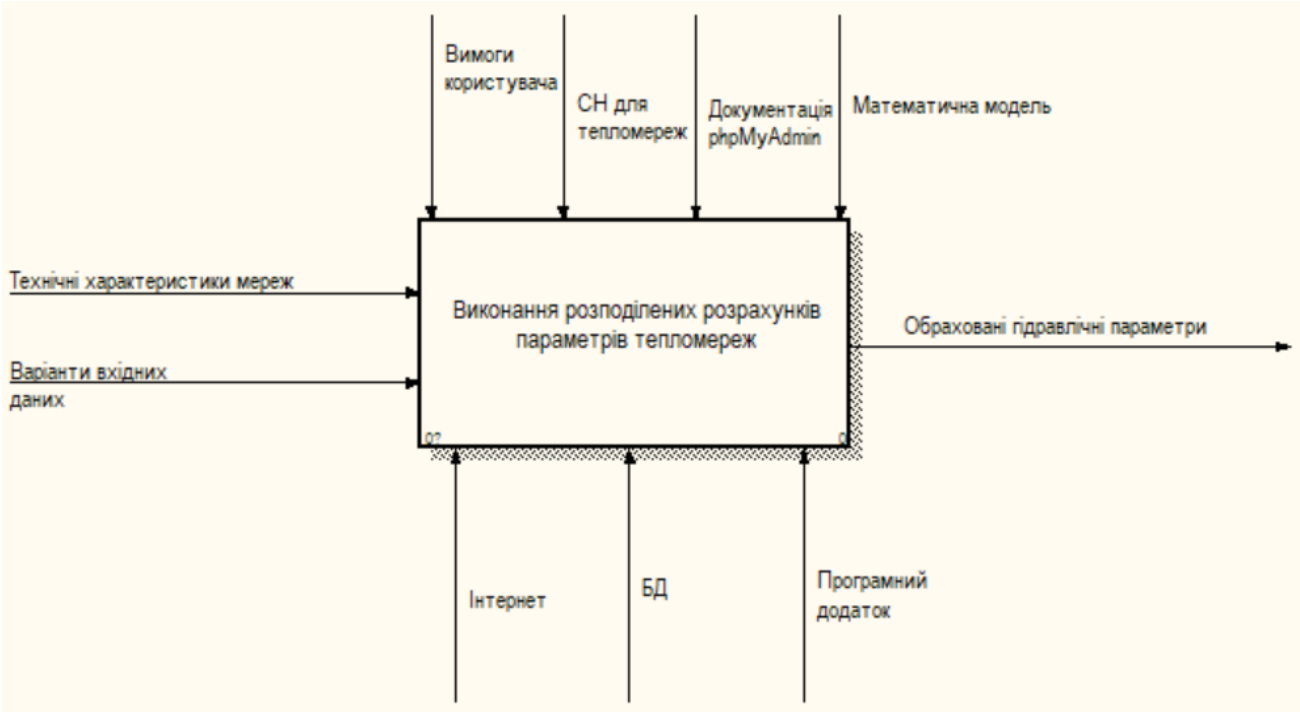


Рисунок 2.1 – Контекстна діаграма

Контекстна діаграма надає загальний опис проекту, тому існує необхідність її декомпозиції для більш детального опису роботи програмного додатку та ознайомлення з послідовністю виконання складових етапів для отримання кінцевого результату.

Контекстна діаграма декомпозується на 2 рівні. Перший рівень складається з 4 блоків, що представлено на рис. 2.2:

– Визначення основних гідравлічних показників;

– Вибір варіанту виконування розрахунків;

– Виконання розрахунку гідравлічних параметрів;

– Взаємодія з базою даних.

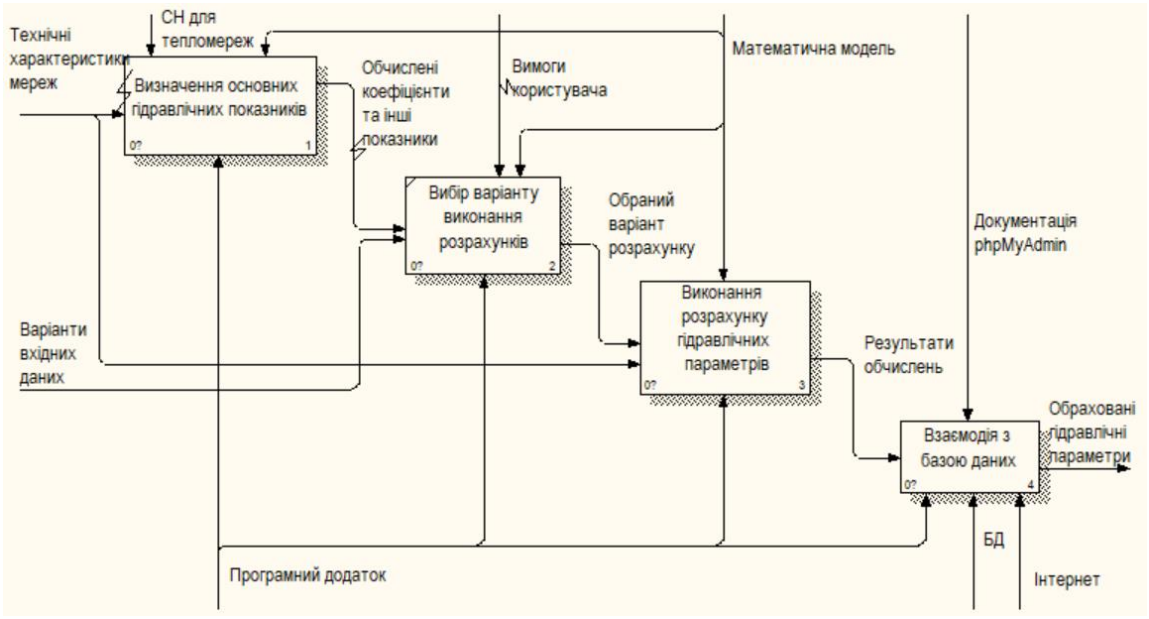


Рисунок 2.2 – Декомпозиція діаграми А–0

Вхідними даними до діяльності «Визначення основних гідравлічних показників» є «Технічні характеристики мереж»; вихідними – «Обчислені коефіцієнти та інші показники»; стрілками контролю – «СН для тепломереж» та «Математична модель»; механізми – «Орограмний додаток».

Вхідними даними до діяльності «вибір варіанту виконання розрахунків» є «Обчислені коефіцієнти та інші показники» та «Варіанти вхідних даних»; вихідними – «Обраний варіант розрахунку»; стрілками контролю – «Вимоги користувача» та «Математична модель»; механізми – «Програмний додаток».

Вхідними даними до діяльності «Виконання розрахунку гідравлічних параметрів» є «Обраний варіант розрахунку» та «Технічні характеристики мереж»; вихідними – «Результати обчислень»; стрілкою контролю – «Математична модель»; механізми – «Програмний додаток».

Вхідними даними до діяльності «Взаємодія з базою даних» є «Результати обчислень»; вихідними – «Обраховані гідравлічні параметри»; стрілками контролю – «Документація phpmyadmin»; механізми – «Програмний додаток», «БД» та «Інтернет».

Діаграму декомпозиції 2-го рівня процесу «Визначення основних гідравлічних показників» представлено на рис.2.3.

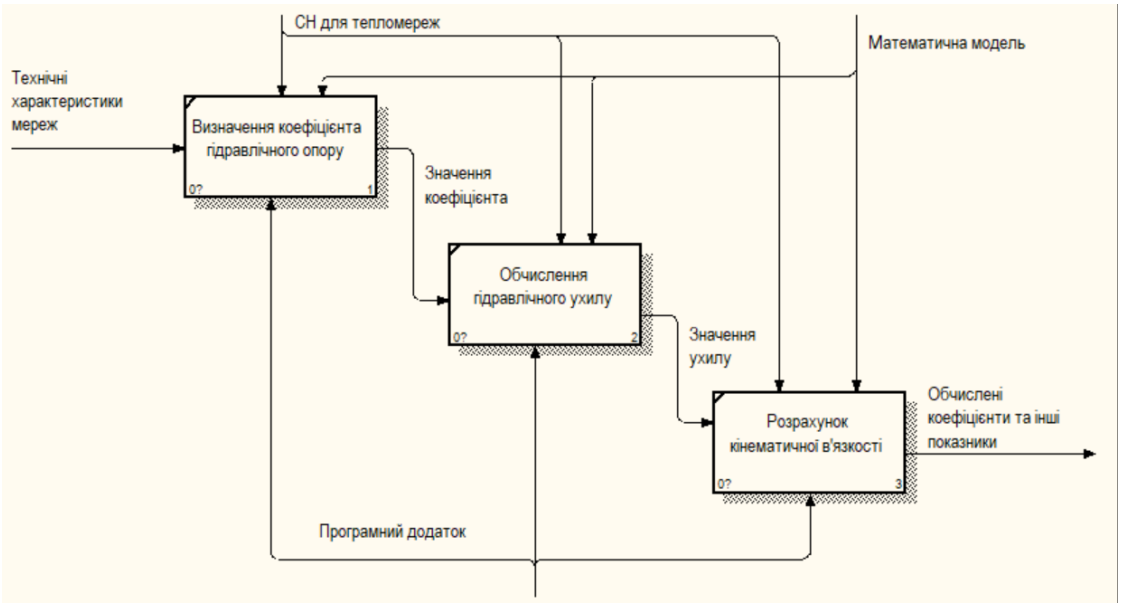


Рисунок 2.3 – Декомпозиція етапу «Визначення основних гідравлічних показників»

Вхідними даними до діяльності «Визначення коефіцієнту гідравлічного опору» є «Технічні характеристики мереж»; вихідними – «Значення коефіцієнта»; стрілками контролю – «СН для тепломереж» та «Математична модель»; механізми – «Програмний додаток».

Вхідними даними до діяльності «Обчислення гідравлічного ухилу» є «Значення коефіцієнту»; вихідними – «Значення ухилу»; стрілками контролю – «СН для тепломереж» та «Математична модель»; механізми – «Програмний додаток».

Вхідними даними до діяльності «Розрахунок кінематичної в’язкості» є «Значення ухилу»; вихідними – «Обчислені коефіцієнти та інші показники»; стрілками контролю – «СН для тепломереж» та «Математична модель»; механізми – «Програмний додаток».

На рисунку 2.4 зображено діаграму декомпозиції процесу «Виконання розрахунку гідравлічних параметрів».

Вхідними даними до діяльності «Визначення напору води H2» є «Обраний варіант розрахунку» та «Технічні характеристики мереж»; вихідними – «H2»; стрілками контролю – «Патематична модель»; механізми – «Програмний додаток».

Вхідними даними до діяльності «Визначення об’єму води Q» є «H2» та «Технічні характеристики мереж»; вихідними – «Q»; стрілками контролю – «Математична модель»; механізми – «Програмний додаток».

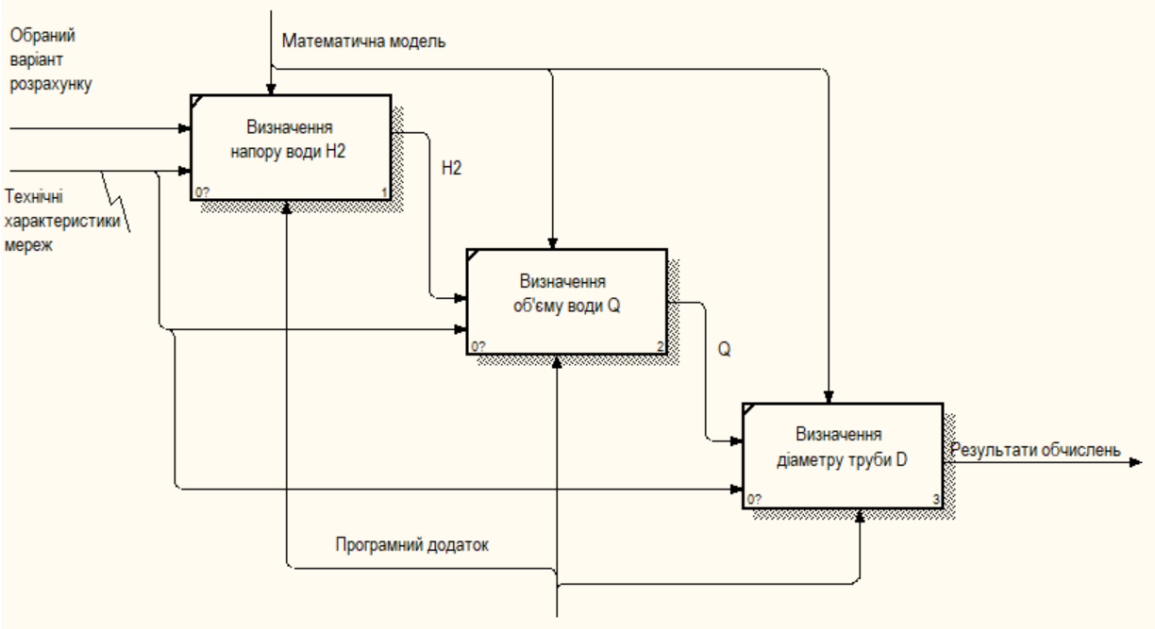


Рисунок 2.4 – Декомпозиція етапу «виконання розрахунку гідравлічних параметрів»

Вхідними даними до діяльності «Визначення діаметру труби D» є «Q» та «Технічні характеристики мереж»; вихідними – «Результати обчислень»; стрілками контролю – «Математична модель»; механізми – «Програмний додаток».

На рисунку 2.5 представлено діаграму декомпозиції процесу «Взаємодія з базою даних».

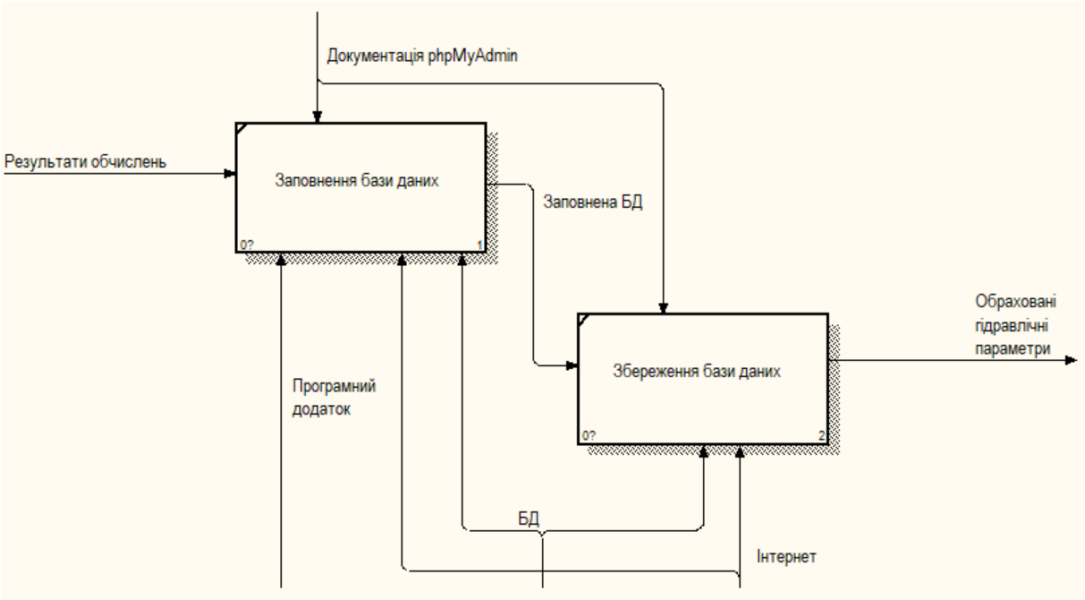


Рисунок 2.5 – Декомпозиція етапу «взаємодія з базою даних»

Вхідними даними до діяльності «заповнення бази даних» є «результати обчислень»; вихідними – «заповнена бд»; стрілками контролю – «документація phpmyadmin»; механізми – «програмний додаток», «бд» та «інтернет».

Вхідними даними до діяльності «збереження бази даних» є «заповнена бд»; вихідними – «обраховані гідравлічні параметри»; стрілками контролю – «документація phpmyadmin»; механізми – «бд» та «інтернет».

2.2Use Cases діаграма

З метою кращого розуміння взаємодії компонентів системи застосовують опис функціональності її складових за допомогою діаграми варіантів використання (use cases diagram). Дана діаграма описує функції, які повинна робити система, відповідно до визначених вимог її поведінки.

Для розробки діаграми варіантів використання програмного додатку були визначені наступні актори:

– User – користувач, який має на меті виконати розподілені розрахунки тпеломережі;

– Ms word – шаблон microsoft office word для запису звіту;
– Ms excel – файл програми microsoft office excel, до якого заносяться результати розрахунків.

Окрім переліку акторів також формується відповідний перелік варіантів використання:

– Converter – конвертер фізичних величин, додатковий інструментарій;
– Calculator – звичайний калькулятор, додатковий інструментарій;
– Pressure – визначення витрат тиску;
– Dew point – визначення температури точки роси;
– Hidraulic calculation – обчислення гідравлічних параметрів;
– Save – збереження отриманого результату;

Відповідно до кожного актора було сформовано перелік сценаріїв взаємодії з програмним додатком. Всі зазначені варіанти використання відповідають технічному завданню (Додаток А).

На рис. 2.5 представлено діаграма варіантів використання для програмного додатку «Gidravlika».

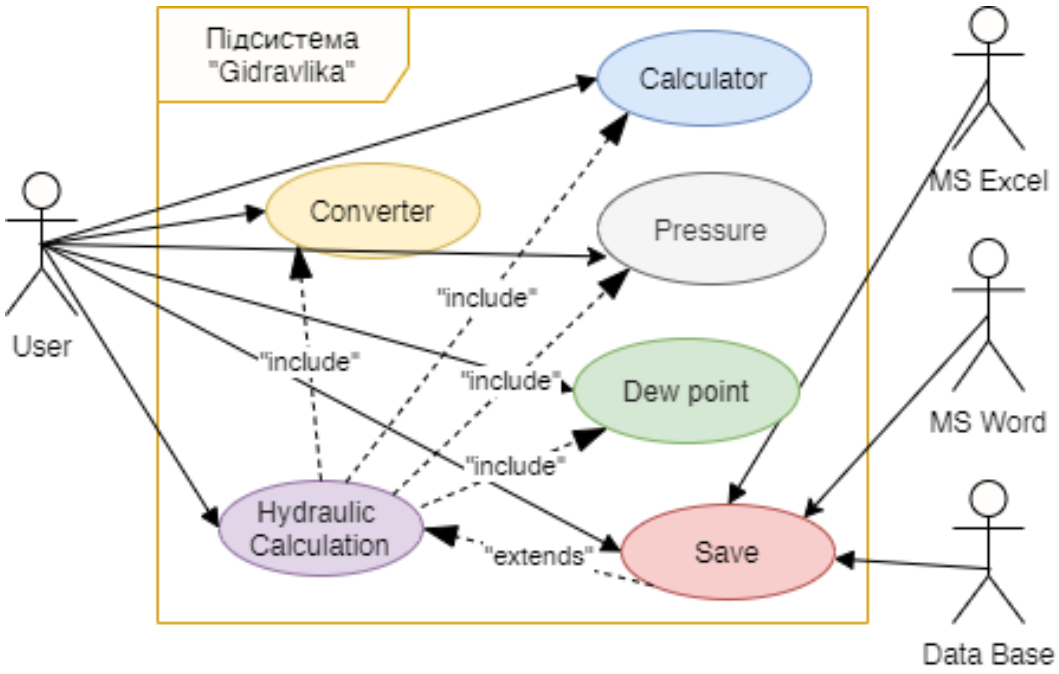


Рисунок 2.5 – Діаграма варіантів використання

2.3Проектування бази даних

Для проектування бази даних зазвичай використовуються модель «сутність-зв’язок», що описує концептуальні схеми, а також визначає дані і відносини між ними. У процесі розробки такої моделі отримуємо перелік сутностей предметної області, їх атрибути, а також взаємозв’язки між ними.

