МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра програмного забезпечення автоматизованих систем
Спеціальність 121 «Інженерія програмного забезпечення»
Освітня програма «Інженерія програмного забезпечення»

«Допущений до захисту»
Завідувач кафедри
_____ проф. Приходько С.Б.
«___» _____ 2022 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти «бакалавр»

на тему: Розробка програмного забезпечення для автоматизації процесів обробки даних системи електронного контролю транспортного складу маршрутного таксі міста Миколаєва

Виконав: студент групи 4151
_____ Семенчук І.М.
(підпис, ПІБ)

Керівник роботи:
доцент каф. ПЗАС, к.т.н., доцент
(посада, науковий ступень вчене звання)
_____ Макарова Л.М
(підпис, ПІБ)

Миколаїв – 2022 р

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра програмного забезпечення автоматизованих систем
Спеціальність 121 «Інженерія програмного забезпечення»
Освітня програма «Інженерія програмного забезпечення»

«ЗАТВЕРДЖУЮ»
Гарант освітньої програми
_____ доц. Макарова Л.М.
(підпис)
«___» _____ 2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ
на здобуття ступеня вищої освіти «бакалавр»

Студенту Семенчуку Івану Миколайовичу
(Прізвище, ім'я, по батькові)

1. Тема роботи: Розробка програмного забезпечення для автоматизації процесів обробки даних системи електронного контролю транспортного складу маршрутного таксі міста Миколаєва

Керівник роботи Макарова Лідія Миколаївна
Затверджені наказом ректора №256-уч від «11» травня 2022 р.

2. Термін подання роботи: 10.06.2022 р.

3. Вихідні дані по роботі:

4. Перелік питань, що належать до розробки (найменування розділів)
Титульний аркуш; Завдання на кваліфікаційну роботу; Анотація (українською та англійською мовами); Зміст; Перелік умовних позначень, символів, одиниць та термінів (при необхідності); Вступ; Опис предметної галузі та постановка задачі; Проєкт програмного забезпечення (ескізний, технічний та робочий); Результати розробки програмного забезпечення; Розділ з охорони праці; Висновки; Список використаних джерел; Додатки (технічне завдання, текст програми, опис програми, інструкція користувача, програма та методика випробувань програмного забезпечення).

5. Перелік презентаційних матеріалів Титульний слайд, Мета кваліфікаційної роботи, Завдання, виконання яких є складовою успішної розробки ПЗ, Функціональне призначення програмного забезпечення, Діаграма варіантів використання, Концептуальна модель, Ескізи інтерфейсу програмного забезпечення, Статичне представлення програмного забезпечення, Динамічне представлення програмного забезпечення, Вибір інструментів розробки, Результати розробки, Висновки, Заключний слайд

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
4	Гурець Н.В., ст. викладач		

7. Дата видачі завдання 11.05.2022 р.

К А Л Е Н Д А Р Н И Й П Л А Н

Номер	Назва етапів роботи	Терміни виконання	Примітка
1.	Підготовка розділу Вступ	24.03.2022	ПП*
2.	Опис та аналіз предметної галузі	28.03.2022	ПП*
3.	Постановка задачі	31.03.2022	ПП*
4.	Ескізний проєкт програмного забезпечення	27.04.2022	
5.	Технічний проєкт програмного забезпечення	06.05.2022	
6.	Робочий проєкт програмного забезпечення	13.05.2022	
7.	Підготовка розділу Результати розробки програмного забезпечення	17.05.2022	
8.	Підготовка розділу з охорони праці	18.05.2022	ПП*
9.	Підготовка розділу Висновки	19.05.2022	
10.	Оформлення списку використаних джерел та додатків	20.05.2022	
11.	Подання на кафедру ПЗАС тексту остаточного варіанту роботи, підписаного її керівником, у роздрукованому та електронному форматі разом із заявами щодо самостійності виконання роботи та ідентичності друкованої та електронної версій роботи (Додатки 1 і 2 «Порядку здійснення заходів з перевірки робіт на наявність текстових збігів/ідентичності/схожості із використанням програмно-технічних засобів», який введений в дію наказом ректора НУК за №20 від 20.01.2020 р.)	10.06.2022	не пізніше, ніж за 14 діб до захисту (згідно п.4.1 зазначеного Порядку)
12.	Підготовка презентації та доповіді	12.06.2022	
13.	Попередній захист роботи на засіданні кафедри ПЗАС	14.06.2022	
14.	Подання на кафедру ПЗАС електронних версій наступних документів у форматі pdf: кваліфікаційної роботи; файлу-опису кваліфікаційної роботи (згідно Додатку до наказу ректора НУК за №287-уч від 19.05.2020 р.); презентації доповіді	24.06.2022	

* - за результатами переддипломної практики (ПП), яка була з 21.03.2022 до 08.05.2022 р.

Студент _____ Семенчук І.М.
(підпис) (ПІБ)

Керівник роботи _____ Макарова Л.М.
(підпис) (ПІБ)

АНОТАЦІЯ

Кваліфікаційна робота присвячена розробці програмного забезпечення для автоматизації процесів обробки даних системи електронного контролю транспортного складу маршрутного таксі міста Миколаєва. Розробка ведеться для комунального підприємства «Міський інформаційно-обчислювальний центр» надалі КП «МІОЦ». Дане програмне забезпечення дозволить працівникам комунального підприємства зменшити об’єм виконуваної роботи, пришвидшити процес обробки даних, зменшити ймовірність виникнення помилок, спростити ведення звітності.

Дана кваліфікаційна робота передбачає розв’язання практичної задачі інженерії програмного забезпечення, що характеризується комплексністю та невизначеністю умов, а саме розробку зазначеного вище програмного забезпечення.

У кваліфікаційній роботі міститься аналіз та опис предметної галузі за темою кваліфікаційної роботи, постановка задачі, проект програмного забезпечення, а також результати розробки програмного забезпечення та розділ з охорони праці.

Кваліфікаційна робота викладена на 93 сторінках друкованого тексту та містить 29 рисунків, 20 таблиць, 5 додаток та список використаних джерел з 21 найменувань.

ABSTRACT

Qualification work is devoted to software development for automation of process of data processing of electronic system management of transport structure of a route taxi in the city Nikolaev. Development is being carried out for the municipal enterprise "Miskyi informatsiyyno-obchislyvalnyi center" abbreviated ME "MIOC". This software will allow employees of the enterprise to reduce the amount of work performed, speed up the data processing, reduce the probability of errors, simplify reporting.

This qualification work involves solving a practical problem of software engineering, which is characterized by complexity and uncertainty of condition namely the development of the above software.

The qualification work contains an analysis and description of the subject area on the topic of qualification work, problem statement, software project, results of software development and the section on labor protection.

The qualification work is presented on 93 pages of printed text and contains 29 figures, 20 tables, 5 appendices and a list of references from 21 names.

ЗМІСТ

ВСТУП

У кваліфікаційній роботі розроблюється програмне забезпечення для автоматизації обробки даних системи електронного контролю транспортного складу маршрутного таксі міста Миколаєва.

Дана кваліфікаційна робота передбачає розв’язання практичної задачі інженерії програмного забезпечення, що характеризується комплексністю та невизначеністю умов.

GPS (англ. Global Positioning System – система глобального позиціонування. Принцип роботи будь-якої навігаційної системи можна описати таким чином: на орбіті літає кілька супутників, які представляють із себе максимально точні літаючі атомні годинники з антеною і сонячною батареєю. Вони випромінюють радіосигнал, в якому міститься точний час. Цей сигнал рухається до приймача зі швидкістю світла. В якості приймача може виступати смартфон, планшет або спеціальний навігатор. Усередині смартфона є мобільний процесор, який відповідає не тільки за обчислення. Однією з важливих частин процесора є модем - він приймає сигнал від веж стільникового зв'язку, Wi-Fi, блютуз і багато іншого. Іноді такий модем не є частиною процесора, а представляє із себе окремий чіп на платі. Коли модем в смартфоні отримує сигнал від GPS супутника, він зіставляє час відправлений супутником і свій час, а після цього визначає різницю. Таким чином, знаючи швидкість сигналу і час, який знадобився на його проходження, приймач обчислює на якій відстані від нього знаходиться супутник.

Маючи три набори даних від трьох різних супутників і знаючи їх точне положення на земній орбіті можна обчислити координати приймача. Супутники рухаються на постійній орбіті з постійною швидкістю і їх координати в кожен відрізок часу відомі з високою точністю. Ці дані зібрані в таблиці, які мають назву "альманахи". Такі таблиці повинен обов'язково мати у своєму розпорядженні будь-який супутниковий приймач до початку змін. У

випадку зі смартфоном, альманах зберігається в прошивці радіомодуля. Маючи мінімум три набори даних з трьох різних супутників з'являється можливість визначити місце розташування приймача. Однак для отримання більш точних результатів завжди використовуються дані четвертого супутника. Таким чином, дані з 4-х супутників – це той мінімальний набір даних, який необхідний для роботи GPS функції. Так, чим більше супутників може бачити пристрій, тим швидше і точніше він зможе визначити своє місце розташування. Саме з цією метою виробники чіпів впроваджують в свої пристрої підтримку максимальної кількості навігаційних систем [1].

З метою впровадження диспетчерського та автоматизованого контролю за роботою рухомого складу, який здійснює пасажирські перевезення на міських автобусних маршрутах загального користування у м. Миколаєві, відповідно до рішення Миколаївського виконавчого комітету Миколаївської міської ради № 155 від 06 березня 2015 року [2], виконкомом міської ради проводиться вивчення та передача в диспетчерський центр координат системи GPS місцезнаходження рухомих одиниць та параметрів їх стану, а також відображення цієї інформації на електронній мапі, керуючись ст. 28 Закону України «Про місцеве самоврядування в Україні».

Дана кваліфікаційна робота передбачає розв’язання практичної задачі інженерії програмного забезпечення, а саме розробку програмного забезпечення для автоматизації процесів обробки даних системи електронного контролю транспортного складу маршрутного таксі міста Миколаєва.

Метою кваліфікаційної роботи є розробка програмного забезпечення, для зменшення об’єму виконуваної ручної роботи працівниками КП «МІОЦ», автоматизації обробки даних системи електронного контролю транспортного складу, автоматизації процесів збереження результатів обробки даних, спрощення процесів ведення звітності, надання ефективного механізму пошуку інформації. ПЗ отримуватиме оновлені дані з серверу, на який надсилається інформація з пристроїв GPS, встановлених на транспортних засобах.

Визначимо завдання, виконання яких є складовою успішної розробки ПЗ:

- проаналізувати предметну галузь;
- виявити вимоги для розроблюваного ПЗ;
- розробити специфікацію модулів ПЗ;
- розробити технічне завдання ПЗ;
- розробити концептуальну модель ПЗ;
- розробити логічну та фізичну моделі ПЗ;
- виконати кодування ПЗ ;
- виконати тестування ПЗ.

1 ОПИС ТА АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ДЛЯ АВТОМАТИЗАЦІЇ ПРОЦЕСІВ ОБРОБКИ ДАНИХ СИСТЕМИ ЕЛЕКТРОННОГО КОНТРОЛЮ ТРАНСПОРТНОГО СКЛАДУ

1.1 Опис та аналіз роботи КП «МІОЦ» з обробки даних системи електронного контролю транспортного складу маршрутного таксі міста Миколаєва

Міська рада міста Миколаєва розташована за адресою: вулиця Адміральська, 20, Миколаїв, Миколаївська область, 54000. Вона вміщує виконавчі органи та комунальні підприємства.

Виконавчі органи Миколаївської міської ради вміщують в себе адміністрації, департаменти, управління та відділи.

Районна державна адміністрація є органом виконавчої влади і входить до системи органів виконавчої влади.

Департамент – це структурний підрозділ, що утворюється для виконання окремих завдань з реалізації державної політики за певними напрямами діяльності центрального органу виконавчої влади, за умови, що в його складі буде не менш як чотири відділи.

Управління – це структурний підрозділ одногалузевого або однофункціонального спрямування. До складу управління входять не менш як два відділи**.** Самостійний відділ утворюється з штатною чисельністю не менш як 5 одиниць**.**

Відділ у складі департаменту (управління) – це структурний підрозділ, що утворюється для виконання завдань за одним напрямом (функцією) діяльності органу виконавчої влади, з штатною чисельністю не менш як 4 одиниці. Відділ очолює начальник (код КП посади «Начальник відділу (у складі управління)»). Такий самий код має й посада «Начальника відділу» [3].

Миколаївська міська рада складається з чотирьох адміністрацій: Заводського, Інгульського, Центрального та Корабельного районів [4].

Одинадцяти департаментів: департамент міського голови, департамент праці та соціального захисту населення, департамент житлово-комунального господарства, департамент фінансів Миколаївської міської ради, департамент архітектури та містобудування, департамент забезпечення діяльності виконавчих органів, юридичний департамент, департамент економічного розвитку, департамент енергетики, енергозбереження та запровадження інноваційних технологій, департамент внутрішнього фінансового контролю, нагляду та протидії корупції, департамент надання адміністративних послуг.

Дванадцяти управлінь: управління комунального майна, управління капітального будівництва, управління державного архітектурно-будівельного контролю, управління освіти, управління охорони здоров'я, управління з питань культури та охорони культурної спадщини, управління у справах фізичної культури і спорту, управління молодіжної політики, управління транспортного комплексу, зв'язку та телекомунікацій, управління земельних ресурсів, управління з питань надзвичайних ситуацій та цивільного захисту населення, управління апарату Миколаївської міської ради.

Дев'яти відділів: відділ кадрів, відділ бухгалтерського обліку, відділ обліку та розподілу житла, відділ з організації оборонної і мобілізаційної роботи та взаємодії з правоохоронними органами, архівний відділ, відділ інформації та використання документів ліквідованих установ (трудовий архів), господарсько-експлуатаційний відділ, технічно-транспортний відділ, відділ стандартизації та впровадження електронного врядування. Також, служба у справах дітей Миколаївської міської ради, патронатна служба міського голови, сектор захисту інформації.

Комунальних підприємств: КП ММР «Миколаївська ритуальна служба», ЖКП ММР «Прибужжя», КП ММР «Стоматологія №3», КП ММР «Миколаївські парки», КП ММР «Капітальне будівництво м. Миколаєва», КВП по організації харчування у навчальних закладах, КП «ДОРОГА», ЖКП

ММР «БРИЗ», МКП «Миколаїв-водоканал», Комунальне спеціалізоване монтажно-експлуатаційне підприємство, КП ММР «Центр захисту тварин», ОКП «Миколаївоблтеплоенерго», КП «Миколаївелектротранс», КП «Дочірнє підприємство стоматологічної поліклініки №2», КП «Миколаївкомунтранс», КП «Обрий-ДКП», КЖЕП Центрального району м. Миколаєва, КП «Госпрозрахункове проектно-виробниче АПБ», КП ММР «ПДЗОВ «Дельфін», КП «ТАЙМСЕТ», КП «Миколаївське міжміське бюро технічної інвентаризації», КП «Дочірнє підприємство стоматологічної поліклініки №1», КП «Міський інформаційно-обчислювальний центр».

КП «Міський інформаційно-обчислювальний центр» – це підприємство, яке займається обслуговуванням комп'ютерів та оргтехніки в миколаївській міській раді. Це включає:

- апаратне та програмне супроводження;
- огляд та аналіз оргтехніки та комп'ютерів на робочих місцях;
- інспектування та виправлення неполадок;
- огляд та аналіз мережевих з'єднань;
- інсталяція, оновлення й налаштування програм;
- консультація з використання програм, серед яких прості прикладні програми, спеціалізовані програми та продукти Microsoft Office;
- установка ОС Windows;
- введення комп'ютерів в домен та надання їм доступу до мережевих дисків та папок;
- налаштування мережевих принтерів та сканерів;
- переписування даних на нові комп'ютери;
- внесення правок до документів і підготовка голосувань. Внесення правок – це внесення змін або доповнень, або втрата чинності, якщо указ за останніми рішеннями більше не дійсний;
- оформлення у печатному вигляді результатів голосувань депутатських сесій.

На жовтень 2021 року транспортна мережа міста Миколаєва обслуговується 43 автобусними, 6 трамвайними та 6 тролейбусними маршрутами.

Обслуговують зазначені маршрути громадського транспорту 12 підприємств – перевізників, а саме:

- комунальне підприємство Миколаївської міської ради «Миколаївелектротранс»;
- комунальне підприємство Миколаївської міської ради «Миколаївпастранс»;
- ТОВ «ЕТАЛОНАВТО»;
- ПП «ПИК»;
- ПП «ІВА»;
- ТОВ «ПРИВАТАВТОЛЮКС»;
- ТОВ ВКФ «ГУРІГ – СЕРВІС»;
- ТОВ «АЛАН – ТЕХНО»;
- ТОВ «ЄВРОТРАНСТЕХСЕРВІС»;
- МПП «ТФТ»;
- ПП «АВТО – ВІОЛА ПЛЮС»;
- ПП «МІС».

Середня кількість транспортних засобів станом на 31.10.2021 по місту складає 565 одиниць [5].

Програма, що буде розроблюватися, призначена для автоматизованої обробки даних системи електронного контролю транспортного складу маршрутного таксі. Система електронного контролю транспортного складу – це комплекс засобів програмного й технічного забезпечення для надсилання, зберігання й обробки даних про рух транспортного складу.

З програмою взаємодіє працівник КП «МІОЦ» – диспетчер, який має свій особистий акаунт для отримання доступу до даних серверу. Після запуску програмне забезпечення автоматично виконує аналіз стану системи GPS

трекінгу, після чого результат автоматично зберігається у вигляді Google Spreadsheets таблиць, до яких має доступ тільки диспетчер і керівник КП «МІОЦ». За бажанням аналіз стану системи може виконуватися автоматично у вказаний час. Опісля диспетчер має можливість виконати аналіз стану маршрутів, які йому потрібні, тобто отримати розширену інформацію про стан системи електронного контролю на транспортному складі певного маршруту, таку як (Реєстраційний номер, IMEI трекера, номер CIM-картки, статус трекера, дата останнього оновлення й тд.), й зберегти результати у вигляді звіту, для подальшого роздрукування або збереження у вигляді Google Spreadsheets таблиць для звітності. Якщо диспетчеру потрібна інформація про певний транспортний засіб, чи засоби, він може скористатися пошуком за ключовими словами й переглянути інформацію про транспортний склад, характеристики якого співпадають з шуканими.

1.2 Аналіз існуючого аналогічного програмного забезпечення

На сьогодні існує достатня кількість програм з обробки й аналізу даних. Зокрема, непоганий огляд найпопулярніших інструментів аналізу даних для бізнесу приведений, наприклад в [6]. Розглянемо декілька самих відомих варіантів.

1) Microsoft Power BI

Microsoft Power BI – це набір інструментів, що використовуються для імпорту, агрегування та представлення даних у формі іммерсивних та простих для розуміння звітів та візуальних елементів [7].

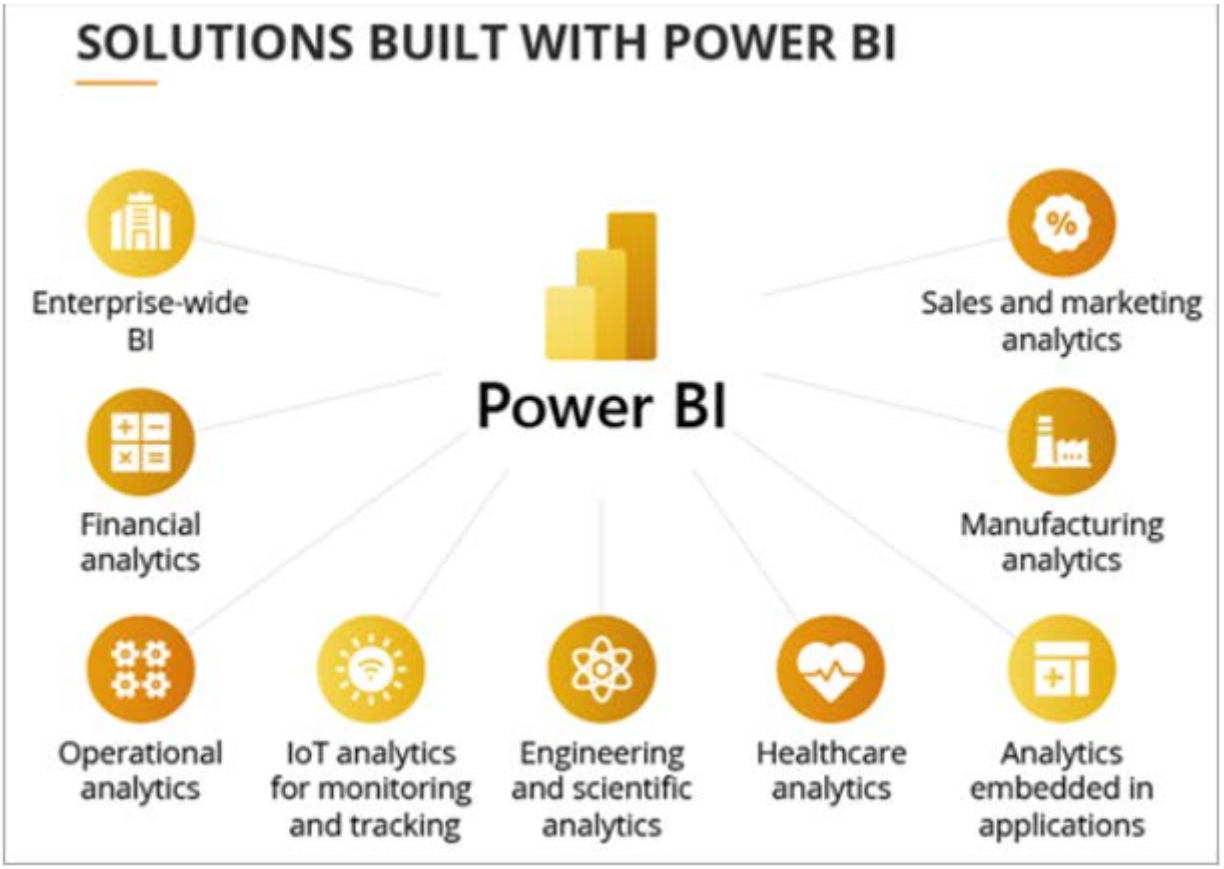


Рисунок 1.1 – Microsoft Power BI [7]

Плюси Microsoft Power BI:

- прийом різних структур даних;
- імпорт даних з широкого спектру джерел даних;
- самостійний прийом реляційних та великих даних;
- складання звітності;
- Візуалізація даних;
- інтеграція з Excel;

Мінуси Microsoft Power BI:

- ціна на Microsoft Power BI від $ 9,99 / користувач / місяць;
- складний в розумінні та використанні;
- потребує знання мови виразів DAX;
- Microsoft Power BI це величезний пакет, в якому є багато інших взаємопов'язаних інструментів.

