Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра комп'ютерних наук

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Інформаційна технологія прогнозування кількості порушень правил дорожнього руху»

Виконав: студент 2-го курсу, групи 1КН-22м,
спеціальності 122 – «Комп'ютерні науки»
_____ Львовський О. О.
(прізвище та ініціали)

Керівник: к.т.н., доц. кафедри КН
_____ Крилик Л. В.
(прізвище та ініціали)
«07» 12 2023 р.

Опонент: к.т.н., доцент каф. АІІТ
_____ Богач І. В.
(прізвище та ініціали)
«07» 12 2023 р.

Допущено до захисту
Завідувач кафедри КН
_____ д.т.н., проф. Яровий А. А.
(прізвище та ініціали)
«08» 12 2023 р.

Вінниця ВНТУ - 2023 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра комп'ютерних наук
Рівень вищої освіти ІІ-й (магістерський)
Галузь знань – 12 Інформаційні технології
Спеціальність – 122 Комп'ютерні науки
Освітньо-професійна програма – Системи штучного інтелекту

ЗАТВЕРДЖУЮ
Завідувач кафедри КН

д.т.н., проф. Яровий А.А.

(підпис)
«29» 08. 2023 року

З А В Д А Н Н Я

НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Львовському Олександру Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи «Інформаційна технологія прогнозування кількості порушень правил дорожнього руху»
керівник роботи Крилик Людмила Вікторівна, к. т. н., доцент.
затверджені наказом вищого навчального закладу від «18 » 09. 2023 року №247.
2. Строк подання студентом роботи 13 листопада 2023 року
3. Вихідні дані до роботи: мова програмування – об'єктно-орієнтована; мінімальна кількість факторів впливу – 3; мінімальна кількість записів в базі даних – 40; мінімальний період для прогнозу – 1 місяць; тип інтерфейсу користувача інтуїтивно-зрозумілий, операційна система сімейства Windows.
4. Зміст текстової частини: вступ, обґрунтування доцільності розробки інформаційної технології прогнозування кількості порушень правил дорожнього руху, моделювання інформаційної технології прогнозування кількості порушень правил дорожнього руху, програмна реалізація інформаційної технології прогнозування кількості порушень правил дорожнього руху, економічна частина, висновки, список використаних джерел, додатки.
5. Перелік ілюстративного матеріалу (з точним зазначенням обв'язкових креслень): структурна схема інформаційної технології; загальна структурна схема; ER-діаграма порушень правил дорожнього руху; UML-діаграма класів; схема алгоритму функціонування модуля прогнозування інформаційної технології прогнозування кількості порушень правил дорожнього руху; діаграма діяльності системи при встановленні з'єднання з базою даних; діаграма діяльності сервера для побудови графіка; приклади роботи програми.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціалита посада консультанта	Підпис, дата	
		завдання видав	виконання прийняв
1–3	Крилик Л. В. к.т.н., доц. каф. КН	29.08.23	16.11.23
4	Адлер О. О., к.т.н, доц. каф. ЕПВМ	16.10.23	20.10.23

7. Дата видачі завдання 29. 08. 2023 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Обгрунтування доцільності розробки	01.09.23 – 07.09.23	Розділ 1
2	Моделювання інформаційної технології прогнозування кількості порушень правил дорожнього руху	08.09.23 – 24.09.23	Розділ 2
3	Програмна реалізація інформаційної технології прогнозування кількості порушень правил дорожнього руху	25.09.23 – 15.10.23	Розділ 3
4	Підготовка економічної частини	16.10.23 - 20.10.23	Розділ 4
5	Апробація результатів дослідження	21.10.23 - 05.10.23	тези доповіді
6	Оформлення пояснювальної записки, ілюстративного матеріалу та презентації	06.11.23 – 10.11.23	Пояснювальна записка, ілюстративний матеріал, презентаці

Студент _____ Львовський О. (підпис) (прізвище та ініціа

Керівник роботи _____ Крилик Л. (підпис) (прізвище та ініціа

АНОТАЦІЯ

УДК 004.8

Львовський О. О. Інформаційна технологія прогнозування кількості порушень правил дорожнього руху. Магістерська кваліфікаційна робота зі спеціальності 122 Комп'ютерні науки, освітня програма – Системи штучного інтелекту. Вінниця: ВНТУ, 2023. 117 с.

На укр. мові. Бібліогр.: 40 назв; рис.: 19; табл. 9.

Дана магістерська кваліфікаційна робота присвячена розробці інформаційної технології прогнозування кількості порушень правил дорожнього руху. Виконано аналіз сучасних програм-аналогів, які використовуються для прогнозування ситуації на дорогах, наведено коротку порівняльну характеристику знайдених програм-аналогів, досліджено методи, які можуть бути використані для реалізації поставленої задачі. Проведено обґрунтування вибору методу для прогнозування кількості порушень правил дорожнього руху та розроблено математичну модель на основі методу ARIMA. Розроблено алгоритм прогнозування кількості порушень правил дорожнього руху та відповідне програмне забезпечення на мові програмування Java, в середовищі IntelliJ IDEA. Аналіз роботи інформаційної технології показав підвищення точності прогнозу кількості порушень правил дорожнього руху.

У розділі економічної частини здійснено оцінювання комерційного потенціалу розробки інформаційної технології прогнозування кількості порушень правил дорожнього руху, проведено оцінювання комерційного потенціалу розробки, спрогнозовано витрати на виконання наукової роботи та впровадження результатів, розраховано період окупності.

Ілюстративна частина складається з 8 плакатів із результатами моделювання.

Ключові слова: інформаційна технологія, прогнозування, правила дорожнього руху, ДТП.

ABSTRACT

Lvovskyi O. O. Information technology for predicting the number of traffic violations. Master's thesis in the specialty 122 Computer science, educational program «Artificial intelligence systems». Vinnytsia: VNTU, 2023. 117 p.

In Ukrainian language. Bibliography: 40 titles; Figures: 19; Table 9.

This master's qualification work is devoted to the development of information technology for predicting the number of traffic violations. An analysis of modern analogue software used to predict the situation on the roads is carried out, a brief comparative description of the found analogue software is given, and methods that can be used to solve the task are investigated. The choice of method for forecasting the number of traffic violations is justified and a mathematical model based on the ARIMA method is developed. An algorithm for predicting the number of traffic violations and the corresponding software in the Java programming language, in the IntelliJ IDEA environment, have been developed. The analysis of the information technology has shown an increase in the accuracy of forecasting the number of traffic violations.

In the economic part of the article, the commercial potential of the development of information technology for predicting the number of traffic violations is assessed, the commercial potential of the development is assessed, the costs of research and implementation of the results are predicted, and the payback period is calculated.

The graphic part consists of 8 posters with modelling results.

Keywords: information technology, forecasting, traffic rules, road accidents.

ЗМІСТ

ВСТУП

Актуальність теми. Прогнозування кількості порушень правил дорожнього руху є актуальною темою досліджень у сфері безпеки дорожнього руху. Завдяки постійному розвитку технологій та наявності великих обсягів даних, використання аналітики і прогностичних моделей може допомогти виявити тенденції, фактори та можливі причини порушень правил дорожнього руху.

Дослідження у цій сфері можуть включати в себе розробку статистичних моделей, машинне навчання та штучний інтелект для прогнозування кількості порушень на основі історичних даних. Це може допомогти органам правопорядку та відповідальним органам розробити стратегії зменшення порушень правил дорожнього руху, спрямовувати ресурси на запобігання та контроль найбільш ймовірних порушень, знижувати кількість аварій та покращувати безпеку на дорогах загалом.

Актуальність досліджень у цій сфері полягає в постійній зміні умов дорожнього руху, поведінки водіїв та дорожньої інфраструктури. Шляхи прогнозування порушень можуть включати аналіз даних про швидкість руху, густоту трафіку, погодні умови, географічні особливості та інші фактори, що впливають на безпеку дорожнього руху.

Варто зазначити, що точність прогнозів може залежати від доступності якісних та актуальних даних, а також від складності моделей і алгоритмів, що застосовуються. Тому дослідження в цій галузі продовжуються для вдосконалення методів прогнозування та покращення ефективності заходів з безпеки дорожнього руху.

Нині існує потреба в науковому аналізі ситуації, прогнозуванні та плануванні заходів безпеки дорожнього руху, проведенні наукових і науково-технічних досліджень у сфері безпеки дорожнього руху, в тому числі: визначення економічної цінності дорожнього руху, втрата життя і здоров'я у

зв’язку з дорожньо-транспортними пригодами та пошкодження господарства внаслідок дорожньо-транспортних пригод.

Актуальність програмної реалізації інформаційної технології прогнозування кількості порушень правил дорожнього руху полягає в поліпшенні безпеки на дорогах, оптимізації ресурсів та вдосконаленні політики безпеки за допомогою аналізу даних та застосування сучасних методологій, як машинного навчання та штучного інтелекту, що можуть забезпечити більш точні та ефективні результати прогнозування

Зв’язок роботи з науковими програмами, планами, темами. Магістерська кваліфікаційна робота виконана відповідно до напрямку наукових досліджень кафедри комп’ютерних наук Вінницького національного технічного університету 22 К1 «Розробка прикладних інтелектуальних інформаційних технологій та систем».

Мета та завдання дослідження. Метою дослідження є підвищення точності прогнозування кількості порушень правил дорожнього руху.

Для досягнення поставленої мети потрібно розв’язати такі **задачі:**

– провести аналіз програм-аналогів та обґрунтувати доцільність розробки інформаційної технології прогнозування кількості порушень правил дорожнього руху;

– розробити структуру інформаційної технології прогнозування кількості порушень правил дорожнього руху;

– розробити математичну модель прогнозування кількості порушень правил дорожнього руху;

– здійснити програмну реалізацію інформаційної технології прогнозування кількості порушень правил дорожнього руху;

– провести тестування програми та проаналізувати отримані результати;

– економічно обґрунтувати доцільність розробки інформаційної технології прогнозування кількості порушень правил дорожнього руху.

Об’єктом дослідження є процес прогнозування кількості порушень правил дорожнього руху.