Провівши аналіз поставлених задач, біло визначено функції, що повинна виконувати система, що моделюється, а саме зберігання вхідних даних та результатів виконуваних обрахунків. Тож отримуємо перелік сутностей :

- Технічні характеристики (input) – явний кандидат на сутність;
- Результати розрахунків (result) – явний кандидат на сутність.

Одразу виникає очевидний зв’язок між визначеними сутностями, а саме «один результат можна отримати з одного набору вхідних даних». З іншої сторони цей зв’язок можна трактувати як «один набір даних використовується для отримання одного результату».

Далі розглянемо властивості сутностей (можливі кандидати в атрибути):

Технічні характеристики

Технічні характеристики, або вхідні дані, мають такі атрибути: довжина трубопроводу, абсолютна шорсткість, температура води та коефіцієнт місцевих опорів.

Результати розрахунку

Результати мають наступні атрибути: напір на початку та в кінці труби, вхідні дані, коефіцієнт гідравлічного опору, гідравлічний уклон, кінематична в’язкість, об’єм води, швидкість води, діаметр трубопроводу.

На рисунку 2.6 представлено концептуальну модель бази даних.

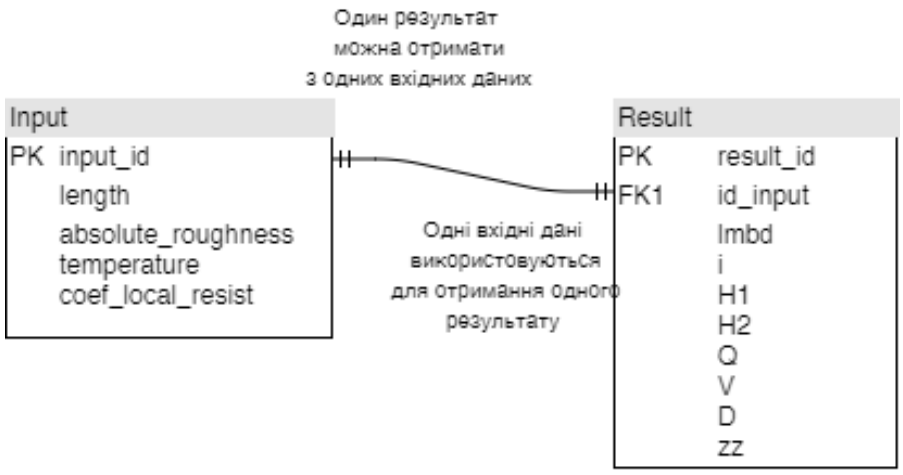


Рисунок 2.6 – Концептуальна модель бази даних

2.4Математична модель

До задач гідравлічного розрахунку належать визначення діаметру теплопроводу, тиску та його втрат в різних точках теплопроводу.

Витрати тиску на ділянках теплопроводу складаються з втрат тиску на тертя (лінійні витрати) і втрати тиску в місцевих опорах:

$$\Delta P = \Delta P_{\text{л}} + \Delta P_{\text{м}} \tag{2.1}$$

Де $\Delta P_{\text{л}}$ – витрати тиску на тертя, па; $\Delta P_{\text{м}}$ – втрати тиску в місцевих опорах, Па.

Витрати тиску на тертя (лінійні втрати) визначаються за наступною формулою:

$$\Delta P_{\text{л}} = \frac{\lambda}{d}\frac{v^2}{2}\rho \cdot l \tag{2.2}$$

Де λ – коефіцієнт гідравлічного опору; l – довжина ділянки трубопроводу, м; d – внутрішній діаметр ділянки трубопроводу, м; υ - швидкість руху теплоносія, м/с; ρ - щільність теплоносія, кг/м3;

Для корретного визначення питомих витрат тиску на тертя необхідним є використання техніко-економічних розрахунків. У даному дипломному проекті перепад тиску в тепловій мережі не заданий, тому пропонується питомі витрати тиску на тертя приймати згідно з санітарними нормами в межах 30-80 па/м.

При заданому перепаді втрати тиску в місцевих опорах визначаються за формулою:

$$\Delta P_{\text{м}} = \xi \frac{v^2}{2} \rho \tag{2.3}$$

Де ξ - коефіцієнт місцевих опорів; υ - швидкість руху теплоносія, м/с; ρ - щільність теплоносія, кг/м3;

Коефіцієнт місцевих опорів ξ представляє собою відношення втрати напору да даному місцевому опорі до швидкісного напору. Для розрахунків використовуються зазвичай вже наявні коефіцієнти відповідно до типу ділянки теплопроводу. Приклади значень коефіцієнтів зображено на рисунку 2.7.

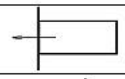
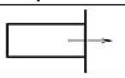
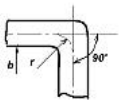
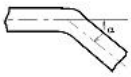
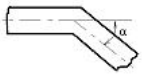
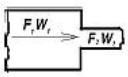
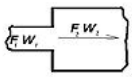
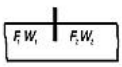
Номер п.п.	Местные сопротивления	Эскиз	Расчет коэффициента ξ	
1	Вход в отверстие с острыми краями		$\xi = 0{,}5$	
2	Выход из канала		$\xi = 1$	
3	Плавный поворот на 90° круглых и квадратных каналов		r/b	ξ
			0,5	1,2
			0,75	0,38
			1	0,19
			2	0,12
			5	0,08
4	То же при угле поворота от 30° до 180°		Значение ξ (п.3) умножается на коэффициент К	
			α^0	К
			30	0,5
			60	0,8
			90	1
			120	1,2
			150	1,3
			180	1,4
5	Резкий поворот прямоугольного канала без закруглений		α^0	ξ
			30	0,6
			60	1
			90	1,2
			120	1,4
			180	1,7
6	Внезапное сужение канала (к скорости W_2)		F_2/F_1	ξ
			0,1	0,5
			0,5	0,3
			0,9	0,1
7	Внезапное расширение канала		F_2/F_1	ξ
			0,1	0,8
			0,5	0,3
			0,9	0,01
8	Частично открытый шибер или заслонка		Степень открытия, %	ξ
			10	230
			30	17
			50	4
			70	1
			90	0,2
			100	0,1

Рисунок 2.7. – Значення коефіцієнта місцевих опорів

Визначення витрати тиску (напору) на ділянках тепломережі виконується по сумі лінійних втрат напору та втрат напору в місцевих опорах:

$$\Delta H = \Delta H_{\text{л}} + \Delta H_{\text{м}} \tag{2.4}$$

Де $\Delta H_{\text{л}} = \frac{\Delta P_{\text{л}}}{\rho g}$ – лінійні втрати напору, м; $\Delta H_{\text{м}} = \frac{\Delta P_{\text{м}}}{\rho g}$ – втрати напору на місцевих опорах, м. Також втрати напору на ділянках можна визначити за наступною формулою:

$$\Delta H = \frac{\Delta P}{\rho g} \quad (2.5)$$

Де ΔP - втрати тиску на ділянках, па; ρ - щільність теплоносія, кг/м3;g – прискорення вільного падіння, м/с^2.

Визначення коефіцієнта гідравлічного опору відбувається за формулою альштуля, так як вона вважається універсальною для розрахунку різних зон руху рідини:

$$\lambda = 0.11\left(\frac{68}{Re} + \frac{\Delta e}{d}\right)^{0.25} \quad (2.6)$$

Де Re – число рейнольдса; d – внутрішній діаметр трубопроводу, м; $\Delta e -$ абсолютна шорсткість, мм.

У свою чергу число рейнольдса визначається як:

$$Re = \frac{VD}{\upsilon} \quad (2.7)$$

Де, V – швидкість потоку рідини, м/с; D – діаметр труби, м; υ – кінематична в'язкість.

Також до розрахунку гідравлічних параметрів відносять визначення об’єму споживаної води, що визначається за формулою:

$$Q = \pi \cdot d^2/4 \cdot v \quad (2.8)$$

Де d – внутрішній діаметр труби, м; v – швидкість водяного потоку, м/с; Q – об’єм споживання води, м3.

Одним з гідравлічних параметрів, що визначає розроблений програмний додаток є гідравлічний уклон, що при постійній швидкості течії і однаковій висоті русла визначається за наступною формулою (яка є виведеною з рівняння бернуллі):

$$i = \frac{H1-H2}{l} \quad (2.9)$$

Де H1 – напір потоку рідині на початку трубопроводу, м; H2 – напір потоку рідини на кінці трубопроводу, м; l – довжина ділянки трубопроводу, м. При інших вхідних даних гідравлічний уклон може розраховуватись за іншою формулою:

$$i = \frac{16\lambda Q^2}{2g\pi^2 D^5} \quad (2.10)$$

Де $\lambda -$ коефіцієнт гідравлічного опору; Q – об'єм споживаної рідини, м^3; g – прискорення вільного падіння, м/с^2; D – діаметр трубопроводу, м.

Окрім головного модулю програмного додатку є додатковий, що відповідає за визначення точки роси при визначених температурі повітря та вологості приміщення:

$$f(T, Rh) = \frac{a \cdot T}{b+T} + \ln\left(\frac{Rh}{100}\right) \quad (2.11)$$

$$T_p = \frac{b \cdot f(T,Rh)}{a - f(T,Rh)} \quad (2.12)$$

Де T_p – температура точки роси, °c; a – константа, що дорівнює 17,27; b – константа, що дорівнює 237,7; T – температура повітря, °C; Rh – відносна вологість повітря, %;

3 РОЗРОБКА ПРОГРАМНОГО ДОДАТКУ ДЛЯ РОЗРАХУНКУ ПАРАМЕТРІВ ТЕПЛОМЕРЕЖ

3.1 Створення прототипу

Для коректної роботи програмного додатку та визначення його інтерфейсу перш за все треба визначити його архітектуру та продумати загальну структуру, щоб запобігти виникненню проблем з навігацією та пошуком інформації при роботі з програмним додатком. На рисунку 3.1 зображено схему взаємодії компонентів програмного додатку «Gidravlika».

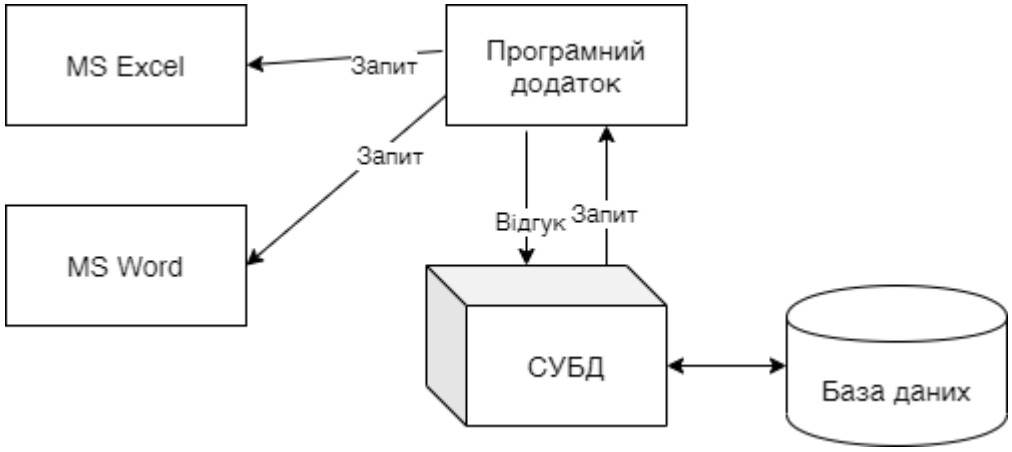


Рисунок 3.1 – Схема взаємодії компонентів додатку

Визначивши як компоненти додатку взаємодіють один з одним, перед розробленням самого програмного додатку, необхідно створити прототип інтерфейсу для визначення кращого розміщення елементів для відповідності зазначеним вимогам. Такий прототип дає загальне уявлення про місцезнаходження основних функціональних блоків (рис. 3.2).

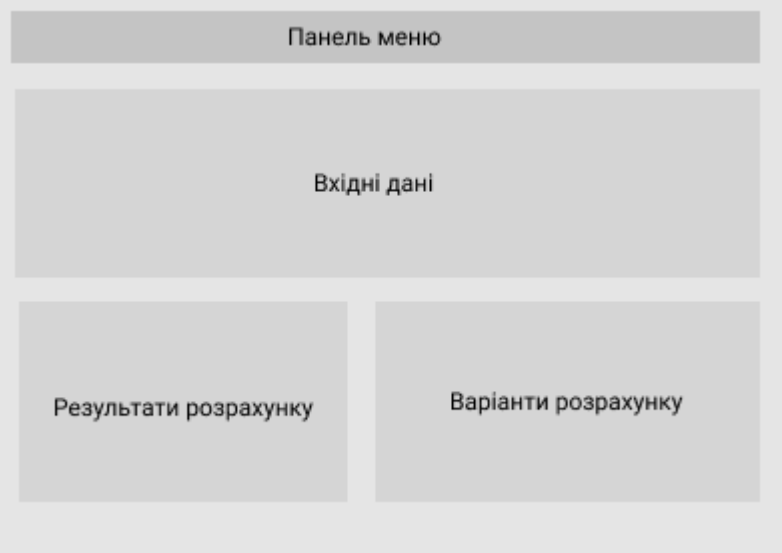


Рисунок 3.2 – Прототип додатку

Панель меню повинна надати користувачу можливість обрати варіант збереження звіту, за що відповідає пункт меню «зберегти звіт», а також обрати варіант додаткового інструментарію для застосування за допомогою пункту меню «інструменти».

До вхідних даних можна віднести довжину ділянки теплопроводу абсолютну шорсткість, яка визначається відповідно до типу ділянки, температура води та коефіцієнт місцевих опорів.

Варіантів розрахунку у даному програмному додатку налічується 4 види, тож за допомогою функціонального блоку «варіанти розрахунку» можливо обрати один з наявних, відповідно до наявних вхідних даних.

Блок «результати розрахунку» містить в собі відповідні поля, що виводять результати обчислень користувачу на екран, до яких належать напори в теплопроводі, обсяг води, що проходить, швидкість води та діаметр теплопроводу. Загальний вигляд прототипу програмного додатку представлено на рисунку 3.3.

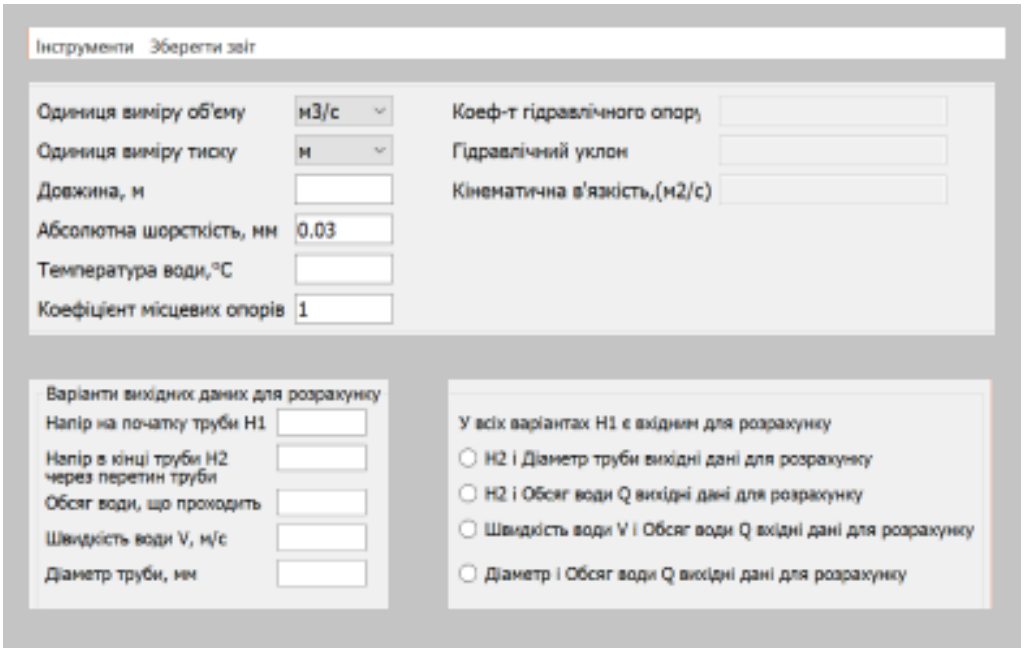


Рисунок 3.3 – Загальний вигляд прототипу додатку

3.2 Розробка бази даних

Відповідно до поставлених задач існує необхідність збереження отриманих результатів та вхідних даних, на основі яких відбувалося виконання розрахунків, до бази даних, структуру якої представлено на рисунку 3.4.

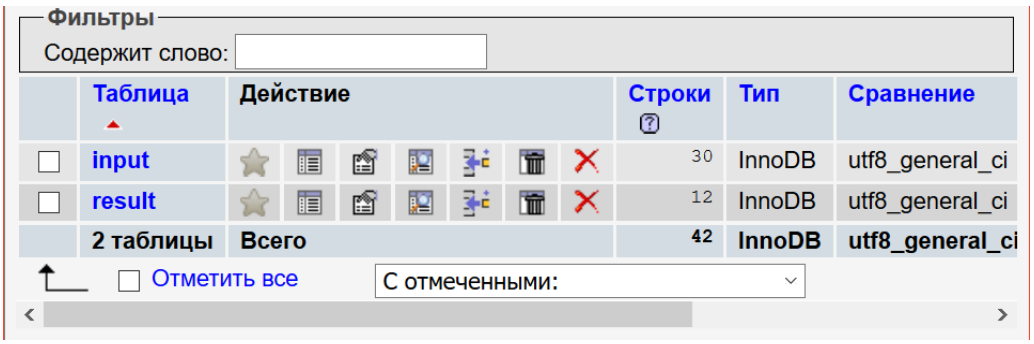


Рисунок 3.4 – Структура бази даних «gidravic»

Таблиця «input» (рис. 3.5) зберігає в собі довжину ділянки теплопроводу, абсолютну шорсткість, температуру води, а також коефіцієнт місцевих опорів, ці дані потрібні для розрахунку основних гідравлічних параметрів.

input_id	length	absolute_roughness	temperature	coef_local_resist
1	1.00	0.03	10.00	1.00
2	10.00	0.03	22.00	1.00
3	10.00	0.03	22.00	1.00
4	1.00	0.03	11.00	1.00
5	1.00	0.03	11.00	1.00
6	1.00	0.03	11.00	1.00
7	1.00	0.03	10.00	1.00

Рисунок 3.5 – Структура таблиці «input»

Всі отримані результати розрахунку гідравлічних параметрів тепломереж зберігаються відповідно у таблиці «result» до складу якої належать вхідні дані, коефіцієнт гідравлічного опору, гідравлічний уклон, кінематична в’язкість, напори на початку та на кінці ділянки трубопроводу, об’єм споживаної води, її швидкість та діаметр трубопроводу (рис 3.6).