2) RapidMiner

RapidMiner – це програмна платформа для підготовки даних, машинного навчання, глибокого навчання, видобутку тексту та інтелектуального розгортання моделі. Він надає всі можливості підготовки даних [8].

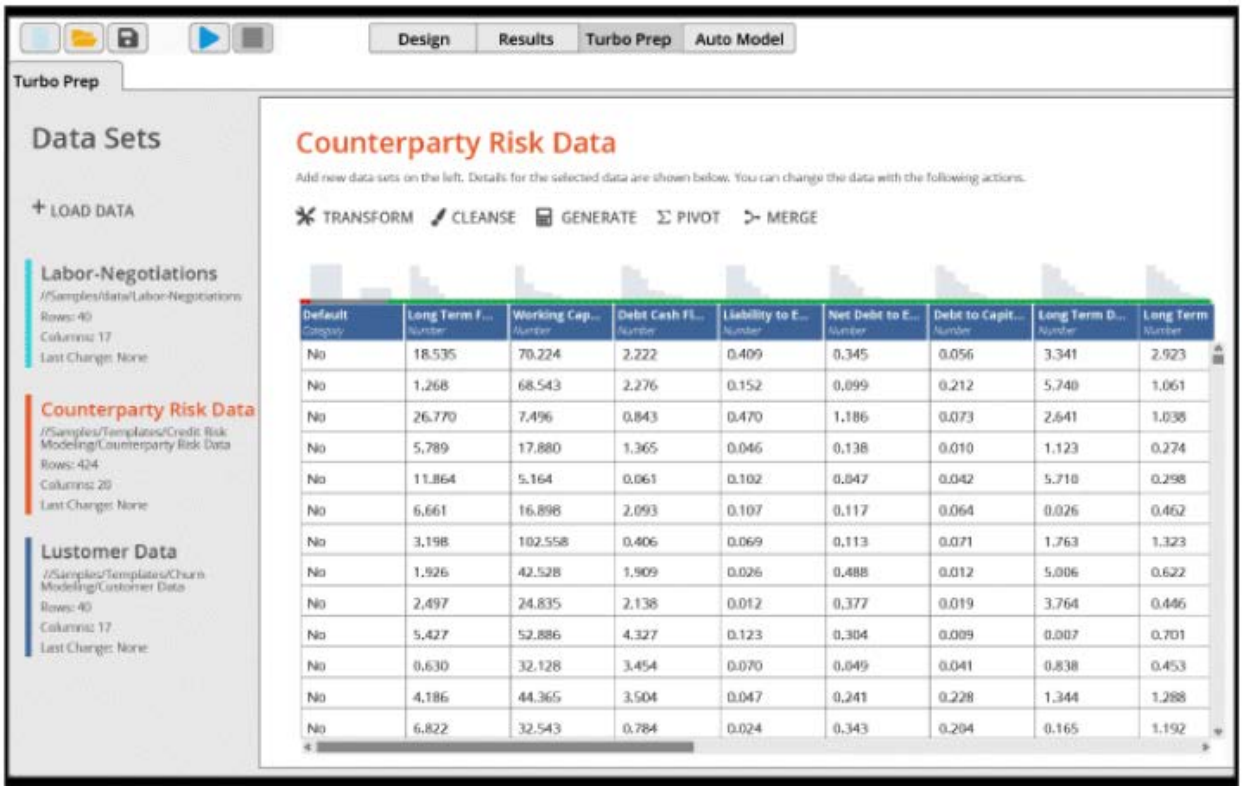


Рисунок 1.2 – інтерфейс RapidMiner [8]

Плюси RapidMiner:

- вбудований контроль безпеки;
- інструмент простий у використанні;
- потужний графічний інтерфейс;
- Radoop позбавляє потреби писати код;
- оптимізація процесів;
- RapidMiner надає п'ять продуктів для аналізу даних, RapidMiner Studio, RapidMiner Auto Model, RapidMiner Turbo Prep, RapidMiner Server і RapidMiner Radoop.

Мінуси RapidMiner:

- висока вартість продукту – мінімальний план 2500 доларів на користувача / рік, максимальний 10000 доларів на користувача / рік;
- відсутня можливість складання звітності;
- відсутня можливість прийому різних структур даних;
- відсутня інтеграція з Excel, або іншими інструментами для роботи з таблицями.

3) Zoho Analytics

Zoho Analytics – це програмне забезпечення для самообслуговування BI та аналізу даних, яке дозволяє за лічені хвилини створювати візуально привабливі візуалізації даних та корисні інформаційні панелі [9].

Рисунок 1.3 – інтерфейс Zoho Analytics [9]

Плюси Zoho Analytics:

- широкий вибір варіантів візуалізації у вигляді діаграм, зведених таблиць;
- розширена аналітика з використанням інтелектуального асистента;

- елегантні мобільні програми для iOS та Android;
- розумне моделювання для підключення відповідних таблиць даних;
- змішування даних з декількох джерел даних.

Мінуси Zoho Analytics:

- вартість продукту – тариф базовий 22 долари на місяць, стандартний 45 доларів, преміум 112 доларів та корпоративний 445 доларів;
- відсутня можливість складання звітності;
- відсутня можливість прийому різних структур даних;
- відсутня інтеграція з Excel, або іншими інструментами для роботи з таблицями.

1.3 Постановка задачі на розробку програмного забезпечення для автоматизації обробки даних системи електронного контролю транспортного складу маршрутного таксі міста Миколаєва

Виконавши аналіз предметної галузі та існуючих аналогів, їх переваг та недоліків, можна зробити висновок, що розробка нового програмного забезпечення буде актуальною, бо матиме такі переваги перед існуючими аналогами:

- програмне забезпечення буде цілком направлене на вирішення завдань КП «МІОЦ»;
- програмне забезпечення потребує мінімальний рівень знань для використання;
- не потребує знань мов програмування або інших спеціальних знань крім розуміння понять предметної галузі;
- програмне забезпечення не є громіздким.

Користувачем даного програмного забезпечення буде диспетчер.

Опис: Чоловік або жінка від 18 до 60 років, закінчив, або навчається в університеті, працює як найманий працівник ПК «МІОЦ» має навички користування комп’ютером, знайомий з предметною галуззю.

Соціальні характеристики: розмовляє українською мовою, досягнув(ла) повноліття, має або здобуває вищу освіту, володіє середнім рівнем в користуванні комп’ютером.

Мотиваційна середа: робоча середа – кабінет, середня мотивація до навчання.

Задача користувача: виконувати покладену на нього роботу, а саме виконувати аналіз стану системи електронного контролю транспортного складу та ведення звітності про його результати.

Робоча середа: ПК, ЖКМ, операційна система – linux (Ubuntu), браузер – Google (Chrome), маніпулятори: мишка, клавіатура, доступ до інтернету.

Проаналізувавши предметну галузь і виділивши профіль потенційного користувача, можна визначити операції, які він може виконувати в рамках, запропонованих йому програмним забезпеченням. Диспетчеру мають бути доступні такі функції:

1) Запустити програму;
2) Авторизуватися;
3) Вибрати файл для збереження результатів;
4) Виконати аналіз стану системи;
5) Вибрати маршрут;
6) Виконати аналіз стану маршруту;
7) Зберегти результати;
8) Виконати пошук;
9) Задати дату автоматичновго виконання програми;
10) Закрити програму.

Також плюсами розроблюваного програмного забезпечення є:

- автоматичне виконання функцій програми;
- можливість задання часу автоматичного виконання програми;

- автоматичне збереження результатів обробки.

Особливостями розроблюваного програмного забезпечення є:

- програмне забезпечення точно виконує задачі встановлені керівництвом ПК «МІОЦ»;

- програмне забезпечення розробляється для КП «МІОЦ» і буде безкоштовним у використанні для його працівників;

- обробка й аналіз даних налаштована на використовувану структуру даних;

- автоматичне зберігання результатів обробки даних у вигляді Google Spreadseets таблиці, що робить дані зручними для перегляду й аналізу;

- відсутність варіантів візуалізації у вигляді діаграм.

Отже, оцінивши всі плюси, переваги й особливості розроблюваного програмного забезпечення можна зробити висновок що розробка є оправданою та доцільною.

Розроблене програмне забезпечення має забезпечити виконання перерахованих нижче функцій:

- автоматичне виконання аналізу стану системи GPS трекінгу після запуску програми.

- автоматичне зберігання результатів обробки даних у вигляді Google Spreadsheets таблиць

- для користувача (диспетчера):

- авторизація акаунту для отримання доступу до даних серверу;

- виконати аналіз системи;

- задати час автоматичного виконання аналізу системи;

- сформувати звіт;

- виконати пошук по даних за ключовими словами.

Докладніше вимоги до програмного забезпечення наведено в технічному завданні у Додатку А.

2 ПРОЕКТ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ АВТОМАТИЗАЦІЇ ПРОЦЕСІВ ОБРОБКИ ДАНИХ СИСТЕМИ ЕЛЕКТРОННОГО КОНТРОЛЮ ТРАНСПОРТНОГО СКЛАДУ

2.1 Ескізний проект програмного забезпечення

На основі аналізу вимог до програмного забезпечення було розроблено ескізний проект програмного забезпечення, представлений моделлю варіантів використання й специфікацією варіантів використання.

2.1.1 Вибір мови моделювання

Для моделювання програмного продукту була обрана мова моделювання UML. UML – уніфікована мова об'єктно-орієнтованого моделювання, використовується у парадигмі об'єктно-орієнтованого програмування. Є невід'ємною частиною уніфікованого процесу розробки програмного забезпечення [10]. UML може бути застосовано на всіх етапах життєвого циклу аналізу бізнес-систем і розробки додатків. Різні види діаграм які підтримуються UML, і найбагатший набір можливостей представлення певних аспектів системи робить UML універсальним засобом опису як програмних, так і ділових систем. Крім того, UML спеціально створювалася для оптимізації процесу розробки програмних систем, що дозволяє збільшити ефективність реалізації програмних систем у кілька разів і помітно поліпшити якість кінцевого продукту.

UML прекрасно зарекомендувала себе в багатьох успішних програмних проектах. Засоби автоматичної генерації кодів дозволяють перетворювати моделі мовою UML у вихідний код об’єктно-орієнтованих мов програмування, що ще більш прискорює процес розробки.

Практично усі CASE-засоби (програми автоматизації процесу аналізу і проектування) мають підтримку UML. Моделі розроблені в UML, дозволяють значно спростити процес кодування і направити зусилля програмістів безпосередньо на реалізацію системи. Діаграми підвищують супроводжуваність проекту і полегшують розробку документації.

2.1.2 Побудова моделі варіантів використання

Диспетчер – працівник що відповідає за ведення звітності про стан системи електронного контролю транспортного складу, має свій особистий акаунт для отримання доступу до даних серверу й Google Spreadsheets файлу в якому збережені попередні результати обробки даних. Диспетчер має можливість виконати аналіз стану маршрутів, які йому потрібні, зберегти результати у вигляді звіту, для подальшого роздрукування або зберегти у вигляді Google Spreadsheets таблиць для звітності. Якщо диспетчеру потрібна інформація про певний транспортний засіб, чи засоби, він може скористатися пошуком за ключовими словами й переглянути інформацію про транспортний склад, характеристики якого співпадають з шуканими.

Профіль користувача наведений у таблиці 2.1

Таблиця 2.1 – Профіль користувача «Працівник КП «МІОЦ»»

Складова	Опис
Типовий представник	Працівник КП «МІОЦ
Опис	Працівник завданням якого є стежити за станом системи електронного контролю транспортного складу.
Тип	Користувач
Відповідальність	Робота з формами звітності
Критерії успіху	Обізнаність в предметній галузі, мінімальні навички в користуванні ОС Linux, наявність обмеженого доступу до даних про водіїв та авто

Основний актор програмного забезпечення – Диспетчер. Виявлені варіанти використання представлені у таблиці 2.2.

Таблиця 2.2 – Виявлені варіанти використання

Код	Основний актор	Назва	Формулювання
D1	Диспетчер	Авторизуватися	Цей варіант використання дозволяє диспетчеру авторизуватися в системі
D2	Диспетчер	Виконати аналіз стану системи	Цей варіант використання дозволяє диспетчеру переглядати дані про стан системи електронного контролю.
D3	Диспетчер	Виконати аналіз стану маршрутів	Цей варіант використання дозволяє диспетчеру отримати розширену інформацію про стан системи електронного контролю на певного маршруту.
D4	Диспетчер	Обрати маршрут	Диспетчер має обрати маршрут, аналіз якого треба виконати
D5	Диспетчер	Обрати файл для збереження результату	Диспетчер має можливість вибрати файл в який буде збережено результати обробки
D6	Диспетчер	Зберегти результат в вигляді таблиці	Диспетчер має можливість зберегти результати аналізу у вигляді Google Spreadsheets таблиць.
D7	Диспетчер	Вказати час автоматичного виконання програми	Цей варіант використання дозволяє диспетчеру виконати аналіз стану системи автоматично у вказаний час
D8	Диспетчер	Виконати пошук	Диспетчер має можливість виконати пошук за ключовими словами.

Модель варіантів використання представлена на рисунку 2.1

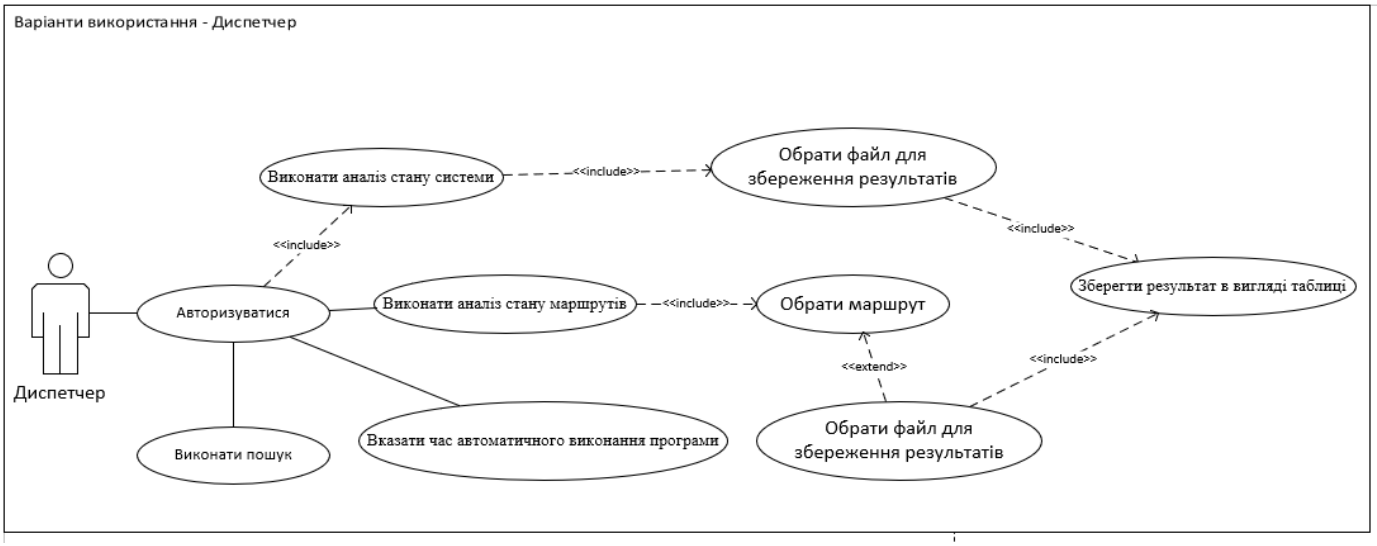


Рисунок 2.1 – Модель варіантів використання для актора диспетчер

Таким чином, ми виявили та описали профіль користувача, обрали головного актора, виявили варіанти використання та побудували модель варіантів використання для актора диспетчер.

2.1.3 Специфікація варіантів використання

Специфікацію варіантів використання розроблюваного програмного забезпечення представлено в таблицях 2.3 – 2.10

Таблиця 2.3 – Специфікація варіанту використання «Авторизуватися»

Складова	Опис
1	2
Опис прецеденту	Даний варіант використання дозволяє диспетчеру авторизуватися в системі електронного контролю.
Дійові особи прецеденту	Диспетчер
Зв’язок з іншими варіантами використання	Має зв’язок стереотипу «include» з варіантом використання «Виконати аналіз системи », має зв’язок з варіантом використання «Виконати аналіз стану маршрутів», має зв’язок з варіантом використання «Вказати час автоматичного виконання програми», має зв’язок з варіантом використання «Виконати пошук»
Передумови	Запуск програми

Продовження табл. 2.3

1	2
Базовий потік	Після запуску програми відкривається вікно авторизації з полям «Логін», «Пароль» та кнопкою «Авторизуватися». Диспетчер вводить в відповідні поля свій логін і пароль й натискає кнопку «Авторизуватися».
Альтернативні потоки	1. Якщо вікно авторизації не відкривається після запуску програми, треба перевстановити програмне забезпечення, у разі не вирішення помилки звернутися за допомогою до спеціаліста. 2. Якщо в поля «логін» «пароль» були введені некоректні дані (спец символи) диспетчеру має бути показано повідомлення про неможливість введення некоректних символів в поля «логін» «пароль» . 3. Якщо користувач ввів не правильні дані, має бути показано повідомлення, що введено не правильний логін чи пароль. 4. Якщо програма запущена й диспетчер намагається ще раз запустити програму, має бути показано повідомлення, що програма вже запущена. 5. Якщо після успішної авторизації головне вікно програми не було відкрито, треба закрити програму й запустити її заново, у разі не вирішення проблеми звернутися за допомогою до спеціаліста.
Постумови	При успішному завершенні прецеденту диспетчеру висвітлюється повідомлення про успішну авторизацію, вікно авторизації закривається й відкривається головне вікно програми.
Особливості	Логін і пароль диспетчер отримує від директора КП «МІОЦ»

Таблиця 2.4 – Специфікація варіанту використання «Виконати аналіз стану системи»

Складова	Опис
1	2
Опис прецеденту	Даний варіант використання дозволяє диспетчеру переглядати дані про стан системи електронного контролю, а саме кількість наявних, активних і неактивних GPS трекерів по кожному маршруту.
Дійові особи прецеденту	Диспетчер

Продовження табл. 2.4

1	2
Зв’язок з іншими варіантами використання	Має зв’язок стереотипу «include» з варіантом використання «Обрати файл для збереження даних», має зв’язок стереотипу «include» з варіантом використання «Авторизуватися»
Передумови	Відкрите головне вікно програми
Базовий потік	Після натискання кнопки «Аналіз стану системи» в лівій половині головного вікна в пункті «Результати» з’являються результати аналізу стану системи.
Альтернативні потоки	1. Якщо після натискання кнопки результати не з’являються, треба закрити програму й запустити її заново, у разі не вирішення проблеми звернутися за допомогою до спеціаліста. 2. Якщо після натискання кнопки в полях результатів отримано не зрозумілі символи, треба закрити програму й запустити її заново, у разі не вирішення проблеми звернутися за допомогою до спеціаліста. 3. Якщо після натискання кнопки в полях результатів отримані данні не є правдивими, треба закрити програму й запустити її заново, у разі не вирішення проблеми звернутися за допомогою до спеціаліста.
Постумови	При успішному завершенні прецеденту постумови відсутні
Особливості	Відсутні

Таблиця 2.5 – Специфікація варіанту використання «Виконати аналіз стану маршрутів»

Складова	Опис
1	2
Опис прецеденту	Даний варіант використання дозволяє диспетчеру отримати розширену інформацію про стан системи електронного контролю на певного маршруту (Реєстраційний номер, IMEI трекера, номер CIM-картки, статус трекера, дата останнього оновлення)
Дійові особи прецеденту	Диспетчер
Зв’язок з іншими варіантами використання	Має зв’язок стереотипу «include» з варіантом використання «Обрати маршрут», має зв’язок стереотипу з варіантом використання «Авторизуватися»

Продовження табл. 2.5

1	2
Передумови	Відкрите головне вікно програми
Базовий потік	Після натискання кнопки «Аналіз стану маршрутів» відкривається вікно обрання маршрутів для аналізу
Альтернативні потоки	1. Якщо після натискання кнопки вікно обрання маршруту не з'явилося, треба закрити програму й запустити її заново, у разі не вирішення проблеми звернутися за допомогою до спеціаліста.
Постумови	При успішному завершенні прецеденту постумови відсутні
Особливості	В вікні обрання маршруту можна обрати відразу декілька маршрутів

Таблиця 2.6 – Специфікація варіанту використання «Обрати маршрут»

Складова	Опис
1	2
Опис прецеденту	Даний варіант використання дозволяє диспетчеру обрати один або декілька маршрутів для отримання аналізу їх стану.
Дійові особи прецеденту	Диспетчер
Зв'язок з іншими варіантами використання	Має зв'язок стереотипу «include» з варіантом використання «Виконати аналіз стану маршрутів», має зв'язок стереотипу «extend» з варіантом використання «Обрати файл для збереження результатів».
Передумови	Відкрите вікно обрання маршрутів
Базовий потік	Ставимо галочку навпроти назви потрібного нам маршруту й натискаємо кнопку «Виконати», після чого вікно обрання маршруту закривається й в лівій половині головного вікна в пункті «Результати» з'являються результати аналізу стану маршруту.
Альтернативні потоки	1. Якщо після відкриття вікна обрання маршруту в списку маршрутів відсутній потрібний маршрут, треба закрити програму й запустити її заново, у разі не вирішення проблеми звернутися за допомогою до спеціаліста. 2. Якщо після відкриття вікна обрання маршруту в списку маршрутів відсутній маршрути, треба закрити програму й запустити її заново, у разі не вирішення проблеми звернутися за допомогою до спеціаліста.