Предметом дослідження є програмні засоби прогнозування кількості порушень правил дорожнього руху, а також точність їх роботи.

Методи дослідження. У роботі використано такі методи наукових досліджень: метод системного аналізу для аналізу структури інформаційної системи; методи аналізу часових рядів; методи експертних оцінок; регресійні методи; методи машинного навчання; методи математичної статистики; методи об’єктно-орієнтованого програмування для автоматизації розрахунків.

Наукова новизна одержаних результатів полягає в наступному: удосконалено інформаційну технологію прогнозування кількості порушень правил дорожнього руху, яка відрізняється від існуючих використанням математичної моделі на основі методу часових рядів ARIMA та аналізу геопросторових даних. Це дозволяє максимально точно визначити кількість порушень правил дорожнього руху, аналізуючи просторові закономірності та передбачаючи ризиковані зони.

Практичне значення одержаних результатів полягає у такому:

1. Удосконалено алгоритм прогнозування кількості порушень правил дорожнього руху.
2. Здійснено програмну реалізацію інформаційної технології прогнозування кількості порушень правил дорожнього руху.

Достовірність теоретичних положень магістерської кваліфікаційної роботи підтверджується строгістю постановки задач, коректним застосуванням математичних методів під час доведення наукових положень, строгим виведенням аналітичних співвідношень, порівнянням результатів з відомими та збіжністю результатів математичного моделювання з результатами, що отримані під час впровадження розроблених програмних засобів.

Особистий внесок здобувача. Результати магістерської кваліфікаційної роботи отримані самостійно. В публікації у співавторстві здобувачу належить дослідження перспектив розробки інформаційної технології прогнозування кількості порушень правил дорожнього руху [1, 2].

Апробація результатів магістерської кваліфікаційної роботи. Результати досліджень було апробовано на LII Науково-технічній конференції підрозділів Вінницького національного технічного університету (НТКП ВНТУ–2023) м. Вінниці у 2023 р [2].

Публікації магістерської кваліфікаційної роботи. За основними результатами досліджень опубліковано тези доповіді на науково-технічних конференціях [1, 2].

1 ОБҐРУНТУВАННЯ ДОЦІЛЬНОСТІ РОЗРОБКИ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ПРОГНОЗУВАННЯ КІЛЬКОСТІ ПОРУШЕНЬ ПРАВИЛ ДОРОЖНЬОГО РУХУ

1.1 Аналіз статистики кількості порушень правил дорожнього руху

Упродовж останніх десятиліть у світі спостерігається стрімке збільшення кількості транспортних засобів та підвищення інтенсивності дорожнього руху, що призводить до збільшення кількості дорожньо-транспортних пригод (ДТП) та їх негативних наслідків. В Україні рівень смертності та травматизму внаслідок ДТП є одним із найвищих в Європі, а рівень організації безпеки дорожнього руху залишається вкрай низьким [3].

Безпека дорожнього руху – це розуміння всіма учасниками дорожнього руху правил дорожнього руху та неухильне їх дотримання. Водії автомобілів, велосипедисти, мотоциклісти та пішоходи зобов’язані знати та дотримуватися правил дорожнього руху.

Безпека дорожнього руху відноситься до заходів, політик, правил та практик, спрямованих на запобігання дорожньо-транспортним пригодам, зменшення ризику постраждалих та забезпечення безпеки всіх учасників дорожнього руху. Основна мета безпеки дорожнього руху полягає в запобіганні травмам, смертям та матеріальним збиткам, які виникають в результаті дорожньо-транспортних пригод. Це досягається шляхом розробки та впровадження ефективних заходів, що охоплюють різні аспекти безпеки на дорозі.

Безпека дорожнього руху є важливим аспектом суспільного життя, оскільки впливає на здоров'я, життя та благополуччя людей. Це вимагає спільних зусиль уряду, організацій, громадськості та кожного окремого учасника дорожнього руху для створення безпечного середовища на дорозі.

Більшість аварій можна було б запобігти заздалегідь, якби всі учасники дорожнього руху дотримувались правил дорожнього руху (ПДР).

За 2022 рік в Україні трапилось понад 18 тис. ДТП із потерпілими, у яких загинула 2791 особа та приблизно 23000 людей отримали травми. Статистика Національної поліції України вказує [4], що у порівнянні з 2021 роком, кількість ДТП, смертей та травмувань на дорогах у 2022 році зменшилась, однак зменшення рівня ДТП, пов’язано з багатьма причинами та факторами, включно зі зменшенням на певних дорогах країни транспортних засобів та, відповідно, інтенсивності дорожнього руху, окупацією територій тощо. Дані за 2022 рік охоплюють лише території, де не було активних бойових дій чи окупації. Таким чином, статистика не є повною щодо Харківської, Луганської, Донецької, Запорізької, Херсонської та Миколаївської областей, частини яких і досі залишаються окупованими. Також статистика вважається неповною через оформлення учасниками ДТП європротоколів, тому наявна інформація лише про випадки з постраждалими. Основними причинами дорожніх аварій стали: перевищення безпечної та встановленої швидкості руху; порушення правил маневрування; проїзду перехресть та пішохідних переходів; недотримання дистанції.

Наявність штучних перешкод на дорогах та відсутність дорожніх знаків, а також відключення світла знизили безпеку дорожнього руху, натомість були проведені заходи з посилення патрулювання ситуації на дорогах за допомогою збільшення кількості автопатрулів, відновлення камер автоматичної фіксації та лазерних вимірювачів швидкості.

В таблиці 1.1 наведено кількість дорожньо-транспортних пригод за причинами у 2022 році за даними патрульної поліції [5].

Таблиця 1.1 – Дорожньо-транспортні пригоди за причинами у 2022 році

Причини	ДТП з загиблими та/або травмованими		
	Усього ДТП	Загинуло осіб	Травмовано осіб
Перевищення безпечної швидкості	7561	1507	9461
Порушення правил маневрування	3846	343	4830
Порушення правил проїзду перехресть	1467	52	2111

Продовження таблиці 1.1

Причини	ДТП з загиблими та/або травмованими		
	Усього ДТП	Загинуло осіб	Травмовано осіб
Порушення правил проїзду пішохідних переходів	1443	111	1450
Недотримання дистанції	843	66	1070
Керування транспортним засобом у стані сп'яніння	790	91	1031
Перехід пішоходів у невстановленому місці	593	155	467
Перевищення встановленої швидкості	469	126	614
Виїзд на смугу зустрічного руху	418	148	699
Невиконання водіями вимог сигналів регулювання	236	9	312
Неочікуваний вихід на проїзну частину	233	43	201
Порушення правил надання безперешкодного проїзду	184	15	257
Порушення правил обгону	184	33	290
Невиконання пішоходами вимог сигналів регулювання	48	10	43
Порушення ПДР пішоходами у стані сп’яніння	47	15	34
Перевтома, сон за кермом	46	20	48
Порушення техніки безпеки пасажиром	46	5	42
Порушення правил зупинки і стоянки транспортного засобу	44	8	44
Порушення правил перевезення пасажирів	43	13	37
Керування несправним транспортним засобом	39	4	57
Порушення правил проїзду залізничних переїздів	16	10	14
Порушення правил перевезення вантажів	10	3	9
Порушення правил буксирування	7	1	6
Порушення правил утримання автодоріг та вулиць	6	2	6
Порушення правил користування зовнішніми світловими приладами транспортних засобів	5	1	8
Порушення вимог ПДР погоничем тварин	2	0	2
Порушення правил проїзду зупинок громадського транспорту	2	0	2
ВСЬОГО по Україні	18 628	2791	23 145

Як видно з таблиці 1.1, надмірна швидкість залишається вбивцею №1 на українських дорогах. За даними Патрульної поліції, у першому півріччі 2022 року понад 44% ДТП сталися саме через перевищення швидкості. Також вона збільшує ризик потрапити у аварію. Адже якщо виникає якась нестандартна ситуація, у водія менше часу – а отже, і можливостей, щоб адекватно відреагувати (до того ж, гальмівний шлях стає довшим). Якщо аварія все ж сталася, швидкість збільшує тяжкість травм, а отже й імовірність смертельних наслідків. Це стосується всіх учасників руху: водіїв авто, пішоходів, велосипедистів, мотоциклістів. За даними ВООЗ, зменшення середньої швидкості руху на 5% може скоротити кількість смертельних ДТП на 30% [6].

У Вінницькій області за рік було зафіксовано 414 ДТП з загиблими або травмованими [7]. Порівняно з 2021 роком кількість таких ДТП зменшилась лише на 1,2%, тобто майже не змінилась, що означає недостатньо вжитих заходів з безпеки дорожнього руху, або збільшення обсягу транспорту.

1.2 Визначення факторів, що впливають на кількість порушень правил дорожнього руху

На початковому етапі впровадження системи автоматичної фіксації адміністративних правопорушень в сфері безпеки дорожнього руху передбачає фіксацію лише таких видів порушень ПДР, як перевищення встановлених швидкісних режимів для транспортних засобів. Зазначені правопорушення фіксуватимуться системами автоматичного фіксування – технічними засобами (пристроями контролю), що забезпечують автоматичне виявлення та фотографування чи відеофіксацію подій, які мають ознаки зазначених правопорушень. Системи автоматичної фіксації порушень правил дорожнього руху встановлюються у місцях (майданчиках), схильних до аварій та місцях підвищеної ймовірності настання дорожньо-транспортних пригод на дорогах, на вулицях за погодженням з компетентним уповноваженим підрозділом Національної поліції. Автоматичні системи фіксації порушень часто

розміщуються на перехрестях, де історія показує підвищений ризик аварій через порушення правил, такі як проїзд на червоне світло або неправильний поворот. Встановлення таких систем у шкільних зонах, де є велика кількість дітей, допомагає виконувати обмеження швидкості та забезпечувати їхню безпеку. Деякі системи встановлюються на ділянках доріг з будівництвом або ремонтом, де водії можуть порушувати обмеження швидкості та правила безпеки. У місцях з великою кількістю пішоходів, які перетинають дорогу, системи автоматичної фіксації порушень можуть допомагати зменшувати ризик ДТП. Іноді системи автоматичної фіксації порушень доцільно встановити на ділянках, де вже було зафіксовано велику кількість порушень та аварій.