Показать все | Количество строк: 50 Фильтровать строки: Поиск в таблице Сортировать по индексу: Ниодного

+ Параметры

result_id	id_input	lmbd	i	H1	H2	Q	V	D	zz
1	3	1.00000000	1.00000000	1.00000000	1.00000000	1.00000	1.00000000	1.00	1.00000000
9	27	0.01720000	0.05000000	4.00000000	3.00000000	0.57087	0.02909000	50.00	0.00000630
10	28	0.01720000	0.05000000	4.00000000	3.00000000	0.57087	0.02909000	50.00	0.00000630
11	29	0.01720000	0.05000000	4.00000000	3.00000000	0.57087	0.02909000	50.00	0.00000630
12	30	0.01720000	0.05000000	4.00000000	3.00000000	0.57087	0.02909000	50.00	0.00000630

Отметить все С отмеченными:

Рисунок 3.6 – Структура таблиці «result»

Для взаємодії бази даних та програмного додатку було створено додатковий функціональний модуль. Фрагмент коду, що дозволяє з’єднатися з базою даних, представлено на рисунку 3.7.

```
def get_connection():
    connection = pymysql.connect(host='127.0.0.1',
                                 user='mysql',
                                 password='mysql',
                                 db='gidravic',
                                 charset='utf8mb4',
                                 cursorclass= pymysql.cursors.DictCursor)
    return connection
```

Рисунок 3.7 – Функція get_connection()

Розроблений функціональний модуль має функції запису даних до таблиць input та result (рис. 3.8), а також функції виконання запитів додавання даних до бази та зчитування їх (рис 3.9).

```
def insert_input(connection, input):
    # Prepare SQL query to INSERT a record into the database.
    sql = """INSERT INTO input (length , absolute_roughness, temperature , coef_local_resist)
                    VALUES ('%d', '%d', '%d', '%d');"""
    execute_query(connection, sql, input)


def insert_result(connection, result, id_input):
    # Prepare SQL query to INSERT a record into the database.
    sql = """INSERT INTO result (id_input, lmbd, i, H1, H2, Q, V, D, zz)
                    VALUES ('%s', '%d', '%d', '%d', '%d', '%d', '%d', '%d', '%d');"""
    data = (id_input, result[0], result[1], result[2], result[3], result[4], result[5], result[6], result[7])
    execute_query(connection, sql, data)
```

Рисунок 3.8 – Функції додавання даних до бази

```
def execute_query(connection, sql, values):
    cursor = connection.cursor()
    try:
       # Execute the SQL command
       cursor.execute(sql % values)
       # Commit your changes in the database
       connection.commit()
    except:
        # Rollback in case there is any error
        #except Exception as e: print(e)
        connection.rollback()


def execute_select(connection, sql):
    cursor = connection.cursor()
    try:
       # Execute the SQL command
       cursor.execute(sql)
       # Commit your changes in the database
       connection.commit()
    except:
        # Rollback in case there is any error
        connection.rollback()
    return cursor
```

Рисунок 3.9 – Функції виконання запитів

3.3 Створення головного модулю програмного додатку

Відповідно до обраного інструментарію, перш за все необхідно створити робоче діалогове вікно за допомогою програмного продукту qt designer. Даний програмний продукт дає змогу зробити макет діалогового вікна, який надалі треба конвертувати в програмний код за допомогою командного рядка. На рисунку 3.10 зображено макет головного діалогового вікна програмного додатку, що містить в собі меню, поля введення та виведення даних, а також функціональні кнопки.

Основною задачею роботи програмного додатку є виконання розподілених розрахунків гідравлічних параметрів тепломережі. За логікою роботи додатку спершу користувач повинен внести до дані до відповідних полей, деякі з них вже заповнені (наприклад, абсолютна шорсткість та коефіцієнт місцевих опорів), тобто користувачу надається одразу дані параметрів, які найчастіше використовуються для розрахунків, але з можливістю редагування. Тобто за потреби користувач може ввести свої необхідні дані.

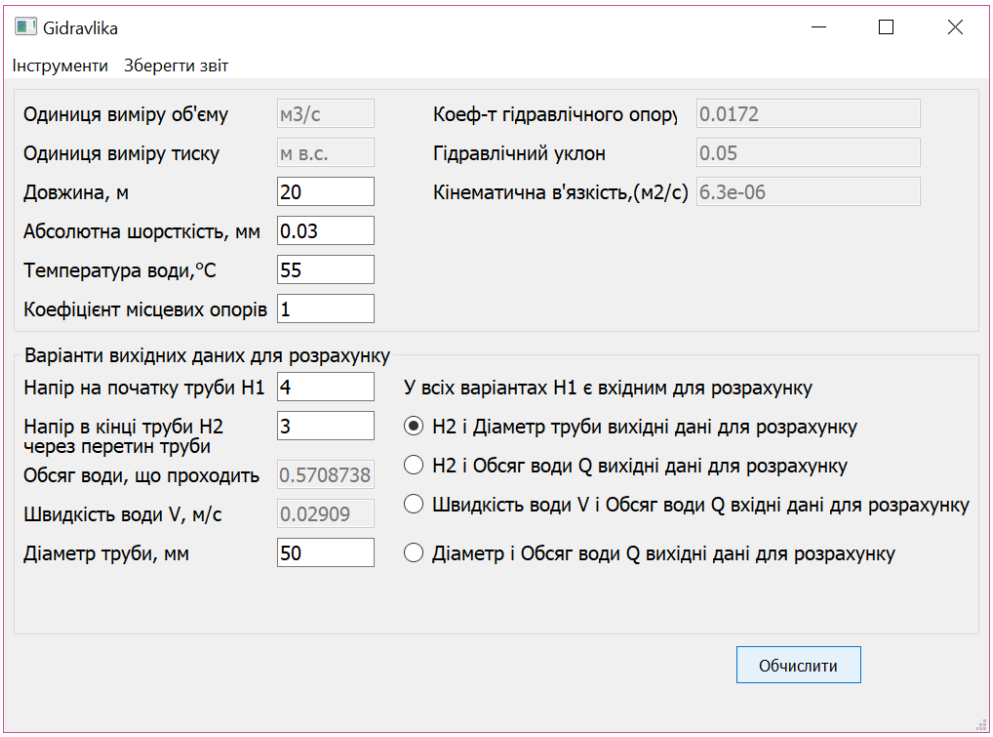


Рисунок 3.10 – Головне діалогове вікно програмного додатку

Файл формату «.py», отриманий після конвертування, стає головним модулем програми. Користувач моє можливість обрати один з чотирьох варіантів виконання розрахунків відповідно до наявних вхідних даних. На рисунку 3.11 зображено варіант виконання розрахунку при наявних значеннях напору наприкінці трубопроводу та його діаметру.

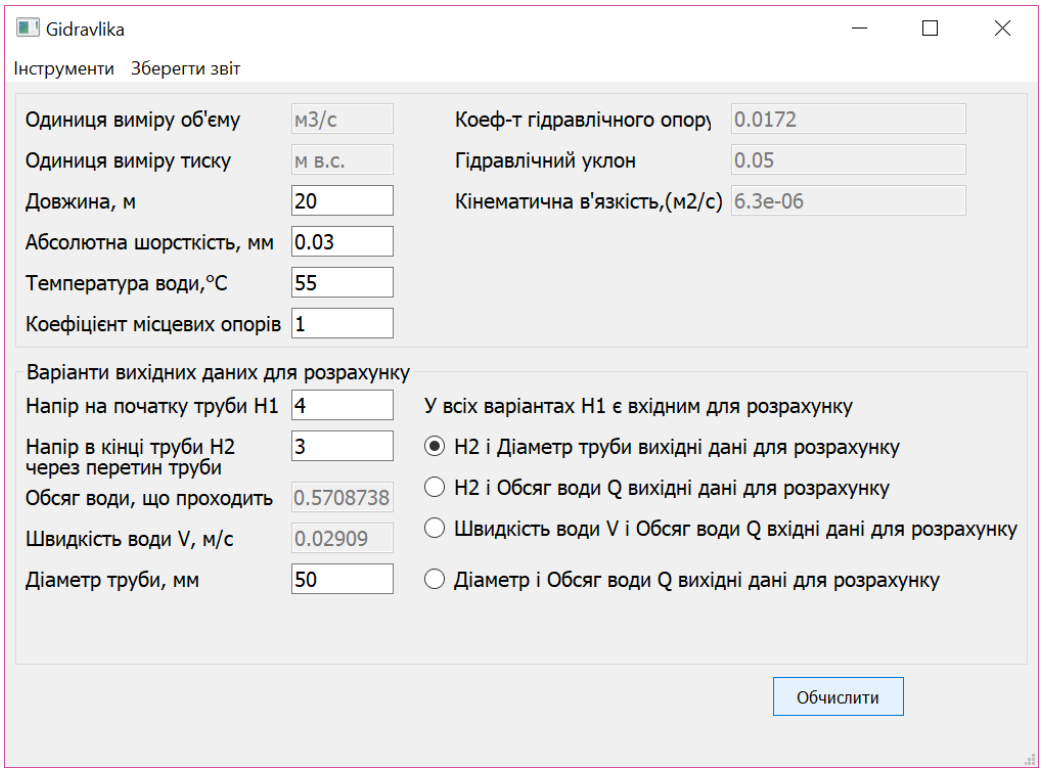


Рисунок 3.11 – Приклад роботи програми

Виконання розрахунку виконується при натисканні кнопки «обчислити», беручи до уваги обраний користувачем варіант виконання обчислень. На рисунку 3.12 представлено фрагмент функції count(), що виконує розрахунок при першому обраному варіанті обчислення гідравлічних параметрів.

При введенні користувачем некоректних даних та при виникненні помилки користувач отримує відповідне повідомлення про виникнення помилки, що зображено на рисунку 3.13.

```
if self.radioButton.isChecked():
    #вычисление гидравлического уклона
    H2 = float(self.lineEdit_9.text())#напор в конце трубы
    i = (H1-H2)/l #гидравлический уклон
    self.lineEdit_3.setText(str(i))
    #-----------------------------
    #вычисление коеффициента гидравлического сопротивления
    d = float(self.lineEdit_12.text())#диаметр трубы
    lmbd = 0.11*pow((e/d), 0.25)
    self.lineEdit_2.setText(str(round(lmbd, 4)))
    #-----------------------------
    #вычисление кинематической вязкости
    zz = 0.0178/10000*(1 + 0.0337*t + 0.000221*t*t)
    self.lineEdit_6.setText(str(round(zz,7)))
    #-----------------------------
    #вычисление скорости потока
    Re = 2320
    v = Re*zz/d
    self.lineEdit_11.setText(str(round(100*v,5)))
    #-----------------------------
    #вычисление объема потока
    Q = 3.14 * pow(d, 2.0) * v/4
    self.lineEdit_10.setText(str(round(Q,7)))

    result = (lmbd, i, H1, H2, Q, v, d, zz)
```

Рисунок 3.12 – Фрагмент функції виконання розрахунку

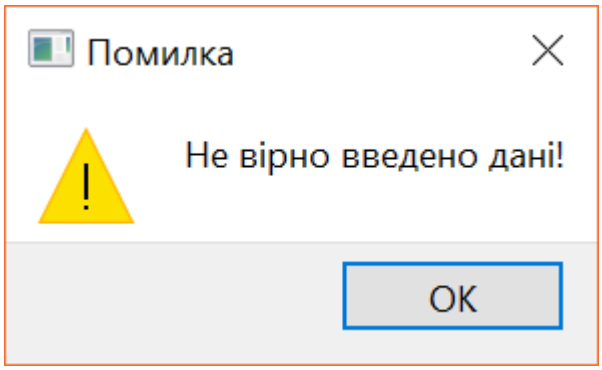


Рисунок 3.13 – Повідомлення про виникнення помилки

Однією з поставлених задач було надання користувачу можливості зберігання даних у вигляді звіту. Використовуючи даний програмний додаток користувач може зберегти отримані дані в форматі «.xlsx» або «.doc», за що відповідають відповідно функції savetoexcel (рис. 3.14) та savetoword(рис. 3.15). Зберігання даних до файлу ексель виконується в одну таблицю, додаючи новий рядок, в свою чергу звіт формату «.docx» може використовуватись для передачі отриманих розрахунків іншим колегам або співробітникам. На рисунках 3.16 та 3.17 зображено приклади збережених звітів у форматах «.xlsx» та «.docx» відповідно.

```
def saveToExcel(self):
    try:

        lmbd = float(self.lineEdit_2.text())#коеффициента гидравлического сопротивления
        i = float(self.lineEdit_3.text())#гидравлического уклона
        zz = float(self.lineEdit_6.text())#кинематической вязкости
        H1 = float(self.lineEdit_8.text())#напор вначале трубы
        H2 = float(self.lineEdit_9.text())#напор вконце трубы
        Q = float(self.lineEdit_10.text())#обьем потока
        V = float(self.lineEdit_11.text())#скорости потока
        D = float(self.lineEdit_12.text())#диаметр трубы


        wb = load_workbook(filename='report.xlsx')
        ws = wb.worksheets[0]
        columnn=ws['A']
        n = len(columnn)
        row = n + 1
        ws.cell(row, column=1).value = lmbd
        ws.cell(row, column=2).value = i
        ws.cell(row, column=3).value = zz
        ws.cell(row, column=4).value = H1
        ws.cell(row, column=5).value = H2
        ws.cell(row, column=6).value = Q
        ws.cell(row, column=7).value = V
        ws.cell(row, column=8).value = D

        wb.save(filename='report.xlsx')
    except:
        msgBox = QMessageBox()
        msgBox.warning(self.centralwidget, 'Помилка ',"Не вірно введено дані!")
```

Рисунок 3.14 – Функція savetoexcel

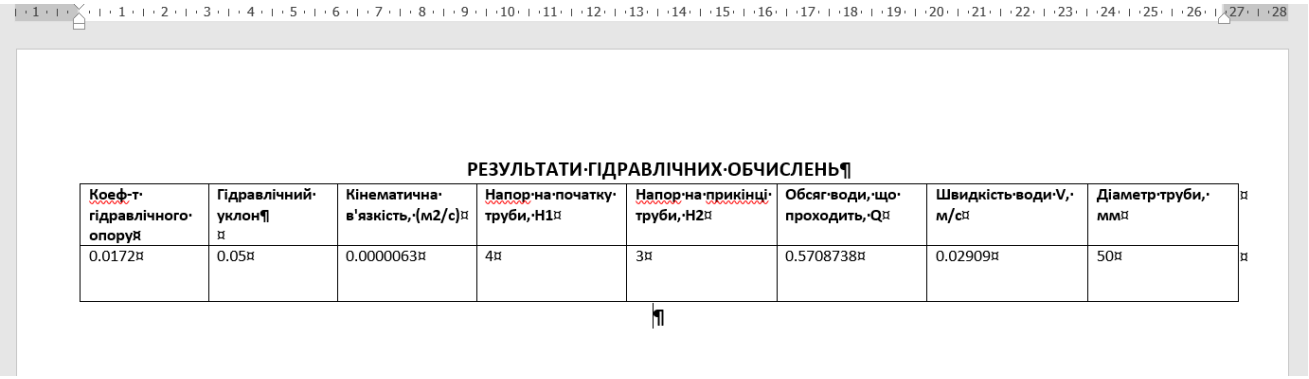
```
def saveToWord(self):
    lmbd = self.lineEdit_2.text()#коеффициента гидравлического сопротивления
    i = self.lineEdit_3.text()#гидравлического уклона
    zz = self.lineEdit_6.text()#кинематической вязкости
    H1 = self.lineEdit_8.text()#напор вначале трубы
    H2 = self.lineEdit_9.text()#напор вконце трубы
    Q = self.lineEdit_10.text()#обьем потока
    V = self.lineEdit_11.text()#скорости потока
    D = self.lineEdit_12.text()#диаметр трубы


    doc = DocxTemplate("report.docx")
    context_Ew = { 'var_lmbd' : lmbd , 'var_i' : i, 'var_zz' : zz , 'var_H1' : H1, 'var_H2' : H2, 'var_Q' : Q, 'var_V' : V, 'var_D' : D}
    doc.render(context_Ew)
    doc.save("Results.docx")
```

Рисунок 3.15 – Функція savetoword

	A	B	C	D	E	F	G	H
1	Коеф-т гідравлічного опору	Гідравлічний уклон	Кінематична в'язкість, (м2/с)	Напор на початку труби, H1	Напор на прикінці труби, H2	Обсяг води, що проходить, Q	Швидкість води V, м/с	Діаметр труби, мм
2	0,0458	0	0,0000024	1	1	0,0044058	0,56125	1
3	0,0543	0,5	0,0000033	2	1	4	1106,77255	0,68
4	0,0543	0,005	0,0000063	1	0,5	3	190,22896	1,42
5	0,0172	0,05	0,0000063	4	3	0,5708738	0,02909	50
6	0,0172	0,05	0,0000063	4	3	0,5708738	0,02909	50
7								

Рисунок 3.16 – Звіт «.xlsx»



РЕЗУЛЬТАТИ·ГІДРАВЛІЧНИХ·ОБЧИСЛЕНЬ¶

Коеф-т· гідравлічного· опору¤	Гідравлічний· уклон¶ ¤	Кінематична· в'язкість,·(м2/с)¤	Напор·на·початку· труби,·H1¤	Напор·на·прикінці· труби,·H2¤	Обсяг·води,·що· проходить,·Q¤	Швидкість·води·V,· м/с¤	Діаметр·труби,· мм¤
0.0172¤	0.05¤	0.0000063¤	4¤	3¤	0.5708738¤	0.02909¤	50¤

¶

Рисунок 3.17 – Звіт «.docx»

Відповідно до умов, що зазначені в технічному завданні, користувачу повинна надаватися можливість використання додаткового інструментарію, до якого належить визначення точки роси, визначення витрат тиску на ділянці (для розрахунку яких використовуються дані, що введені користувачем до головного вікна програми), конвертер фізичних величин, та калькулятор.