Продовження табл. 2.6

1	2
Альтернативні потоки	3. Якщо не було обрано жодного маршруту, диспетчер має отримати повідомлення, що для виконання аналізу треба обрати хоча б один маршрут. 4. Якщо не можливо поставити галочку напроти потрібного маршруту, треба закрити програму й запустити її заново, у разі не вирішення проблеми звернутися за допомогою до спеціаліста. 5. Якщо після натискання кнопки «Виконати» вікно обрання маршруту не закривається треба закрити програму й запустити її заново, у разі не вирішення проблеми звернутися за допомогою до спеціаліста. 6. Якщо після натискання кнопки «Виконати» в лівій половині головного вікна в пункті «Результати» не з’являються результати аналізу маршруту. треба закрити програму й запустити її заново, у разі не вирішення проблеми звернутися за допомогою до спеціаліста.
Постумови	При успішному завершенні прецеденту в лівій половині головного вікна в пункті «Результати» з’являються результати аналізу стану маршруту.
Особливості	В вікні обрання маршруту можна обрати відразу декілька маршрутів

Таблиця 2.7 – Специфікація варіанту використання «Обрати файл для збереження результату»

Складова	Опис
1	2
Опис прецеденту	Диспетчер має можливість вибрати файл в який буде збережено результати обробки.
Дійові особи прецеденту	Диспетчер
Зв’язок з іншими варіантами використання	Має зв’язок стереотипу «include» з варіантом використання «Зберегти результат у вигляді таблиці», має зв’язок стереотипу «extend» з варіантом використання «Обрати маршрут».
Передумови	Відкрито головне вікно програми, в лівій половині вікна в пункті «Результати» наявні результати аналізу стану маршруту або аналізу стану системи.

Продовження табл. 2.7

1	2
Базовий потік	Натискаємо кнопку «Вибрати файл збереження». Відкривається вікно вибору документа. Натискаємо на кружечок напроти потрібного нам ID документу. Натискаємо на кнопку «Обрати».
Альтернативні потоки	1. Якщо після відкриття вікна обрання маршруту в списку маршрутів відсутній потрібний нам ID документу, треба закрити програму й запустити її заново, у разі не вирішення проблеми звернутися за допомогою до спеціаліста. 2. Якщо після відкриття вікна обрання маршруту в списку маршрутів відсутні ID документи, треба закрити програму й запустити її заново, або додати їх в ручну, у разі не вирішення проблеми звернутися за допомогою до спеціаліста. 3. Якщо не можливо вибрати потрібний ID документу, треба закрити програму й запустити її заново, у разі не вирішення проблеми звернутися за допомогою до спеціаліста. 4. Якщо після натискання кнопки «Обрати» вікно обрання маршруту не закривається треба закрити програму й запустити її заново, у разі не вирішення проблеми звернутися за допомогою до спеціаліста.
Постумови	При успішному завершенні прецеденту постумови відсутні
Особливості	Відсутні

Таблиця 2.8 – Специфікація варіанту використання «Зберегти результат в вигляді таблиці»

Складова	Опис
1	2
Опис прецеденту	Диспетчер має можливість зберегти результати аналізу у вигляді Google Spreadsheets таблиць для ведення звітності або подальшого роздрукування.
Дійові особи прецеденту	Диспетчер
Зв’язок з іншими варіантами використання	Має зв’язок стереотипу «include» з варіантом використання «Обрати файл для збереження результатів»

Продовження табл. 2.8

1	2
Передумови	Відкрито вікно вибору документу й вибраний ID документу для збереження результатів
Базовий потік	Натискаємо на кружечок напроти потрібного нам ID документу. Натискаємо на кнопку «Обрати».
Альтернативні потоки	1. Якщо після натискання кнопки «Обрати» вікно обрання маршруту не закривається треба закрити програму й запустити її заново, у разі не вирішення проблеми звернутися за допомогою до спеціаліста. 2. Якщо не було обрано жодного ID документу, диспетчер має отримати повідомлення, що для виконання аналізу треба обрати ID документу.
Постумови	При успішному завершенні прецеденту диспетчеру буде показано повідомлення про успішне збереження результатів обробки. Вікно вибору файлу закривається
Особливості	Відсутні

Таблиця 2.9 – Специфікація варіанту використання «Вказати час атоматичного виконання програми»

Складова	Опис
1	2
Опис прецеденту	Цей варіант використання дозволяє диспетчеру виконати аналіз стану системи автоматично у вказаний час.
Дійові особи прецеденту	Диспетчер
Зв’язок з іншими варіантами використання	Має зв’язок з варіантом використання «Авторизуватися»
Передумови	Відкрито головне вікно програми
Базовий потік	Вводимо дату й час в поле «Дата виконання», натискаємо кнопку «Задати»
Альтернативні потоки	1. Якщо вказана дата та час мають не вірний формат, має бути показано повідомлення про невірний формат вказаних даних. 2. Якщо вказані дата та час виконання є минулим часом відносно часу виконання програми, має бути показано повідомлення, що вказані дата та час належать минулому часу, операція неможлива.

Продовження табл. 2.9

1	2
Постумови	При успішному завершенні прецеденту диспетчеру буде показано повідомлення про успішне задання дати автоматичного виконання програми
Особливості	Відсутні

Таблиця 2.10 – Специфікація варіанту використання «Виконати пошук»

Складова	Опис
Опис прецеденту	Диспетчер має можливість виконати пошук за ключовими словами й переглянути інформацію про транспортний склад, характеристики якого співпадають з шуканими.
Дійові особи прецеденту	Диспетчер
Зв’язок з іншими варіантами використання	Має зв’язок з варіантом використання «Авторизуватися»
Передумови	Відкрито головне вікно програми
Базовий потік	Записуємо ключове слово в полу «Ключові слова» та натискаємо кнопку «Знайти». в лівій половині головного вікна в пункті «Результати» з’являються результати пошуку
Альтернативні потоки	1. Якщо поле «Ключові слова» пусте, при натисненні на кнопку «Знайти» має бути показано повідомлення, що поле «Ключові слова» пусте. 2. Якщо пошук за ключовими словами не є успішним, має бути показано повідомлення, збігів за вказаними ключовими словами не знайдено.
Постумови	При успішному завершенні прецеденту в лівій половині головного вікна в пункті «Результати» з’являються результати пошуку
Особливості	Відсутні

В даних таблицях приведені та описані короткий опис прецеденту, дійові особи, зв’язки з іншими варіантами використання, передумови виконання, базові потоки, альтернативні потоки, постумови та особливості вказаних вище прецедентів.

2.1.4 Розробка концептуальної моделі

Концептуальна модель сутність-зв’язок – це різновид блок-схеми, де показано, як різні "сутності" (люди, об'єкти, концепції тощо) пов'язані між собою всередині системи. ER-діаграми найчастіше застосовуються для проектування та налагодження реляційних баз даних у сфері освіти, дослідження та розробки програмного забезпечення та інформаційних систем для бізнесу [11].

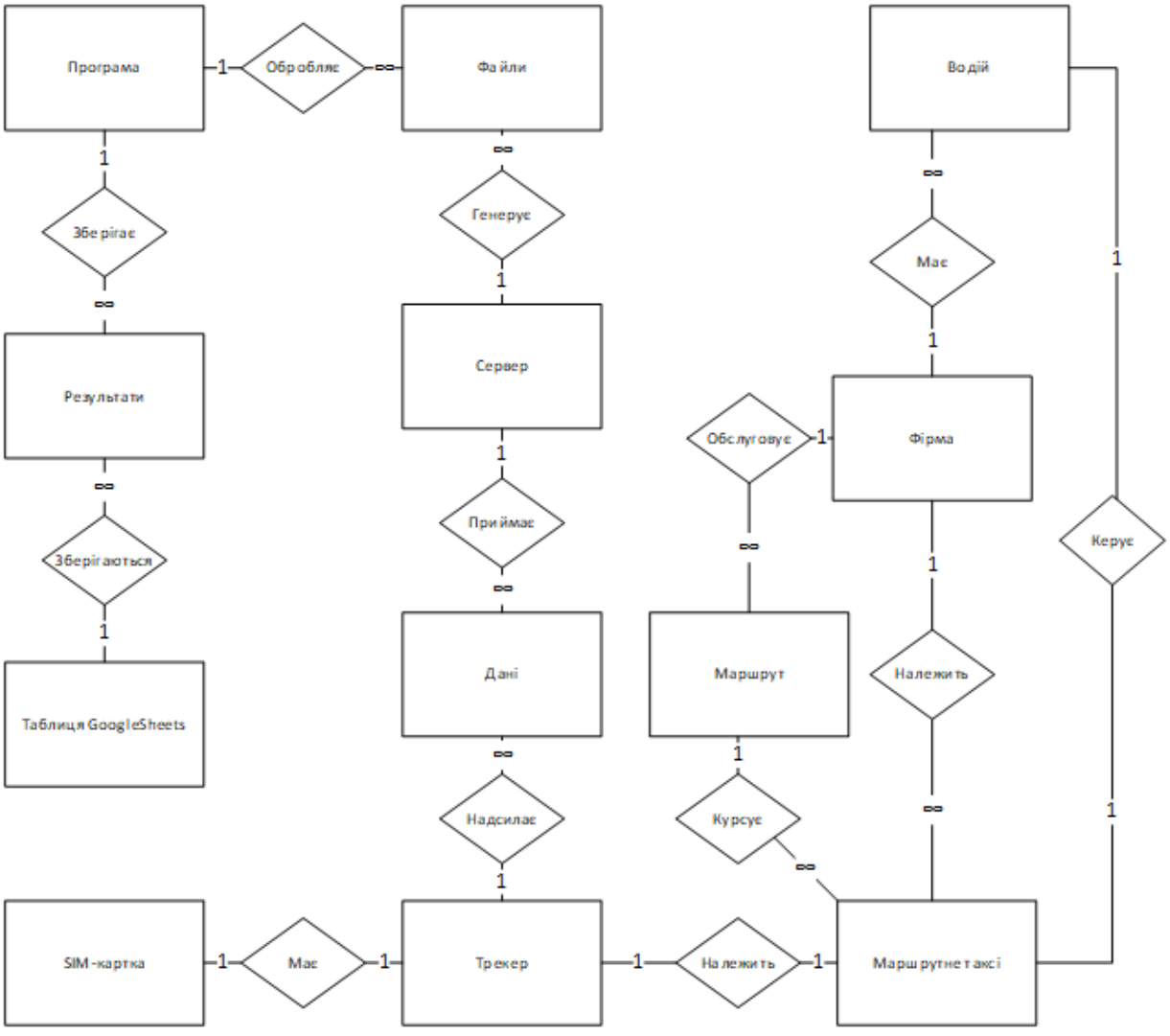


Рисунок 2.2 – Концептуальна модель програмного забезпечення

Атрибути сутностей

Водій: ПІБ водія, номер телефону, номер авто, назва фірми.

Фірма: Назва фірми, код ЄДРПОУ, адреса фірми, номери маршрутів.

Маршрут: номер маршруту, назва фірми, номера авто.

Маршрутне таксі: номер авто, марка авто, модель авто, стан, назва фірми, номер маршруту, IMEI_трекера.

Трекер: IMEI_трекера**,** модель трекера**,** версія прошивки**,** стан**,** номер SIM-картки**,** номер авто.

SIM-картка: номер SIM-картки, PIN-код, PUM-код, IMEI_трекера

Дані: Id, attributes, deviceId, protocol, deviceTime, outdated, valid, latitude, altitude, speed, course, accuracy, network, status, lastUpdate, phone, model, category, disabled.

Сервер: serverTime, fixTime.

Файли:

Positions.json: Id, priority, sat, event, di1, out1, out2, motion, workMode, rssi, adc1, pdop, hdop, power, operator, distance, totalDistance, deviceId, type, protocol, serverTime, deviceTime, fixTime, outdated, valid, latitude, longitude, altitude, speed, course, address, accuracy, network.

Devices.json: Id, route, groupId, name, uniqueId, status, lastUpdate, positionId, geofenceIds, phone, model, contact, category, disabled**.**

Програма: logIn(), get_data(), system status analysis(), route status analysis(), write_to_googlesheet().

Результати: write_to_googlesheet(), result[][]

Таблиця Google Sheets: result [][]

2.1.5 Розробка ескізу інтерфейсу

Проробивши аналіз предметної галузі, сформувавши профілі і сценарії дій користувачів, визначивши функціональність програми можна скласти

схему навігаційної системи, яка зображена на рисунку 2.3.

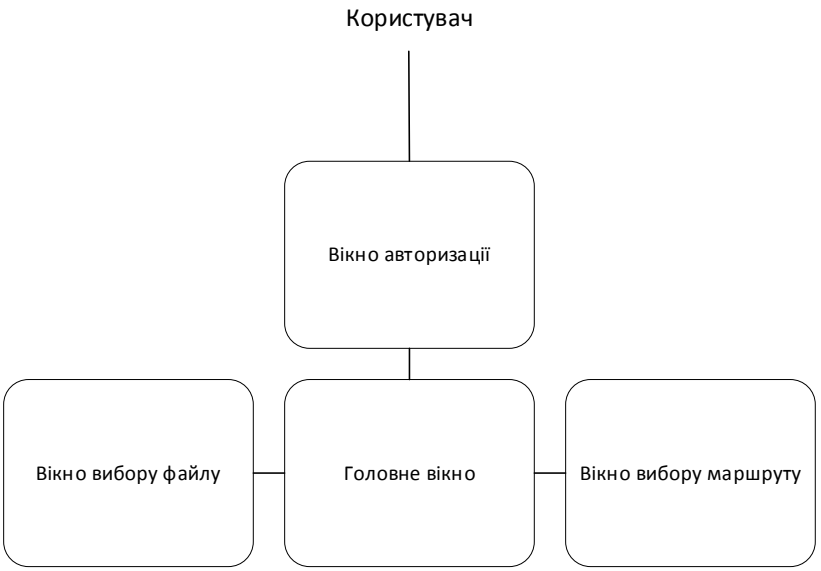


Рисунок 2.3 – Схема навігаційної системи

Користувацький інтерфейс є важливою частиною програмного забезпечення. Від нього залежить ефективність роботи працівника й точність управління машиною з сторони людини [12].

На прототипі можливо тестувати сприйняття системи користувачем та її основну логіку. Користь паперової версії прототипування у виключній простоті модифікації за результатами тестування й в можливості безболісно знаходити представників цільової аудиторії. На паперовому етапі прототип може пережити багато виправлень і, відповідно, багато версій [13]. Прототипи вікон програми приведено на рисунках 2.4 – 2.7.

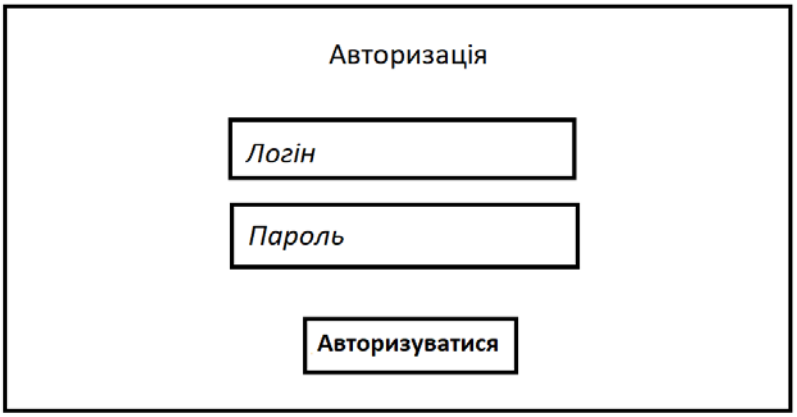


Рисунок 2.4 – Прототип вікна авторизації

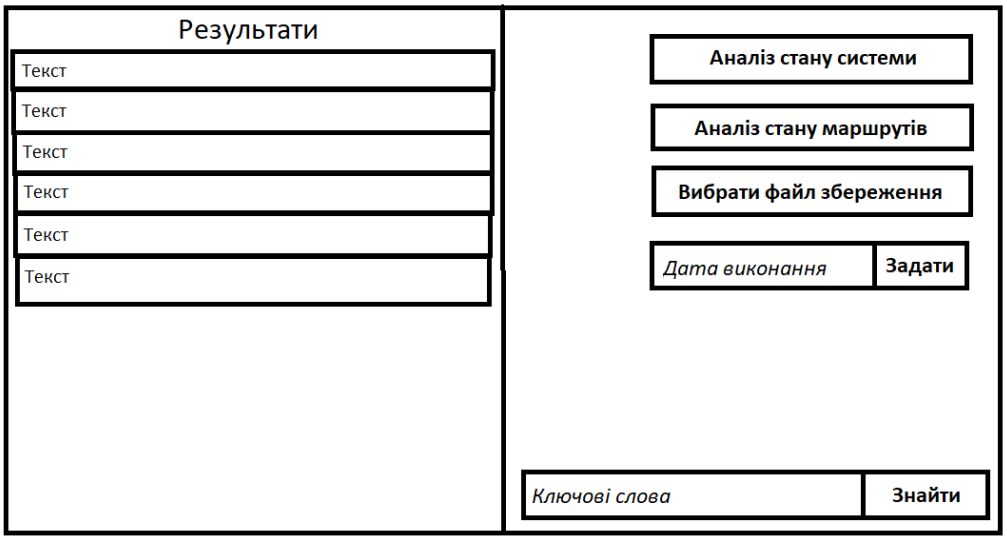


Рисунок 2.5 – Прототип головного вікна програми

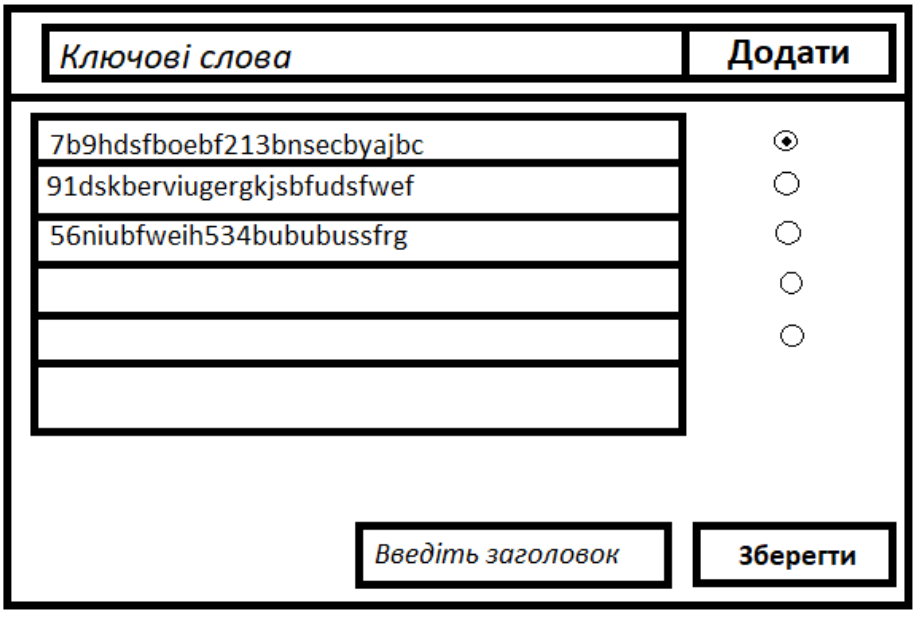


Рисунок 2.6 – Прототип вікна вибору документа

Рисунок 2.7 – Прототип вікна обрання маршруту

Підберемо стилі для програмного інтерфейсу на прикладі вікон «Авторизація» (див. рис. 2.8) та «Головне вікно» (див рис 2.9).

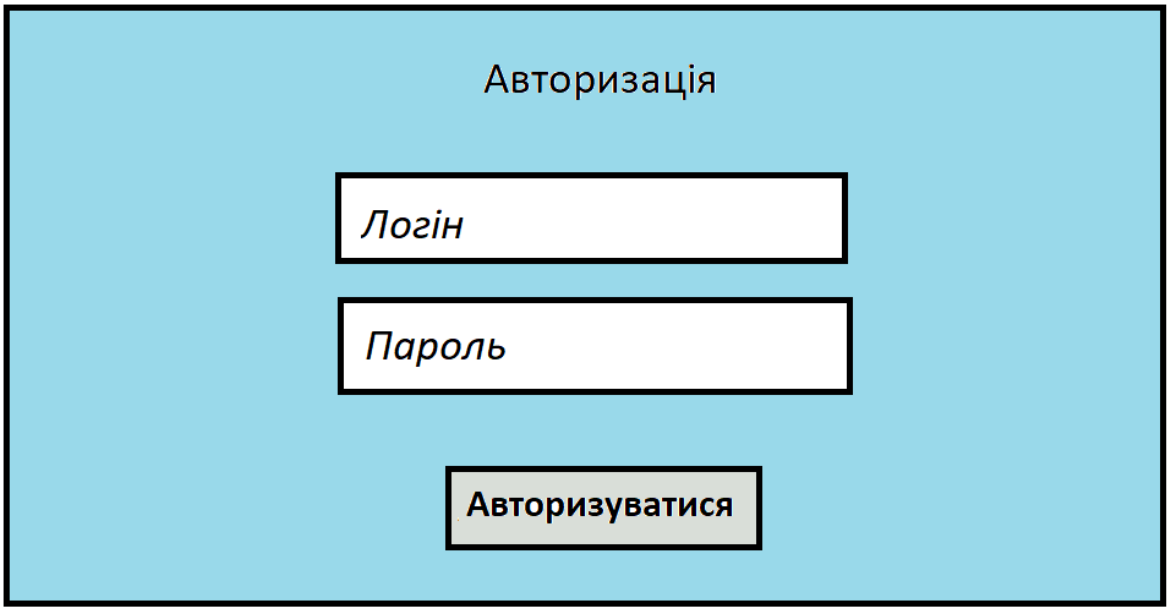


Рисунок 2.8 – Підбирання стилю для вікна авторизації

Рисунок 2.9 – Підбирання стилю для головного вікна

Фінальний варіант стилю оформлення вікон програми наведено на рисунках 2.10 та 2.11