Встановлення цих систем допомагає відстежувати порушення правил та надавати докази для проведення правових дій проти порушників. У багатьох випадках це сприяє підвищенню безпеки на дорогах та зменшенню кількості ДТП.

Використовуючи дані інформаційно-телекомунікаційної системи «Інформаційний портал Національної поліції України» [4] було визначено топ-5 порушень правил дорожнього руху. Найпоширенішими порушеннями є перевищення швидкості та проїзд на заборонений сигнал (ч. 1, ст. 122). За цією статтею мають право оштрафувати водіїв, які рухаються зі швидкістю, більшою дозволеної на 20 км/год, тих, хто не виконує вимоги дорожніх знаків, перетинає суцільну лінію дорожньої розмітки, перевозить вантаж, проводить буксирування, зупиняється, проїжджає пішоходні переходи з порушенням правил дорожнього руху. Також за цією статтею можуть оштрафувати при ігноруванні пішоходів на нерегульованих переходах чи за рух транспорту по тротуарах, чи пішохідних доріжках.

На другому місці в рейтингу топ-5 порушень – ігнорування вимог про зупинку транстпортного засобу (ч. 2, ст. 122). Ця частина статті передбачає накладення штрафу через такі порушення: недотримання правил проїзду перехрестя, зупинки транспортних засобів; проїзд на заборонений сигнал світлофора чи жест регулювальника; недотримання правил обгону та руху в

зустрічному напрямку, зменшення безпечної дистанції або інтервалу, стоянка на проїжджій частині; порушення правил обгону та зустрічного дорожнього руху; порушення правил використання приладів зовнішнього освітлення або попереджувальних сигналів під час старту або зміни напрямку руху; використання наведених пристроїв та їх перебудови з порушенням наявних вимог; порушення правил використання мобільних пристроїв та засобів зв’язку; порушення правил поведінки.

Третє місце серед порушень правил дорожнього руху передбачає порушення правил користування ременями безпеки або мотошоломами.

Наступне часте порушення – керування автомобілем при відсутності водійського посвідчення, свідоцтва про реєстрацію транспортного засобу або страхового полісу.

Закриває топ-5 статей порушення, які можуть зумовити пошкодження транспортного засобу, вантажу, інфраструктури, доріг, вулиць, переїздів тощо.

Аналіз кількості ДТП, які сталися в Україні за 10 місяців минулого року та порівняно з 2021 роком, показує, що встановлення камер для фіксації порушень ПДР не вплинуло на кількість ДТП. Свідчення про це надає платформа для роботи з відкритими даними Opendatabot з посиланням на дані Нацполіції [8].

Досить багато факторів, що залежать від водія, спричиняють порушення правил дорожнього руху та провокують ДТП.

Фактори, пов'язані з водієм:

- недотримання правил дорожнього руху (перевищення швидкості, проїзд на червоне світло, неправильне виконання обгону та інші порушення правил);
- неправильна поведінка водіїв (агресивна їзда, водіння у стані алкогольного або наркотичного сп'яніння, використання мобільних пристроїв під час керування та інші небезпечні дії).

Фактори, пов'язані з дорогами та інфраструктурою:

- погана якість дорожнього покриття (ями, тріщини, нерівності та інші дефекти, які можуть сприяти виникненню аварій);

- розмітка доріг (незрозумілі або відсутні знаки, неправильне позначення смуг руху та інші проблеми в розмітці);

- відсутність або недостатня ефективність дорожньої сигналізації (відсутність або погана видимість світлофорів, неправильне функціонування сигналів та інші проблеми з сигналізацією).

Фактори, пов'язані з умовами дорожнього руху:

- густота трафіку (велика кількість транспортних засобів на дорозі може створювати ускладнення та збільшувати ризик аварій);

- погодні умови (дощ, туман, ожеледиця, снігопад та інші погодні умови можуть знижувати видимість та погіршувати умови руху, що призводить до аварій).

Технічний стан транспортних засобів:

- несправність автомобілів (проблеми з гальмами, шинами, світловою сигналізацією та інші технічні недоліки можуть призвести до аварій).

Соціально-економічні фактори:

- рівень освіти та культура дорожнього руху (недостатня освіченість водіїв щодо правил дорожнього руху та безпеки на дорозі може призводити до аварій);

- економічний статус (відсутність коштів на технічне обслуговування автомобілів або вибір більш дешевих, але менш безпечних транспортних засобів можуть збільшувати ризик аварій).

Ці чинники часто взаємодіють між собою, і уникнення їх впливу на дорожньо-транспортні пригоди вимагає комплексного підходу, включаючи поліпшення інфраструктури, забезпечення освіти та свідомості водіїв, посилення контролю та підвищення вимог до безпеки дорожнього руху.

Перераховані фактори мають різний вплив на ситуацію, в деяких випадках високий, в інших – низький. Також потрібно розуміти, що є психологічний фактор, який неможливо передбачити, а тому і враховувати. Однак зважаючи на ті фактори, які можна врахувати, розглянемо методи, які допоможуть вирішити задачу прогнозування кількості порушень правил дорожнього руху.

Потрібно зазначити, що швидкість має вплив як на ризик, так і на тяжкість ДТП. Співвідношення між швидкістю та пошкодженням має вирішальне значення для вразливих учасників дорожнього руху, таких як пішоходи та велосипедисти.

Наприклад, на швидкості 30 км/год водій має побачити ситуацію і вирішити загальмувати за 16 метрів до пішохода, а на швидкості 50 км/год ця відстань становить 35 метрів. Також потрібно враховувати, що на мокрому асфальті гальмівний шлях збільшується на 25%, а на льоду – на 50%. Рухаючись зі швидкістю 60 км/год, водій зможе прийняти рішення проїхавши в середньому 25 метрів до того, як натисне на гальма, що на вулиці міста може означати летальний результат.

Отже, при збільшенні швидкості руху автомобіля зона фокусування уваги водія звужується, а тривалість огляду зменшується (рис. 1.1).

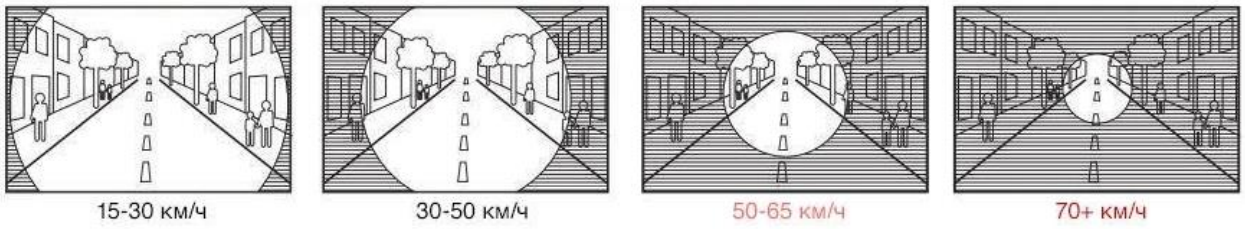

Рисунок 1.1 – Залежність зони фокусування від швидкості

Зменшення зони концентрації при підвищення швидкості підтверджує підвищення інтенсивності роботи водія, при чому зменшення цієї зони здійснюється за рахунок периферійних зон, а це означає що інформація поза зоною концентрації може не сприймаєтися водіями. При швидкості понад 50 км/год водій не може враховувати обставини, що відбуваються на тротуарі з відстані близько 25 метрів [9].

Рішенням проблем з дорожньо-транспортними пригодами та зменшення їх негативних наслідків може бути обґрунтоване системною роботою з усіма факторами, що впливають на аварійність. Необхідно розробити та послідовно впроваджувати комплексні заходи щодо стану доріг, забезпечення їх засобами

зниження аварійності, сучасним технологічним контролем за дотриманням дорожнього законодавства та роз'яснювальними заходами для перехожих. Фактично потрібно переходити від індивідуальних заходів до створення цілісної системи, яка повністю підтримуватиме безпеку дорожнього руху та мінімізуватиме наслідки ДТП як для кожного, так і для суспільства в цілому.

Більшість країн світу давно зрозуміли потребу в проведенні робіт з дорожньою інфраструктурою для забезпечення організації руху, що сприятиме усуненню можливих помилок людей. Не можна чекати від людей ідеальної поведінки та звинувачувати лише їх у спричиненні аварій або нещасних випадків на дорозі. Існує потреба в організації руху таким чином, щоб помилки людей не призвели до фатальних наслідків [10].

1.3 Порівняння існуючих моделей та методів, що застосовуються для прогнозування кількості порушень правил дорожнього руху

Прогнозування кількості порушень правил дорожнього руху може використовувати різні математичні моделі та методи, залежно від наявної інформації та поставленої задачі. Наведемо кілька типових підходів, які можуть застосовуватися в таких випадках:

Регресійні моделі. Можна побудувати математичну модель, яка передбачає кількість порушень дорожніх правил на основі історичних даних. Використовуючи методи регресії, можна аналізувати залежність між різними факторами, такими як час доби, погодні умови, типи доріг, густота трафіку та інші, та прогнозувати кількість порушень на основі цих факторів.

Часові ряди. Якщо доступні часові дані про кількість порушень правил дорожнього руху протягом певного періоду, можна використати методи аналізу часових рядів, такі як ARIMA (авторегресійна інтегрована ковзна середня) або SARIMA (сезонна авторегресійна інтегрована ковзна середня), для прогнозування майбутніх значень кількості порушень.

Методи машинного навчання. Методи машинного навчання, такі як нейронні мережі, дерева рішень, ансамблеві моделі (наприклад, випадковий ліс) та інші, також можуть використовуватися для прогнозування кількості порушень [11]. Зазвичай використовуються як статистичні характеристики, такі як попередня кількість порушень, так й інші ознаки, які можуть бути корисними для прогнозування, наприклад, типи доріг, географічне розташування тощо.