Розрахунок точки роси може знадобитися користувачу при розрахунку або визначенні належних умов обслуговування тепломереж, з метою запобігання накопичення вологи в неналежних місцях. Щоб виконати розрахунок необхідно обрати відповідний пункт меню, а саме «інструменти» –> «температура точки роси». На рисунку 3.18 зазначено функцію виконання відповідних розрахунків, а результат роботи даного інструментарію представлено на рисунку 3.19. При введенні некоректних даних користувач отримує відповідне повідомлення про виникнення помилки.

```
def count(self):
    try:
        T = float(self.lineEdit.text())#температура
        Rh = float(self.lineEdit_2.text())#влажность
        a = 17.27#константа
        b = 237.7#константа
        f = a*T/(b+T) + math.log(Rh/100)
        Tp = b*f/(a-f)
        s = str(round(Tp,2)) + " °C"
        self.label_res1.setText(s)
    except:
        msgBox = QMessageBox()
        msgBox.warning(self.centralwidget, 'Помилка ',"Не вірно введено дані!")
```

Рисунок 3.18 – Функція визначення точки роси

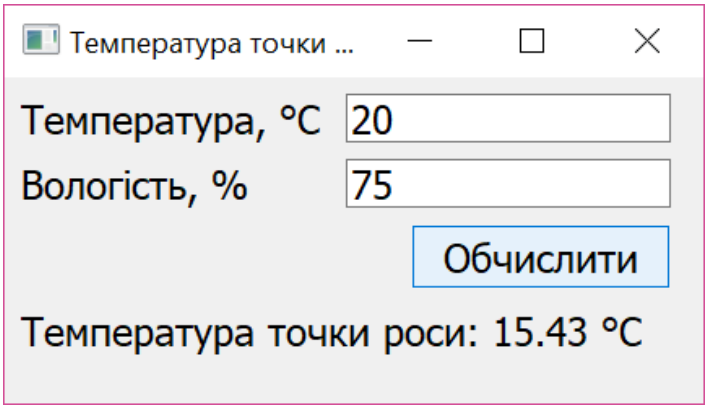


Рисунок 3.19 – Результат визначення температури точки роси

Однією з задач виконання розрахунку гідравлічних параметрів тепломереж є визначення витрат тиску на ділянці. Даний програмний додаток надає таку можливість виконання розрахунку за допомогою відповідного пункту меню «інструменти» –> «витрати тиску». Для виконання розрахунків використовуються останні отримані дані, що були введені до головного модулю програми та збережені у базі даних. Кнопка «обчислити» виконує зчитування даних з бази даних та виконує відповідні розрахунки, виводячи отриманий результат у відповідні поля. На рисунку 3.20 представлено код функції, що виконує розрахунок, відповідно на рисунку 3.21 представлено результат її виконання.

```
def count2(self):
    connection = connutils.get_connection()
    cursor = connection.cursor()

    cursor.execute("SELECT * FROM input ORDER BY input_id DESC LIMIT 1")
    records = cursor.fetchall()
    row = records[0]
    l = row["length"]
    ro = 1000
    e = row["absolute_roughness"]
    z = row["coef_local_resist"]
    v = 0.02909
    d = 50
    lmbd = 0.11*pow((e/d), 0.25)

    Pl = lmbd*v*v*ro*l/(2*d)
    Pm = z*v*v*ro/2

    P = Pl+Pm
    self.lineEdit.setText(str(round(Pl,5)))
    self.lineEdit_2.setText(str(round(Pm,2)))
    self.lineEdit_3.setText(str(round(P,2)))
```

Рисунок 3.20 – Функція визначення витрат тиску

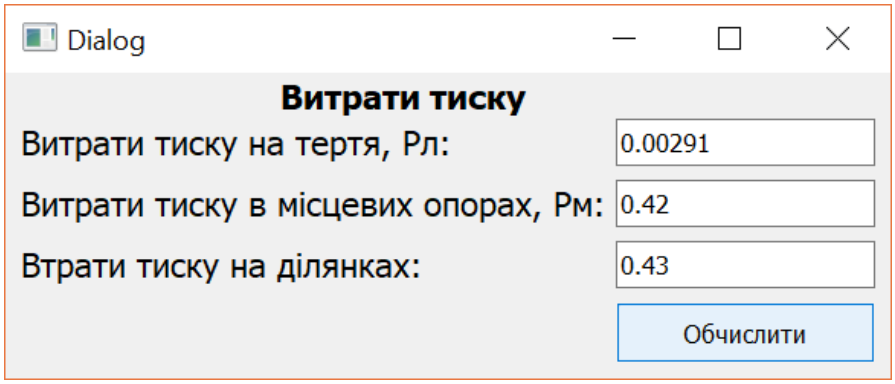


Рисунок 3.21 – Результат обчислення витрат тиску

Також як додатковий інструментарій необхідно було створити калькулятор для виконання звичайних математичних операцій для випадку, коли користувачу настає потреба шось обрахувати, але немає під рукою потрібного пристрою, тоді в нагоді стане цей калькулятор. Щоб його використати, треба натиснути пункт меню «інструменти» –> «калькулятор». На рисунку 3.22 зображено приклад роботи калькулятора, на рисунку 3.23 приведено код функції розрахунку

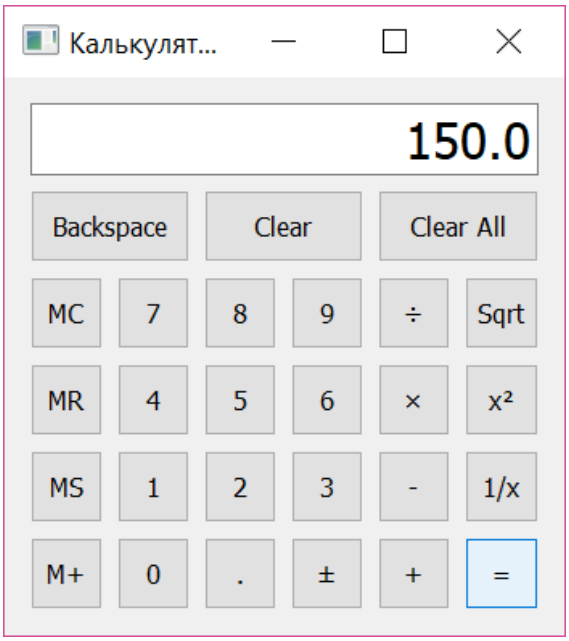


Рисунок 3.22 – Результат виконання множення

```
def multiplicativeOperatorClicked(self):
    clickedButton = self.sender()
    clickedOperator = clickedButton.text()
    operand = float(self.display.text())

    if self.pendingMultiplicativeOperator:
        if not self.calculate(operand, self.pendingMultiplicativeOperator):
            self.abortOperation()
            return

        self.display.setText(str(self.factorSoFar))
    else:
        self.factorSoFar = operand

    self.pendingMultiplicativeOperator = clickedOperator
    self.waitingForOperand = True
```

Рисунок 3.23 – Функція множення

Після проведення манипуляцій, що зазначено вище, було розроблено програмний додаток, що відповідає вимогам, визначеним у Додутку А. Програмний код створених файлів представлено в Додатку В. Інструкцію користувача представлено в Додатку Г.

ВИСНОВКИ

У процесі виконання даної дипломної роботи було розроблено програмний додаток для виконання розподілених розрахунків гідравлічних параметрів тепломереж. Даний програмний додаток було розроблено відповідно до поставлених вимог:

- Визначення діаметрів теплопроводів;
- Розрахунок напору в різних точках тепломережі;
- Визначення витрат тиску на різних ділянках;
- Наявність допоміжного інструментарію для обчислень;
- Можливість збереження результату у вигляді звіту в форматі .xlsx.

Провівши аналіз предметної області, що включає в себе дослідження необхідності створення програмного додатку, а також аналіз аналогів. Ґрунтуючись на отриманих результатах, було прийнято рішення щодо доцільності створення програмного додатку для виконання розрахунків гідравлічних параметрів теплових мереж, оскільки сфера теплоенергетики в україні потребує значних покращень як технічних показників, так і технологічних.

Після проведення попередніх досліджень було сформовано мету створення даного дипломного проекту, а саме створення програмного додатку «Gidravlika» для виконання розподілених розрахунків параметрів тепломереж для різних режимів їх роботи, що полегшує роботу працівників, обслуговуючих сферу теплоенергетики.

Як результат аналізу аналогів було сформовано певний перелік необхідних функцій для реалізації, а також обрано необхідний інструментарій для роботи над проектом, що включає в себе мову програмування python 3, текстовий редактор з відкритим вхідним кодом atom, додатковий модуль qt gui для розробки інтерфейсу користувача та субд mysql.

Було проведено необхідне проектування програмного додатку. На даному етапі біли створені функціональні діаграми нотації IDEF0 для демонстрації етапів роботи програмного додатку, er-діаграма для демонстрації структури бази даних, а

також діаграма use case для демонстрації взаємодії акторів та варіантів використання програмного додатку. Відповідно до операційної логіки роботи програмного додатку було виконано математичне моделювання гідравлічного розрахунку тепломереж.

Перед початком створення модулів програмного додатку було розроблено прототип інтерфейсу робочого вікна, а реалізовано його було засобами програми QT Designer. За допомогою бібліотек функцій та надбудов мори програмування Python 3.8 було виконано розробку основного функціоналу програмного додатку, а також необхідних додаткових інструментів, що були визначені технічнім завданням. При введенні некоректних даних для виконання розрахунків користувач отримує відповідне повідомлення про виникнення помилки. Після того, як програмний додаток було остаточно розроблено, було виконано його тестування на визначення коректності виконання розрахунків.

Отже, після виконання всіх необхідних етапів було розроблено програмний додаток, що матиме попит серед диспетчерів та інших співробітників, що обслуговують тепломережі та теплопостачання, а також дозволить пришвидшити їх роботу.

СПИСОК ЛІТЕРАТУРИ

1) Немнитов, В.А. Разработка информационной системы поддержки принятия решений для оптимизации систем теплоснабжения [Текст]/В.А. Немнитов, С.М. Терехов// Вестник ТГТУ.-Тамбов, 2018.-С.216-227

2) Косяков, С.В. Повышение эффективности эксплуатации систем централизованного теплоснабжения на основе применения информационной системы мониторинга тепловых сетей [Текст]/С.В. Косяков, А.М. Садыков, В.В. Сенников, В.В. Смирнов// Вестник ИГЭУ. – Иваново, 2018. – С.57-66

3) Алексенко, О.В. Визначення параметрів теплової мережі мікрорайону на основі комплексної адаптивної моделі [Текст]/О.В. Алексенко, В.Г. Неня, Ю.В. Парфенеко, М.М. Проклова// Вестник ХНТУ. – Харків, 2014.-С.108-113

4) Різник Е. Дослідження сучасного стану тепломереж в Україні[Текст]/Е. Різник, І.Б. Федишин//Природничі гуманітарні науки. Актуальні питання. – Тернопіль, 2015. – С.80-81

5) Ефективні конструкції магістральних тепломереж [Електронний ресурс] – режим доступу: https://bitly.su/Q49mFe

6) Перспективи розвитку теплової енергетики [Електронний ресурс] – режим доступу: https://bitly.su/QHpdDTxN

7) Экономическая эффективность информатизации инженерных коммуникаций [Електронний ресурс] – режим доступу: https://bitly.su/hgNtDgyH

8) Korf Hydraulics - программа для расчета гидравлических сетей [Електронний ресурс] – режим доступу: https://bitly.su/uezDBC

9) Умная вода [Електронний ресурс] – режим доступу: http://smartwater.su/

10) VALTEC инженерная техника [Електронний ресурс] – режим доступу: https://valtec.ru/document/calculate/

11) Python [Електронний ресурс] – режим доступу: https://bitly.su/MnaUO

12) PyQt5 Tutorial [Електронний ресурс] – режим доступу: https://bitly.su/CZ28

13) SQL [Електронний ресурс] – режим доступу: http://progopedia.ru/language/sql/

14) Преимущества Иерархической Структуры Работ (WBS) для менеджеров ИТ проектов [Електронний ресурс] – режим доступу: https://habr.com/ru/post/327872/

15) Діаграми „сутність-зв”язок”: призначення, місце застосування, правила побудови, erd-стандарти. Сутності, відношення та зв’язки в нотації чена [електронний ресурс] – режим доступу: https://bitly.su/iuz40

16) Кузьмин е. В. Управление проектами с использованием microsoft project 2013: лабораторный практикум / е. В. Кузьмин. - самара: пгути, 2016. –151 с.

17) Методологія функціонального моделювання sadt [електронний ресурс] – режим доступу: https://bitly.su/kqfq542

18) Методология idef0 - учебная и научная деятельность анисимова владимира викторовича [електронний ресурс] – режим доступу: https://cutt.ly/6ynzpev

ДОДАТОК А

ТЕХНІЧНЕ ЗАВДАННЯ

На розробку програмного додатку «Gidravlika»

Назва програмного продукту **«gidravlika».**
Область застосування програмного продукту – програмний додаток призначений для виконання розрахунку гідравлічних показників тепломереж.
Об'єкт, у якому використовують програму – ТЕС, або інші компанії, що обслуговують інженерні комунікації у сфері енергопостачання.

1 основи для розробки
Розробка виконується на основі завдання, виданого викладачем секції інформаційних технологій проектування.

2 призначення розробки
Розробка має надавати користувачу можливість оброблювати, переглядати, а також зберігати в базі даних отримані результати розрахунків та бути універсальною для різних тепломереж.
Назва організації: ковальчук владислав, група ІТ-61-8, кафедра комп'ютерних наук.
Тема проекту: «програмний додаток розподілених обчислень параметрів гідравлічних теплових мереж»

3 вимоги до програмного додатку
Програмний додаток повинен бути реалізований у вигляді віконного додатку з підключенням до баз даних, що забезпечує виконання функціональних можливостей з пункту 3.3.

3.1 вимоги до програмного додатку
Програмний додаток має бути розроблено мовою програмування python 3 з використанням qt gui для створення інтерфейсу користувача, а також мати підключення до баз даних. Створення самої бази даних, таблиць та запитів до них реалізується за допомогою мови запитів sql.

3.2 вимоги до програмного забезпечення
Для належної роботи програмного додатку необхідно постійне інтернет з'єднання, а також наявність встановленого веб-серверу для використання баз даних та виконання необхідних запитів. Для роботи з субд необхідною є наявність будь-якого веб-браузера (google chrome, intertnet explorer або будь-який інший).

3.3 вимоги до функціональних характеристик
Програмний додаток повинен виконувати наступні функції:

- Вибір варіанту виконання розрахунків параметрів гідравлічних тепломереж;
- Збереження даних;
- Наявність калькулятора як додаткового інструмента;
- Можливість розрахунку температури точки роси як додаткового інструмента;
- Збереження результатів розрахунків у вигляді звіту;
- визначення втрат тиску на різних ділянках мережі;

- Визначення діаметрів теплопроводів;

4 перелік програмної документації

- Опис проекту продукту із використанням uml-діаграм;
- Макет інтерфейсу програмного продукту;
- Програмний код розробки програмного додатку;
- Технічне завдання;
- Інструкція користувача;

5 порядок виконання робіт і етапи розробки

Стадії та етапи розробки повинні складатися з наступних пунктів:

- Постановка задачі;
- Розроблення технічного завдання
- Планування роботи: побудова мережевого графіку та діаграми ганта;
- Розроблення моделей програмного додатку;
- Розроблення функціональних модулів програмного додатку та інтерфейсу користувача;
- Провести тестування програмного додатку;
- Розроблення інструкції користувача;
- Оформлення пояснювальної записки;
- Здача пояснювальної записки та розробленого програмного продукту;
- Презентація роботи та її захист.

6 порядок контролю та приймання

Контроль коректності функціонування та придатності програмного продукту здійснюється секцією інформаційні технології проектування кафедри комп’ютерних наук на підставі наданої пояснювальної записки до дипломної роботи та програмних файлів. Контроль ходу виконання проекту здійснюється на підставі календарного плану виконання дипломної роботи:

- Перевірка завдання дипломної роботи.
- Перевірка технічного завдання.
- Перевірка мережевого графіка та діаграми ганта.
- Перевірка математичної моделі програмного додатку.
- Перевірка структури програмного продукту
- Перевірка наявності функціоналу, базових маніпуляцій та інтерфейсу.
- Перевірка інструкції користувача.
- Здача пз.
- Захист.

ДОДАТОК Б
ПЛАНУВАННЯ РОБІТ

Ідентифікація мети іт-проекту методом smart.

Метою даного дипломного проекту є створення програмного додатку, що надає можливість користувачеві провести розрахунки гідравлічних показників теплових мереж, із застосуванням мови програмування python, де користувач може обрати, для якого саме режиму буде проведено обчислення. Для біль чіткої ідентифікації мети рекомендовано використовувати критерій коректності мети – метод smart, результати деталізації якого відображено в таблиці Б.1.

Таблиця Б.1 – деталізація мети проекту методом smart

Specific (конкретна)	Розробити програмний додаток для розподілених обчислень гідравлічних параметрів тепломережі.
Measurable (вимірювання)	Результатом роботи даного проекту є оцінка замовника, адже проект є некомерційним.
Achievable (досяжна, узгоджена)	Оскільки розробник володіє всіма необхідними навичками в реалізації програмних додатків за допомогою засобів python та mysql, а також ознайомлений з необхідним інструментарієм, мету можна вважати досяжною.
Relevant (реалістична)	Для реалізації поставлених задач наявні всі необхідні програмні засоби, а саме редактор коду atom, веб-сервер open server та доступ до мережі інтернет.
Time-related (обмежена в часі)	Програмний додаток розробляється в обмеженому періоді часу, відповідно до наяваного календарного плану.

Опис фази розробки іт-проекту. Розподіл задач на менші та більш керуємі частини спрощує процес управління проектом. Для реалізації цього процесу доцільно використовувати ієрархічну структуру робіт wbs (work breakdown structure), що надає можливість структурувати кожен етап проекту та брати до уваги всі поставлені задачі. Необхідність використовування wbs-діаграм полягає в тому, що вони допомагають правильно організувати проекти, розподілити обов'язки, а також, що не менш важливо, вірно оцінити витрати, ризики та час виконання робіт [14]. На рисунку Б.1 зображено декомпозовану wbs-діаграму.

Планування структури організації, для впровадження готового проекту. Після побудови wbs-діаграми необхідно розробити організаційну структуру виконавців, а саме obs-структуру проекту (organizational breakdown structure), що являє собою графічне представлення всіх учасників проекту, а також відповідальних осіб. Список виконавців, що реалізують даний проект, представлено в таблиці Б.2. На рисунку Б.2 представлено obs-структуру даного проекту.

Таблиця Б.2 – виконавці проекту

Роль	Ім'я	Проектна роль
Розробник, дизайнер, тестувальник	Ковальчук в.а.	Відповідає за розробку функціональних модулів програмного додатку, інтерфейсу користувача, а також його тестування.
Менеджер проекту	Неня в.г.	Відповідає за формування завдання проекту та виконання термінів реалізації всіх етапів проекту.

Побудова календарного графіку виконання ІТ – проекту. На підставі розроблених wbs та obs діаграм будується календарний графік робіт для реального представлення про тривалість виконання робіт з урахуванням вихідних та святкових днів. Для цього зазвичай використовується діаграма ганта, що дозволяє візуалізувати розклад виконання робіт. Діаграму ганта для цього проекту було побудовано за допомогою програми ms project та представлено на рисунку Б.3,

список робіт зазначено на рисунку Б.4. На рис. Б.5- б.10 представлено мережевий графік.