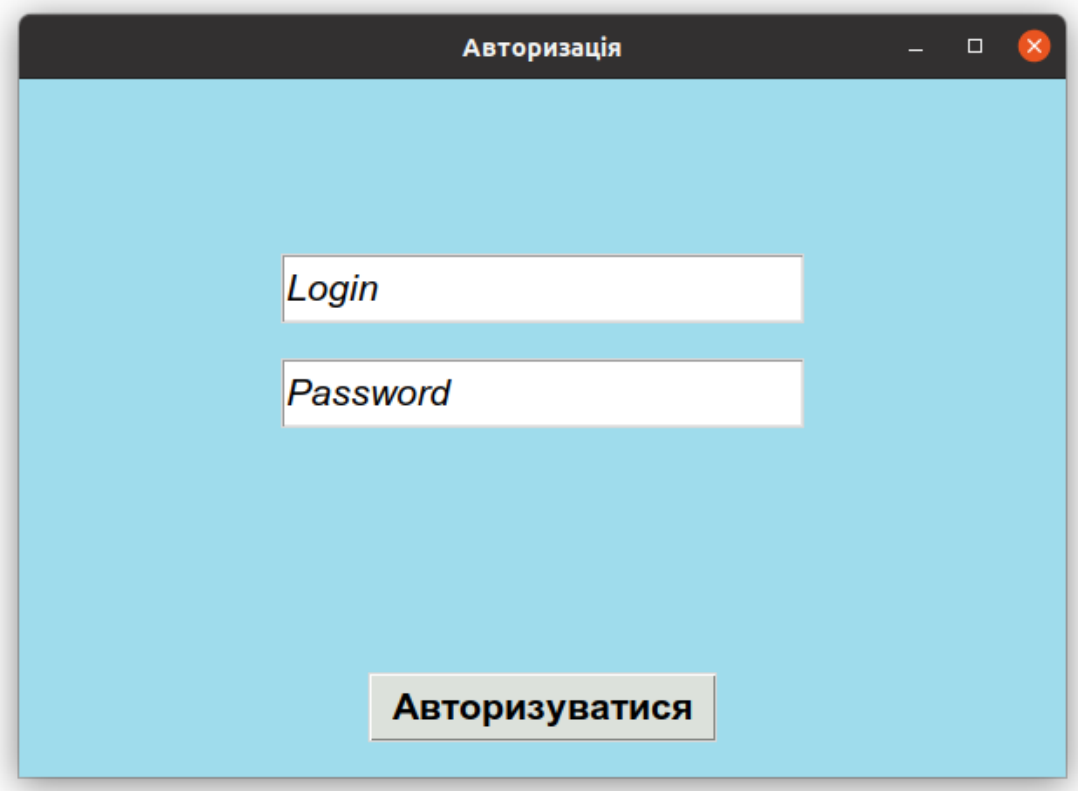


Рисунок 2.10 – Фінальний варіант вікна авторизації

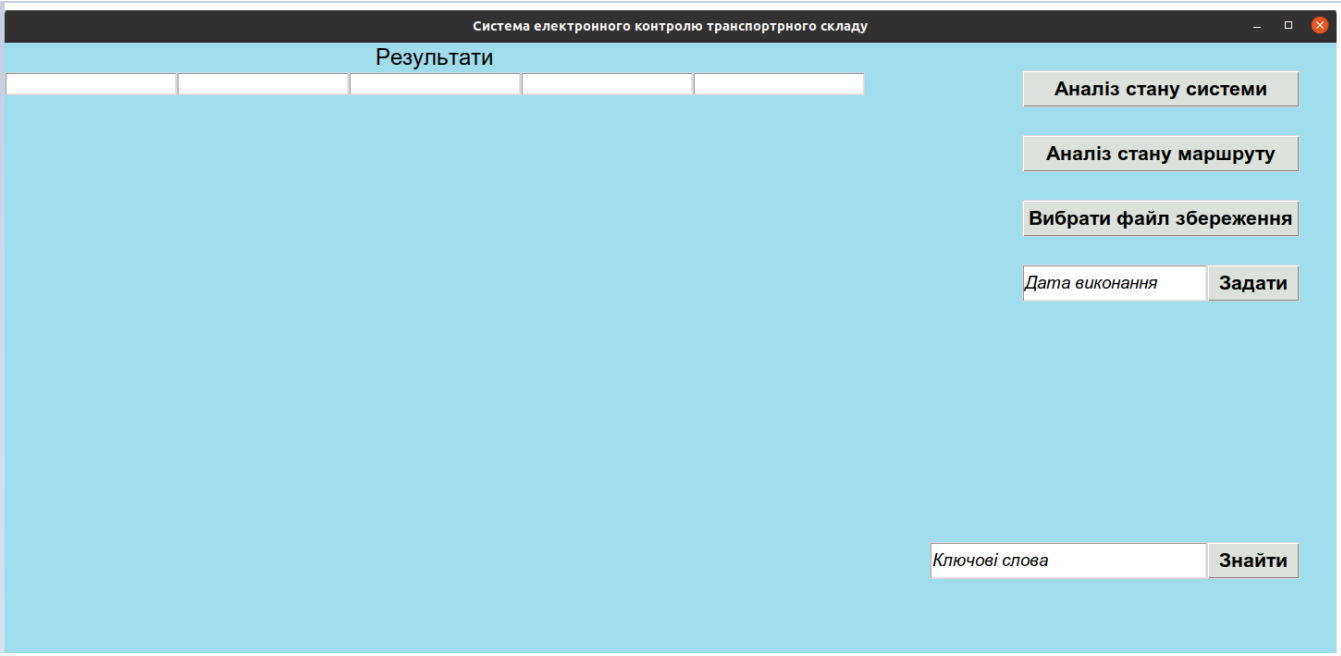


Рисунок 2.11 – Фінальний варіант головного вікна

Підібраний стиль складається з не нав’язливої, зручної для перегляду інформації та палітри кольорів. При підбирання стилю перевагу було віддано мінімалістичному типу за чіткість, ясність і зрозумілість.

2.2 Технічний проект програмного забезпечення

Технічний проект розроблений на основі аналізу ескізного проекту програмного забезпечення. Результати розробки технічного проекту є статична модель програмного забезпечення представлена діаграмою класів й динамічна модель представлена діаграмою діяльності.

2.2.1 Діаграма класів програмного забезпечення

На рисунку 2.12 представлена статична модель програмного забезпечення – діаграма класів, розроблена на основі моделі варіантів використання

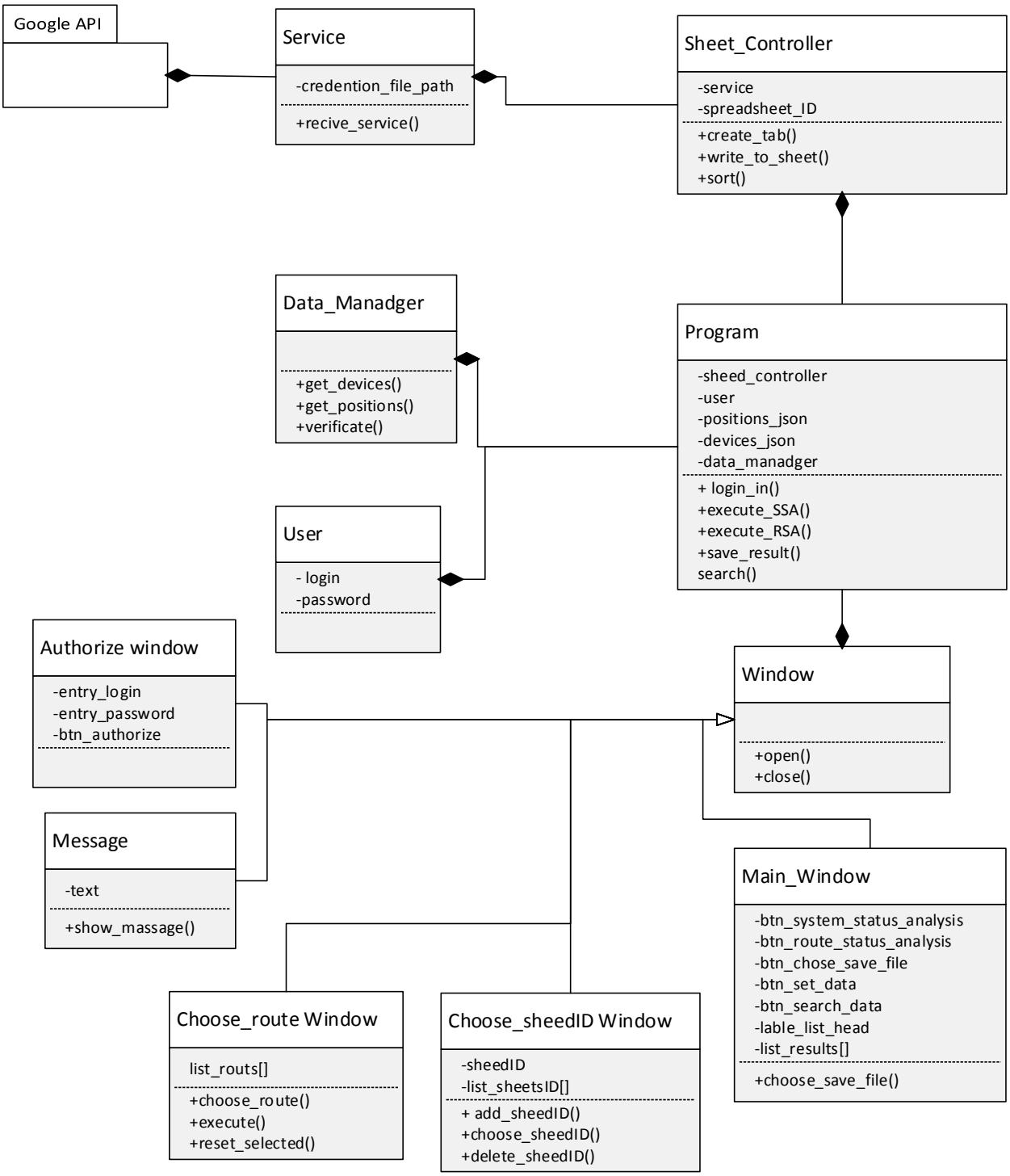


Рисунок 2.12 – Статична модель програмного забезпечення

Розроблена діаграма класів складається з 12 класів і одного пакету Google API.

2.2.2 Діаграма діяльності програмного забезпечення

Динамічна модель програмного забезпечення розроблена на основі статичної моделі та діаграми варіантів використання.

Діаграма діяльності для прецеденту «Авторизуватися» представлена на рисунку 2.13

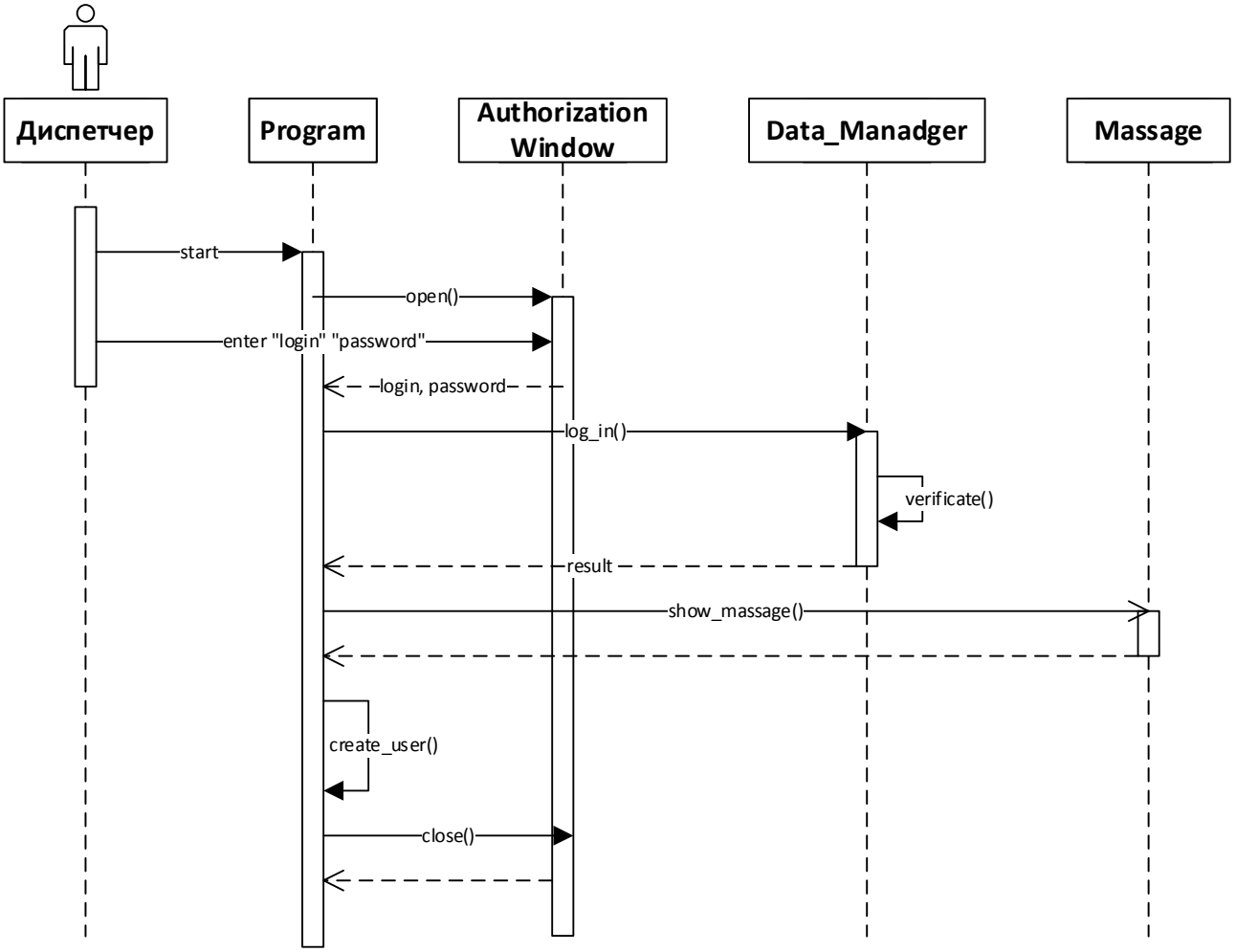


Рисунок 2.13 – Діаграма діяльності для прецеденту «Авторизуватися»

Діаграма діяльності для прецеденту «Виконати аналіз системи» представлена на рисунку 2.14

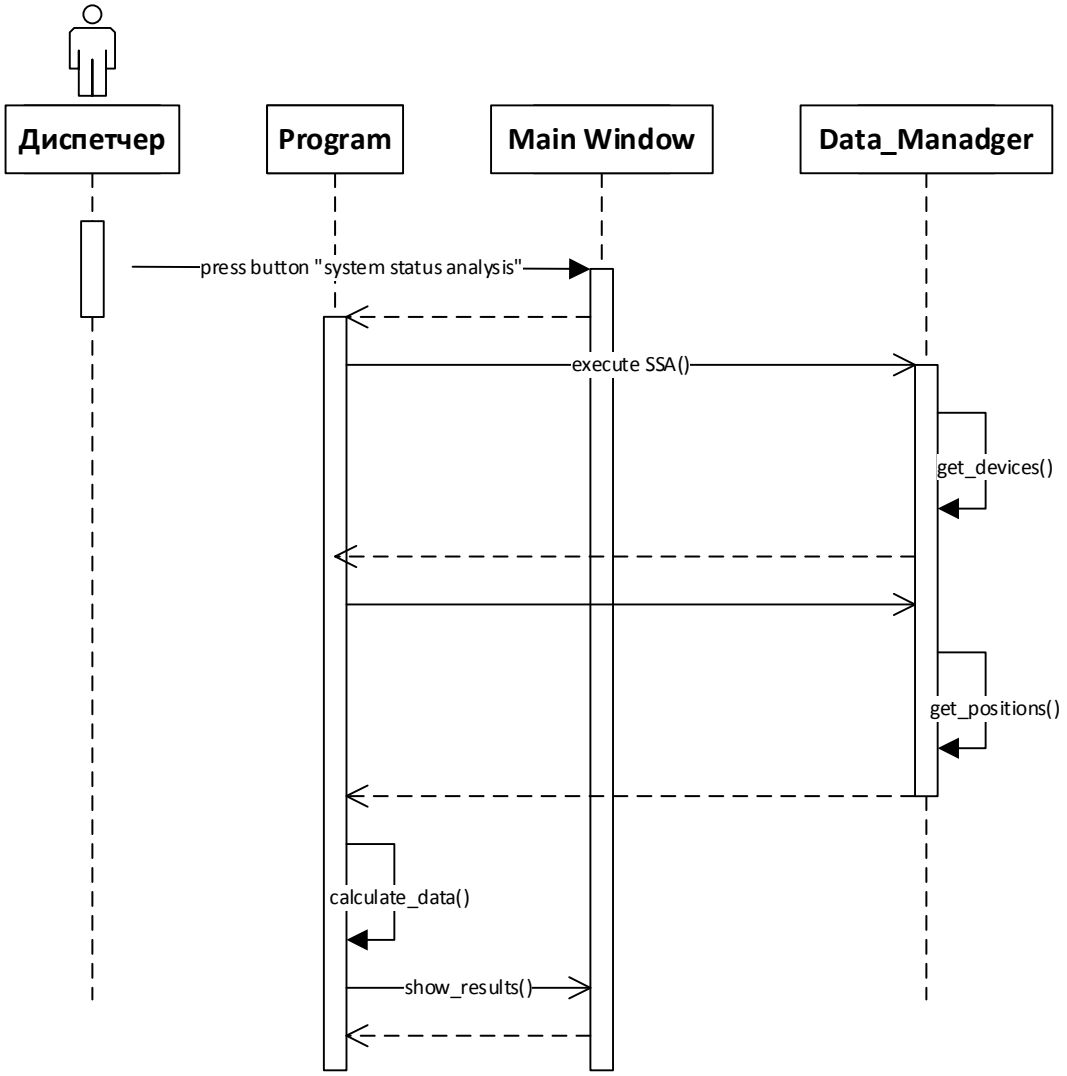


Рисунок 2.14 – Діаграма діяльності для прецеденту «Виконати аналіз системи»

Діаграма діяльності для прецеденту «Виконати аналіз маршруту» представлена на рисунку 2.15

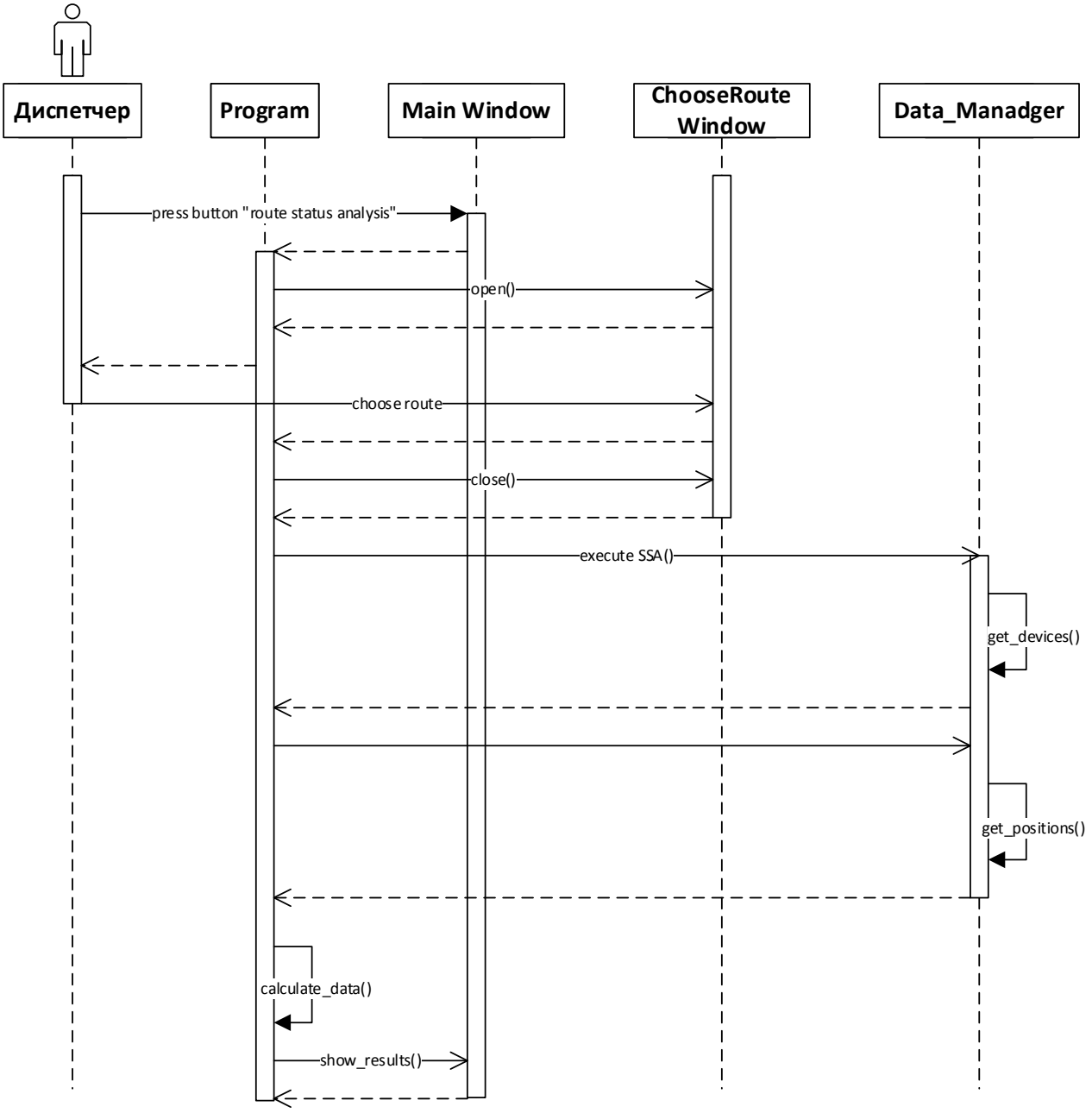


Рисунок 2.15 – Діаграма діяльності для прецеденту «Виконати аналіз маршруту»

В даному підпункті була розглянута динамічна модель програмного забезпечення на прикладі діаграм діяльності варіантів використання «Авторизуватися», «Виконати аналіз стану системи» та «Виконати аналіз стану маршруту»

2.3 Робочий проект програмного забезпечення

В робочому проекті програмного забезпечення представлені вибір і обґрунтування інструментів розробки й реалізація та тестування основних класів еквівалентності програмного забезпечення.

2.3.1 Вибір та обґрунтування інструментів розробки

Вибір мови й інструментів розробки виконується відповідно умов поставлених замовником та відображених у підрозділі 1.3 та ТЗ.

Python – динамічна інтерпретована об’єктно-орієнтована скриптова мова програмування із строгою динамічною типізацією. Розроблена в 1990 році голандським програмістом Гвідо ван Россумом . Python – багатоцільова мова програмування, яка дозволяє писати код, що добре читається. Відносний лаконізм мови Python дозволяє створити програму, яка буде набагато коротше свого аналога, написаного на іншій мові [14].

Python – багатоплатформова мова програмування. Це означає, що програми на Python можна запускати в різних операційних системах без будь-яких змін.

Ще однією перевагою Python є його стандартна бібліотека, яка встановлюється разом з Python і містить готові інструменти для роботи з операційною системою, веб-сторінками, базами даних, різними форматами даних, для побудови графічного інтерфейсу програм тощо.

Програми, написані на мові програмування Python, можуть бути як невеликими скриптами, так і складними системами.

API – це інтерфейс, за допомогою якого додаток або операційна система комп'ютера може використовувати сторонні функції чи програми [15].