Аналіз геопросторових даних. Якщо доступні дані про місцеположення порушень правил дорожнього руху (наприклад, GPS-координати), можна використовувати геопросторовий аналіз для прогнозування майбутніх зон або ділянок, де ймовірність порушень є високою. Можуть застосовуватися методи геостатистики, класифікації та кластеризації для аналізу просторових закономірностей та передбачення ризикових зон.

Прогнозування – процес передбачення майбутнього стану об’єкта або явища на основі аналізу минулого та теперішнього, систематично оцінює інформацію про якісні та кількісні характеристики обраного об’єкта чи явища в майбутньому [12]. У цьому підрозділі надано алгоритмічне та математичне забезпечення для вирішення завдань аналізу набору математичних моделей для найбільш точної реалізації прогнозу.

Лінійна регресія − це статистичний метод, який використовується для вивчення взаємозв'язків між залежною змінною (змінною, яку ви намагаєтеся передбачити) та однією чи кількома незалежними змінними (факторами), які служать для прогнозування залежної змінної. Метод лінійної регресії передбачає, що існує лінійна залежність між незалежними змінними і залежною змінною [13].

Лінійна регресія передбачає, що залежна змінна залежить від незалежних змінних за допомогою лінійної функції. Це означає, що зміни в незалежних змінних пов'язані зі змінами в залежній змінній лінійним чином. У лінійній регресії є коефіцієнти, які описують цю лінійну залежність. Коефіцієнти регресії вказують, наскільки змінюється залежна змінна при зміні кожної незалежної змінної.

Модель лінійної регресії створюється так, щоб мінімізувати помилки між фактичними та передбаченими значеннями залежної змінної. Ця мінімізація робить модель найкращою лінійною апроксимацією вихідних даних. Лінійна регресія використовується для прогнозування значень залежної змінної на основі значень незалежних змінних. Також вона допомагає аналізувати взаємозв'язки та вплив незалежних змінних на залежну. Методи оцінки лінійної регресії допомагають визначити, наскільки модель є надійною та які точності можна досягти у прогнозуванні.

Лінійна регресія є одним із найпоширеніших методів в статистиці та аналізі даних, і вона широко використовується в різних галузях, включаючи економіку, фінанси, медицину, науку про дані, соціологію та інші. Важливо враховувати, що лінійна регресія припускає наявність лінійних залежностей та може бути некорисною, якщо залежність не є лінійною.

У простій задачі регресії (один x і один y) модель описується рівнянням (1.1).

$$y_t = \beta_0 + \beta_1 x_1 + \varepsilon_t, \tag{1.1}$$

де β_0 і β_1 – коефіцієнти регресії,

ε – помилка моделі,

t – момент часу.

У випадку, коли існує більше одного входу x, лінію називають площиною або гіперплощиною. Модель прогнозування харектеризується рівнянням (1.2):

$$y_t = \beta_0 + \beta_1 x_{1,t} + \beta_2 x_{2,t} + \cdots + \beta_n x_{n,t} + \varepsilon. \tag{1.2}$$

Візуальне представлення лінійної регресії наведено на рисунку 1.2.

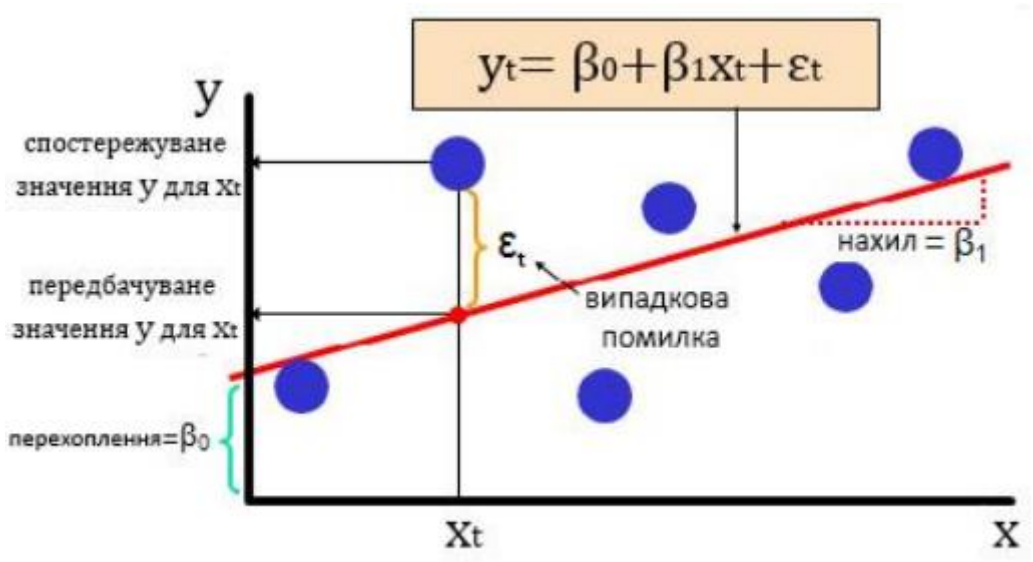

Рисунок 1.2 – Візуальне представлення лінійної регресії

У виападку, коли коефіцієнт дорівнює нулю, він ефективно усуває вплив вхідних змінних на модель а, отже, на передбачення, зроблене з моделі ($0 \cdot x = 0$). Існують вдосконалення для вивчення лінійної моделі, що називаються методами регуляризації. Вони намагаються як мінімізувати похибку квадрата моделі на навчальних даних (використовуючи звичайні найменші квадрати), так і зменшити складність моделі (наприклад, кількість або абсолютне значення суми всіх коефіцієнтів моделі).

Метод дерева рішень є одним із популярних методів машинного навчання, який використовується для задач класифікації та регресії. Цей метод заснований на структурі дерева, де кожен вузол представляє питання або гілку, а листя дерева представляють клас або значення, яке ми намагаємося передбачити. Метод дерева рішень має кілька важливих особливостей:

Розгалуження: дерево починається з кореневого вузла, який представляє всі дані в наборі даних. На кожному кроці алгоритм вибирає найкращу ознаку (фактор), за допомогою якої можна розділити дані на меншу кількість підгруп. Цей процес триває, доки ми не досягнемо критерію зупинки, такого як максимальна глибина дерева або мінімальна кількість прикладів в аркуші.

Класифікація та регресія: дерева рішень можна використовувати як для задач класифікації, де вони призначають класи кожному листу, так і для задач регресії, де вони прогнозують числові значення.

Вибір найкращої функції для розгалуження зазвичай базується на кількох критеріях. Найпоширеніші критерії включають ентропію, індекс Джині та середньоквадратичну помилку.

Прагнення до простоти: дерева рішень зазвичай спрямовані на створення простої моделі, яку можна інтерпретувати, яку легко зрозуміти людям.

Обробка відсутніх даних: метод дерева рішень може обробляти відсутні дані, розглядаючи їх як окремий варіант або пропускаючи об’єкти з відсутніми даними.

Здатність вирішувати складні завдання: дерева рішень можуть розгалужувати дані та долати складні зв’язки між функціями.

Дерева рішень можуть надавати оцінку важливості атрибутів, яка допомагає визначити, які фактори мають найбільший вплив на результат.

Метод дерева рішень має велику кількість застосувань у різних сферах, таких як бізнес, медицина, наука про дані та багато інших. Він може бути ефективним інструментом для класифікації та прогнозування великої кількості завдань. Однак важливо мати на увазі, що дерева рішень схильні до перенавчання (панельної реконструкції), і їх продуктивність можна покращити за допомогою широкого спектру методів регулярізації та оптимізації (рис.1.3).

Одним з найкращих і найбільш широко використовуваних методів навчання вчителів є алгоритми на основі дерева [14]. Вони дозволяють моделюванню передбачати з більшою точністю, стабільністю та легшою інтерпретацією. На відміну від інших алгоритмів підготовки вчителів, алгоритм дерева рішень також можна використовувати для розв’язування задач регресії та класифікації. Такі методи, як дерева рішень, випадковий ліс, покращення градієнта, широко використовуються в багатьох типах проблем науки про дані. Загальна причина використання дерева рішень полягає у створенні моделі навчання, яку можна використовувати для прогнозування класу або значення

цільових змінних шляхом вивчення правил прийняття рішень, отриманих з попередніх (навчальних) даних.

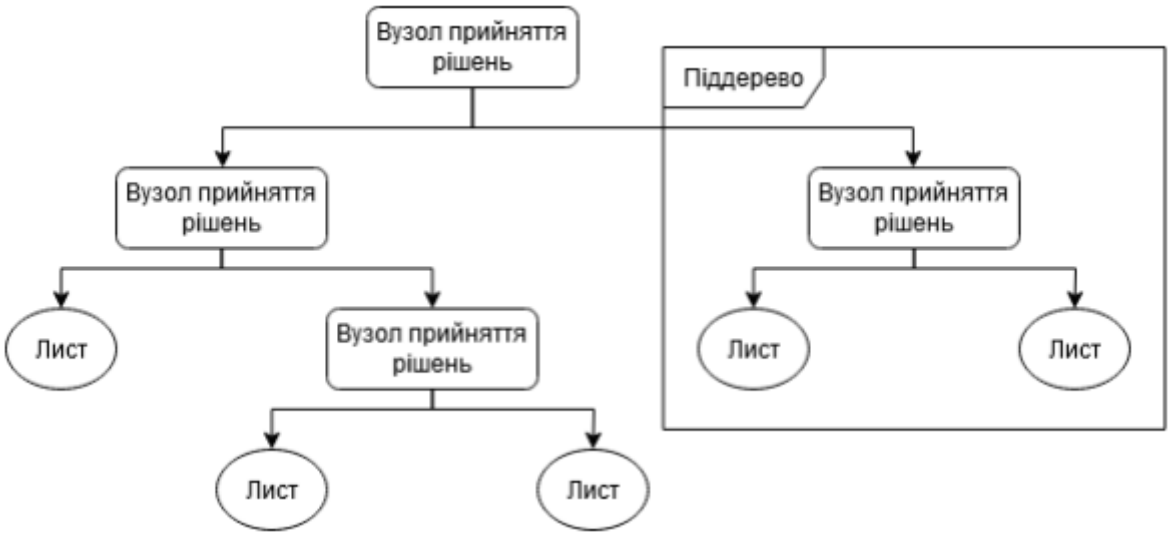

Рисунок 1.3 – Візуальне представлення дерева рішень

Основна ідея будь-якого алгоритму дерева рішень полягає в наступному:

1. Проведення вибору найкращого атрибуту за допомогою параметрів вибору атрибутів (ASM), щоб розділити записи.