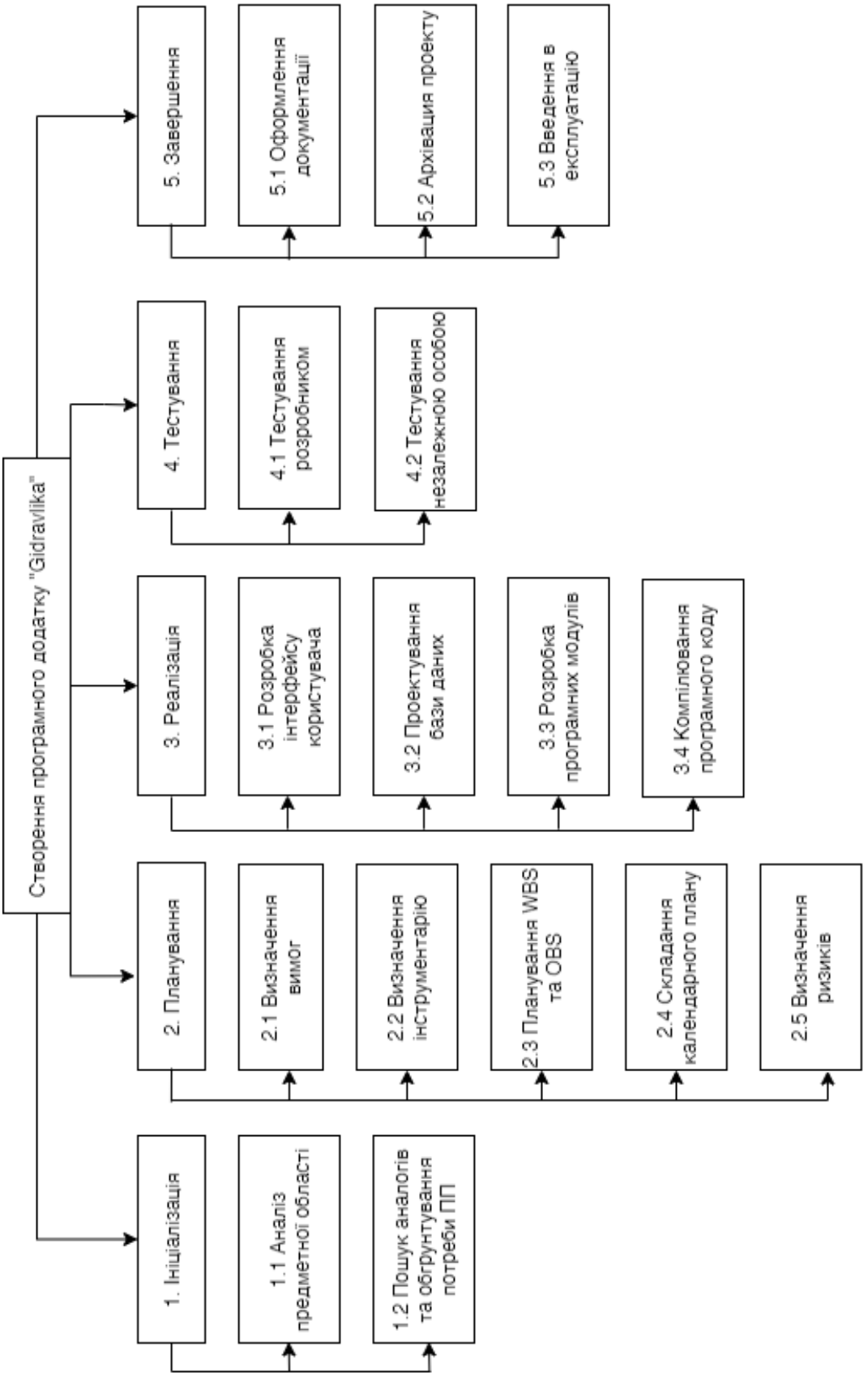


Рисунок Б.1 – WBS-діаграма дипломного проекту

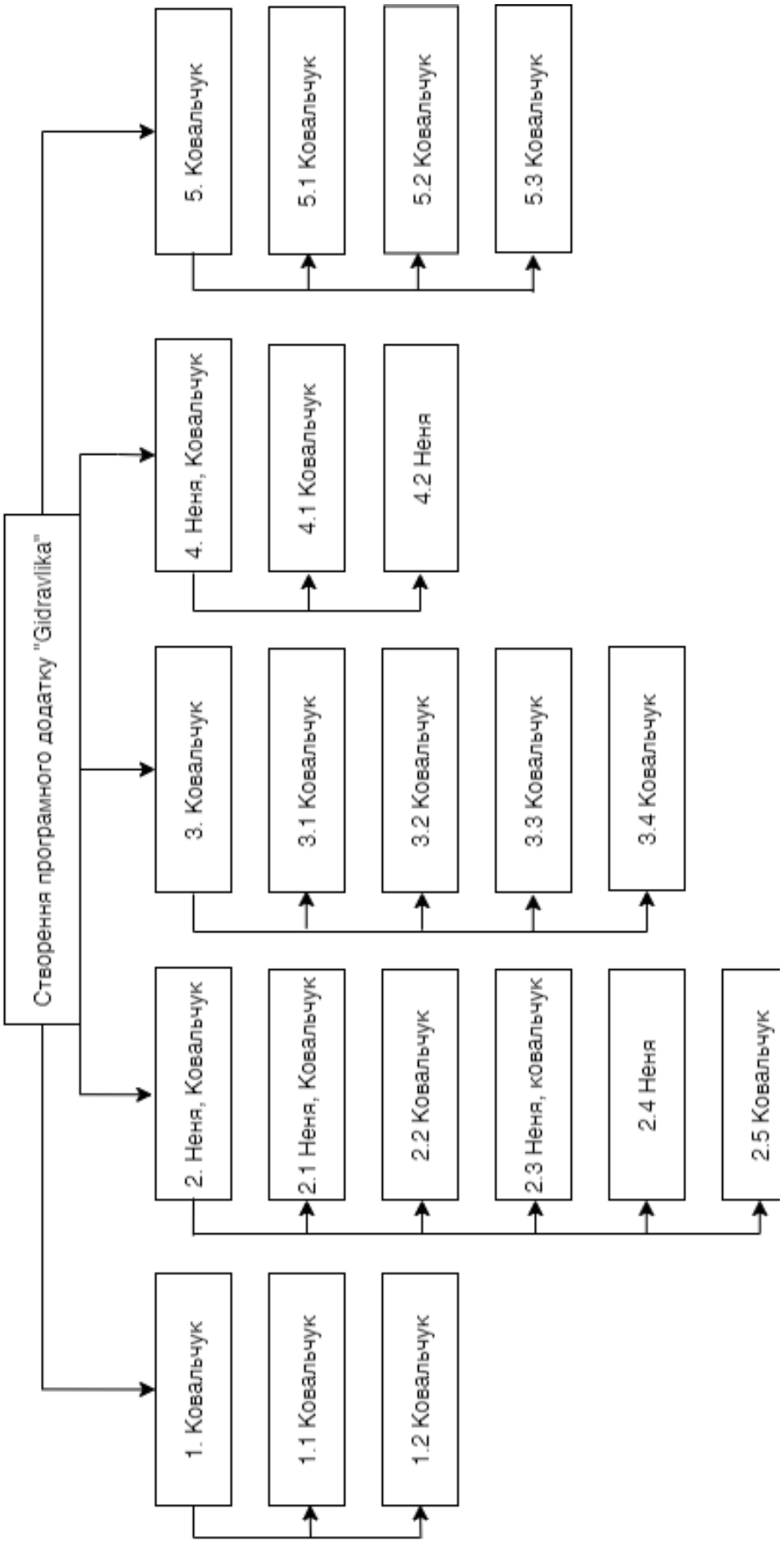


Рисунок Б.2 – OBS-діаграма дипломного проекту

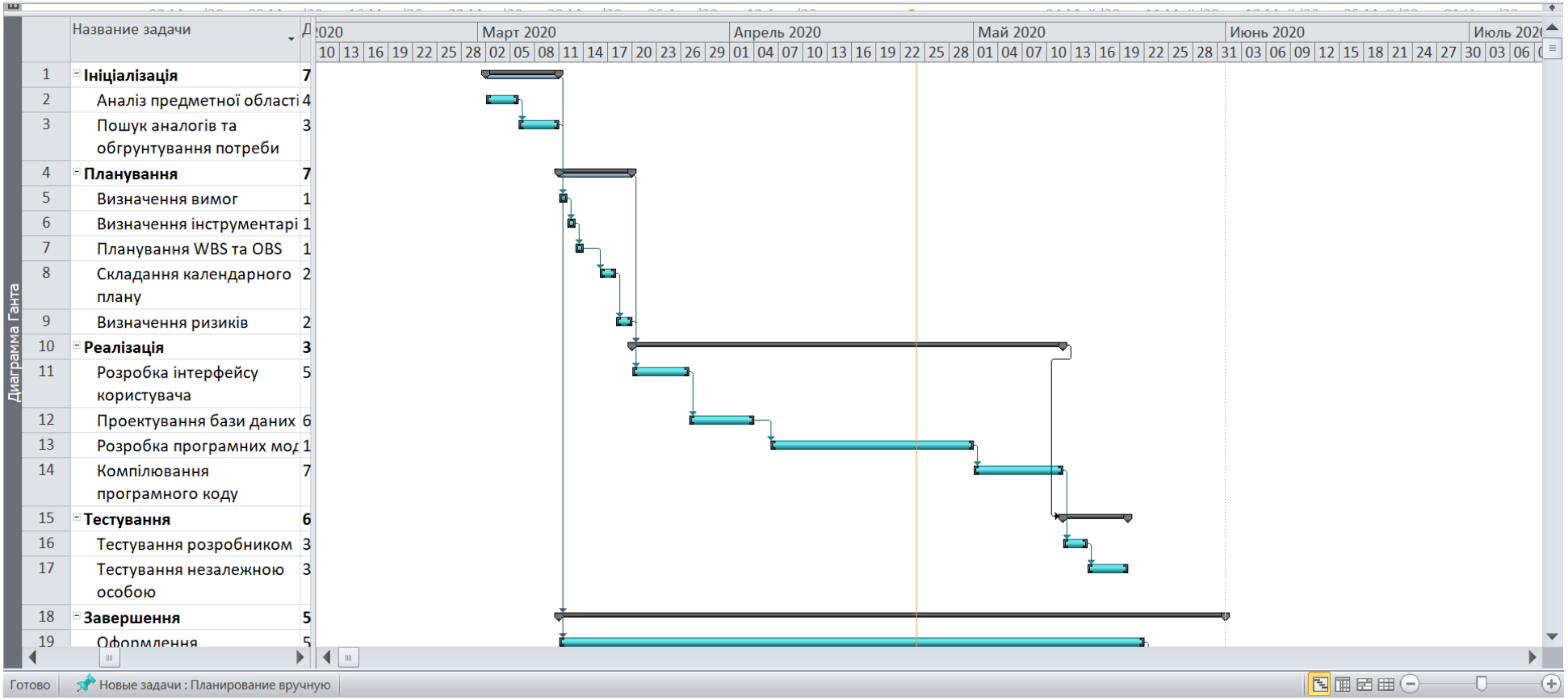
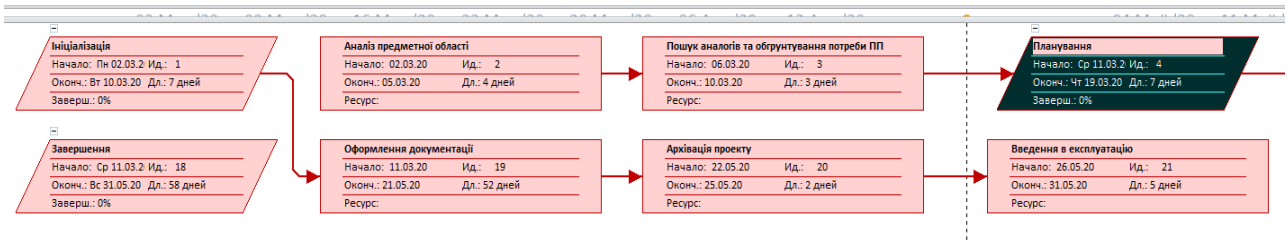


Рисунок Б.3 – Діаграма Ганта для даного проекту

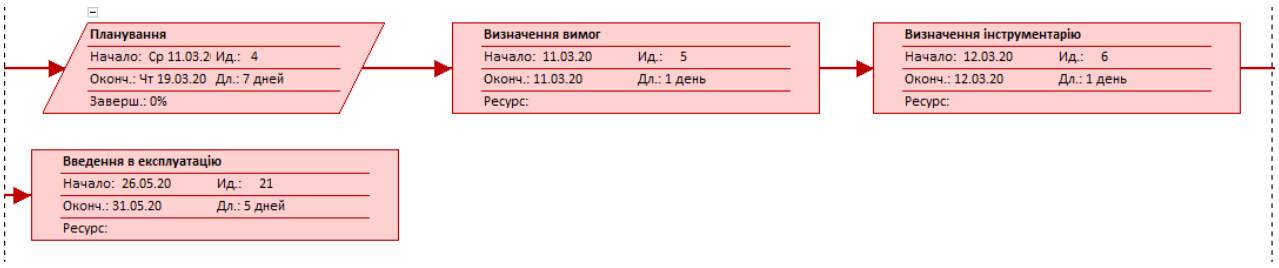
Диаграмма Ганта

	Режим задачи	Название задачи	Длительно	Начало	Окончание	Предшественни
1		**Ініціалізація**	**7 дней**	**Пн 02.03.20**	**Вт 10.03.20**	
2		Аналіз предметної області	4 дней	Пн 02.03.20	Чт 05.03.20	
3		Пошук аналогів та обгрунтування потреби	3 дней	Пт 06.03.20	Вт 10.03.20	2
4		**Планування**	**7 дней**	**Ср 11.03.20**	**Чт 19.03.20**	**1**
5		Визначення вимог	1 день	Ср 11.03.20	Ср 11.03.20	3
6		Визначення інструментарі	1 день	Чт 12.03.20	Чт 12.03.20	5
7		Планування WBS та OBS	1 день	Пт 13.03.20	Пт 13.03.20	6
8		Складання календарного плану	2 дней	Пн 16.03.20	Вт 17.03.20	7
9		Визначення ризиків	2 дней	Ср 18.03.20	Чт 19.03.20	8
10		**Реалізація**	**37 дней**	**Пт 20.03.20**	**Пн 11.05.20**	**4**
11		Розробка інтерфейсу користувача	5 дней	Пт 20.03.20	Чт 26.03.20	9
12		Проектування бази даних	6 дней	Пт 27.03.20	Пт 03.04.20	11
13		Розробка програмних мод	19 дней	Пн 06.04.20	Чт 30.04.20	12
14		Компілювання програмного коду	7 дней	Пт 01.05.20	Пн 11.05.20	13
15		**Тестування**	**6 дней**	**Вт 12.05.20**	**Вт 19.05.20**	**10**
16		Тестування розробником	3 дней	Вт 12.05.20	Чт 14.05.20	14
17		Тестування незалежною	3 дней	Пт 15.05.20	Вт 19.05.20	16
18		**Завершення**	**58 дней**	**Ср 11.03.20**	**Вс 31.05.20**	**1**
19		Оформлення документації	52 дней	Ср 11.03.20	Чт 21.05.20	1
20		Архівація проекту	2 дней	Пт 22.05.20	Пн 25.05.20	19
21		Введення в експлуатацію	5 дней	Вт 26.05.20	Вс 31.05.20	20

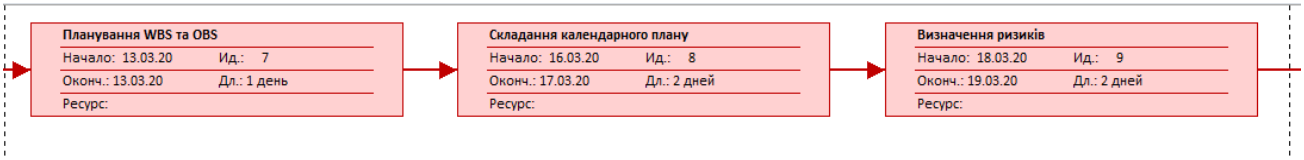
Рисунок б.4 – список робіт для діаграми ганта



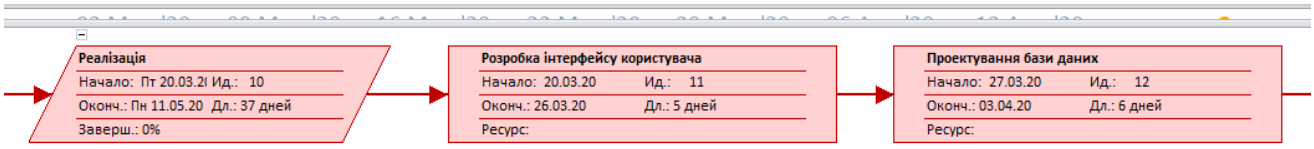
Б.5 – мережевий графік проекту



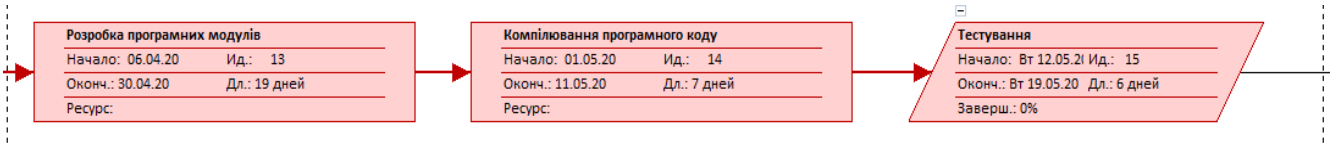
Б.6 – продовження мережевого графіку



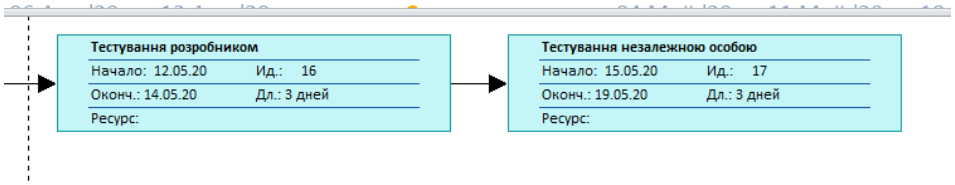
Б.7 – продовження мережевого графіку



Б.8 – продовження мережевого графіку



Б.9 – продовження мережевого графіку



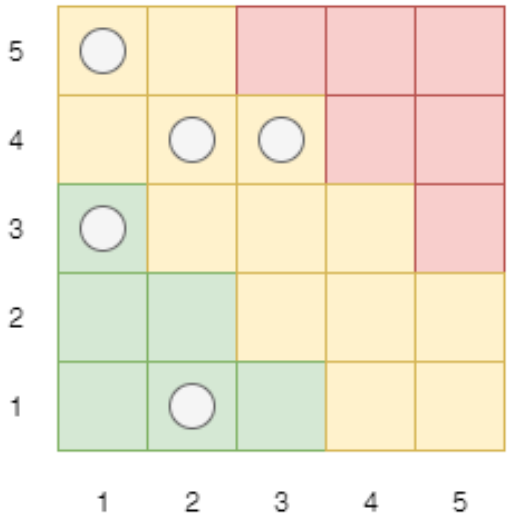
Б.10 – продовження мережевого графіку

Планування ризиків проекту. На сьогодні в управлінні проектами при строго встановленому бюджеті є необхідність формалізованого керування ризиками, що включає в себе планування управління ризиками та моніторинг ризиків під час виконання проекту. Ризики даного дипломного проекту представлено в таблиці Б.3.