Google API – це набір інтерфейсів прикладного програмування, що дозволяє клієнту взаємодіяти з інтегрованими сервісами. Це дає можливість

створювати прості програми для складніших програмних рішень на основі розташування для Інтернету, iOS та Android [16].

JSON (JavaScript Object Notation) – простий формат обміну даними, що є зручним як для читання та написання людиною, так і для парсінгу та генерації комп'ютером. Він базується на підмножині мови програмування JavaScript стандарту ECMA-262 3rd Edition – December 1999. JSON – це текстовий формат, повністю незалежний від мови реалізації, але він використовує конвенції, знайомі програмистам C-подібних мов, таких як C, C++, C#, Java, JavaScript, Perl, Python та багатьох інших. Ці властивості роблять JSON ідеальною мовою для обміну даними [17].

Google Sheets (Google SpreadSheets – це онлайн-додаток, за допомогою якого ви можете створювати та форматувати таблиці, а також працювати над ними спільно з іншими користувачами [18].

2.3.2 Реалізація та тестування основних класів еквівалентності програмного забезпечення

Тестування основних класів еквівалентності представлене тестуванням варіантів використання «Авторизуватися», «Виконати пошук» стратегією чорної скриньки методом еквівалентного розбиття**.** Чорний ящик (Black Box) – цей метод заснований на тому, що тестувальник має доступ до ПЗ тільки через ті ж інтерфейси, що і замовник або користувач, або зовнішні інтерфейси, що дозволяють іншому комп’ютеру або іншому процесу підключитися до системи для тестування. Еквівалентне розбиття: Розробка тестів методом чорного ящика, в якому тестові сценарії створюються для перевірки елементів еквівалентної області. Як правило, тестові сценарії розробляються для покриття кожній області як мінімум один раз [19].

Тестування функції «Авторизуватися».

Після запуску програми, диспетчеру висвітлюється віконна форма авторизації рис.2.10, на якій потрібно пройти процедуру авторизації. Форма складається з двох полів для вводу тексту – «логін» і «пароль» та кнопки «Авторизуватися».

Класи еквівалентності наведені в таблиці 2.11.

Таблиця 2.11 – Класи еквівалентності функції «Авторизуватися »

Вхідні умови	Правильні класи еквівалентності	Неправильні класи еквівалентності
Логін	1.Символи й числа, окрім спец. символів	2. Спец символи, пусте поле
Пароль	1.Символи й числа, окрім спец. символів	2. Спец символи, пусте поле

Тести з правильних класів еквівалентності наведено в таблиці 2.12.

Таблиця 2.12 – Тести з правильних класів еквівалентності

№ Тесту	№ покритогокласу еквівалентності	Вхідні дані	Результат
1	1	Login12345	В результаті тестування з’являється напис «Авторизація виконана успішно»
	1	Password12345	

Тести з неправильних класів еквівалентності наведено в таблиці 2.13

Таблиця 2.13 – Тести з неправильних класів еквівалентності

№ Тесту	№ покритого класу еквівалентності	Вхідні дані	Результат
1	2	login**123	В результаті тестування з’являється повідомлення «Помилка у форматі даних»
2	4	Password 123	В результаті тестування з’являється повідомлення «Помилка у форматі даних»
3	5	пустота	В результаті тестування з’являється повідомлення «Помилка у форматі даних»
4	6	символи пробілу	В результаті тестування з’являється повідомлення «Помилка у форматі даних»

Тестування функції «Виконати пошук».

Після успішної авторизації диспетчеру відкривається головне вікно програми, яке представлене на рис.2.11. В цьому вікні диспетчер може виконати пошук за ключовими словами. В правій нижній частині вікна знаходиться форма для введення ключових слів «Ключові слова» та кнопка «Знайти».

Класи еквівалентності наведені в таблиці 2.14.

Таблиця 2.14 – Класи еквівалентності функції «Виконати пошук»

Вхідні умови	Правильні класи еквівалентності	Неправильні класи еквівалентності
Ключові слова	1.Символи й числа, символ «=», формули виду (характеристика=стан)	2. Спец. символи, окрім «=», пусте поле, символи пробілу або табуляції

Тести з правильних класів еквівалентності наведено в таблиці 2.15.

Таблиця 2.15 – Тести з правильних класів еквівалентності

№ Тесту	№ покритогокласу еквівалентності	Вхідні дані	Результат
1	1	status=online	В результаті тестування результати пошуку з'являються на віконній формі в полі «Результати»

Тести з неправильних класів еквівалентності наведено в таблиці 2.16.

Таблиця 2.16 – Тести з неправильних класів еквівалентності

№ Тесту	№ покритого класу еквівалентності	Вхідні дані	Результат
1	2	“online”	В результаті тестування з’являється повідомлення «Помилка у форматі даних»
2	4	=	В результаті тестування з’являється повідомлення «Помилка у форматі даних»
3	5	пустота	В результаті тестування з’являється повідомлення «Помилка у форматі даних»
4	6	символи пробілу	В результаті тестування з’являється повідомлення «Помилка у форматі даних»

2.3.3 Випробування програмного забезпечення

Програму і методику випробувань розробленого програмного забезпечення наведено в Додатку Д. На основі інструкцій, що наведені в цьому додатку, проведемо випробування розробленого програмного забезпечення.

Перевірка працездатності програми виконується згідно з вимогами до функціональних характеристик ПЗ. Перевірка вважається завершеною у разі відповідності складу і послідовності виконаних дій вимогам до функціональних характеристик. В процесі тестування була перевірена функціональність за темою «Розробка програмного забезпечення для автоматизації процесів обробки даних системи електронного контролю транспортного складу маршрутного таксі міста Миколаєва»». У табл. 2.17 вказано перелік випробувань основних функціональних можливостей.

Таблиця 2.17 – Перелік випробувань основних функціональних можливостей

№	Дія	Очікуваний результат	Результат перевірки	Зауваження
1	2	3	4	5
1	Перевірка правильності введення даних при авторизації	Програма повинна перевірити введені дані	Виконано	
2	Перевірка правильності введення даних при виконанні пошуку	Програма повинна перевірити коректність введених даних	Виконано	
3	Перевірка правильності додавання нового ID документу	Програма повинна перевірити коректність введених даних	Виконано	

Продовження табл. 2.17

1	2	3	4	5
4	Перевірка правильності обрання маршрутів	Програма повинна перевірити правильність обраного маршруту	Виконано	
5	Перевірка правильності вказання дати автоматичного виконання програми	Програма повинна перевірити правильність вказаної дати й часу виконання програми	Виконано	

Повний перелік випробування основних функціональних можливостей програмного забезпечення наведений в Додатку Д.

3 РЕЗУЛЬТАТИ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ АВТОМАТИЗАЦІЇ ПРОЦЕСІВ ОБРОБКИ ДАНИХ СИСТЕМИ ЕЛЕКТРОННОГО КОНТРОЛЮ ТРАНСПОРТНОГО СКЛАДУ

У даній кваліфікаційній роботі було виконано аналіз та постановку задачі до розробки програмного забезпечення для автоматизації процесів обробки даних системи електронного контролю транспортного складу маршрутного таксі міста Миколаєва, розроблено проект програмного забезпечення для автоматизації процесів обробки даних системи електронного контролю транспортного складу, а також представлені результати розробки.

У даній кваліфікаційній роботі було вирішено практичну задачу інженерії програмного забезпечення, що характеризується комплексністю та невизначеністю умов.

Задача розробленого ПЗ – автоматизувати процес обробки даних системи електронного контролю транспортного складу, зменшити об’єм виконуваної роботи працівниками КП «МІОЦ», пришвидшити процес обробки даних, зменшити ймовірність виникнення помилок, спростити ведення звітності.

Також у даній кваліфікаційній роботі було розроблено технічне завдання, опис програми, програми та методики випробувань програмного продукту та інструкцію користувача за темою кваліфікаційної роботи «Розробка програмного забезпечення для автоматизації процесів обробки даних системи електронного контролю транспортного складу маршрутного таксі міста Миколаєва».

Результати розробки програмного забезпечення повністю відповідають поставленій меті. В ході роботи над кваліфікаційною роботою було виявлено основні переваги та недоліки існуючих виробів аналогів, були внесені додаткові фактори, що якісно впливають на терміни та ефективність

виконання закладених функцій.

Головне вікно розробленої програми представлено на рисунку 3.1

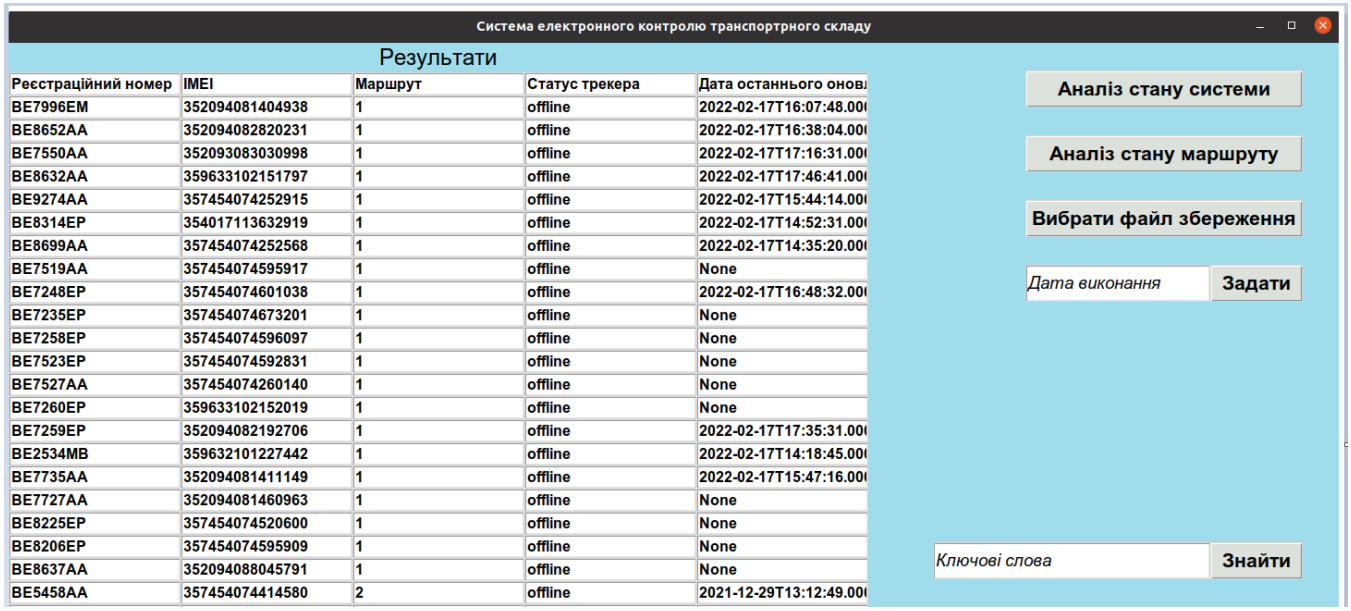


Рисунок 3.1 – Головне вікно програми

Більш детально опис розробленого програмного забезпечення та інструкція користувача наведені у додатках до кваліфікаційної роботи.

4 ОХОРОНА ПРАЦІ

4.1 Основні положення Закону України «Про охорону праці»

Охорона праці – це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження життя, здоров'я і працездатності людини у процесі трудової діяльності [20] .

Охорона праці містить три основних складових частини: правові норми трудового законодавства, виробничу санітарію, гігієну та техніку безпеки, а також протипожежний захист і електробезпеку .

Мета охорони праці – забезпечення безпечних, нешкідливих і сприятливих умов праці через вирішення багатьох складних завдань, основними з яких є:

- проектування підприємств, технологічних процесів і конструювання обладнання з обов'язковим виконанням вимог охорони праці;

- знаходження оптимальних співвідношень між різними факторами виробничого середовища, що дозволяє забезпечити мінімум несприятливого впливу їх на здоров'я працівників;

- встановлення, законодавче оформлення визначених норм кожного з несприятливих або небезпечних факторів, систематичний король за їх застосуванням;

- розробка конкретних заходів щодо покращення умов праці та забезпечення її безпеки на основі застосування у виробництві новітніх досягнень науки 1 техніки;

- застосування раціональних засобів захисту працівників від впливу несприятливих факторів виробничого середовища, а також втілення організаційних заходів, які нейтралізують або послаблюють ступінь їх впливу на організм людини;

- розробка та застосування методів і засобів оцінки ефективності заходів з охорони праці, що плануються і здійснюються .

Законодавство про працю містить норми і вимоги з техніки безпеки і виробничої санітарії, норми, що регулюють робочий час і час відпочинку, звільнення та переведення на іншу роботу, норми праці щодо жінок, молоді, гігієнічні норми і правила тощо.

Правовою основою законодавства щодо охорони праці є Конституція України, Закони України: «Про охорону праці», «Про охорону здоров'я», «Про пожежну безпеку», «Про використання ядерної енергії та радіаційний захист», «Про забезпечення санітарного та епідеміологічного благополуччя населення», а також Кодекс законів про працю України.

Основоположним законодавчим документом в галузі охорони праці є Закон України «Про охорону праці» [21], дія якого поширюється на всі підприємства, установи і організації незалежно від форм власності та видів їх діяльності, на усіх громадян, які працюють, а також залучені до Праці на цих підприємствах.

Верховна Рада України 14 жовтня 1992 року прийняла Закон України «Про охорону праці». Цей Закон визначає основні положення щодо реалізації конституційного права громадян про охорону їх життя і здоров'я в процесі трудової діяльності, регулює за участю відповідних державних органів відносини між власником підприємства, установи і організації або уповноваженим ним органом і працівником з питань безпеки, гігієни праці та виробничого середовища і встановлює єдиний порядок організації охорони праці в Україні.

4.2 Аналіз шкідливих та небезпечних факторів на робочому місці

Шкідливі виробничі фактори – фактори середовища і трудового процесу, які можуть викликати професійну патологію, тимчасове або стійке

зниження працездатності. В таблиці 4.1 перераховані шкідливих та небезпечних виробничих факторів.

Таблиця 4.1 – Перелік шкідливих та небезпечних виробничих факторів

Найменування факторів	Можливі джерела їх виникнення	Характер дії
Небезпека ураження електричним струмом	Мережа живлення	Небезпечний
Пожежонебезпечність приміщень	Наявність матеріалів, що загорають і джерел запалення (електроапаратура)	Небезпечний та шкідливий
Електромагнітне випромінювання в тому числі і рентгенівське	ЕПТ (Дисплей є джерелом рентгенівського, радіочастотного, ультрафіолетового й інфрачервоного випромінювання та випромінювання звукового діапазону)	Шкідливий
Статична електрика	ЕПТ монітору і діелектрична поверхня екрана	Шкідливий
Іонізація повітря	Статична електрика і рентгенівське випромінювання	Шкідливий
Підвищений рівень шуму	Шум створюється перетворювачем напруги ЕОМ, її технічною периферію, а також людьми, що працюють в аудиторії	Шкідливий
Несприятлива освітленість	Недостатнє штучне і природне освітлення	Шкідливий
Незадовільні параметри мікроклімату	Незадовільний стан системи опалення і вентиляції	Шкідливий
Психофізіологічні напруження	Монотонність праці, перенапруженість зорових аналізаторів	Шкідливий

Робота на персональному комп'ютері супроводжується постійною і значною напругою функцій зорового аналізатора. Розлад органів зору різко збільшується при роботі понад чотири годин на день .

Нервово-емоційне напруження при роботі на ПК виникає внаслідок дефіциту часу, великого об'єму і щільності інформації, особливостей

діалогового режиму спілкування людини і ПК, відповідальності за безпомилковість інформації. Тривала робота на дисплеї, особливо в діалоговому режимі, може призвести до нервово-емоційного перенапруження, порушення сну, погіршення стану, зниження концентрації уваги та працездатності, хронічного головного болю, підвищеної збудливості нервової системи, депресії.

Підвищені статичні і динамічні навантаження у користувачів ПК призводять до скарг на болі в спині, шийному відділі хребта і руках. З усіх нездужань, обумовлених роботою на комп'ютерах, частіше зустрічаються ті, які пов'язані з використанням клавіатури. У період виконання операцій введення даних кількість дрібних стереотипних рухів кистей і пальців рук за зміну може перевищити 60 тис., що відповідно до гігієнічної класифікації праці відноситься до категорії шкідливих і небезпечних. Більшість працюючих рано чи пізно починають пред'являти скарги на болі в шиї і спині. Ці нездужання накопичуються поступово і отримали назву «синдром тривалих статичних навантажень».

Іншою причиною виникнення СДСН може бути тривале перебування в положенні «сидячи», яке призводить до сильного перенапруження м'язів спини і ніг. Основною причиною перенапруги м'язів спини і ніг є нераціональна висота робочої поверхні столу і сидіння, відсутність опорної спинки і підлокітників, незручне розміщення монітора, клавіатури та документів, відсутність підставки для ніг.

Ще одним небезпечним фактором може стати недостатність освітлення, що приводить до напруги зору, ослабляє увагу, приводить до настання передчасної стомленості. Надмірно яскраве освітлення викликає засліплення, роздратування і різь в очах. неправильний напрям світла на робочому місці може створювати різкі тіні, відблиски, дезорієнтувати працюючого. Всі ці причини можуть привести до нещасного випадку або профзахворювань, тому такий важливий правильний розрахунок освітленості.

Тривала і інтенсивна робота на комп'ютері може стати джерелом важких професійних та звичайних захворювань таких як:

- неправильна постава – це призводить до подальшого розвитку викривлення хребта сколіозу, лордозу, кіфозу, а як результат головні болі, болі в області шиї і всього хребта, болі в області таза. Неправильно положення ніг може привести до артриту (запалення суглобів), артрозу (деформації);

- тунельний синдром – найвідоміше захворювання людей працюють за комп'ютером. Воно ж синдром зап'ястного каналу. Тунельний синдром проявляється після кількох годин напруженої роботи за комп'ютером;

- геморой - хвороба аналогічна варикозного розширення вен. Причиною хвороби може бути застій крові у венах;

- розвиток короткозорості – через те, що екран монітора за контрастом вище, ніж навколишні об'єкти розвивається короткозорість;

- порушення фокусування – наслідком напруженої роботи за монітором є порушення фокусування, яка може бути викликана перенапруженням очних м'язів;

- сухість очей – через рефлексу, заснованого на тому, що при погляді на джерело світла очей починає менше моргати виникає «обсушування» рогівки ока яке призводить до очних болів.

4.3 Заходи щодо зменшення впливу шкідливих факторів роботи за ПК

Працюючи за комп’ютером, рекомендуємо дотримуватися правил тривалості роботи, правильної постави, розміру шрифтів та зображень, вимог до приміщення тощо.

Принципи правильної роботи за комп’ютером:

- у робочому приміщенні (кімнаті), де встановлені комп’ютери, щодня потрібно виконувати вологе прибирання;

- приміщення, у якому знаходяться комп’ютери, потрібно провітрювати щогодини;

- після кожного часу роботи рекомендується робити десяти хвилинну перерву, яку зручно суміщати з провітрюванням. За будь-яких умов безперервна робота за комп’ютером для дорослої людини не повинна перевищувати двох годин. Під час перерви не варто читати або дивитися телевізор;

- необхідно постійно слідкувати за станом екрану монітора: він має бути чистим, без плям та пилу;

- потрібно слідкувати за поставою: ноги повинні твердо стояти на підлозі чи на спеціальній підставці; стегна повинні бути розташовані під прямим кутом до тулуба, а гомілки – під прямим кутом до стегон; сидіти потрібно прямо або злегка нахилившись вперед; відстань від очей до екрану монітора – не менше 55-60 см; центр екрану має знаходитися на рівні очей чи трохи нижче; рекомендується хоча б раз на день виконувати гімнастику для очей;

- щоб попередити „синдром сухого ока”, потрібно моргати кожні 3-5 секунд;

- не використовувати звичайний телевізор замість монітору;

- у процесі роботи рекомендується періодично (приблизно раз на 20-30 хвилин) переводити погляд з екрану на найбільш віддалений предмет у кімнаті, а ще краще – на віддалений об’єкт за вікном;

- якщо з’явилося відчуття втоми, напруження, сонливості, тяжкості в очах, потрібно припинити роботу та хоча б трохи відпочити.

Рекомендовані умови для роботи за комп'ютером:

- твердий стілець. край стільця не повинен тиснути на судини під колінами.;

- відстань до монітора повинна бути 50-70см;

- перерву в сидячій роботі, 15-20 хвилин кожні 1-2 години;

- правильне освітлення робочого місця. при слабкому світлі очі напружуються і болять;

- потрібно закривати очі для відпочинку. час від часу відводите очі вбік, щоб дати відпочити своєму зору.

ВИСНОВКИ

У даній кваліфікаційній роботі було виконано аналіз та постановку задачі до розробки програмного забезпечення для автоматизованої обробки даних системи електронного контролю транспортного складу маршрутного таксі міста Миколаєва, розроблено проект програмного забезпечення, а також представлені результати розробки та спеціальний розділ з охорони праці.

У даній кваліфікаційній роботі було вирішено практичну задачу інженерії програмного забезпечення, що характеризується комплексністю та невизначеністю умов.

Задача вказаного ПЗ – автоматизувати обробку даних системи електронного контролю транспортного складу, автоматизувати процес збереження результатів обробки даних, спростить процес ведення звітності, надавати ефективний механізм пошуку інформації.

Також у даній кваліфікаційній роботі було розроблено технічне завдання, опис програми, програми та методики випробувань програмного продукту та інструкцію користувача за темою кваліфікаційної роботи «Розробка програмного забезпечення для автоматизованої обробки даних системи електронного контролю транспортного складу маршрутного таксі міста Миколаєва».

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Система глобального позиціонування GPS і як він працює на мобільних телефонах. URL: http://gsm-ka.com.ua/ua/chto-takoe-gps-i-kak-on-rabotaet-na-mobilnykh-telefonakh/ (дата звернення: 27.05.22).

2. Рішення Миколаївського виконавчого комітету №155 від 06.03.15. URL: https://mkrada.gov.ua/documents/22370.html (дата звернення: 17.05.22).

3. Структурні підрозділи держорганів. URL: https://i.factor.ua/ukr/journals/bb/2020/may/issue-19/article-108565.html (дата звернення: 17.05.22).

4. Виконавчі органи Миколаївської міської ради. URL: https://mkrada.gov.ua/content/vikonavchiy-komitet.html (дата звернення: 17.05.22).

5. Транспортна інфраструктура міста Миколаєва. URL: https://mkrada.gov.ua/content/transport-tarifi-marshruti-rozklad.html (дата звернення: 17.05.22).

6. 10 Best Data Analysis Tools. URL: https://uk.myservername.com/10-best-data-analysis-tools (дата звернення: 17.05.22).