2. Перетворення атрибуту на вузол пряйняття рішеннь та розкладання набору даних на менші підмножини.

3. Побудова дерева, рекурсивне повторення процесу для кожного дочірнього елемента, поки не буде виконана одна з умов:

- усі кортежі належать до одного значення атрибута;

- більше атрибутів немає;

- більше випадків немає.

Підвищення точності цієї моделі забезпечується градієнтним збільшення (GBM). Моделі машинного навчання можуть бути пристосовані до даних окремо або набором. Набір являє собою комбінацію простих окремих моделей, які разом створюють нову, потужнішу модель.

Цільова оцінка для кожного випадку в даних залежить від того, як зміна прогнозу в цьому випадку впливає на загальну помилку прогнозу:

- якщо невелика зміна прогнозу для випадку призводить до значного зменшення похибки, тоді важливий наступний цільовий результат для цього випадку. Прогнозування нової моделі ближче до її цілей зменшить помилку;

- якщо невелика зміна прогнозу в цьому випадку не змінює помилку, то наступний цільовий бал для цього випадку дорівнює нулю. Зміна цього прогнозу не зменшує помилку.

Моделі часових рядів використовують минулі рухи змінних для прогнозування їх майбутніх значень [15]. Моделі часових рядів – це математичні структури, які використовуються для опису та прогнозування часових рядів. Ці моделі є важливим інструментом у статистичному аналізі та прогнозуванні динаміки різних явищ, таких як фінансові показники, економічні показники, погода, виробництво тощо.

Модель часового ряду Бокса-Дженкінса, також відома як ARIMA (Autoregressive Integrated Moving Average), є широко використовуваною моделлю для аналізу та прогнозування часових рядів. Ця модель була введена Джорджем Боксом та Гвідо Дженкінсом, і вона включає в себе авторегресійні (AR), інтегровані (I) та ковзні середні (MA) компоненти. Стаціонарний ряд Y_t позначається $ARMA$ (p, q), якщо виконується рівняння (1.3).

$$Y_t - \varphi_1 Y_t - 1 - \cdots - \varphi_p Y_t - p = \varepsilon_t + \theta_1 \varepsilon_t - 1 + \cdots + \theta_1 \varepsilon_t - q. \quad (1.3)$$

де ε_t – білий шум і відсутній загальний фактор між авторегресивним многочленом і ковзним середнім многочленом [16].

Основні етапи методології Бокса-Дженкінса:

1. Ідентифікація моделі: визначення стаціонарності часового ряду, здійснення диференціації, якщо ряд не є стаціонарним.

2. Оцінка параметрів: визначення підходящих значень параметрів AR, I та MA під час аналізу автокореляційних та часткових автокореляційних функцій (ACF та PACF).

3. Перевірка моделі: застосування отриманих параметрів до моделі ARIMA та аналіз отриманого ряду.

4. Прогнозування: застосування побудованої моделі для прогнозування майбутніх значень часового ряду.

Метод ARIMA може бути використаний для моделювання широкого спектру часових рядів, включаючи ті, що мають тренд, сезонність та інші характеристики. Модель може ефективно враховувати тренд та сезонні зміни в часових рядах, що робить її корисною для прогнозування рядів з складними залежностями. Метод ARIMA дозволяє проводити діагностику моделі, включаючи аналіз залишкових помилок, для перевірки адекватності побудованої моделі. Параметри ARIMA (авторегресії, інтеграції, ковзних середніх) мають інтерпретацію, що полегшує розуміння внутрішньої структури моделі. Якщо часовий ряд має стаціонарну структуру, то підбір параметрів може бути здійснений автоматично, що робить ARIMA відносно легким у використанні.

Методологія ARIMA намагається описати рух стаціонарного часового ряду за допомогою параметрів так званих «авторегресивних і ковзних середніх». Вони називаються параметрами AR (авторегресивні параметри) і параметрами MA (ковзне середнє). Модель AR з одним параметром можна записати у вигляді рівняння (1.4).

$$X(t) = A(1)X(t-1) + E(t), \tag{1.4}$$

де $X(t)$ – досліджуваний часовий ряд,

$A(1)$ – автоматичний параметр першого порядку,

$X(t-1)$ – часовий ряд, що відставав під час першого періода,

$E(t)$ – термін помилки модулі.

За допомогою технології ARIMA можна створювати моделі, які включають як авторегресію, так і ковзні середні. Ці моделі часто мають назву «змішані моделі». Попри те, що прогнозування стає при цьому складнішим, структура може фактично краще імітувати серію та створювати більш точний прогноз [17].

Отже, для прогнозування кількості порушень правил дорожнього руху, метод ARIMA найкраще підходить, оскільки він дозволяє побудувати тренд-модель, що включає дані про попередні порушення як вектор вхідних даних, а також дані про майбутні порушення як вихідний вектор. Також доцільно використовувати розширення методу, що враховує сезонні коливання в часових рядах. Цей метод може бути ефективним для прогнозування кількості порушень правил дорожнього руху, оскільки часто існує сезонність в таких данних (наприклад, підвищення кількості порушень у вихідні дні чи під час певних подій). Отже, доцільно використовувати розширену модель ARIMA для розробки інформаційної технології прогнозування кількості порушень правил дорожнього руху.

1.4 Огляд сучасних продуктів для прогнозування кількості порушень правил дорожнього руху

Нині існує потреба в науковому забезпеченні аналізу ситуації, прогнозуванні та плануванні заходів з безпеки дорожнього руху, проведення наукових і науково-технічних досліджень у сфері безпеки дорожнього руху, у тому числі: визначення економічної оцінки вартості дорожнього руху; пошкодження господарства внаслідок дорожньо-транспортних пригод.

Прогнозування дорожньо-транспортних пригод, травм і загиблих є важливим завданням для планувальників безпеки дорожнього руху. Ці прогнози зазвичай корисні для кращого розуміння тенденцій аварій та ефективності існуючих заходів безпеки. Тобто для людей, здантий забезпечити безпеку представляє інтерес оцінити поточну політику та заходи безпеки, розглядаючи майбутні тенденції аварій та вживаючи коригуючих заходів.

Проаналізуємо програмне забезпечення, яке дозволяє прогнозувати ситуації на дорогах для вибору прототипного рішення.

Першою розглянемо програму Aimsun. Aimsun є програмним засобом для моделювання дорожнього руху та прогнозування ДТП (рис. 1.4). Він

використовує агентне моделювання, щоб досліджувати поведінку водіїв та вплив різних факторів на безпеку дорожнього руху. Приклад використання програмного забезпечення для моделювання порушень транспортних засобів і дослідження їх впливу на ризик ДТП на швидкісних магістралях показав, що вплив порушень зростає з інтенсивністю трафіку [18].

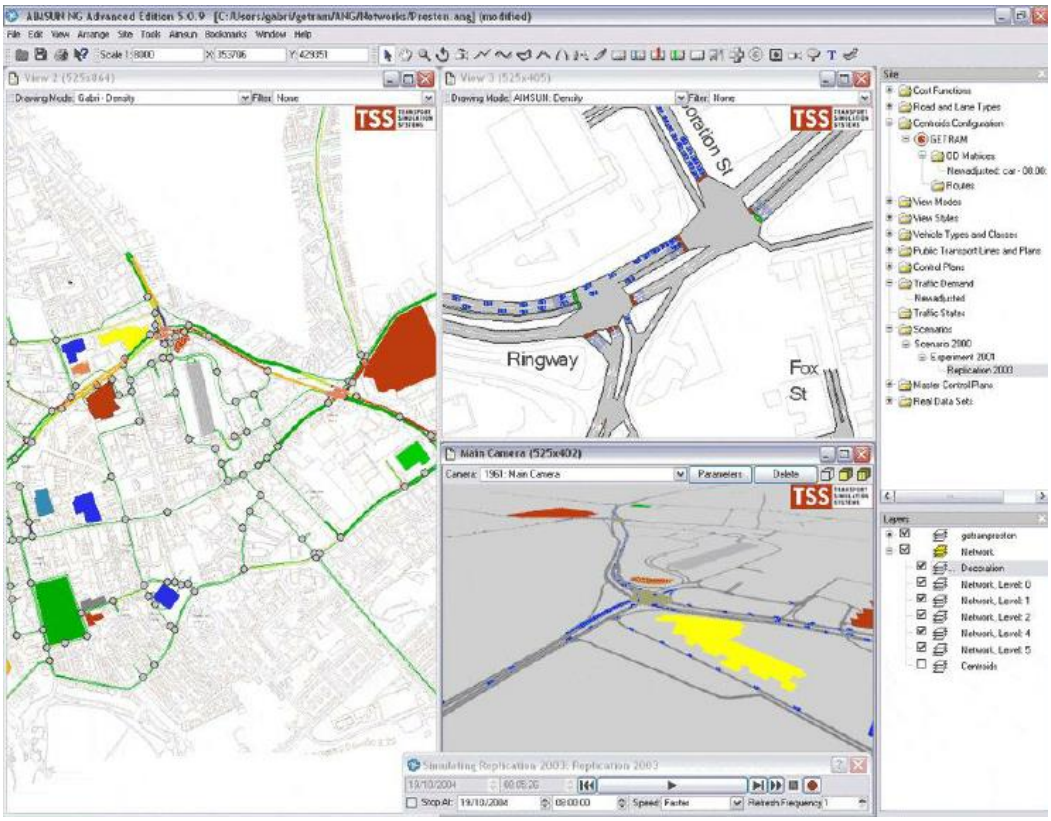

Рисунок 1.4 – Приклад роботи Aimsun

Загалом, автомобільні симулятори Aimsun пропонують кілька стратегій управління дорожнім рухом, які дозволяють користувачам змінювати умови мережі руху, впливати на поведінку водія або моделювати події в мережі руху.