На основі даних класифікації ризиків, була побудована матриця ризиків (рис.Б.11)

Таблиця Б.3 – класифікація ризиків

Назва ризику	Ймовірність	Вплив ризику
Технічні ризики (несправність устаткування, відсутність інтернету, ін.)	3	4
Некоректно та неповно складене тз	1	3
Технологічні ризики (ризики невірно обраного програмного забезпечення проекту, тощо)	2	4
Ризик трудових ресурсів (недостатня кваліфікація розробника, його хвороба і т.д.)	1	5
Некоректне тестування	2	1



Б.11 – матриця ризиків

Наступним етапом було визначення ступінь дії ризиків та рівні ризиків, виконано їх оцінку для кожного ризику в проекті. Результати розрахунків представлено в таблиці Б.4.

Таблиця Б.4 – визначення ступенів та рівнів ризиків

№	Назва ризику	Ймовірність ризику	Ранг ризику	Рівень ризику	Ступінь дії
1	Технічні ризики (несправність устаткування, відсутність інтернету, ін.)	3	12	Недопустимий	Істотний
2	Некоректно та неповно складене тз	1	3	Допустимий	Проігнорувати
3	Технологічні ризики (ризики невірно обраного програмного забезпечення проекту, тощо)	2	8	Оправданий	Незначний
4	Ризик трудових ресурсів (недостатня кваліфікація розробника, його хвороба і т.д.)	1	5	Оправданий	Незначний
5	Некоректне тестування	2	2	Допустимий	Проігнорувати

ДОДАТОК В
ФАЙЛИ КОДУ РЕАЛІЗАЦІЇ

Файл коду connutils.py

```
Import logging

Import pymysql.cursors
From pyqt5.qtwidgets import qmessagebox

Logging.getlogger().setlevel(logging.debug)


Def get_connection():
  connection = pymysql.connect(host='127.0.0.1',
                    user='mysql',
                    password='mysql',
                    db='gidravic',
                    charset='utf8mb4',
                    cursorclass= pymysql.cursors.dictcursor)
  return connection


Def insert_input(connection, input):
  # prepare sql query to insert a record into the database.
  sql = """insert into input (length , absolute_roughness, temperature , coef_local_resist)
          values ('%d', '%d', '%d', '%d');"""
  execute_query(connection, sql, input)


Def insert_result(connection, result, id_input):
  # prepare sql query to insert a record into the database.
  sql = """insert into result (id_input, lmbd, i, h1, h2, q, v, d, zz)
          values ('%s', '%d', '%d', '%d', '%d', '%d', '%d', '%d', '%d');"""
  data = (id_input, result[0], result[1], result[2], result[3], result[4], result[5], result[6],
result[7])
  execute_query(connection, sql, data)


Def execute_query(connection, sql, values):
  cursor = connection.cursor()
  try:
    # execute the sql command
    cursor.execute(sql % values)
    # commit your changes in the database
    connection.commit()
  except:
     # rollback in case there is any error
     #except exception as e: print(e)
     connection.rollback()
```

```
        Def execute_select(connection, sql):
          cursor = connection.cursor()
          try:
            # execute the sql command
            cursor.execute(sql)
            # commit your changes in the database
            connection.commit()
          except:
             # rollback in case there is any error
             connection.rollback()
return cursor
```

Файл коду press.py

```
        From pyqt5 import qtcore, qtgui, qtwidgets
        Import connutils

        Class ui_dialog2(object):

          def count2(self):
            connection = connutils.get_connection()
            cursor = connection.cursor()

            cursor.execute("select * from input order by input_id desc limit 1")
            records = cursor.fetchall()
            row = records[0]
            l = row["length"]
            ro = 1000
            e = row["absolute_roughness"]
            z = row["coef_local_resist"]

            cursor.execute("select * from result order by result_id desc limit 1")
            records2 = cursor.fetchall()
            row2 = records2[0]
            d = row2["d"]
            v = row2["v"]
            lmbd = 0.11*pow((e/d), 0.25)
            pl = lmbd*v*v*ro*l/(2*d)
            pm = z*v*v*ro/2
            p = pl+pm
            self.lineedit.settext(str(round(pl,5)))
            self.lineedit_2.settext(str(round(pm,2)))
            self.lineedit_3.settext(str(round(p,2)))
          def setupui(self, dialog):
            dialog.setobjectname("dialog")
            dialog.resize(580, 200)
            self.label = qtwidgets.qlabel(dialog)
            self.label.setgeometry(qtcore.qrect(180, 0, 200, 31))
            font = qtgui.qfont()
```

```
        font.setpointsize(11)
        font.setbold(true)
        font.setweight(75)
        self.label.setfont(font)
        self.label.setobjectname("label")
        self.label_2 = qtwidgets.qlabel(dialog)
        self.label_2.setgeometry(qtcore.qrect(10, 30, 400, 31))
        font = qtgui.qfont()
        font.setpointsize(11)
        self.label_2.setfont(font)
        self.label_2.setobjectname("label_2")
        self.label_3 = qtwidgets.qlabel(dialog)
        self.label_3.setgeometry(qtcore.qrect(10, 70, 400, 31))
        font = qtgui.qfont()
        font.setpointsize(11)
        self.label_3.setfont(font)
        self.label_3.setobjectname("label_3")
        self.label_4 = qtwidgets.qlabel(dialog)
        self.label_4.setgeometry(qtcore.qrect(10, 110, 400, 31))
        font = qtgui.qfont()
        font.setpointsize(11)
        self.label_4.setfont(font)
        self.label_4.setobjectname("label_4")
        self.lineedit = qtwidgets.qlineedit(dialog)
        self.lineedit.setgeometry(qtcore.qrect(400, 30, 170, 31))
        self.lineedit.setobjectname("lineedit")
        self.lineedit_2 = qtwidgets.qlineedit(dialog)
        self.lineedit_2.setgeometry(qtcore.qrect(400, 70, 170, 31))
        self.lineedit_2.setobjectname("lineedit_2")
        self.lineedit_3 = qtwidgets.qlineedit(dialog)
        self.lineedit_3.setgeometry(qtcore.qrect(400, 110, 170, 31))
        self.lineedit_3.setobjectname("lineedit_3")
        self.count_button1 = qtwidgets.qpushbutton(dialog)
        self.count_button1.setgeometry(qtcore.qrect(400, 150, 170, 40))
        self.count_button1.setobjectname("count_button1")
        self.count_button1.clicked.connect(self.count2)
        self.retranslateui(dialog)
        qtcore.qmetaobject.connectslotsbyname(dialog)

    def retranslateui(self, dialog):
        _translate = qtcore.qcoreapplication.translate
        dialog.setwindowtitle(_translate("dialog", "dialog"))
        self.label.settext(_translate("dialog", "витрати тиску"))
        self.label_2.settext(_translate("dialog", "витрати тиску на тертя, рл:"))
        self.label_3.settext(_translate("dialog", "витрати тиску в місцевих опорах, рм:"))
        self.label_4.settext(_translate("dialog", "втрати тиску на ділянках:"))
        self.count_button1.settext(_translate("dialog", "обчислити"))

If __name__ == "__main__":
    import sys
    app = qtwidgets.qapplication(sys.argv)
```

```
        dialog = qtwidgets.qmainwindow()
        ui = ui_dialog2()
        ui.setupui(dialog)
        dialog.show()
sys.exit(app.exec_())
```

Файл коду rosa_temp.py

```
From pyqt5 import qtcore, qtgui, qtwidgets
Import math
From pyqt5.qtwidgets import qmessagebox

Class ui_dialog(object):

  def count(self):
    try:
      t = float(self.lineedit.text())#температура
      rh = float(self.lineedit_2.text())#влажность
      a = 17.27#константа
      b = 237.7#константа
      f = a*t/(b+t) + math.log(rh/100)
      tp = b*f/(a-f)
      s = str(round(tp,2)) + " °c"
      self.label_res1.settext(s)
    except:
      msgbox = qmessagebox()
      msgbox.warning(self.centralwidget, 'помилка ',"не вірно введено дані!")



  def setupui(self, dialogtemperature):
    dialogtemperature.setobjectname("dialogtemperature")
    dialogtemperature.resize(430, 200)
    self.centralwidget = qtwidgets.qwidget(dialogtemperature)
    self.centralwidget.setobjectname("centralwidget")
    font = qtgui.qfont()
    font.setpointsize(12)
    dialogtemperature.setfont(font)
    self.label_temper = qtwidgets.qlabel(self.centralwidget)
    self.label_temper.setgeometry(qtcore.qrect(10, 10, 200, 30))
    font = qtgui.qfont()
    font.setpointsize(12)
    self.label_temper.setfont(font)
    self.label_temper.setobjectname("label_temper")
    self.label_vol = qtwidgets.qlabel(self.centralwidget)
    self.label_vol.setgeometry(qtcore.qrect(10, 50, 200, 30))
    font = qtgui.qfont()
    font.setpointsize(12)
    self.label_vol.setfont(font)
    self.label_vol.setobjectname("label_vol")
    self.lineedit = qtwidgets.qlineedit(self.centralwidget)
```

```
            self.lineedit.setgeometry(qtcore.qrect(210, 10, 200, 30))
            font = qtgui.qfont()
            font.setpointsize(12)
            self.lineedit.setfont(font)
            self.lineedit.setobjectname("lineedit")
            self.lineedit_2 = qtwidgets.qlineedit(self.centralwidget)
            self.lineedit_2.setgeometry(qtcore.qrect(210, 50, 200, 30))
            self.lineedit_2.setobjectname("lineedit_2")
            self.label_temp_ros = qtwidgets.qlabel(self.centralwidget)
            self.label_temp_ros.setgeometry(qtcore.qrect(10, 140, 350, 30))
            self.label_temp_ros.setobjectname("label_temp_ros")
            self.label_res1 = qtwidgets.qlabel(self.centralwidget)
            self.label_res1.setgeometry(qtcore.qrect(300, 140, 110, 30))
            self.label_res1.setobjectname("label_res1")
            self.count_button1 = qtwidgets.qpushbutton(self.centralwidget)
            self.count_button1.setgeometry(qtcore.qrect(250, 90, 160, 40))
            self.count_button1.setobjectname("count_button1")
            self.count_button1.clicked.connect(self.count)
            dialogtemperature.setcentralwidget(self.centralwidget)

            self.retranslateui(dialogtemperature)
            qtcore.qmetaobject.connectslotsbyname(dialogtemperature)

        def retranslateui(self, dialogtemperature):
            _translate = qtcore.qcoreapplication.translate
            dialogtemperature.setwindowtitle(_translate("dialogtemperature", "температура точки роси"))
            self.label_temper.settext(_translate("dialogtemperature", "температура, °c"))
            self.label_vol.settext(_translate("dialogtemperature", "вологість, %"))
            self.label_temp_ros.settext(_translate("dialogtemperature", "температура точки роси:"))
            self.label_res1.settext(_translate("dialogtemperature", "результат"))
            self.count_button1.settext(_translate("dialogtemperature", "обчислити"))

    If __name__ == "__main__":
        import sys
        app = qtwidgets.qapplication(sys.argv)
        dialogtemperature = qtwidgets.qmainwindow()
        ui = ui_dialog()
        ui.setupui(dialogtemperature)
        dialogtemperature.show()
sys.exit(app.exec_())
```