7. Сайт Microsoft Power BI. URL: https://www.scnsoft.com/services/business-intelligence/microsoft/power-bi#power-bi-in-brief (дата звернення: 17.05.22).

8. Сайт RapidMiner. https://rapidminer.com/platform/automated-data-science/ (дата звернення: 17.05.22).

9. Сайт Zoho Analytics. URL:https://www.zoho.com/analytics/ (дата звернення: 17.05.22).

10. Уніфікована мова моделюваннння UML. URL: http://www.znannya.org/?view=uml (дата звернення: 17.05.22).

11. Модель «сутність – зв’язок». URL: https://www.lucidchart.com/pages/ru/erd-диаграмма (дата звернення: 17.05.22).

12. Користувацький інтерфейс. URL: https://ela.kpi.ua/bitstream/123456789/38430/1/Zvolikevych_magistr.pdf (дата звернення: 17.05.22).

13. Поняття інтерфейсу. Типи користувацького інтерфейсу та етапи їх розробки. URL: https://allcompositions.at.ua/temy_1-2.pdf (дата звернення: 17.05.22).

14. Мізюк О. Путівник мовою програмування Python. URL: https://pythonguide.rozh2sch.org.ua/ (дата звернення: 17.05.22).

15. Що таке API. URL: https://support.google.com/admob/answer/10284360?hl=uk (дата звернення: 17.05.22).

16. Google API. URL: https://habr.com/ru/hub/google_api/ (дата звернення: 17.05.22).

17. Вступ в JSON. URL: https://www.json.org/json-uk.html (дата звернення: 17.05.22).

18. Як працювати з додатком Google SpreadSheets. URL: https://support.google.com/docs/answer/6000292?hl=ru&co=GENIE.Platform%3DDesktop (дата звернення: 17.05.22).

19. Тестування методом «Чорної скриньки». URL: https://qalearning.com.ua/theory/lectures/material/requirements-testing-methods-equivalence/ (дата звернення: 17.05.22).

20. Охорона та умови праці. URL: https://umanupszn.gov.ua/2017/03/15/ponyattya-okhoroni-praci/ (дата звернення: 17.05.22).

21. Закон України «Про охорону праці». URL: http://www.jobs.ua/ukr/pravo/labour_protection/lib-article-209/ (дата звернення: 17.05.22).

Додаток А Технічне завдання

1 Вступ

Програмне забезпечення для автоматизованої обробки даних системи електронного контролю транспортного складу маршрутного таксі міста Миколаєва. Програма призначена для автоматизованої обробки даних системи електронного контролю транспортного складу маршрутного таксі. Система електронного контролю транспортного складу – це комплекс засобів програмного й технічного забезпечення для надсилання, зберігання й обробки даних про рух транспортного складу.

2 Підстави для розробки

Розробка ведеться по замовленню комунального підприємства «Міський інформаційно-обчислювальний центр», скорочено КП «МІОЦ».

3 Призначення розробки

3.1 Експлуатаційне призначення

Автоматизація процесів обробки даних, зменшення об’єму ручної виконуваної роботи працівниками КП «МІОЦ», зменшення ймовірності виникнення помилок, спрощення процесу ведення звітності.

3.2 Функціональне призначення

- обробка вхідних даних з серверу;
- автоматичне збереження результатів обробки у вигляді таблиць;
- формування звітів у вигляді таблиць;
- зберігання й використання інформації.

4 Вимоги до програмного забезпечення

4.1 Вимоги до функціональних характеристик

Програмне забезпечення має реалізовувати наступні функції:

- аналіз стану системи GPS трекінгу – підрахунок кількості активних та не активних трекерів на всіх маршрутах, формування списків;

- задавання дати й часу автоматичного виконання аналіз стану системи GPS трекінгу;

- автоматичне зберігання результатів – зберігання інформації, отриманої в результаті аналізу даних, в вигляді Google Spreadsheets таблиць;

- аналіз маршрутів – зібрати інформацію про діючий на маршруті транспорт (Реєстраційний номер, IMEI трекера, номер CIM-картки, статус трекера, дата останнього оновлення);

- пошук – знаходження GPS-трекеру й інформації про нього через ключові слова.

4.2 Організація вхідних та вихідних даних:

Вхідні дані:

- дані отримані з пристроїв введенні інформації (мишка та клавіатура);

- файли типу . json з інформацію про трекери й їх місцезнаходження (positions.json, devices.json).

Вихідні дані:

- звіти з результатами обробки даних.

4.3 Вимоги до надійності

Програмне забезпечення має виконувати свої функції без виникнення збоїв при умові стабільної й коректної роботи, комп’ютера, встановленої на ньому операційної системи, веб-браузера й інтернет з’єднання.

У разі виникнення проблем або збоїв в роботі комп’ютера чи операційної системи, перезавантаження програми має відновити нормальну роботу програмного продукту.

В разі якщо збій виник в момент обробки даних або збереження результатів обробки, отриманий прогрес має бути відкинутий, а отримані результати видалені, обробка даних має початися спочатку для отримання й

збереження точного результату.

У випадку, якщо коректна робота програмного продукту не можлива, користувач має отримати відповідне повідомлення.

Якщо користувач програми при введені даних використовує некоректну інформацію, користувачеві має бути виведено повідомлення про помилку в веденні даних.

4.4 Умови використання

Необхідним є комп’ютер з доступом в інтернет розташований в закритому, провітрюваному приміщенні з вологістю повітря не більше 50% й температурою приблизно 20 градусів Цельсія (+- 5 градусів Цельсія) для забезпечення стабільної роботи обладнання.

Необхідні навички користувача для використання програми:

- мінімальні навички в користуванні комп’ютером;
- мінімальні навички в користуванні веб-браузером;
- достатні навички для користування Google sheets:

4.5 Вимоги до складу параметрів технічних засобів

Для коректної роботи програмного забезпечення комп’ютер повинен відповідати наступним мінімальним системним вимогам:

- CPU: Intel Core i3– 2.93 МГц;
- RAM: 4 GB;
- HDD: 2 GB вільного місця;
- доступ до інтернету.

4.6 Вимоги до інформаційної й програмної сумісності

Для коректної роботи програмного забезпечення необхідна ОС Linux дистрибутив Ubuntu версії не нижче 20.04 з встановленим інтерпретатором Python 3, встановленим пакетом virtualen та налаштованим віртуальним оточенням (env). Веб-браузер Google Chrome версії не нижче 100.0. В браузері

мають бути встановлені два розширення: API Google Drive і API Google Spreadsheets.

5 Вимоги до програмної документації

Програмна документація повинна містити:

- технічне завдання (ТЗ);
- текст програми;
- опис програми;
- інструкцію користувача;
- програму і методику випробувань ПЗ.

6 Стадії та етапи розробки

Стадії та етапи розробки представлені в таблиці А.1.

Таблиця А.1 – Стадії та етапи розробки програмного забезпечення

Стадії розробки	Етапи робіт	Термін виконання робіт	
1 Технічне завдання	1.1Обгрунтування необхідності розробки програми	14.02.22	28.02.22
	1.2 Розробка технічного завдання	01.03.22	20.03.22
	1.3 Затвердження технічного завдання	21.03.22	27.03.22
2 Ескізний проект	2.1 Розробка ескізного проекту	28.03.22	10.04.22
	2.2 Затвердження ескізного проекту	11.04.22	17.04.22
3 Технічний проект	3.1 Розробка технічного проекту	18.04.22	30.04.22
	3.2 Затвердження технічного проекту	01.05.22	05.05.22
4 Робочий проект	4.1 Розробка програми	06.05.22	15.05.22
	4.2 Розробка програмної документації	16.05.22	17.05.22
	4.3 Випробування програми	18.05.22	20.05.22

7 Порядок контролю та приймання

Контроль за аналізом та проектуванням кожної окремої частини програмного забезпечення відбувається на кожному етапі з урахуванням вимог, визначених у технічному завдані. Кожна стадія розробки повинна бути представлена в зазначені строки та узгоджена із замовником.

Приймання проводиться відповідно до програми і методики випробувань і документується за допомогою протоколу проведення випробувань. У разі знаходження помилок під час приймання програмного забезпечення, складається акт про знайдені помилки, який підписується представниками замовника і розробника і затверджуються керівником організації–замовника та організації–розробника. Розробник повинен, на протязі не більше ніж 2 тижні, виправити зазначені помилки і оповістити замовника про повторне проведення перевірки.

Додаток Б Код програми

```
import httplib2
import apiclient.discovery
import requests
from oauth2client.service_account import ServiceAccountCredentials
from datetime import timedelta, datetime
import tkinter
from tkinter import *
from tkinter import messagebox
from tkinter import ttk


# Количество дней, по истечению которых устройство считается
# не активным считая с даты последней активности
DELTA_TIME = 5


def show_message(text):
    messagebox.showinfo("Повідомлення", text)


class ProgramController(object):


    def startProgram(self):
        self.auth_window = AuthorizationWindow(self)          # створюємо вікно авторизації
        self.auth_window.mainloop()                           # виводимо вікно на екран

        if( not self._user._auth_status):                     # якщо статус авторизації = false
            return                                            # перервати виконання програм
        else:
            fc1 = FileController('https://tracking.mkrada.gov.ua/api/devices')
            fc2 = FileController('https://tracking.mkrada.gov.ua/api/positions')
            self.devices_json = fc1.get_file_json(self._user.get_login(), self._user.get_password())
            self.positions_json = fc2.get_file_json(self._user.get_login(), self._user.get_password())
            del fc1
            del fc2
            self.main_window = MainWindow(self)               # відкриваємо головне вікно програми
            self.main_window.mainloop()                       # виводимо вікно на екран


    def authorize(self, login, password):
        fc = FileController('https://tracking.mkrada.gov.ua/api/devices')
        res = fc.get_acsses(login, password)

        if not res:
            show_message('Помилка. Вказано невірний логін, або пароль')
            del fc
        else:
            self._user = User(login, password)
            self._user.set_auth_status(True)
            show_message('Авторизація виконана успішно!')
            self.auth_window.destroy()
```

```
        del fc

    def execute_SSA(self, root):
        table = [['Маршрут', 'Всього трекерів', 'Активних', 'Відомих', 'Не відомих']]            # Перша строчка в таблиці

        for dev in self.devices_json:                                                    # Проходимся по струкрурам даних в json файлі
            route = self.find_Route(str(dev['attributes']['Route']), str(dev['groupId']))          # Повертає true якщо маршрут наявний в системі,

            if not route:
                print('Find new Group ID!!!!!!!')
                continue

            if self.find_Row(table, route):                                              # Якщо в table є строка з маршрутом = route
                self.add_Data(table, route, str(dev['id']), str(dev['lastUpdate']))          # Додати в цю строку нові дані
            else:
                self.add_Row(table, route, str(dev['id']), str(dev['lastUpdate']))           # Інакше додати в table нову строчку


        self.clear_grid(root)                                                            # Очистити grid
        lable_head = Label(root, text="Результати", fg="#000000", bg="#9fdcec", font=("Arial","18"))       # Заголовок в першому рядку grid
        lable_head.grid(row=0, column=2, sticky="N")
        self.write_grid(root, table)                                                     # Записати в grid таблицю


    def execute_RSA(self, second_window, main_window, lst_var):
        table = [['Реєстраційний номер', 'IMEI', 'Маршрут', 'Статус трекера', 'Дата останнього оновлення']]

        for i in range(len(lst_var)):
            if lst_var[i].get() == 1:
                row = second_window.grid_slaves(row=i+1)
                group_id = main_window._controller.find_groupID(row[0].get())
                for dev in self.devices_json:
                    if str(dev['groupId']) == group_id:
                        table.append([str(dev['name']), str(dev['uniqueId']), str(dev['attributes']['Route']), str(dev['status']), str(dev['lastUpdate'])])

        main_window._controller.clear_grid(main_window)
        lable_head = Label(main_window, text="Результати", fg="#000000", bg="#9fdcec", font=("Arial","18"))
        lable_head.grid(row=0, column=2, sticky="N")
        main_window._controller.write_grid(main_window, table)
        second_window.destroy()


    def find(self, key_value):
        if(key_value==''):
            messagebox.showinfo("Пошук", "Поле для ключових слів пусте!")
        else:
            table = [['Name', 'UniqueId', 'Route', 'Status', 'LastUpdate']]
            for dev in self.devices_json:
                string = str(dev)
                string = string.replace('\', '')
```

```
            string = string.replace(':', '=')
            string = string.replace(' ', '')
            if key_value in string:
                table.append([str(dev['name']),
                              str(dev['uniqueId']),
                              str(dev['attributes']['Route']),
                              str(dev['status']),
                              str(dev['lastUpdate'])])

        if(len(table) > 1):
            self.clear_grid(self.main_window)
            lable_head = Label(self.main_window, text="Результати", fg="#000000", bg="#9fdcec",
font=("Arial","18"))
            lable_head.grid(row=0, column=2, sticky="N")
            self.write_grid(self.main_window, table)
        else:
            messagebox.showinfo("Пошук", "Збігів по заданому ключу не знайдено")


    def find_Route(self, route, group_id):                    # Повертає номер маршуту, якщо маршрут
не знайдено повертає false
        if route == '0':                                      # Якщо номер маршрут = 0 - авто знято з маршруту
            return 'Снят с маршрута'
        else:
            file = open('Route_GroupId.txt', 'r')             # Відкриваємо файл, в якому зберігається
відвідність route і groupId
            for line in file:                                 # Проходимо по файлу
                lst = line.split(':')                         # Розбиваємо строку на дві підстроки, приклад
'15:120' -> ['15', '120\n']
                if lst[1][:-1] == group_id:                   # [:-1] - це друга підстрока без символу'\n'
                    return lst[0]                             # Повертаємо номер маршруту
        return False                                          # Якщо маршрут не знайдено повертає false


    def find_Row(self, table, route):                         # Повертає номер строки в таблиці table з
маршрутом = route
        for i in range(len(table)):
            if table[i][0] == route:                          # Приклад строки ['route', 0, 0, 0, 0]
                return i
        return False                                          # Якщо строчка з маршрутом = route не знайдено,
повертає false


    def add_Row(self, table, route, device_id, date):         # add new array to sheet for new route
        table.append([route, 0, 0, 0, 0])
        self.add_Data(table, route, device_id, date)


    def add_Data(self, table, route, device_id, date):        # Додає дані про трекер в відповідний
маршрут
        i = self.find_Row(table,route)                        # індекс строки з маршрут = route
        if not self.check_tracker_info(device_id):            # check file positions.json not contain info
about device
            table[i][1] += 1
            table[i][4] += 1                                  # device don`t send any data ['route', 0, 0, 0, 0+1]
        else:
            table[i][1] += 1
```

```
            table[i][3] += 1                                    # file contain info, device was sending data['route',
1,0,0+1,0]
            if self.check_refresh(date):
                table[i][2] += 1                                # device send data at last DELTA_TIME days ['route',
1, 0+1,0,0]
        pass


    def check_tracker_info(self, device_id):            # return True if file positions.json contain info about device
        for pos in self.positions_json:
            if str(pos['deviceId']) == device_id:
                return True
        return False                        # return False if info was not find


    def check_refresh(self, date):
        # transform date format from file to date format datetime
        lastUpdate = datetime.strptime(date, '%Y-%m-%dT%H:%M:%S.%f%z').replace(tzinfo=None)

        if datetime.now() - lastUpdate < timedelta(DELTA_TIME):
            return True
        else:
            return False


    def write_grid(self, root, table):                      # записати таблицю в grid
        total_rows = len(table)                             # кількість строчок
        total_columns = len(table[0])                       # кількість стовпців

        for i in range(total_rows):
            for j in range(total_columns):
                e = Entry(root, width=20, fg='black', font=('Arial',12,'bold'))     # створює поле для внесення даних
                e.grid(row=i+1, column=j)                   # розташовуємо поле в grid
                e.insert(END, table[i][j])                  # записуємо в поле дані з таблиці


    def clear_grid(self, root):                             #  grid
        for e in root.grid_slaves():
            e.grid_forget()


    def find_groupID(self, route):
        file = open('Route_GroupId.txt', 'r')
        for line in file:
            lst = line.split(':')               # example '15:120' -> ['15', '120\n']
            if lst[0] == route:
                return lst[1][:-1]              # string without last element '\n'


    def check_refresh(self, date):
        # transform date format from file to date format datetime
        lastUpdate = datetime.strptime(date, '%Y-%m-%dT%H:%M:%S.%f%z').replace(tzinfo=None)

        if datetime.now() - lastUpdate < timedelta(DELTA_TIME):
            return True
        else:
            return False
```

```
    def save_results(self, second_window, title_name, variable):
        row = second_window.grid_slaves(row=variable+1)
        file_id = row[0].get()
        table = []

        table_size = self.main_window.grid_size()
        table_row_count = table_size[1]

        for i in range(1, table_row_count):
            row = self.main_window.grid_slaves(row=i)
            table.append([row[4].get(), row[3].get(), row[2].get(), row[1].get(), row[0].get()])

        if title_name == 'Введіть заголовок' or title_name == '':
            title = datetime.now().strftime("%d.%m.%Y")
        else:
            title = title_name

        sc = SheetsConroller(file_id)
        sc.create_tab(title)
        sc.write_sheet(title, table)
        sc.sort(title, len(table))
        second_window.destroy()
        show_message('Результати успішно збережено')


class User():
    _auth_status = False

    def __init__(self, login, password):
        self._login = login
        self._password = password

    def get_auth_status(self):
        return self._auth_status

    def set_auth_status(self, status):
        self._auth_status = status

    def get_login(self):
        return self._login

    def set_login(self, login):
        self._login = login

    def get_password(self):
        return self._password

    def set_password(self, password):
        self._password = password

    login = property(get_login, set_login, "Свойство логин")
    password = property(get_password, set_password, "Свойство пароль")


class SheetsConroller(object):
```

```
    CREDENTIALS_FILE = 'creds.json'


    def __init__(self, spreadsheet_id):
        self._spreadsheet_id = spreadsheet_id
        self._service = self.receive_service()


    def receive_service(self):                              # Авторизуемся и получаем service — экземпляр доступа
к API
        credentials = ServiceAccountCredentials.from_json_keyfile_name(
            self.CREDENTIALS_FILE,
            ['https://www.googleapis.com/auth/spreadsheets',
             'https://www.googleapis.com/auth/drive'])
        httpAuth = credentials.authorize(httplib2.Http())
        return apiclient.discovery.build('sheets', 'v4', http=httpAuth)


    def create_tab(self, title):  # Создать новый аркуш
        self._service.spreadsheets().batchUpdate(
            spreadsheetId=self._spreadsheet_id,
            body={
                'requests': [{
                    'addSheet': {
                        'properties': {
                            'title': title,
                            'tabColor': {
                                'red': 0.44,
                                'green': 0.99,
                                'blue': 0.50
                            }
                        }
                    }
                }]
            }).execute()


    def write_sheet(self, title, table):                            # Записать данные масива sheet в google-sheets
        r = title + '!' + 'A1:' + 'E' + str(len(table))
        self._service.spreadsheets().values().batchUpdate(
            spreadsheetId=self._spreadsheet_id,
            body={
                "valueInputOption": "USER_ENTERED",
                "data": [
                    {"range": r,
                     "majorDimension": "ROWS",
                     "values": table},
                ]
            }
        ).execute()


    def sort(self, title, len_table):       # Сортировка
        spreadsheet = self._service.spreadsheets().get(spreadsheetId=self._spreadsheet_id).execute()
        sheet_id = None
        for _sheet in spreadsheet['sheets']:
```

```
            if _sheet['properties']['title'] == title:
                sheet_id = _sheet['properties']['sheetId']

        self._service.spreadsheets().batchUpdate(spreadsheetId=self._spreadsheet_id, body={
            "requests": [
                {
                    "sortRange": {
                        "range": {
                            "sheetId": sheet_id,
                            "startRowIndex": 1,
                            "endRowIndex": len_table,
                            "startColumnIndex": 0,
                            "endColumnIndex": 5
                        },
                        "sortSpecs": [
                            {
                                "dataSourceColumnReference": {
                                    "name": "A"
                                },
                                "sortOrder": "ASCENDING"
                            }
                        ]
                    }
                }
            ]
        }).execute()


class FileController(object):


    def __init__(self, file_path):
        self._file_path = file_path


    def get_file_json(self, login, password):
        return requests.get(self._file_path, auth=(login, password)).json()


    def get_acsses(self, login, password):
        return requests.get(self._file_path, auth=(login, password))


    def __del__(self):
        print('delete fc')


class MainWindow(Tk):

    def __init__(self, controller):

        self._controller = controller
        super().__init__()
        self.title("Система електронного контролю транспортрного складу")
        self.geometry("1440x900")

        self.configure(bg='#9fdcec')
```

```
        data = StringVar()
        search = StringVar()

        lable_head = Label(self, text="Результати", fg="#000000", bg="#9fdcec", font=("Arial","18"))
        lable_head.grid(row=0, column=2, sticky="N")

        #SSA - system status analysis
        #SRA - route status analysis
        #SSF - select save file
        btn_SSA = Button(self, text="Аналіз стану системи", background="#dce1db", foreground="#000000", padx="20", pady="8", font=("Arial","16","bold"), command=lambda: self._controller.execute_SSA(self))
        btn_RSA = Button(self, text="Аналіз стану маршруту", background="#dce1db", foreground="#000000", padx="20", pady="8", font=("Arial","16","bold"), command=lambda: SelectRouteWindow(self))
        btn_SSF = Button(self, text="Вибрати файл збереження", background="#dce1db", foreground="#000000", padx="20", pady="8", font=("Arial","16","bold"), command=lambda: SelectSaveFile(self))
        btn_setData = Button(self, text="Задати", background="#dce1db", foreground="#000000", padx="20", pady="8", font=("Arial","16","bold"))
        btn_search = Button(self, text="Знайти", background="#dce1db", foreground="#000000", padx="20", pady="8", font=("Arial","16","bold"), command=lambda: self._controller.find(search_entry.get()))

        data_entry = Entry(self, textvariable=data, font=("Arial","14","italic"))
        search_entry = Entry(self, textvariable=search, font=("Arial","14","italic"))
        data_entry.insert(0,'Дата виконання')
        search_entry.insert(0,'Ключові слова')


        btn_SSA.place(height=40, width=300, x=1100, y=30, bordermode=OUTSIDE)
        btn_RSA.place(height=40, width=300, x=1100, y=100, bordermode=OUTSIDE)
        btn_SSF.place(height=40, width=300, x=1100, y=170, bordermode=OUTSIDE)
        btn_setData.place(height=40, width=100, x=1300, y=240, bordermode=OUTSIDE)
        btn_search.place(height=40, width=100, x=1300, y=540, bordermode=OUTSIDE)


        data_entry.place(height=40, width=200, x=1100, y=240)
        search_entry.place(height=40, width=300, x=1000, y=540)

        table = [['', '', '', '', '']]                          # Пустая строка в grid, для першого відкриття вікна
        self._controller.write_grid(self, table)


class AuthorizationWindow(Tk):

    def __init__(self, controller):

        self._controller = controller
        super().__init__()
        self.title("Авторизація")
        self.geometry("600x400")
        self.configure(bg='#9fdcec')

        login = StringVar()
        password = StringVar()

        login_entry = Entry(self, textvariable=login, font=("Arial","16","italic"))
        password_entry = Entry(self, textvariable=password, font=("Arial","16","italic"))
```

```
        login_entry.insert(0,'Login')
        password_entry.insert(0,'Password')

        login_entry.place(height=40, width=300, x=150, y=100)
        password_entry.place(height=40, width=300, x=150, y=160)

        btn_authorization = Button(self, text="Авторизуватися", background="#dce1db", foreground="#000000",
padx="20", pady="8", font=("Arial","16","bold"),
                                                                command=lambda:
self._controller.authorize(login_entry.get(), password_entry.get()))
        btn_authorization.place(height=40, width=200, x=200, y=340, bordermode=OUTSIDE)


class SelectRouteWindow(Toplevel):

    def __init__(self, master = None):

        super().__init__(master = master)                # master - main_window
        self._controller = master._controller            # self - second_window
        self.title("Маршрути")
        self.geometry("720x900")
        self.configure(bg='#9fdcec')

        lable_routes = Label(self,text="Маршрути", width=30, fg='black', bg="#9fdcec", font=("Arial","18"))
        lable_pick = Label(self,text="Обрати", width=15, fg='black', bg="#9fdcec", font=("Arial","18"))
        lable_routes.grid(row=0, column=0, sticky="N")
        lable_pick.grid(row=0, column=1, sticky="N")

        btn_cancel = Button(self, text="Повернутися", background="#dce1db", foreground="#000000", padx="20",
pady="8", font=("Arial","16","bold"), command=lambda: self.cancel())
        btn_reset = Button(self, text="Скинути вибране", background="#dce1db", foreground="#000000",
padx="20", pady="8", font=("Arial","16","bold"), command=lambda: self.reset(lst_var))
        btn_execute = Button(self, text="Виконати", background="#dce1db", foreground="#000000", padx="20",
pady="8", font=("Arial","16","bold"), command=lambda: self._controller.execute_RSA(self, master, lst_var))

        btn_cancel.place(height=40, width=200, x=25, y=800, bordermode=OUTSIDE)
        btn_reset.place(height=40, width=200, x=250, y=800, bordermode=OUTSIDE)
        btn_execute.place(height=40, width=200, x=475, y=800, bordermode=OUTSIDE)

        lst_routs = []
        file = open('Route_GroupId.txt', 'r')

        for line in file:
            lst = line.split(':')              # example '15:120' -> ['15', '120\n']
            lst_routs.append(lst[0])

        total_rows = len(lst_routs)
        lst_var = []

        for i in range(total_rows):
            lst_var.append(IntVar())

        for i in range(total_rows):
            e = Entry(self, width=30, fg='black', font=('Arial',12,'bold'))
            c = Checkbutton(self, variable=lst_var[i], onvalue=1, offvalue=0).grid(row=i+1,column=1)
            e.grid(row=i+1,column=0)
            e.insert(END, lst_routs[i])
```

```
    def cancel(self):
        self.destroy()

    def reset(self, lst_var):
        for i in range(len(lst_var)):
            lst_var[i].set(0)


class SelectSaveFile(Toplevel):


    def __init__(self, master = None):

        super().__init__(master = master)
        self._controller = master._controller
        self.title("Обрати файл")
        self.geometry("620x600")
        self.configure(bg='#9fdcec')

        fileId = StringVar()
        title = StringVar()

        entry_title = Entry(self, textvariable=title, width=20, fg='black', font=("Arial","14","italic"))
        entry_title.insert(0,'Введіть заголовок')
        entry_title.place(x=230, y=565)

        entry_addFile = Entry(self, textvariable=fileId, width=45, fg='black', font=("Arial","14","italic"))
        entry_addFile.insert(0,'Введь Id нового файлу')
        entry_addFile.grid(row=0,column=0)

        btn_add = Button(self, text="Додати", background="#dce1db", foreground="#000000", padx="30",
pady="2", font=("Arial","16","bold"), command=lambda: self.add_file(master, entry_addFile.get()))
        btn_add.grid(row=0,column=1)

        btn_save = Button(self, text="Зберегти", background="#dce1db", foreground="#000000", padx="20",
pady="2", font=("Arial","16","bold"), command=lambda: self._controller.save_results(self, entry_title.get(),
dote.get()))
        btn_save.place(x=450,y=560,width=150)

        file = open('GoogleSpreadsheetsId.txt', 'r')
        lst_fileId = []
        for line in file:
            lst_fileId.append(line[:-1])

        dote = IntVar()
        for i in range(len(lst_fileId)):
            e = Entry(self, width=50, fg='black', font=('Arial',12,'bold'))
            r = Radiobutton(self, bg='#9fdcec',highlightthickness=0, value=i, variable=dote, pady=10).grid(row=i+1,
column=1, sticky="N")
            e.grid(row=i+1,column=0, padx=10, pady=10, sticky="W")
            e.insert(END, lst_fileId[i])
        file.close()

    def reopen_window(self, master):
        SelectSaveFile(master)
        self.destroy()
```

```
    def add_file(self, master, fileID):
        file = open('GoogleSpreadsheetsId.txt', 'a')
        file.write(fileID+'\n')
        file.close()
        self.reopen_window(master)


def show_message(text):
    messagebox.showinfo("Повідомлення", text)


def main():
    program = ProgramController()
    program.startProgram()


if __name__ == "__main__":
        main()
```