Стратегії управління дорожнім рухом також можна використовувати для моделювання заторів на дорогах за періодичних або одноразових умов руху. У цій технічній примітці пояснюється, як зменшити затори на дорозі за допомогою таких дій:

A. Вимкнення резервної смуги.

B. Зміна плану контролю.

C. Зміна швидкості.

D. Активація/деактивація дій щодо трафіку на основі даних виявлення за допомогою тригерів.

E. Активація моделі співпраці Turn.

VISSIM є програмним засобом для моделювання руху транспортних потоків та прогнозування ДТП. Він дозволяє створювати віртуальні моделі доріг та транспортних засобів і аналізувати їх поведінку з точки зору безпеки дорожнього руху (рис. 1.5).

Рисунок 1.5 – Приклад використання VISSIM

Програмне забезпечення виконує такі завдання:

1) побудова транспортної мережі будь-якої складності з урахуванням індивідуальних і швидкісних особливостей дорог та вулиць;
2) вибір оптимальної схеми організації руху на перехресті;
3) оцінка пропускної здатності для кожного варіанта руху;
4) моделювання регульованих і нерегульованих перехресть;
5) прогнозування виникнення заторів;
6) моделювання та оптимізація роботи світлосигнальних пристроїв;
7) моделювання та аналіз пішохідного руху;

8) широкий спектр аналізу: для відрізків, транспортних засобів, пішоходів, світлосигнальних пристроїв, маршрутів громадського транспорту, перехрестя в цілому;

9) створення презентаційних матеріалів у вигляді відеороликів [19].

TransCAD є програмним пакетом для географічного аналізу та моделювання транспортних систем [20]. Він може використовуватися для прогнозування ДТП шляхом моделювання руху транспортних потоків, оцінки безпеки дорожнього руху та ідентифікації зон з високим ризиком.

Основні можливості та характеристики TransCAD включають:

Моделювання транспортних мереж: TransCAD дозволяє створювати та аналізувати цифрові моделі транспортних мереж, включаючи дороги, автомагістралі, масовий транспорт, велосипедні та пішохідні шляхи, залізниці тощо.

Аналіз трафіку: програма допомагає визначити потоки транспорту, часи руху, швидкості і завантаження дорожніх мереж. Вона може бути використана для прогнозування та оптимізації трафіку.

Планування та оптимізація маршрутів: TransCAD дозволяє розробляти оптимальні маршрути для транспортних засобів і громадського транспорту, враховуючи різні фактори, такі як час в дорозі, витрати на пальне, затори тощо.

Аналіз громадського транспорту: програма допомагає вивчати і оптимізувати мережі громадського транспорту, розраховувати розклади та визначати рейси.

Географічна інформаційна система (ГІС): TransCAD має вбудовані функції ГІС, що дозволяють працювати з географічними даними, створювати карти і проводити аналіз на основі просторової інформації.

Аналіз природних катастроф і безпеки дорожнього руху: програма може бути використана для аналізу впливу природних катастроф на транспортну інфраструктуру та безпеку дорожнього руху.

Інтеграція з іншими джерелами даних: TransCAD може імпортувати та експортувати дані з інших джерел, таких як бази даних, географічні інформаційні системи та джерела транспортних даних (рис. 1.6).

Скриптінг та автоматизація: програма підтримує скриптінг, що дозволяє користувачам створювати власні автоматизовані процеси та розширювати функціональність за допомогою скриптів і плагінів.

Рисунок 1.6 – Приклад роботи TransCAD

SAS Visual Analytics є програмним засобом для аналізу даних, включаючи дані про ДТП. Він може використовуватися для виявлення тенденцій, залежностей та прогнозування кількості ДТП на основі статистичних методів та аналізу даних [21].

Функції SAS Visual Analytics:

1) виявлення даних завдяки інтерактивній взаємодії та візуалізації (анімовані та мережеві діаграми, теплові карти, кореляційні матриці, дерево рішень тощо). Використання встановлених форм та графіків, створення власних уявлень;

2) бізнес-аналітика без залучення IT-фахівців, що дозволяє побудувати найбільш актуальне подання даних та вибрати оптимальний прогноз;

3) доступ до даних через мобільний додаток;

4) візуальний аналіз географічних даних на карті;

5) імпорт та експорт даних, а також можливість автоматичного оновлення та синхронізація;

6) управління обліковими записами з можливістю моніторингу активності користувачів.

Також розглянемо програмне забезпечення PTV Visum. PTV Visum - це програмне забезпечення для моделювання дорожнього руху та транспортних систем. PTV Visum розробляється компанією PTV Group і використовується для аналізу, планування і оптимізації різних аспектів дорожнього руху та транспортних систем. Це програмне забезпечення можна назвати базовим для проектування транспортних стратегій, інтегрованих транспортних схем, а також інтегрованих схем управління рухом. Приклад роботи PTV Visum наведено на рисунку 1.7.

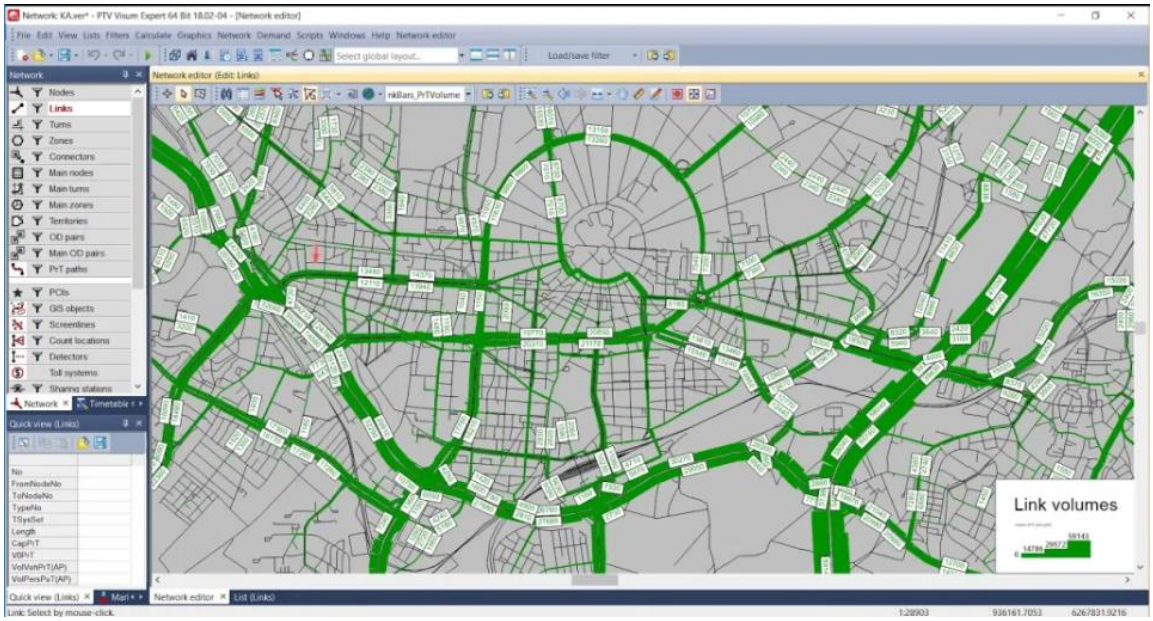

Рисунок 1.7 – Приклад роботи PTV Visum

PTV Visum включає наступні функції [22]:

- створення та налаштування моделей дорожнього руху для різних типів доріг і розташувань;

- аналіз обсягів руху, швидкостей та інших параметрів для прогнозування трафіку в різний час і на різних ділянках;

- визначення ефективних транспортних мереж для різних умов і завдань;

- розробка та аналіз оптимальних маршрутів для транспортних засобів;

- вивчення інтенсивності руху, перевантажень, пропускної спроможності і інших параметрів транспортних потоків;

- встановлення та аналіз зон для різних типів транспорту;

- можливість інтеграції PTV Visum з іншими програмами для більш широкого аналізу та вирішення завдань.

Завудяки модулю Safety є можливість імпорту даних про аварії та проектування цієї інформації на мережу доріг. Даний модуль Safety «Модель генерації ДТП» дає можливість простій статистичній моделі прогнозувати ДТП на перехрестях та на дорогах за допомогою статистики ДТП. Для створення такої моделі враховуються лише часткові дані мережі та аварій.

Якщо прогнозована кількість аварій у певному місці значно менша за реальну кількість, це означає концентрацію аварій, що можна зменшити зміною локальних умов.

HERE Traffic Analytics [23] – це сервіс компанії HERE Technologies, який надає аналітику трафіку та пов'язану інформацію. HERE Technologies спеціалізується на геолокаційних технологіях, картографії та пов'язаних інноваціях у сфері транспорту. Продукт дає можливість надавати інформацію про обсяги руху, швидкість руху транспортних засобів та інші параметри трафіку для різних ділянок доріг. Завдяки HERE Technologies можна здійснити аналіз імовірних транспортних заторів і прогнозування трафіку на основі історичних даних та реального часу. Він також надає рекомендації для оптимальних маршрутів на основі поточних умов трафіку, вивчає географію руху транспортних засобів, визначає популярні маршрути і точки призначення.

Також доступна інтеграція з іншими послугами. Наприклад, взаємодія з іншими сервісами HERE Technologies, такими як HERE Maps, для надання повного спектру геолокаційних та картографічних рішень.

Також цей програмний продукт дозволяє користувачам зрозуміти, що відбувається на асфальтованих дорогах, використовуючи потужну інтелектуальну інформацію про місцезнаходження та, наприклад, переглядати середню швидкість у місті за вибраний період. Це дає змогу приймати кращі обґрунтовані рішення щодо будівництва доріг, руху транспорту, управління мережею доріг та землекористування (рис. 1.8).