Файл коду gidr.py

```
From pyqt5 import qtcore, qtgui, qtwidgets
Import math

From calculator import *
From rosa_temp import *
From press import *
```

```
Import connutils

Import pandas as pd
Import xlsxwriter
From openpyxl import *
From docxtpl import docxtemplate
Import pymysql.cursors

From pyqt5.qtwidgets import qmessagebox

Class ui_mainwindow(object):

  def radioonclicked(self):
    if self.radiobutton.ischecked():
      self.lineedit_9.setenabled(true)
      self.lineedit_10.setenabled(false)
      self.lineedit_11.setenabled(false)
      self.lineedit_12.setenabled(true)
    elif self.radiobutton_2.ischecked():
      self.lineedit_9.setenabled(true)
      self.lineedit_10.setenabled(true)
      self.lineedit_11.setenabled(false)
      self.lineedit_12.setenabled(false)
    elif self.radiobutton_3.ischecked():
      self.lineedit_9.setenabled(false)
      self.lineedit_10.setenabled(true)
      self.lineedit_11.setenabled(true)
      self.lineedit_12.setenabled(false)
    elif self.radiobutton_4.ischecked():
      self.lineedit_9.setenabled(false)
      self.lineedit_10.setenabled(true)
      self.lineedit_11.setenabled(false)
      self.lineedit_12.setenabled(true)


  def savetoexcel(self):
    try:

      lmbd = float(self.lineedit_2.text())#коеффициента гидравлического
сопротивления
      i = float(self.lineedit_3.text())#гидравлического уклона
      zz = float(self.lineedit_6.text())#кинематической вязкости
      h1 = float(self.lineedit_8.text())#напор вначале трубы
      h2 = float(self.lineedit_9.text())#напор вконце трубы
      q = float(self.lineedit_10.text())#объем потока
      v = float(self.lineedit_11.text())#скорости потока
      d = float(self.lineedit_12.text())#диаметр трубы


      wb = load_workbook(filename='report.xlsx')
      ws = wb.worksheets[0]
```

```
                columnmn=ws['a']
                n = len(columnmn)
                row = n + 1
                ws.cell(row, column=1).value = lmbd
                ws.cell(row, column=2).value = i
                ws.cell(row, column=3).value = zz
                ws.cell(row, column=4).value = h1
                ws.cell(row, column=5).value = h2
                ws.cell(row, column=6).value = q
                ws.cell(row, column=7).value = v
                ws.cell(row, column=8).value = d

                wb.save(filename='report.xlsx')
            except:
                msgbox = qmessagebox()
                msgbox.warning(self.centralwidget, 'помилка ',"не вірно введено дані!")

        def savetoword(self):
            lmbd = self.lineedit_2.text()#коеффициента гидравлического сопротивления
            i = self.lineedit_3.text()#гидравлического уклона
            zz = self.lineedit_6.text()#кинематической вязкости
            h1 = self.lineedit_8.text()#напор вначале трубы
            h2 = self.lineedit_9.text()#напор вконце трубы
            q = self.lineedit_10.text()#объем потока
            v = self.lineedit_11.text()#скорости потока
            d = self.lineedit_12.text()#диаметр трубы


            doc = docxtemplate("report.docx")
            context_ew = { 'var_lmbd' : lmbd , 'var_i' : i, 'var_zz' : zz , 'var_h1' : h1, 'var_h2' : h2, 'var_q'
: q, 'var_v' : v, 'var_d' : d}
            doc.render(context_ew)
            doc.save("results.docx")


        def count (self):

            try:
                l = float(self.lineedit.text())#длина
                e = float(self.lineedit_4.text())#абсолютная шероховатость
                t = float(self.lineedit_5.text())#температура воды
                z = float(self.lineedit_7.text())#коеффициент местных сопротивлений
                connection = connutils.get_connection()
                cursor = connection.cursor()


                input = (l, e, t, z)
                connutils.insert_input(connection, input)

                h1 = float(self.lineedit_8.text())#напор вначале трубы
```

```
if self.radiobutton.ischecked():
    #вычисление гидравлического уклона
    h2 = float(self.lineedit_9.text())#напор в конце трубы
    i = (h1-h2)/l #гидравлический уклон
    self.lineedit_3.settext(str(i))
    #----------------------------
    #вычисление коэффициента гидравлического сопротивления
    d = float(self.lineedit_12.text())#диаметр трубы
    lmbd = 0.11*pow((e/d), 0.25)
    self.lineedit_2.settext(str(round(lmbd, 4)))
    #----------------------------
    #вычисление кинематической вязкости
    zz = 0.0178/10000*(1 + 0.0337*t + 0.000221*t*t)
    self.lineedit_6.settext(str(round(zz,7)))
    #----------------------------
    #вычисление скорости потока
    re = 2320
    v = re*zz/d
    self.lineedit_11.settext(str(round(100*v,5)))
    #----------------------------
    #вычисление объема потока
    q = 3.14 * pow(d, 2.0) * v/4
    self.lineedit_10.settext(str(round(q,7)))

    result = (lmbd, i, h1, h2, q, v, d, zz)

if self.radiobutton_2.ischecked():
    #----------------------------
    #вычисление гидравлического уклона
    h2 = float(self.lineedit_9.text())#напор в конце трубы
    i = (h1-h2)/l #гидравлический уклон
    self.lineedit_3.settext(str(i))
    #----------------------------
    #вычисление кинематической вязкости
    zz = 0.0178/10000*(1 + 0.0337*t + 0.000221*t*t)
    self.lineedit_6.settext(str(round(zz,7)))
    #----------------------------
    #вычисление коэффициента гидравлического сопротивления
    re = 2320
    lmbd = 0.11*pow((68/re + e), 0.25)
    self.lineedit_2.settext(str(round(lmbd, 4)))
    #----------------------------
    #вычисление скорости потока
    q = float(self.lineedit_10.text())
    v = math.pow((16*q*9.8*9.8*abs(h2-h1)/(lmbd*lmbd*l*l*3.14)),0.2)
    self.lineedit_11.settext(str(round(100*v,5)))
    #----------------------------
    #вычисление диаметра трубы
    d = math.sqrt(4*q/(3.14*v))
    self.lineedit_12.settext(str(round(d, 2)))
```

```
    result = (lmbd, i, h1, h2, q, v, d, zz)


if self.radiobutton_3.ischecked():
    #----------------------------
    #вычисление кинематической вязкости
    zz = 0.0178/10000*(1 + 0.0337*t + 0.000221*t*t)
    self.lineedit_6.settext(str(round(zz,7)))
    #----------------------------
    #вычисление диаметра трубы
    q = float(self.lineedit_10.text())
    v =  float(self.lineedit_11.text())
    d = math.sqrt((4*q)/(4*v))
    self.lineedit_12.settext(str(round(d,2)))
    #----------------------------
    #вычисление коеффициента гидравлического сопротивления
    lmbd = 0.11*pow((e/d), 0.25)
    self.lineedit_2.settext(str(round(lmbd, 4)))
    #----------------------------
    #вычисление напора в конце трубы н2
    h = lmbd*l*math.pow(v,2)/(d*2*9.8)
    h2 = h1-h#напор в конце трубы
    self.lineedit_9.settext(str(round(h2,2)))
    #----------------------------
    #вычисление гидравлического уклона
    i = (h1-h2)/l #гидравлический уклон
    self.lineedit_3.settext(str(i))

    result = (lmbd, i, h1, h2, q, v, d, zz)

if self.radiobutton_4.ischecked():
    #----------------------------
    #вычисление кинематической вязкости
    zz = 0.0178/10000*(1 + 0.0337*t + 0.000221*t*t)
    self.lineedit_6.settext(str(round(zz,7)))
    #----------------------------
    #вычисление скорости потока
    q = float(self.lineedit_10.text())
    d =  float(self.lineedit_12.text())
    v = q*4/(3.14*math.pow(d,2))
    #----------------------------
    #вычисление коеффициента гидравлического сопротивления
    lmbd = 0.11*pow((e/d), 0.25)
    self.lineedit_2.settext(str(round(lmbd, 4)))
    #----------------------------
    #вычисление напора в конце трубы н2
    h = lmbd*l*math.pow(v,2)/(d*2*9.8)
    h2 = h1-h#напор в конце трубы
    self.lineedit_9.settext(str(round(h2,2)))
    #----------------------------
    #вычисление гидравлического уклона
```

```
                i = (h1-h2)/l #гидравлический уклон
                self.lineedit_3.settext(str(i))

                result = (lmbd, i, h1, h2, q, v, d, zz)

            result = (float(self.lineedit_2.text()), float(self.lineedit_3.text()),
float(self.lineedit_8.text()), float(self.lineedit_9.text()), float(self.lineedit_10.text()),
float(self.lineedit_11.text()), float(self.lineedit_12.text()), float(self.lineedit_6.text()))
            print (result)
            sql =("select input_id from input order by input_id desc limit 1")
            cursor.execute(sql)
            id = cursor.fetchone()
            print(id)
            connutils.insert_result(connection, result, id['input_id'])

          except:
            msgbox = qmessagebox()
            msgbox.warning(self.centralwidget, 'помилка ',"не вірно введено дані!")

        def openrosa(self):
          self.window = qtwidgets.qmainwindow()
          self.ui = ui_dialog()
          self.ui.setupui(self.window)
          self.window.show()

        def openpress(self):
          self.window = qtwidgets.qmainwindow()
          self.ui = ui_dialog2()
          self.ui.setupui(self.window)
          self.window.show()

        def opencalculator(self):
          dialog = calculator(self)
          self.dialogs.append(dialog)
          dialog.show()

        def openconverter(self):
          self.window = qtwidgets.qmainwindow()
          self.ui = ui_dialog3()
          self.ui.setupui(self.window)
          self.window.show()

        def setupui(self, mainwindow):
          mainwindow.setobjectname("mainwindow")
          mainwindow.resize(1010, 700)
          self.centralwidget = qtwidgets.qwidget(mainwindow)
          self.centralwidget.setobjectname("centralwidget")
          self.groupbox = qtwidgets.qgroupbox(self.centralwidget)
          self.groupbox.setgeometry(qtcore.qrect(10, 10, 990, 250))
          font = qtgui.qfont()
          font.setpointsize(10)
```

```
self.groupbox.setfont(font)
self.groupbox.settitle("")
self.groupbox.setobjectname("groupbox")
self.label = qtwidgets.qlabel(self.groupbox)
self.label.setgeometry(qtcore.qrect(10, 10, 250, 30))
font = qtgui.qfont()
font.setpointsize(10)
self.label.setfont(font)
self.label.setobjectname("label")


self.combobox_3 = qtwidgets.qcombobox(self.groupbox)
self.combobox_3.setgeometry(qtcore.qrect(270, 10, 100, 30))
self.combobox_3.setobjectname("combobox_3")

self.combobox_3.additems(["м3/с", "л/с", "м3/ч"])

self.label_2 = qtwidgets.qlabel(self.groupbox)
self.label_2.setgeometry(qtcore.qrect(10, 90, 250, 30))
self.label_2.setobjectname("label_2")
self.lineedit = qtwidgets.qlineedit(self.groupbox)
self.lineedit.setgeometry(qtcore.qrect(270, 90, 100, 30))
self.lineedit.setobjectname("lineedit")

self.label_3 = qtwidgets.qlabel(self.groupbox)
self.label_3.setgeometry(qtcore.qrect(430, 10, 250, 30))
self.label_3.setobjectname("label_3")
self.lineedit_2 = qtwidgets.qlineedit(self.groupbox)
self.lineedit_2.setenabled(false)
#self.lineedit_2.settext("0.1153")

self.lineedit_2.setgeometry(qtcore.qrect(700, 10, 230, 30))
self.lineedit_2.setobjectname("lineedit_2")
self.label_4 = qtwidgets.qlabel(self.groupbox)
self.label_4.setgeometry(qtcore.qrect(10, 50, 280, 30))
self.label_4.setobjectname("label_4")
self.combobox_2 = qtwidgets.qcombobox(self.groupbox)
self.combobox_2.setgeometry(qtcore.qrect(270, 50, 100, 30))
self.combobox_2.setobjectname("combobox_2")
self.combobox_2.additems(["м", "атм.", "па"])

self.label_5 = qtwidgets.qlabel(self.groupbox)
self.label_5.setgeometry(qtcore.qrect(430, 50, 290, 30))
self.label_5.setobjectname("label_5")
self.lineedit_3 = qtwidgets.qlineedit(self.groupbox)
self.lineedit_3.setenabled(false)
self.lineedit_3.setgeometry(qtcore.qrect(700, 50, 230, 30))
self.lineedit_3.setobjectname("lineedit_3")
#self.lineedit_3.settext("0.005")

self.label_6 = qtwidgets.qlabel(self.groupbox)
```

```
self.label_6.setgeometry(qtcore.qrect(10, 130, 250, 30))
self.label_6.setobjectname("label_6")
self.lineedit_4 = qtwidgets.qlineedit(self.groupbox)
self.lineedit_4.setgeometry(qtcore.qrect(270, 130, 100, 30))
self.lineedit_4.setobjectname("lineedit_4")
self.lineedit_4.settext("0.03")

self.label_7 = qtwidgets.qlabel(self.groupbox)
self.label_7.setgeometry(qtcore.qrect(10, 170, 250, 30))
self.label_7.setobjectname("label_7")
self.lineedit_5 = qtwidgets.qlineedit(self.groupbox)
self.lineedit_5.setgeometry(qtcore.qrect(270, 170, 100, 30))
self.lineedit_5.setobjectname("lineedit_5")
self.label_8 = qtwidgets.qlabel(self.groupbox)
self.label_8.setgeometry(qtcore.qrect(430, 90, 290, 30))
self.label_8.setobjectname("label_8")
self.lineedit_6 = qtwidgets.qlineedit(self.groupbox)
self.lineedit_6.setenabled(false)
#self.lineedit_6.settext("0.0000005")

self.lineedit_6.setgeometry(qtcore.qrect(700, 90, 230, 30))
self.lineedit_6.setobjectname("lineedit_6")
self.label_9 = qtwidgets.qlabel(self.groupbox)
self.label_9.setgeometry(qtcore.qrect(10, 210, 250, 30))
self.label_9.setobjectname("label_9")
self.lineedit_7 = qtwidgets.qlineedit(self.groupbox)
self.lineedit_7.setgeometry(qtcore.qrect(270, 210, 100, 30))
self.lineedit_7.setobjectname("lineedit_7")
self.lineedit_7.settext("1")

self.groupbox_2 = qtwidgets.qgroupbox(self.centralwidget)
self.groupbox_2.setgeometry(qtcore.qrect(10, 270, 990, 300))
font = qtgui.qfont()
font.setpointsize(10)
self.groupbox_2.setfont(font)
self.groupbox_2.setobjectname("groupbox_2")
self.label_10 = qtwidgets.qlabel(self.groupbox_2)
self.label_10.setgeometry(qtcore.qrect(10, 30, 250, 30))
self.label_10.setobjectname("label_10")
self.label_11 = qtwidgets.qlabel(self.groupbox_2)
self.label_11.setgeometry(qtcore.qrect(10, 70, 250, 30))
self.label_11.setobjectname("label_11")
self.label_12 = qtwidgets.qlabel(self.groupbox_2)
self.label_12.setgeometry(qtcore.qrect(10, 120, 250, 30))
self.label_12.setobjectname("label_12")
self.label_13 = qtwidgets.qlabel(self.groupbox_2)
self.label_13.setgeometry(qtcore.qrect(10, 160, 250, 30))
self.label_13.setobjectname("label_13")
self.label_14 = qtwidgets.qlabel(self.groupbox_2)
self.label_14.setgeometry(qtcore.qrect(10, 200, 250, 30))
self.label_14.setobjectname("label_14")
```

```
    self.lineedit_8 = qtwidgets.qlineedit(self.groupbox_2)
    self.lineedit_8.setgeometry(qtcore.qrect(270, 30, 100, 30))
    self.lineedit_8.setobjectname("lineedit_8")
    self.lineedit_9 = qtwidgets.qlineedit(self.groupbox_2)
    self.lineedit_9.setgeometry(qtcore.qrect(270, 70, 100, 30))
    self.lineedit_9.setobjectname("lineedit_9")
    self.lineedit_10 = qtwidgets.qlineedit(self.groupbox_2)
    self.lineedit_10.setgeometry(qtcore.qrect(270, 120, 100, 30))
    self.lineedit_10.setobjectname("lineedit_10")
#   self.lineedit_10.setenabled(false)
#   self.lineedit_10.settext("0.0000824")

    self.lineedit_11 = qtwidgets.qlineedit(self.groupbox_2)
    self.lineedit_11.setgeometry(qtcore.qrect(270, 160, 100, 30))
    self.lineedit_11.setobjectname("lineedit_11")
#   self.lineedit_11.setenabled(false)
#   self.lineedit_11.settext("0.02916")

    self.lineedit_12 = qtwidgets.qlineedit(self.groupbox_2)
    self.lineedit_12.setgeometry(qtcore.qrect(270, 200, 100, 30))
    self.lineedit_12.setobjectname("lineedit_12")
    self.label_15 = qtwidgets.qlabel(self.groupbox_2)
    self.label_15.setgeometry(qtcore.qrect(10, 90, 200, 30))
    self.label_15.setobjectname("label_15")
    self.label_16 = qtwidgets.qlabel(self.groupbox_2)
    self.label_16.setgeometry(qtcore.qrect(400, 30, 500, 30))
    self.label_16.setobjectname("label_16")
    self.radiobutton = qtwidgets.qradiobutton(self.groupbox_2)
    self.radiobutton.setgeometry(qtcore.qrect(400, 70, 600, 30))
    self.radiobutton.setobjectname("radiobutton")
    self.radiobutton.toggled.connect(self.radioonclicked)

    self.radiobutton_2 = qtwidgets.qradiobutton(self.groupbox_2)
    self.radiobutton_2.setgeometry(qtcore.qrect(400, 110, 600, 30))
    self.radiobutton_2.setobjectname("radiobutton_2")
    self.radiobutton_2.toggled.connect(self.radioonclicked)

    self.radiobutton_3 = qtwidgets.qradiobutton(self.groupbox_2)
    self.radiobutton_3.setgeometry(qtcore.qrect(400, 150, 600, 30))
    self.radiobutton_3.setobjectname("radiobutton_3")
    self.radiobutton_3.toggled.connect(self.radioonclicked)

    self.radiobutton_4 = qtwidgets.qradiobutton(self.groupbox_2)
    self.radiobutton_4.setgeometry(qtcore.qrect(400, 200, 600, 30))
    self.radiobutton_4.setobjectname("radiobutton_4")
    self.radiobutton_4.toggled.connect(self.radioonclicked)

    self.pushbutton = qtwidgets.qpushbutton(self.centralwidget)
    self.pushbutton.setgeometry(qtcore.qrect(750, 580, 130, 40))
    self.pushbutton.setobjectname("pushbutton")
    self.pushbutton.clicked.connect(self.count)
```

```
mainwindow.setcentralwidget(self.centralwidget)
self.menubar = qtwidgets.qmenubar(mainwindow)
self.menubar.setgeometry(qtcore.qrect(0, 0, 651, 18))
self.menubar.setobjectname("menubar")
self.menu = qtwidgets.qmenu(self.menubar)
self.menu.setobjectname("menu")
self.menu_2 = qtwidgets.qmenu(self.menubar)
self.menu_2.setobjectname("menu_2")
#self.menu_2.addactiont(self.openrosa)
mainwindow.setmenubar(self.menubar)
self.statusbar = qtwidgets.qstatusbar(mainwindow)
self.statusbar.setobjectname("statusbar")
mainwindow.setstatusbar(self.statusbar)
self.action = qtwidgets.qaction(mainwindow)
self.action.setobjectname("action")
self.action.triggered.connect(self.opencalculator)
self.dialogs = list()


self.action_3 = qtwidgets.qaction(mainwindow)
self.action_3.setobjectname("action_3")
self.action_5 = qtwidgets.qaction(mainwindow)
self.action_5.setobjectname("action_5")
self.action_5.triggered.connect(self.openrosa)
self.action_7 = qtwidgets.qaction(mainwindow)
self.action_7.setobjectname("action_7")
self.action_7.triggered.connect(self.openpress)
self.action_9 = qtwidgets.qaction(mainwindow)
self.action_9.setobjectname("action_9")
self.action_9.triggered.connect(self.savetoexcel)
self.action_11 = qtwidgets.qaction(mainwindow)
self.action_11.setobjectname("action_11")
self.action_11.triggered.connect(self.savetoword)
self.menu.addaction(self.action)
self.menu.addseparator()
self.menu.addaction(self.action_3)
self.menu.addseparator()
self.menu.addaction(self.action_5)
self.menu.addseparator()
self.menu.addaction(self.action_7)
self.menu_2.addaction(self.action_9)
self.menu_2.addseparator()
self.menu_2.addaction(self.action_11)
self.menu_2.addseparator()
self.menubar.addaction(self.menu.menuaction())
self.menubar.addaction(self.menu_2.menuaction())

self.retranslateui(mainwindow)
qtcore.qmetaobject.connectslotsbyname(mainwindow)
```

```
        def retranslateui(self, mainwindow):
            _translate = qtcore.qcoreapplication.translate
            mainwindow.setwindowtitle(_translate("mainwindow", "gidravlika"))
            self.label.settext(_translate("mainwindow", "одиниця виміру об\'єму"))
            self.label_2.settext(_translate("mainwindow", "довжина, м"))
            self.label_3.settext(_translate("mainwindow", "коеф-т гідравлічного опору"))
            self.label_4.settext(_translate("mainwindow", "одиниця виміру тиску"))
            self.label_5.settext(_translate("mainwindow", "гідравлічний уклон"))
            self.label_6.settext(_translate("mainwindow", "абсолютна шорсткість, мм"))
            self.label_7.settext(_translate("mainwindow", "температура води,°c"))
            self.label_8.settext(_translate("mainwindow", "кінематична в\'язкість,(м2/с)"))
            self.label_9.settext(_translate("mainwindow", "коефіцієнт місцевих опорів, ξ"))
            self.groupbox_2.settitle(_translate("mainwindow", "варіанти вихідних даних для розрахунку"))
            self.label_10.settext(_translate("mainwindow", "напір на початку труби h1"))
            self.label_11.settext(_translate("mainwindow", "напір в кінці труби h2"))
            self.label_12.settext(_translate("mainwindow", "обсяг води, що проходить"))
            self.label_13.settext(_translate("mainwindow", "швидкість води v, м/с"))
            self.label_14.settext(_translate("mainwindow", "діаметр труби, мм"))
            self.label_15.settext(_translate("mainwindow", "через перетин труби"))
            self.label_16.settext(_translate("mainwindow", "у всіх варіантах h1 є вхідним для розрахунку"))
            self.radiobutton.settext(_translate("mainwindow", "h2 і діаметр труби вихідні дані для розрахунку"))
            self.radiobutton_2.settext(_translate("mainwindow", "h2 і обсяг води q вихідні дані для розрахунку"))
            self.radiobutton_3.settext(_translate("mainwindow", "швидкість води v і обсяг води q вхідні дані для розрахунку"))
            self.radiobutton_4.settext(_translate("mainwindow", "діаметр і обсяг води q вихідні дані для розрахунку"))
            self.pushbutton.settext(_translate("mainwindow", "обчислити"))
            self.menu.settitle(_translate("mainwindow", "інструменти"))
            self.menu_2.settitle(_translate("mainwindow", "зберегти звіт"))
            self.action.settext(_translate("mainwindow", "калькулятор"))
            self.action_3.settext(_translate("mainwindow", "конвертер фізичних величин"))
            self.action_5.settext(_translate("mainwindow", "температура точки роси"))
            self.action_7.settext(_translate("mainwindow", "витрати тиску"))
            self.action_9.settext(_translate("mainwindow", "excel - файл"))
            self.action_11.settext(_translate("mainwindow", "word - файл"))


    If __name__ == "__main__":
        import sys
        app = qtwidgets.qapplication(sys.argv)
        mainwindow = qtwidgets.qmainwindow()
        ui = ui_mainwindow()
        ui.setupui(mainwindow)
        mainwindow.show()
sys.exit(app.exec_())
```