Додаток В Опис програми

1 Загальні відомості

1.1 Позначення і найменування програми

Найменування програмного забезпечення – «Розробка програмного забезпечення для автоматизації процесів обробки даних системи електронного контролю транспортного складу маршрутного таксі міста Миколаєва». Розробка ведеться по замовленню комунального підприємства «Міський інформаційно-обчислювальний центр» та на підставі завдання на кваліфікаційну роботу, виданого кафедрою ПЗАС НУК імені адмірала Макарова.

Позначення програми «Mini-bus MIOC».

1.2 Програмне забезпечення, необхідне для функціонування програми

Система повинна задовольняти вказаним вимогам на комп'ютері наступної мінімальної комплектації:

- CPU: Intel Core i3– 2.93 МГц;
- RAM: 4 GB;
- HDD: 2 GB вільного місця;
- доступ до інтернету.

Для повноцінного функціонування системи необхідно використовувати ОС Linux дистрибутив Ubuntu версії не нижче 20.04 з встановленим інтерпретатором Python3.

1.3 Мови програмування

Програмний додаток «Mini-bus MIOC» розроблено мовою програмування Python3. Для розробки програмного забезпечення обрали бібліотеку для розробки програм з графічним інтерфейсом на мові Python –

Tkinter – це багатоплатформна графічна бібліотека інтерфейсів на основі засобів Tk, поширювана з відкритими вихідними текстами, написана Стіном Лумхольтом і Гвідо ван Россумом. Входить у стандартну бібліотеку Python. Також для розробки було використано набір інтерфейсів прикладного програмування Google API, який дозволяє клієнту взаємодіяти з інтегрованими сервісами. Модулі, реалізовані на інших мовах програмування не використовуються.

2. Функціональне призначення

2.1 Класи розв’язування завдань та призначення програми

Функціональним призначенням програмного забезпечення для автоматизації процесів обробки даних системи електронного контролю транспортного складу маршрутного таксі міста Миколаєва є автоматизація процесів:

- виконання аналізу стану системи GPS трекінгу – підрахунок кількості активних та не активних трекерів на всіх маршрутах, формування списків;
- задавання дати й часу автоматичного виконання аналіз стану системи GPS трекінгу;
- автоматичне зберігання результатів – зберігання інформації, отриманої в результаті аналізу даних, в вигляді Google Spreadsheets таблиць;
- аналіз маршрутів – зібрати інформацію про діючий на маршруті транспорт (Реєстраційний номер, IMEI трекера, номер CIM-картки, статус трекера, дата останнього оновлення);
- пошук – знаходження GPS-трекеру й інформації про нього через ключові слова.

3 Опис логічної структури

3.1 Структура програми

Програмний додаток «Mini-bus MIOC» має дві складові частини: клієнтська частина – графічний інтерфейс та база даних – програмне забезпечення, що забезпечую зберігання даних та їх видачу в момент запиту.

Логічна структура програми представлена взаємодією основних модулів програми: модулями и програми можна представити варіанти використання на діаграмі варіантів використання, тому логічною моделлю програми можна представити діаграму варіантів використання.

3.2 Алгоритм програми

Робота програмного додатку «Mini-bus MIOC» представлена взаємодією між клієнтською частиною та сервером. Алгоритм роботи програми є наступним:

1) Користувач взаємодіє з програмним додатком.

2) Програмний додаток звертається до серверу (якщо це потрібно) та відсилає результат.

3) Програмний додаток відображає отриманий результат.

3.3 Використовувані методи

Програмний додаток - «Mini-bus MIOC» розроблено мовою Python3, що є об’єктно-орієнтованою.

4 Технічні засоби, що використовуються

Програмний додаток «Mini-bus MIOC» потребує від комп’ютеру, на якому він буде встановлений, наступних характеристик, які треба розглядати як мінімальні:

- CPU: Intel Core i3– 2.93 МГц;
- RAM: 4 GB;

- HDD: 2 GB вільного місця;
- доступ до інтернету.

Для повноцінного функціонування системи необхідно використовувати ОС Linux дистрибутив Ubuntu версії не нижче 20.04 з встановленим інтерпретатором Python 3, встановленим пакетом virtualen та налаштованим віртуальним оточенням (env). Веб-браузер Google Chrome версії не нижче 100.0. В браузері мають бути встановлені два розширення: API Google Drive і API Google Spreadsheets.

5 Виклик та завантаження

Виклик та завантаження програмного додатку «Mini-bus MIOC» відбувається запуском відповідно названого виконуваного файлу - Mini-bus MIOC.exe. Запуск програмного додатку «Mini-bus MIOC» іншими способами неможливий.

6 Вхідні дані

Вхідні дані: введення даних здійснюється за допомогою пристроїв введення даних (клавіатура та миша). Дані або вводяться вручну, або обираються із представлених списків.

7 Вихідні дані

Вихідні дані: виведення даних здійснюється за допомогою пристроїв виведення даних (монітор)

Додаток Г Інструкція користувача

Дана інструкція призначена для ознайомлення користувачів з функціями ПЗ та полегшення роботи з ним.

1) Для запуску програми потрібно натиснути на ярлик з написом «Mini-bus MIOC» або відкрити його будь-яким іншим чином.

2) Після запуску з’явиться вікно в якому потрібно ввести свій логін та пароль. У випадку введення правильного логіну та паролю буде виведено повідомлення про успішну авторизацію та відкрите головне вікно програми, інакше користувач побачить повідомлення про помилку.

Приклад заповнення полів вводу форми авторизації користувачів в ПЗ наведено на рисунку Г.1.

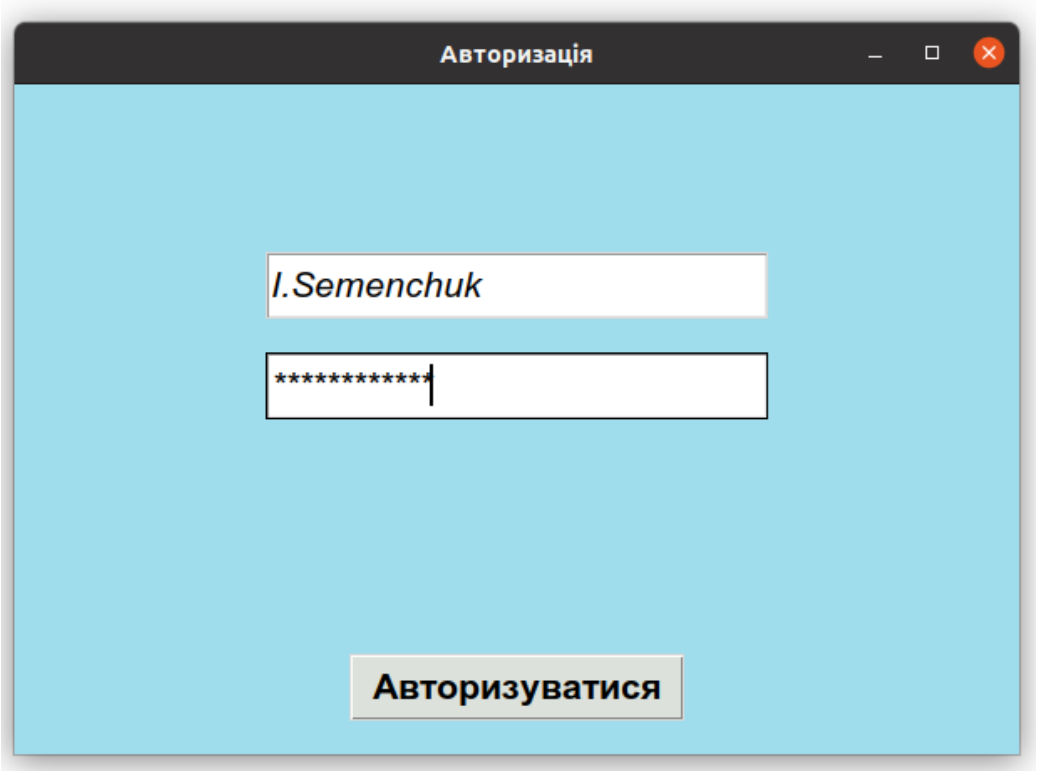


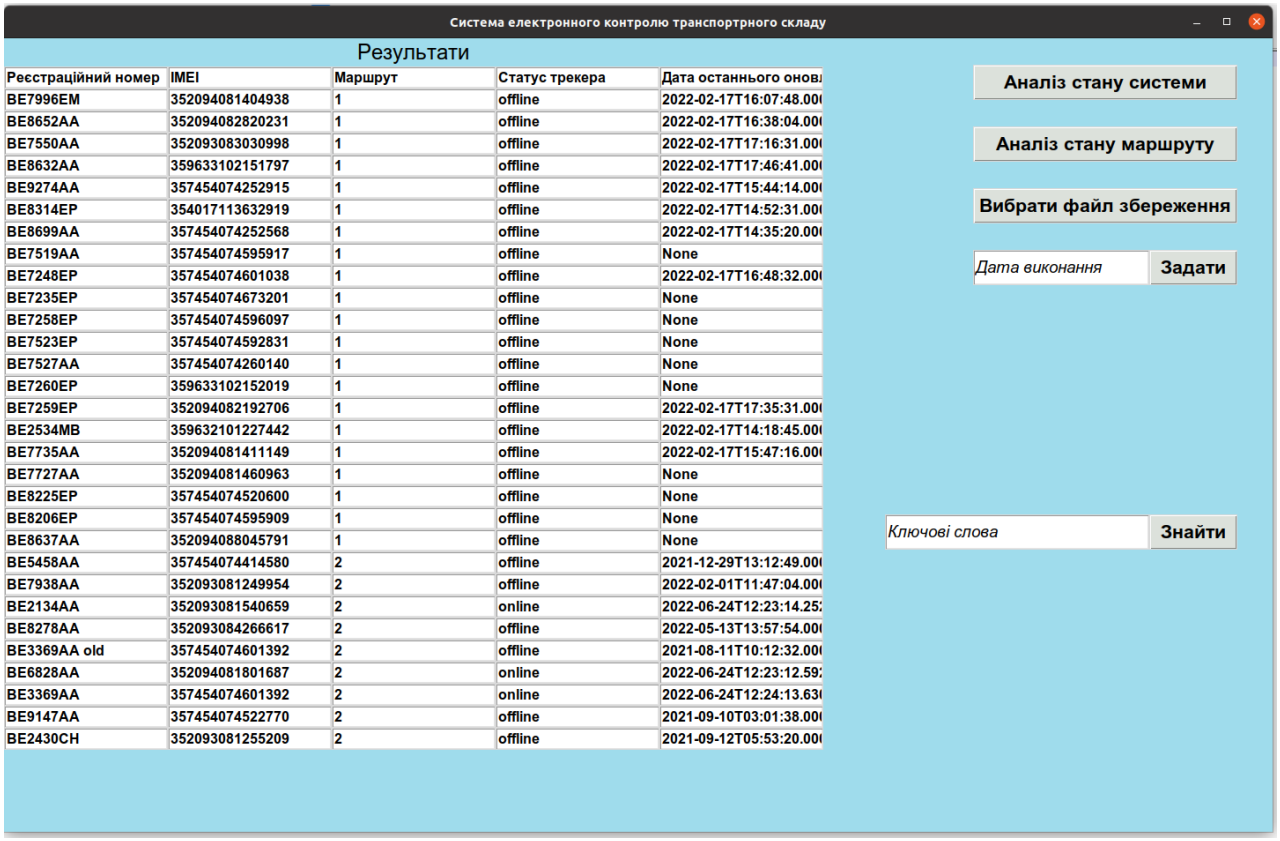
Рисунок Г.1 – Форма авторизації користувачів

Після відкриття головного вікна програми будуть доступні наступні функції:

1) Виконати аналіз стану системи

Для того аби виконати аналіз стану системи, треба на головному вікні програми натиснути кнопку «Аналіз стану системи», після чого в лівій частині головного вікна програми в розділі «Результати» буде висвітлено результати аналізу стану системи.

Приклад результату виконання аналізу стану системи наведено на рисунку Г.2



Реєстраційний номер	IMEI	Маршрут	Статус трекера	Дата останнього онов
BE7996EM	352094081404938	1	offline	2022-02-17T16:07:48.00
BE8652AA	352094082820231	1	offline	2022-02-17T16:38:04.00
BE7550AA	352093083030998	1	offline	2022-02-17T17:16:31.00
BE8632AA	359633102151797	1	offline	2022-02-17T17:46:41.00
BE9274AA	357454074252915	1	offline	2022-02-17T15:44:14.00
BE8314EP	354017113632919	1	offline	2022-02-17T14:52:31.00
BE8699AA	357454074252568	1	offline	2022-02-17T14:35:20.00
BE7519AA	357454074595917	1	offline	None
BE7248EP	357454074601038	1	offline	2022-02-17T16:48:32.00
BE7235EP	357454074673201	1	offline	None
BE7258EP	357454074596097	1	offline	None
BE7523EP	357454074592831	1	offline	None
BE7527AA	357454074260140	1	offline	None
BE7260EP	359633102152019	1	offline	None
BE7259EP	352094082192706	1	offline	2022-02-17T17:35:31.00
BE2534MB	359632101227442	1	offline	2022-02-17T14:18:45.00
BE7735AA	352094081411149	1	offline	2022-02-17T15:47:16.00
BE7727AA	352094081460963	1	offline	None
BE8225EP	357454074520600	1	offline	None
BE8206EP	357454074595909	1	offline	None
BE8637AA	352094088045791	1	offline	None
BE5458AA	357454074414580	2	offline	2021-12-29T13:12:49.00
BE7938AA	352093081249954	2	offline	2022-02-01T11:47:04.00
BE2134AA	352093081540659	2	online	2022-06-24T12:23:14.25
BE8278AA	352093084266617	2	offline	2022-05-13T13:57:54.00
BE3369AA old	357454074601392	2	offline	2021-08-11T10:12:32.00
BE6828AA	352094081801687	2	online	2022-06-24T12:23:12.59
BE3369AA	357454074601392	2	online	2022-06-24T12:24:13.63
BE9147AA	357454074522770	2	offline	2021-09-10T03:01:38.00
BE2430CH	352093081255209	2	offline	2021-09-12T05:53:20.00

Рисунок Г.2 – Результати виконання аналізу системи

2) Виконати аналіз стану маршруту

Щоб виконати аналіз стану певного маршруту, треба на головному вікні програми натиснути кнопку «Аналіз стану маршрутів» після чого поверх головного вікна відкриється нове вікно «Вікно вибору маршруту». Для того, щоб обрати маршрут, треба навпроти номеру потрібного маршруту поставити галочку в стовпці «Обрати». Після того як потрібний маршрут обрано

натиснути кнопку «Виконати» після чого «Вікно вибору маршруту» автоматично закриється і контроль повернеться до головного вікна програми в лівій частині якого в розділі «Результати» буде висвітлено результати аналізу стану маршруту.