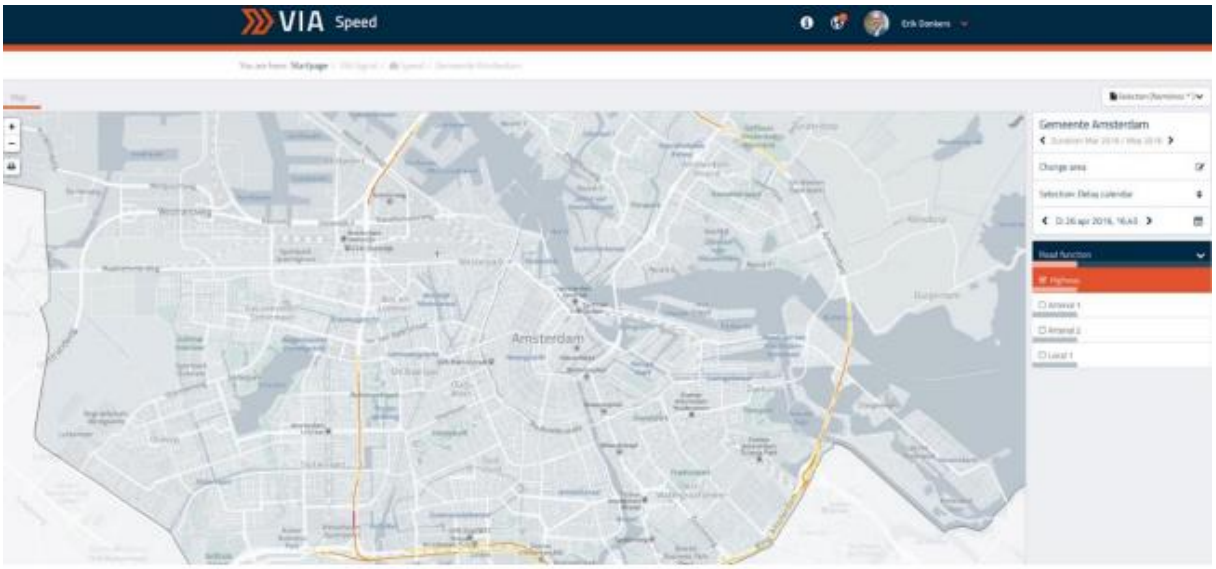

Рисунок 1.8 – Приклад роботи HERE Traffic Analytics

Розглянутий продукт прогнозує затори та інтенсивність руху в короткостроковій перспективі шляхом аналізу доступних даних, але не може передбачити кількість порушень ПДР.

Тому, проаналізувавши додатки для прогнозування дорожніх ситуацій, було вирішено вибрати PTV Visum як прототип, оскільки з його допомогою можна симулювати кількість порушень і ДТП, враховуючи деякі недоліки в розглянутому додатку, як точність прогнозування (оскільки додаток не націлений на високу точність) було вирішено розробити просту у використанні інформаційну технологію, яка забезпечить підвищення точності прогнозування кількості порушень ПДР.

1.5 Формулювання вимог та постановка задачі

Задача цієї роботи полягає у розробці інформаційної технології прогнозування кількості порушень правил дорожнього руху.

Вимоги до користувача:

- можливість вибору періоду прогнозування;
- можливість вибору камери;
- можливість перегляду інформації у вигляді графіка;
- завершення сесії (вихід).

Вимоги до адміністратора:

- можливість редагування інформації;
- можливість перегляду статистики;
- можливість додавання нової інформації (камер);
- можливість видалення інформації;

Вимоги до сервера:

- зберігання даних користувача через зв’язок з базою даних;
- визначення типу користувача;
- видача потрібних даних за запитом від адміністратора.

Вхідними даними роботи є період прогнозування і камера.

Вихідними даними є прогнозована кількість порушень правил дорожнього руху на вибраний період для вибраної камери.

Отже, потрібно розробити інформаційну технологію, в якій буде реалізовано алгоритм прогнозування кількості порушень правил дорожнього руху. Завдяки розробленій інформаційній технології відбувається автоматичне опрацювання всієї доступної вхідної інформації, виводиться результат у вигляді графіку з прогнозованою кількістю порушень правил дорожнього руху на всі дні вказаного проміжку.

1.6 Висновок до розділу 1

У цьому розділі було проведено аналіз статистики порушень правил дорожнього руху, що призвело до визначення необхідності акцентувати увагу на безпеці дорожньої інфраструктури. Також проведено дослідження впливу різних факторів на кількість порушень правил дорожнього руху і виявлено основні причини дорожньо-транспортних пригод із постраждалими. Проведено аналіз сучасних аналогів, які використовуються для прогнозування ситуацій на дорогах. Надано короткий огляд основних функцій цих програм і під час вибору прототипу було вирішено обрати «PTV Visum», оскільки цей інструмент дозволяє моделювати кількість порушень та ДТП. Однак, зважаючи на недоліки, такі як точність прогнозування, виникла потреба в розробці інформаційної технології, яка покращить точність прогнозування кількості порушень правил дорожнього руху. Досліджено методи прогнозування кількості порушень і вибір було зроблено на користь моделей часових рядів. Зокрема, визначено, що модель ARIMA є оптимальною, оскільки вона дозволяє враховувати минулі рухи змінних для прогнозування майбутніх значень. Сформульовано вимоги та поставлено задачу для подальшої розробки.

2 МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ПРОГНОЗУВАННЯ КІЛЬКОСТІ ПОРУШЕНЬ ПРАВИЛ ДОРОЖНЬОГО РУХУ

2.1 Обґрунтування вибору методу для прогнозування кількості порушень правил дорожнього руху

Перш за все потрібно проаналізувати дані та врахувати всі фактори, що впливають на прогноз. В цьому буде корисним метод аналізу геопросторових даних. Використання аналізу геопросторових даних для прогнозування кількості порушень правил дорожнього руху може бути дуже корисним підходом, оскільки цей метод дозволяє враховувати просторові залежності та контекст у прогнозуванні.

Геопросторовий аналіз дозволяє враховувати взаємозв'язки між різними місцями на дорозі. Завдяки цьому можна зробити аналіз, чи є певні ділянки доріг більш схильними до порушень правил дорожнього руху, і враховувати це при прогнозуванні. Геопросторовий аналіз також дозволяє враховувати сезонні коливання у кількості порушень на певних ділянках доріг. Наприклад, визначити, що певні дороги мають більше порушень під час годин пік.

Вплив факторів середовища також враховується завдяки геопросторовому аналізу. Такими факторами можуть бути погодні умови, стан доріг, освітлення тощо. Можна дослідити, як ці фактори впливають на кількість порушень на конкретній ділянці дороги. Геопросторовий аналіз дозволяє ідентифікувати місця, де можуть бути задіяні конкретні стратегії безпеки, такі як встановлення додаткових дорожніх знаків, камер спостереження, обмеження швидкості тощо.

Врахування інших геопросторових даних, як ГІС (геоінформаційні системи), з даними про порушення дорожніх правил мають місце для кращого розуміння взаємозв'язків та прогнозування [24]. Геопросторовий аналіз дозволяє ідентифікувати «гарячі точки» − місця на дорозі, де кількість порушень правил

дорожнього руху надзвичайно висока. Це дозволяє правоохоронним органам та іншим зацікавленим сторонам вживати дієві заходи для зменшення порушень.

Прогноз на основі геопросторового аналізу може бути корисним інструментом для покращення безпеки на дорогах, спрямованим на зменшення кількості порушень правил дорожнього руху та аварій.

Загальною метою використання геопросторового аналізу є створення більш точних та інформативних прогнозів кількості порушень правил дорожнього руху, що може сприяти покращенню безпеки та оптимізації управління дорожнім рухом.

Використання методу ARIMA (Autoregressive Integrated Moving Average) для прогнозування кількості порушень правил дорожнього руху може використовуватися завдяки таким причинам:

Часовий характер даних. ARIMA − це статистичний метод, який спеціалізується на аналізі та прогнозуванні часових рядів. Кількість порушень дорожнього руху часто має часовий характер і може варіюватися впродовж часу. ARIMA добре підходить для таких ситуацій [16].

Статистична складність. ARIMA дозволяє аналізувати складні часові ряди, включаючи залежності між попередніми значеннями та поточними значеннями. Це може бути корисним при аналізі факторів, що впливають на кількість порушень дорожнього руху.

Детерміновані та стохастичні компоненти. ARIMA розбиває часовий ряд на детерміновані та стохастичні компоненти. Це дозволяє враховувати та моделювати різні фактори, які впливають на кількість порушень, включаючи сезонність та тренди.

Зрозуміла модель. ARIMA − це стандартна модель з великою кількістю ресурсів та літератури, що допомагає вивчити і розуміти. Це робить ARIMA привабливим вибором для дослідників та аналітиків, які не обов'язково є експертами у складних методах прогнозування.

Практичний досвід. ARIMA вже успішно використовувався для прогнозування в різних галузях, включаючи фінанси, економіку, медицину та

інше. Є багато прикладів використання ARIMA для аналізу та прогнозування часових рядів.

2.2 Розробка математичної моделі прогнозування кількості порушень правил дорожнього руху

Методологія ARIMA намагається описати динаміку стаціонарного часового ряду, використовуючи параметри, відомі як «авторегресивні» та «ковзне середнє». Вони позначаються AR (авторегресивні) і MA (ковзне середнє). Модель AR з одним параметром може бути виражена у вигляді рівняння (2.1).

$$X(t) = A(1)X(t-1) + E(t), \tag{2.1}$$

де $X(t)$ – досліджуваний часовий ряд,

$A(1)$ – автоматичний параметр першого порядку,

$X(t-1)$ – часовий ряд, що відставав під час першого періода,

$E(t)$ – термін помилки моделі.

Це означає, що для кожного значення $X(t)$ може бути пояснення у вигляді функції його попереднього значення $X(t-1)$ з додаванням випадкової помилки $E(t)$. Якщо значення $A(1)$ дорівнює 0,3, це вказує на те, що поточне значення ряду має зв’язок лише з 30% його попереднього значення з додаванням випадкової помилки. Це також означає, що ряд може мати зв'язок з більш ніж одним попереднім терміном. Для вираження цього використовується наступна формула (2.2).

$$X(t) = A(1)X(t-1) + A(2)X(t-2) + E(t). \tag{2.2}$$

Отже, це означає, що значення в ряду формується як комбінація двох попередніх рядів, до яких додається випадкова помилка. Такий тип моделі відомий як авторегресивна модель другого порядку.