Файл коду calculator.py

```
Import math

From pyqt5.qtcore import qt
From pyqt5.qtwidgets import (qapplication, qgridlayout, qlayout, qlineedit,
      qsizepolicy, qtoolbutton, qwidget)


Class button(qtoolbutton):
   def __init__(self, text, parent=none):
      super(button, self).__init__(parent)

      self.setsizepolicy(qsizepolicy.expanding, qsizepolicy.preferred)
      self.settext(text)

   def sizehint(self):
      size = super(button, self).sizehint()
      size.setheight(size.height() + 20)
      size.setwidth(max(size.width(), size.height()))
      return size


Class calculator(qwidget):
   numdigitbuttons = 10

   def __init__(self, parent=none):
      super(calculator, self).__init__(parent)

      self.pendingadditiveoperator = ''
      self.pendingmultiplicativeoperator = ''

      self.suminmemory = 0.0
      self.sumsofar = 0.0
      self.factorsofar = 0.0
      self.waitingforoperand = true

      self.display = qlineedit('0')
      self.display.setreadonly(true)
      self.display.setalignment(qt.alignright)
      self.display.setmaxlength(15)

      font = self.display.font()
      font.setpointsize(font.pointsize() + 8)
      self.display.setfont(font)

      self.digitbuttons = []

      for i in range(calculator.numdigitbuttons):
         self.digitbuttons.append(self.createbutton(str(i),
               self.digitclicked))

      self.pointbutton = self.createbutton(".", self.pointclicked)
```

```
self.changesignbutton = self.createbutton(u"\n{plus-minus sign}",
    self.changesignclicked)

self.backspacebutton = self.createbutton("backspace",
    self.backspaceclicked)
self.clearbutton = self.createbutton("clear", self.clear)
self.clearallbutton = self.createbutton("clear all", self.clearall)

self.clearmemorybutton = self.createbutton("mc", self.clearmemory)
self.readmemorybutton = self.createbutton("mr", self.readmemory)
self.setmemorybutton = self.createbutton("ms", self.setmemory)
self.addtomemorybutton = self.createbutton("m+", self.addtomemory)

self.divisionbutton = self.createbutton(u"\n{division sign}",
    self.multiplicativeoperatorclicked)
self.timesbutton = self.createbutton(u"\n{multiplication sign}",
    self.multiplicativeoperatorclicked)
self.minusbutton = self.createbutton("-", self.additiveoperatorclicked)
self.plusbutton = self.createbutton("+", self.additiveoperatorclicked)

self.squarerootbutton = self.createbutton("sqrt",
    self.unaryoperatorclicked)
self.powerbutton = self.createbutton(u"x\n{superscript two}",
    self.unaryoperatorclicked)
self.reciprocalbutton = self.createbutton("1/x",
    self.unaryoperatorclicked)
self.equalbutton = self.createbutton("=", self.equalclicked)

mainlayout = qgridlayout()
mainlayout.setsizeconstraint(qlayout.setfixedsize)

mainlayout.addwidget(self.display, 0, 0, 1, 6)
mainlayout.addwidget(self.backspacebutton, 1, 0, 1, 2)
mainlayout.addwidget(self.clearbutton, 1, 2, 1, 2)
mainlayout.addwidget(self.clearallbutton, 1, 4, 1, 2)

mainlayout.addwidget(self.clearmemorybutton, 2, 0)
mainlayout.addwidget(self.readmemorybutton, 3, 0)
mainlayout.addwidget(self.setmemorybutton, 4, 0)
mainlayout.addwidget(self.addtomemorybutton, 5, 0)

for i in range(1, calculator.numdigitbuttons):
   row = ((9 - i) / 3) + 2
   column = ((i - 1) % 3) + 1
   mainlayout.addwidget(self.digitbuttons[i], row, column)

mainlayout.addwidget(self.digitbuttons[0], 5, 1)
mainlayout.addwidget(self.pointbutton, 5, 2)
mainlayout.addwidget(self.changesignbutton, 5, 3)

mainlayout.addwidget(self.divisionbutton, 2, 4)
```

```
    mainlayout.addwidget(self.timesbutton, 3, 4)
    mainlayout.addwidget(self.minusbutton, 4, 4)
    mainlayout.addwidget(self.plusbutton, 5, 4)

    mainlayout.addwidget(self.squarerootbutton, 2, 5)
    mainlayout.addwidget(self.powerbutton, 3, 5)
    mainlayout.addwidget(self.reciprocalbutton, 4, 5)
    mainlayout.addwidget(self.equalbutton, 5, 5)
    self.setlayout(mainlayout)

    self.setwindowtitle("калькулятор")

def digitclicked(self):
    clickedbutton = self.sender()
    digitvalue = int(clickedbutton.text())

    if self.display.text() == '0' and digitvalue == 0.0:
        return

    if self.waitingforoperand:
        self.display.clear()
        self.waitingforoperand = false

    self.display.settext(self.display.text() + str(digitvalue))

def unaryoperatorclicked(self):
    clickedbutton = self.sender()
    clickedoperator = clickedbutton.text()
    operand = float(self.display.text())

    if clickedoperator == "sqrt":
        if operand < 0.0:
            self.abortoperation()
            return

        result = math.sqrt(operand)
    elif clickedoperator == u"x\n{superscript two}":
        result = math.pow(operand, 2.0)
    elif clickedoperator == "1/x":
        if operand == 0.0:
            self.abortoperation()
            return

        result = 1.0 / operand

    self.display.settext(str(result))
    self.waitingforoperand = true

def additiveoperatorclicked(self):
    clickedbutton = self.sender()
    clickedoperator = clickedbutton.text()
```

```
    operand = float(self.display.text())

    if self.pendingmultiplicativeoperator:
        if not self.calculate(operand, self.pendingmultiplicativeoperator):
            self.abortoperation()
            return

        self.display.settext(str(self.factorsofar))
        operand = self.factorsofar
        self.factorsofar = 0.0
        self.pendingmultiplicativeoperator = ''

    if self.pendingadditiveoperator:
        if not self.calculate(operand, self.pendingadditiveoperator):
            self.abortoperation()
            return

        self.display.settext(str(self.sumsofar))
    else:
        self.sumsofar = operand

    self.pendingadditiveoperator = clickedoperator
    self.waitingforoperand = true

def multiplicativeoperatorclicked(self):
    clickedbutton = self.sender()
    clickedoperator = clickedbutton.text()
    operand = float(self.display.text())

    if self.pendingmultiplicativeoperator:
        if not self.calculate(operand, self.pendingmultiplicativeoperator):
            self.abortoperation()
            return

        self.display.settext(str(self.factorsofar))
    else:
        self.factorsofar = operand

    self.pendingmultiplicativeoperator = clickedoperator
    self.waitingforoperand = true

def equalclicked(self):
    operand = float(self.display.text())

    if self.pendingmultiplicativeoperator:
        if not self.calculate(operand, self.pendingmultiplicativeoperator):
            self.abortoperation()
            return

        operand = self.factorsofar
        self.factorsofar = 0.0
```

```
        self.pendingmultiplicativeoperator = ''

    if self.pendingadditiveoperator:
        if not self.calculate(operand, self.pendingadditiveoperator):
            self.abortoperation()
            return

        self.pendingadditiveoperator = ''
    else:
        self.sumsofar = operand

    self.display.settext(str(self.sumsofar))
    self.sumsofar = 0.0
    self.waitingforoperand = true

def pointclicked(self):
    if self.waitingforoperand:
        self.display.settext('0')

    if "." not in self.display.text():
        self.display.settext(self.display.text() + ".")

    self.waitingforoperand = false

def changesignclicked(self):
    text = self.display.text()
    value = float(text)

    if value > 0.0:
        text = "-" + text
    elif value < 0.0:
        text = text[1:]

    self.display.settext(text)

def backspaceclicked(self):
    if self.waitingforoperand:
        return

    text = self.display.text()[:-1]
    if not text:
        text = '0'
        self.waitingforoperand = true

    self.display.settext(text)

def clear(self):
    if self.waitingforoperand:
        return

    self.display.settext('0')
```

```
        self.waitingforoperand = true

    def clearall(self):
        self.sumsofar = 0.0
        self.factorsofar = 0.0
        self.pendingadditiveoperator = ''
        self.pendingmultiplicativeoperator = ''
        self.display.settext('0')
        self.waitingforoperand = true

    def clearmemory(self):
        self.suminmemory = 0.0

    def readmemory(self):
        self.display.settext(str(self.suminmemory))
        self.waitingforoperand = true

    def setmemory(self):
        self.equalclicked()
        self.suminmemory = float(self.display.text())

    def addtomemory(self):
        self.equalclicked()
        self.suminmemory += float(self.display.text())

    def createbutton(self, text, member):
        button = button(text)
        button.clicked.connect(member)
        return button

    def abortoperation(self):
        self.clearall()
        self.display.settext("####")

    def calculate(self, rightoperand, pendingoperator):
        if pendingoperator == "+":
            self.sumsofar += rightoperand
        elif pendingoperator == "-":
            self.sumsofar -= rightoperand
        elif pendingoperator == u"\n{multiplication sign}":
            self.factorsofar *= rightoperand
        elif pendingoperator == u"\n{division sign}":
            if rightoperand == 0.0:
                return false

            self.factorsofar /= rightoperand

        return true


If __name__ == '__main__':
```

```
        import sys

        app = qapplication(sys.argv)
        calc = calculator()
        calc.show()
sys.exit(app.exec_())
```

ДОДАТОК Г

ІНСТРУКЦІЯ КОРИСТУВАЧА

1. Запуск додатку

Для коректного використання інформаційної технології користувачу потрібно мати повний пакет файлів (gidr.exe, report.xlsx, report.docx). Також користувачу потрібно впевнитись, що всі необхідні файли знаходяться в одній папці з виконуваним файлом додатку, існує інтернет підключення, а також відповідна база даних gidravic.sql додана до субд.

2. Виконання програми

Після запуску додатку зявляється основне діалогове вікно (рис.Г.1). Де за замовчуванням користувачу пропонується використовувати для розрахунків абсолютну шорсткість зі значенням 0.03 та коефіцієнт місцевих опорів рівний 1.

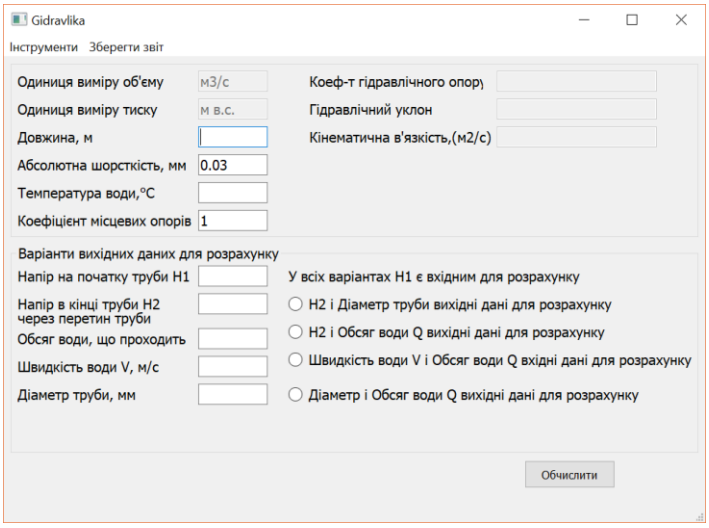


Рисунок Г.1 – головне вікно додатку

Для виконання розрахунків користувачу пропонується ввсети наявні в нього дані та обрати один з 4 варіантів вхідних даних, що він має. В залежності від вхідних даних програмний додаток буде виконувати розрахунки параметрів, яких бракує. Далі натиснути кнопку «обчислити». Як результат на головному вікну користувачу відобразяться всі отримані результати (рис. Г.2), а також дані заносяться до бази даних.

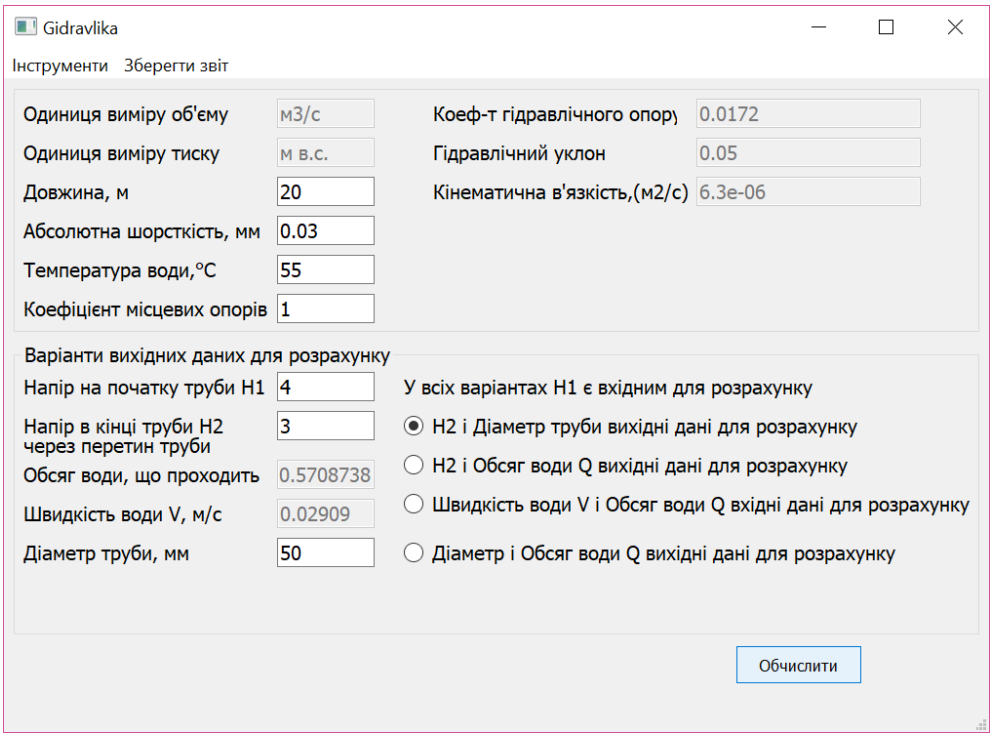


Рисунок Г.2 – результат виконання розрахунків

У разі, якщо користувач ввів некоректні дані, або виникла помилка при виконанні розрахунків, користувач отримує відповідне повідомнення про помилку (рис. Г.3)

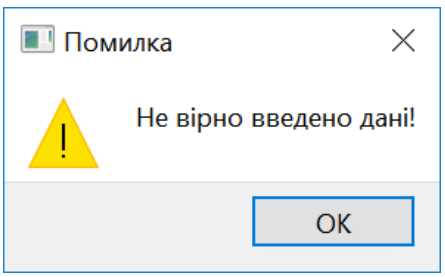


Рисунок Г.3 – повідомлення про помилку

Далі для виконання розрахунків витрат тиску в залежності від введених даних необхідно обрати пункт меню «інструменти» –> «витрати тиску» та натиснути кнопку «обчислити». Для виконання даних розрахунків програмний додаток використовує останні розраховані дані з головного вікна додатку. Результат виконання таких розрахунків представлено на рисунку Г.4.

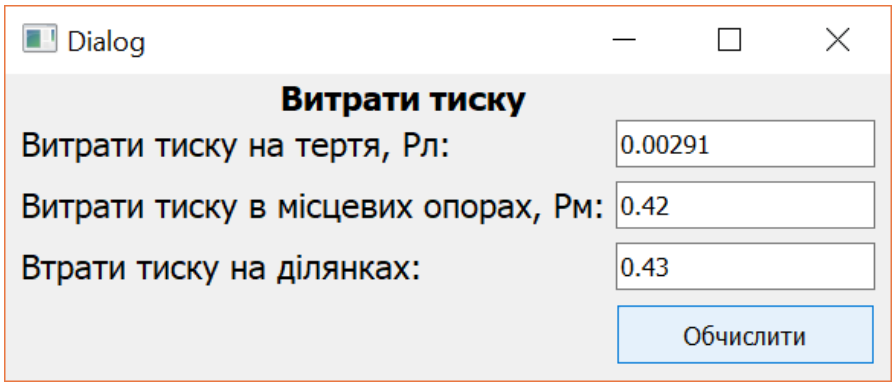


Рисунок Г.4 – результат виконання розрахунку втрат тиску

Для визначення температури точки роси необхідно обрати пункт меню «інструменти» –> «температура точки роси».відкриється відповідне діалогове вікно, до якого треба внести тепературу та вологість повітря, натиснут кнопку «обчислити». Результат виконання таких обчислень представлено на рисунку г.5. При виникненні помилки обчислення користувач отримує відповідне повідомлення.

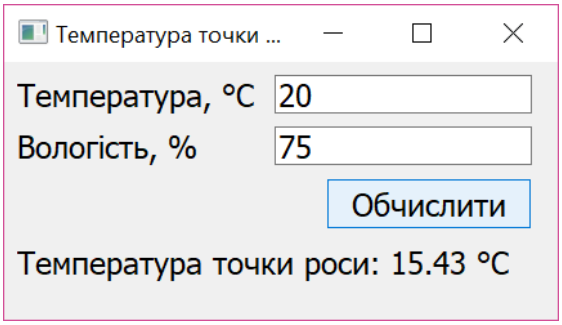


Рисунок Г.5 – результат виконання розрахунку температури точки роси

Як допоміжний інструментарій користувачу надається можливість використовувати каль кулятор, доступ до якого можна отримати, натиснувши пункт меню «інструменти» –> «калькулятор». Функціонал даний калькулятор має точно як звичайний. На рисунку Г.6 представлено результат виконання множення.

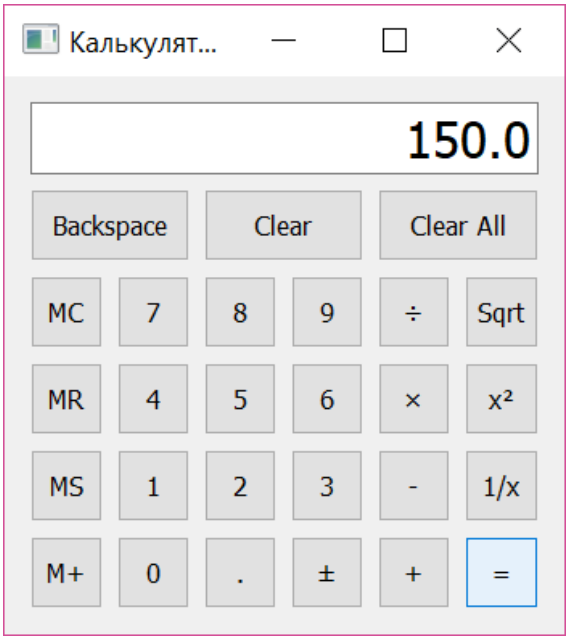
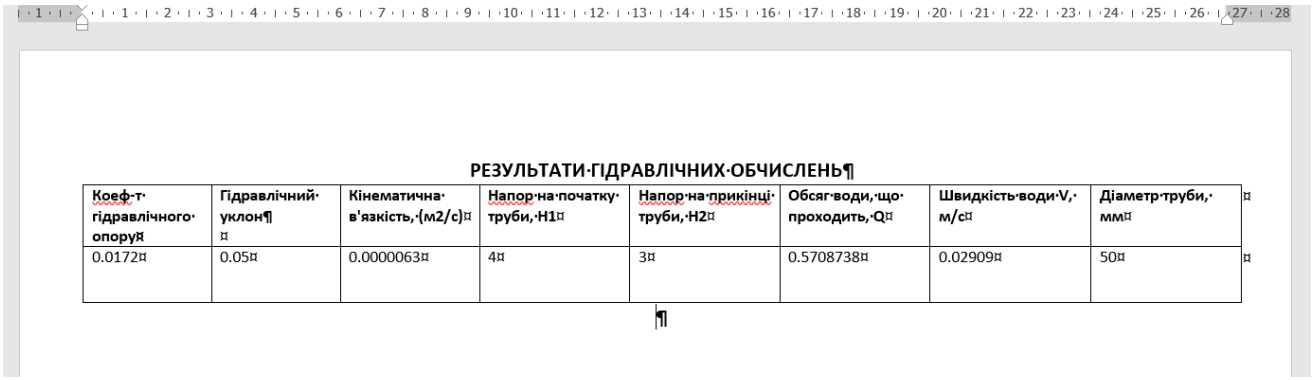


Рисунок Г.6 – результат виконання множення

Далі для збереження отриманих результатів виконання розподілених розрахунків користувач має можливість обрати варіант збереження звіту, натиснувши пункт меню «зберегти звіт». Результат зберігання звіту, натиснувши пункт меню «excel-файл», представлено на рисунку Г.7. Результат зберігання звіту, натиснувши пункт меню «word-файл», представлено на рисунку Г.8.

	A	B	C	D	E	F	G	H
1	Коеф-т гідравлічного опору	Гідравлічний уклон	Кінематична в'язкість, (м2/с)	Напор на початку труби, H1	Напор на прикінці труби, H2	Обсяг води, що проходить, Q	Швидкість води V, м/с	Діаметр труби, мм
2	0,0458	0	0,0000024	1	1	0,0044058	0,56125	1
3	0,0543	0,5	0,0000033	2	1	4	1106,77255	0,68
4	0,0543	0,005	0,0000063	1	0,5	3	190,22896	1,42
5	0,0172	0,05	0,0000063	4	3	0,5708738	0,02909	50
6	0,0172	0,05	0,0000063	4	3	0,5708738	0,02909	50
7								

Рисунок Г.7 – звіт «.xlsx»



РЕЗУЛЬТАТИ·ГІДРАВЛІЧНИХ·ОБЧИСЛЕНЬ¶

Коеф-т· гідравлічного· опору¤	Гідравлічний· уклон¶ ¤	Кінематична· в'язкість,·(м2/с)¤	Напор·на·початку· труби,·Н1¤	Напор·на·прикінці· труби,·Н2¤	Обсяг·води,·що· проходить,·Q¤	Швидкість·води·V,· м/с¤	Діаметр·труби,· мм¤
0.0172¤	0.05¤	0.0000063¤	4¤	3¤	0.5708738¤	0.02909¤	50¤

¶

Рисунок Г.8 – звіт «.docx»