Приклад вибору маршрутів та результат аналізу їх стану наведено на рисунку Г.3 результати аналізу на рисунку Г.4

Рисунок Г.3 – Приклад вибору маршрутів 1Авт та 2Авт

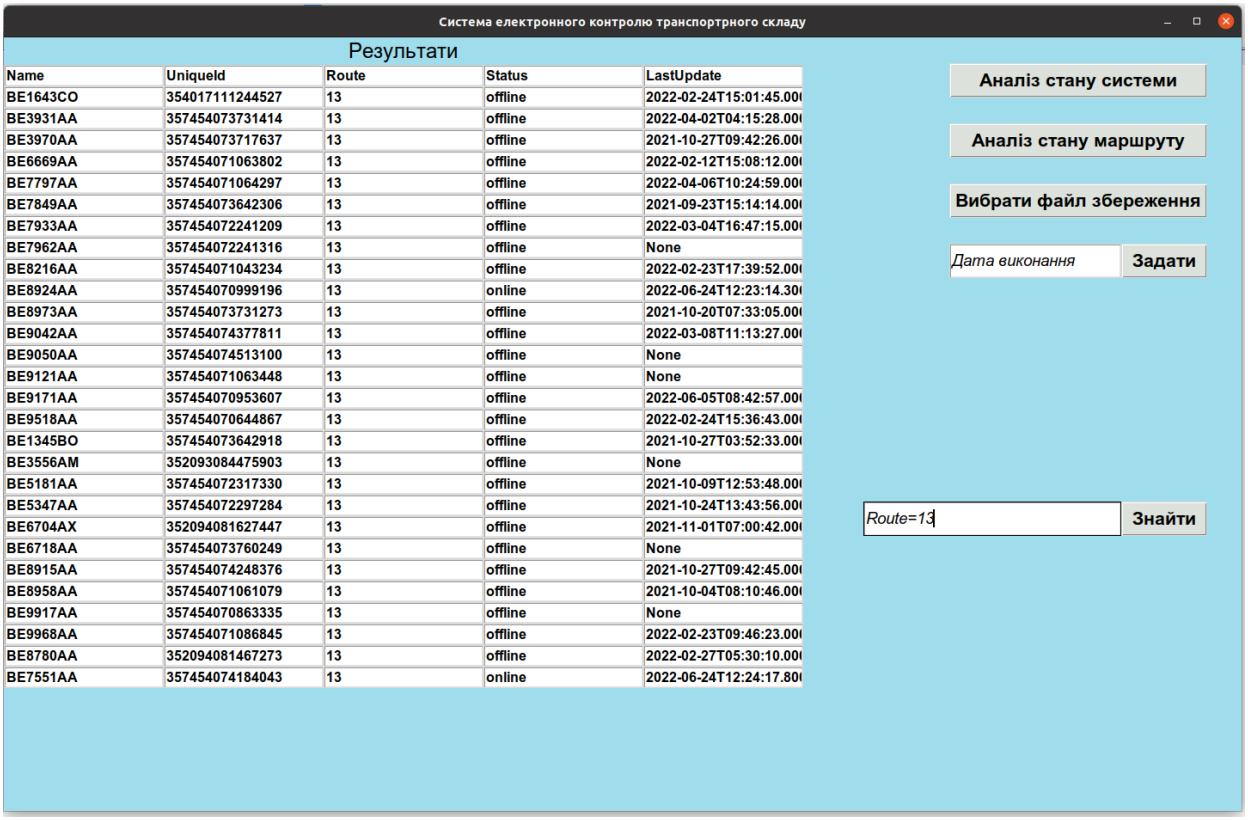


Рисунок Г.4 – Результати аналізу станів маршрутів 1Авт та 2Авт

3) Вибрати файл для збереження результатів

Щоб вибрати файл в який будуть збережені результати виконаного аналізу, треба на головному вікні програми натиснути кнопку «Вибрати файл збереження» після чого поверх головного вікна відкриється нове вікно «Вікно вибору файлу». Для того, щоб обрати файл, треба навпроти id потрібного файлу поставити точку в стовпці «Обрати».

Приклад вибору файлу збереження наведено на рисунку Г.5

Рисунок Г.5 – Приклад вибору файлу збереження

4) Додати файл збереження

Щоб додати до списку файлів новий файл, треба на головному вікні програми натиснути кнопку «Вибрати файл збереження» після чого поверх головного вікна відкриється нове вікно «Вікно вибору файлу». В верхній частині відкритого вікна міститься поля для введення даних «id документу» вводимо в нього id потрібного файлу й натискаємо кнопку «Додати». У разі успішного додання нового файлу програма покаже повідомлення про успішне додання, в іншому випадку повідомлення про помилку.

Результати додання нового файлу збереження представлено на рисунку Г.6

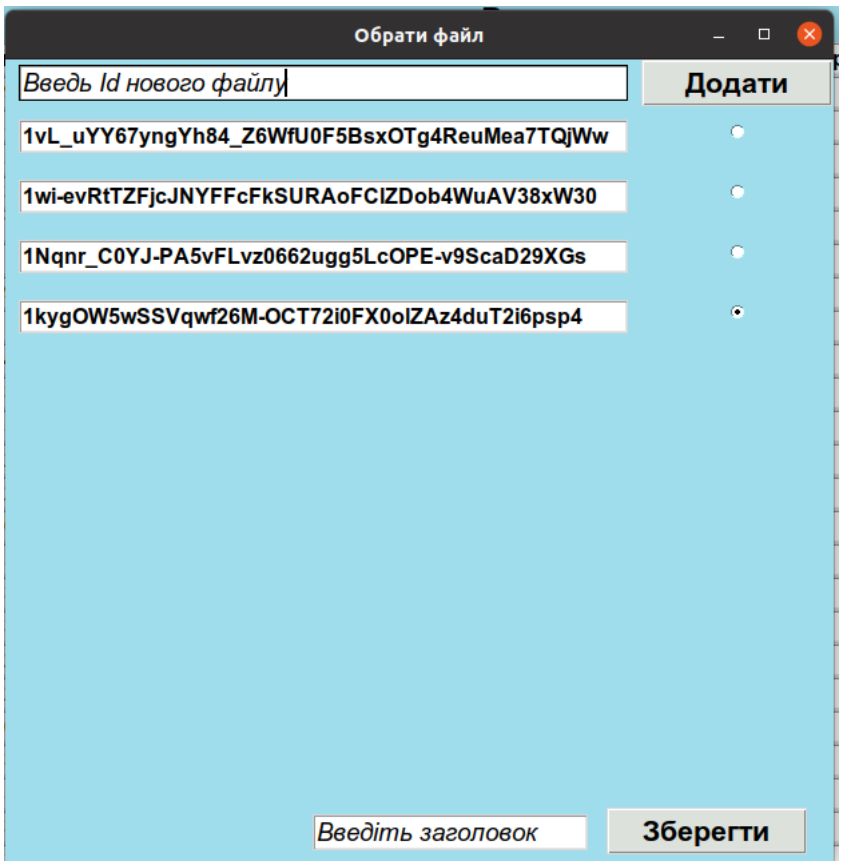


Рисунок Г.6 - Результати додання нового файлу збереження

5) Зберегти результати

Щоб зберегти результати обробки даних, спочатку треба вибрати файл для збереження даних. Після того як потрібний файл обрано, заповнити поле «Введіть заголовок» потрібний вам заголовок та натиснути кнопку «Зберегти» після чого користувачеві буде показано повідомлення про успішне завершення збереження, «Вікно вибору файлу» автоматично закриється і контроль повернеться до головного вікна програми. Якщо заголовок не буде введено програма автоматично згенерує заголовок в вигляді дати виконання.

Результати збереження наведено на рисунках Г7 та Г8

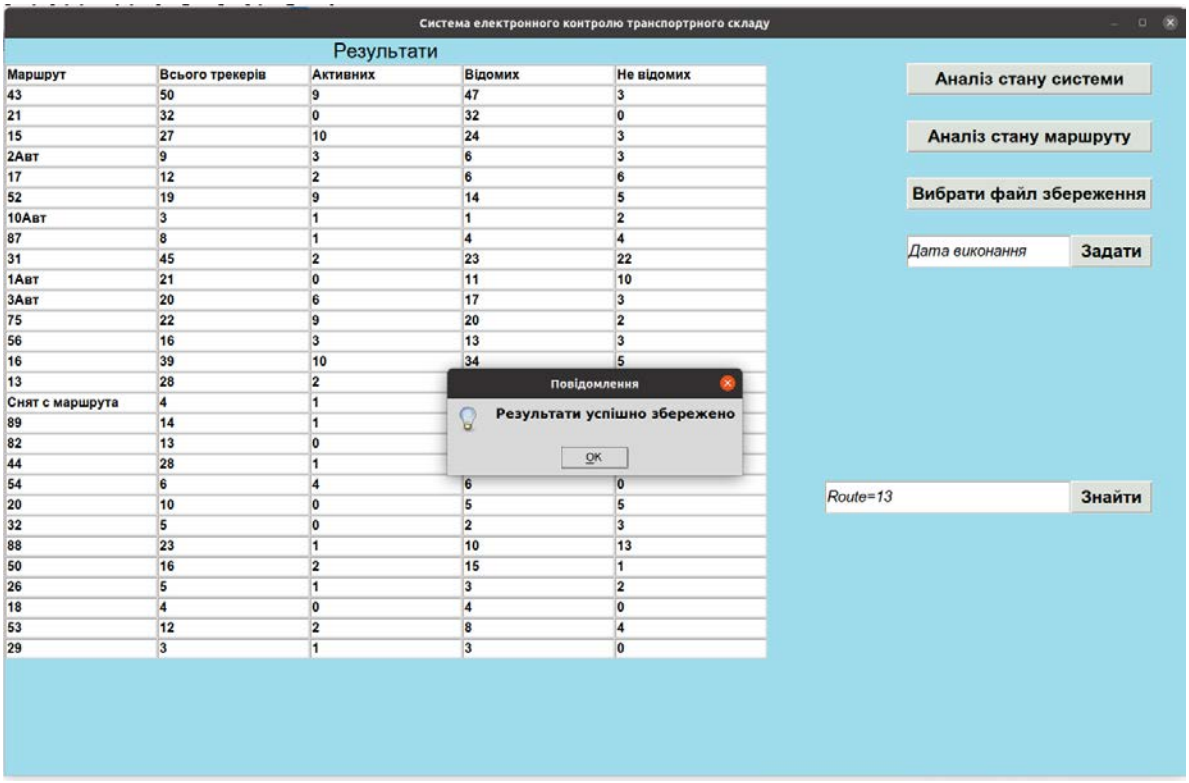


Рисунок Г.7 – Повідомлення про успішне збереження

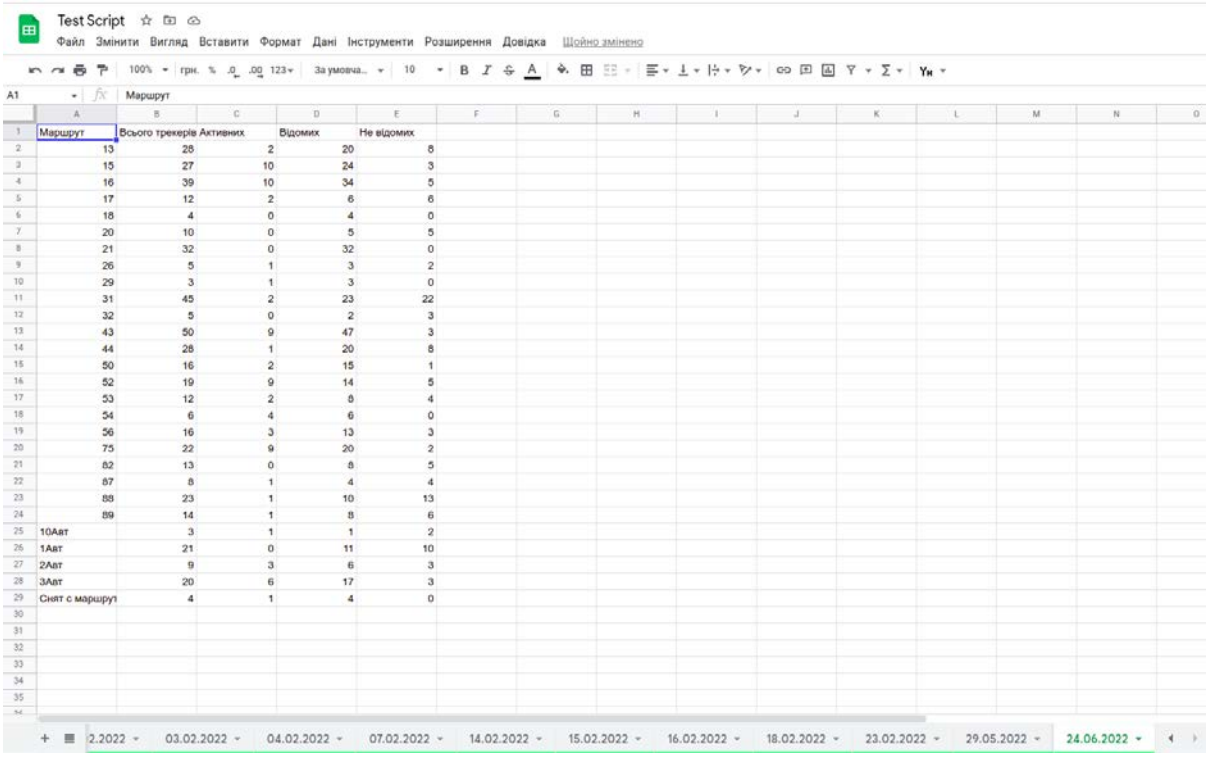


Рисунок Г.8 – Результат збереження даних в обраному файлі

6) Виконати пошук

Щоб виконати пошук за ключовими словами, треба на головному вікні програми в поле для введення даних «Ключові слова», що знаходиться в правому нижньому кутку вікна ввести формулу чи ключове слово, які вам треба знайти й натиснути кнопку «Знайти». Приклад формули: status=online чи Route=13. Приклад пошуку по ключовому слову: BE1111AP (номер авто). Якщо пошук по заданому ключеві не дав результатів, користувач отримає повідомлення, що збігів не знайдено, якщо в ведені даних формат даних був вказаний не правильно, користувач отримає повідомлення про не правильний формат введення даних.

Приклад виконання пошуку за формулою наведено на рисунку Г.9

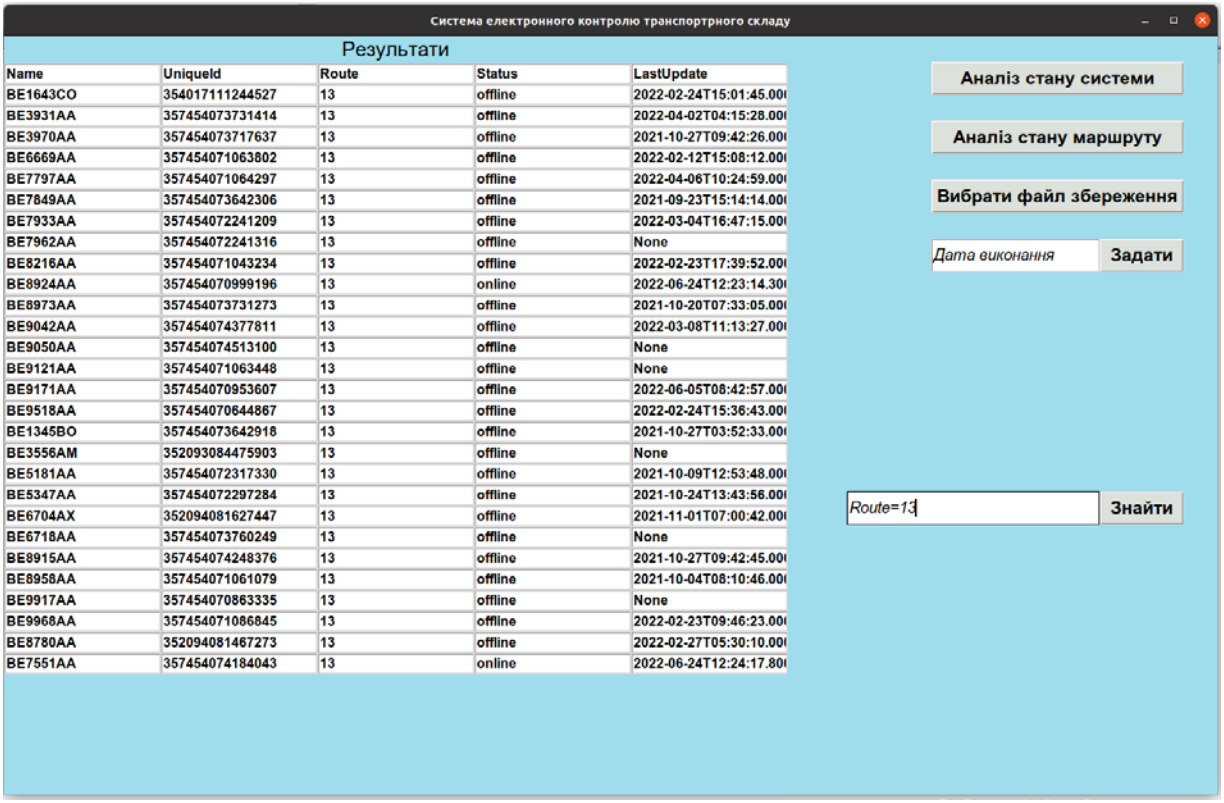


Рисунок Г.9 – Приклад виконання пошуку за формулою «Route=13»

7) Задати дату автоматичного виконання програми;

Щоб задати дату автоматичного виконання програми, треба на

головному вікні програми в правій частині вікна в поле для введення даних «дата виконання» вказати дату в форматі дд/мм/рррр/гг:хх:сс та натиснути кнопку «Задати». Приклад введення дати: 21.09.2022:08.00. Якщо дата введена користувачем відноситься до часу який вже відбувся, тобто належить минулому, користувач отримає повідомлення про помилку введення даних, якщо користувач введе дату в не вірному форматі, отримає повідомлення про помилку в форматі введених даних.

Приклад заповнення поля «дата виконання» представлено на рисунку Г.10

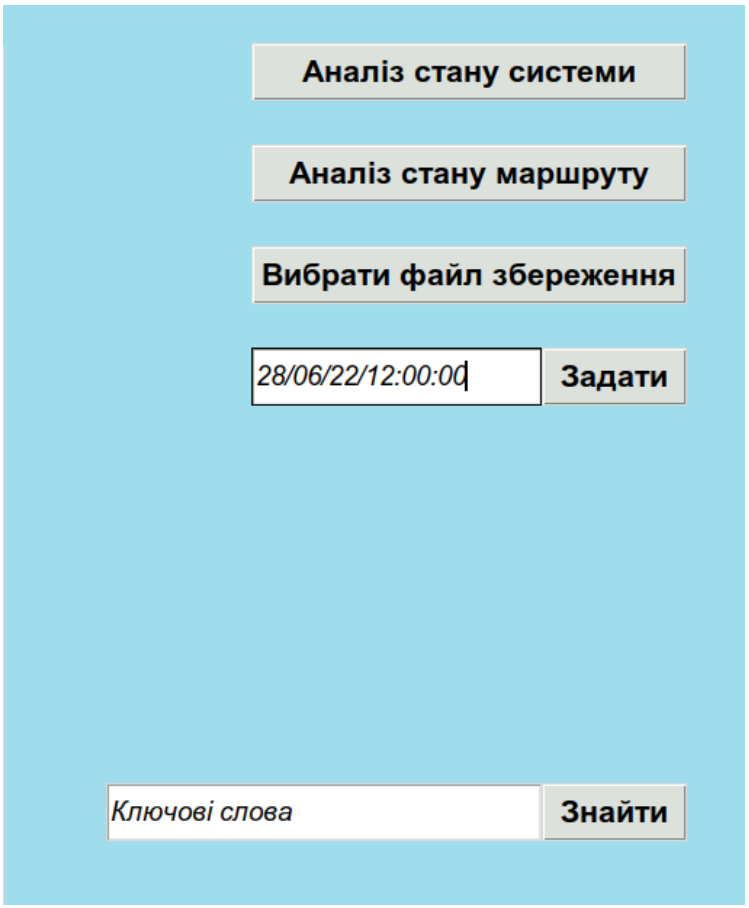


Рисунок Г.10 – Приклад заповнення поля «дата виконання»

Додаток Д Програма та методика випробувань програмного забезпечення

1 Об’єкт випробування

Об’єктом випробування є програмне забезпечення для автоматизації процесів обробки даних системи електронного контролю транспортного складу маршрутного таксі міста Миколаєва. Розробка ведеться по замовленню комунального підприємства «Міський інформаційно-обчислювальний центр» та на підставі завдання на кваліфікаційну роботу, виданого кафедрою ПЗАС НУК імені адмірала Макарова.

2. Мета випробування

Метою проведення випробування є перевірка працездатності даного програмного забезпечення, перевірка відповідності характеристик та вимог розробленої програми, викладених у технічному завдані, приведення прикладу роботи та отримання відповідних навичок.

3. Вимоги до додатка

При проведенні випробувань функціональні характеристики (можливості) програми підлягають перевірці на відповідність вимогам, викладеним у пункті «Вимоги до функціональних характеристик» технічного завдання.

4. Вимоги до програмної документації

Програмна документація повинна включати в себе наступні документи:

1) Текст програми.

2) Опис програми.

3) Програма та методика випробувань.

4) Інструкція користувача.

5 Засоби і порядок випробувань

5.1 Технічні засоби, що використовуються під час випробувань

Склад технічних засобів, що використовуються під час випробувань:

- CPU: Intel Core i3– 2.93 МГц;
- RAM: 4 GB;
- HDD: 2 GB вільного місця;
- доступ до інтернету.

5.2 Програмні засоби, що використовуються під час випробувань

Склад програмних засобів, що використовуються під час випробувань:

- ОС Linux дистрибутив Ubuntu версії не нижче 20.04;
- інтерпретатор Python 3;
- пакет virtuale;
- Веб-браузер Google Chrome версії не нижче 100.0.;
- розширення: API Google Drive і API Google Spreadsheets.

5.3 Порядок проведення випробувань

Порядок проведення випробувань складається з перевірки вимог до функціональних характеристик програмного продукту та перевірки вимог до програмної документації.

6 Методи випробувань

6.1 Методика проведення перевірки вимог до програмної документації

Перевірка дотримання вимог програмної документації на програмний продукт виконується візуально представником служби, відповідальної за експлуатацію. У ході перевірки зіставляється склад і комплектність програмної документації, представленої розробником, з переліком програмної документації, наведеним у пункті «Склад програмної документації, пропонованої на випробування» цього документа.

Перевірка вважається завершеною у випадку відповідності складу та комплектності програмної документації, представленої розробником, переліком програмної документації, наведеному у зазначеному вище пункті.

6.2 Методика проведення перевірки вимог до функціональних характеристик програмного продукту

Перевірка працездатності програми виконується згідно з пунктом «Вимоги до функціональних характеристик» Додатку А Технічне завдання. Перевірка вважається завершеною у разі відповідності складу і послідовності виконаних дій пункту «Вимоги до функціональних характеристик». В процесі тестування була перевірена функціональність за темою «Розробка програмного забезпечення для автоматизації процесів обробки даних системи електронного контролю транспортного складу маршрутного таксі міста Миколаєва». У табл. Д. 1, що наведена нижче, вказано перелік випробувань основних функціональних можливостей.

Таблиця Д. 1- Методика випробування

№	Дія	Очікуваний результат	Результат перевірки	Зауваження
1	2	3	4	5
1	Перевірка правильності введення даних при авторизації	Програма повинна перевірити введені дані	Виконано	
2	Перевірка правильності введення даних при виконанні пошуку	Програма повинна перевірити коректність введених даних	Виконано	
3	Перевірка правильності додавання нового ID документу	Програма повинна перевірити коректність введених даних	Виконано	

Продовження табл. Д-1

1	2	3	4	5
4	Перевірка правильності обрання маршрутів	Програма повинна перевірити правильність обраного маршруту	Виконано	
5	Перевірка правильності вказання дати автоматичного виконання програми	Програма повинна перевірити правильність вказаної дати й часу виконання програми	Виконано	
6	Перевірка правильності вказання заголовку таблиці в файлі збереження	Програма повинна перевірити правильність вказаного заголовку таблиці в обраному файлі збереження	Виконано	
7	Перевірка не дублювання відкриття головних вікон програми	Програма повинна перевірити правильність відкриття головних вікон програми з не можливість їх дублювання		