Для моделі типу ковзного середнього використовується інша концепція. Параметри цієї моделі пов'язані з подіями в минулому періоді t із випадковими помилками, що відбулися у попередніх періодах, тобто $E(t-1)$, $E(t-2)$ застосовують замість $X(t-1), X(t-2)$.

Методика ARIMA також дозволяє конструювати моделі, які містять як авторегресію, так і ковзні середні [17]. Ці моделі часто називають «змішаними моделями». Навіть якщо це ускладнює інструмент прогнозування, структура може більш точно імітувати ряд і надавати більш точний прогноз. З чистої моделі випливає, що структура складається лише з параметрів AR або MA, а не з обох.

Моделі, розроблені за цим підходом, часто називають моделями ARIMA, так як в них міститься комбінація авторегресії (AR), інтегрування (I) – для процесу зворотного диференціювання для прогнозування та операцій ковзного середнього (MA). Модель ARIMA зазвичай називається $ARIMA$ (p, d, $ARIMA$), де p – порядок компонентів авторегресії, d – кількість операторів диференціювання, і найвищий порядок ковзної середньої. Наприклад, $ARIMA$(2,1,1) позначає авторегресивну модель другого порядку з компонентом ковзного середнього першого порядку, ряди якого були диференційовані один раз, щоб бути перетвореними в стаціонарний ряд.

Далі розглянемо безпеку руху на перехрестях, враховуючи ймовірність виникнення дорожньо-транспортних пригод на перехрестях з урахуванням додаткових параметрів [16]. Для конфліктних точок, таких як перехрестя, які представляють певні ризики для руху при інтенсивності потоку, обчислення проводять за відповідною формулою:

$$q_i = K_i M_i N_i \cdot 25 \cdot 10^{-7} / K_p, \tag{2.3}$$

де K_i – відносна небезпека конфліктної точки;

M_i, N_i – інтенсивність потоків, що рухаються в протилежних напрямках, автомобіль за годину;

K_p – коефіцієнт змін за рік.

Число порушень, які спричиняють дорожньо-транспортну пригоду визначають за формулою:

$$q_n = K_n\left(\sum_{i=1}^{n} N_i + \sum_{j=1}^{k} M_j\right) \cdot 10^{-2}, \quad (2.4)$$

де K_n – небезпека ДТП;

$\sum_{i=1}^{n} N_i$, $\sum_{j=1}^{k} M_j$ – сумарна інтенсивність руху на перехресті.

Ймовірність виникнення аварії на регульованих перехрестях визначають за формулою

$$G_p = -0{,}468 + q_n + \sum_{i=1}^{n} q_i, \quad (2.5)$$

де G_p – ймовірність аварійності на регульованому перехресті, ДТП/год.;

n – кількість точок конфлікту.

В той час як на нерегульованому перетині доріг використовується формула, отримана в результаті регресійного аналізу:

$$G_n = 0.0025 + 0.92 \cdot 10^{-3} \sum_{i=1}^{n} \left(I_n^{\frac{1}{4}} \cdot I_T\right), \quad (2.6)$$

де G_n – число аварій за участі пішоходів на рік;

I_n – інтенсивність руху пішоходів;

I_T – інтенсивність руху транспорту;

n – кількість переходів на перехресті.

Для визначення загальної кількості аварійних ситуацій на рік, обчислені значення G_p та G_n додаються.

Обрахувати рівень безпеки дорожнього руху на перехрестях можна з використанням формули коефіцієнта аварійності

$$K_a = \frac{G \cdot K_r \cdot 10^7}{25\left(\sum_{i=1}^{n} N_i + \sum_{j=1}^{k} M_j\right)}. \quad (2.7)$$

Якщо коефіцієнт аварійності менше 3, то таке перехрестя вважається безпечним; від 3 до 8 – не дуже безпечне; коефіцієнт в межах від 8 до 12 – небезпечне; коли коефіцієнт дорівнює 12 – дуже небезпечне.

Знайдені параметри слід враховувати при побудові моделі прогнозування кількості порушень правил дорожнього руху для локацій, що знаходяться на перехрестях.

Отже, запропонована математична модель, що враховує ймовірність аварійності на перехрестях дає змогу знайти ефективне рішення завдання, що забезпечить збільшення точності прогнозування.

2.3 Вибір бази даних для інформаційної технології прогнозування кількості порушень правил дорожнього руху

Вибір бази даних є критично важливим етапом при розробці інформаційної технології прогнозування кількості порушень правил дорожнього руху. Правильно обрана база даних допоможе зберігати, організовувати та ефективно отримувати доступ до інформації, що є основою для аналізу та прогнозування порушень.

Для вибору бази даних потрібно переконатися, що обрана база даних може вміщати та обробляти очікуваний обсяг даних, які включають в себе історичні дані про порушення правил дорожнього руху, дані про дорожні умови та інші відомості. Важливо, щоб база даних була досить швидкою для виконання операцій читання і запису, особливо якщо потрібна обробка даних в реальному часі або близька до реального часу обробка даних. Для ефективного

прогнозування порушень потрібна можливість виконувати складні запити та аналізувати дані. Обрана база даних має підтримувати ці функції. Має бути можливість легко розширювати базу даних при збільшенні обсягу даних або обчислювальних ресурсів. Забезпечення конфіденційності та цілісності даних є критичним аспектом. Важливо вибрати базу даних, яка підтримує засоби захисту даних.

Отже, потрібно використовувати сучасну базу даних для забезпечення надійного, швидкого та зручного введення, видалення, управління та маніпулювання інформацією. Використання такої бази даних та її системи управління значно скоротить час виконання відповідних операцій, які раніше виконувались вручну.

База даних − це організована колекція даних, яка зберігається та оброблюється з метою ефективного доступу, управління та аналізу інформації. Бази даних використовуються для зберігання і структурування великих обсягів даних, таких як текст, числа, зображення, аудіо та інші форми інформації [25].

ER-діаграма для бази даних зображена на рисунку 2.1.

Рисунок 2.1 – ER-діаграма порушень правил дорожнього руху

Для опису предметної галузі «Порушення правил дорожнього руху» необхідні такі сутності: дорожній випадок, учасник, порушення, камера, прогноз. В універсальне відношення потрібно включити атрибути, що описують ці сутності. Отже, універсальне відношення для цієї бази даних буде мати такий вигляд: R (номер випадка, локація, дата, час, тяжкість, погодні умови, інтенсивність руху пішоходів, інтенсивність руху транспорту, ПІБ учасника, вік, стать, тип порушення, обставини, вулиця, наявність перехрестя, число переходів, локація прогнозу, дата прогнозу, кількість порушень).

Ступінь універсального відношення – 19.

Вибір реляційної бази даних та мови маніпулювання даними SQL для інформаційної технології прогнозування кількості порушень правил дорожнього руху є досить доцільним і обґрунтованим.

Реляційні бази даних і мова SQL добре підходять для зберігання та обробки структурованих даних, які характерні для інформації про порушення правил дорожнього руху. Завдяки простоті використання можна створити таблиці для різних типів даних, таких як інформація про авто, дорожні умови, час та інше.

SQL надає потужний інструмент для виконання аналізу даних. Це дає змогу легко створювати складні запити, які допоможуть вибирати, агрегувати та аналізувати дані для прогнозування порушень правил дорожнього руху.

Реляційні бази даних підтримують цілісність даних за допомогою обмежень, ключів і зовнішніх ключів. Це важливо для того, щоб база даних була цілісною і надійною.

Реляційна база даних має підтримку транзакції, що забезпечить атомарність операцій та запобіжність невідновлюваним збоям даних.

SQL і реляційні бази даних мають величезну кількість ресурсів та документацію, що допоможе розробляти та налагоджувати інформаційну систему.

При потребі роботи з великим обсягом даних, розробник зможе знайти в реляційних базах даних зручні інструменти для адміністрування та масштабування бази даних.

Зважаючи на ці переваги, вибір реляційної бази даних та SQL для інформаційної технології прогнозування порушень правил дорожнього руху може значно полегшити розробку та аналіз даних і підтримувати надійність та інтеграцію розроблюваного проекту.

Для взаємодії з реляційною базою даних була обрана мова перетворення даних SQL. Це обумовлено кількома перевагами цієї мови, такими як: незалежність від конкретної бази даних; стандартизованість, легкість доступу.

SQL відповідає принципам ACID (атомарність, узгодженість, ізольованість, довговічність), що забезпечує цілісність бази даних. Це означає, що транзакції чітко визначають, як вони взаємодіють з базою даних, щоб уникнути конфліктів та зберегти консистентність. Однією з важливих переваг є можливість користувача отримувати доступ до даних у базі, не маючи повної інформації про структуру таблиці. SQL надає абстракцію для роботи з даними, що полегшує їхнє використання. Крім того, кожен рядок в таблиці містить унікальний набір даних для визначених стовпців, що робить SQL ефективним і зручним інструментом для роботи з категоріями даних в базі [26].

SQL − це мова для роботи з реляційними базами даних, і вона володіє величезним набором функцій і операцій для вибору, фільтрації, групування та аналізу даних. Ви зможете легко виконувати складні запити для вибору необхідної інформації та створення аналітичних операцій. Мова SQL дозволяє легко агрегувати дані, такі як знаходження середнього значення, суми, кількості тощо. Це дуже корисно для аналізу та прогнозування порушень правил дорожнього руху. SQL є стандартизованою мовою для роботи з реляційними базами даних, що дозволяє розробникам легко розуміти та підтримувати код. Це особливо важливо, коли ви працюєте в команді або маєте передбачену підтримку системи після завершення дипломної роботи.

Недоліки: труднощі з відхиленням від стандартів, труднощі в роботі з ієрархічними структурами.

Отже, використання мови маніпулювання даними SQL є дуже доцільним для інформаційної технології прогнозування кількості порушень правил

дорожнього руху. SQL надає потужний та стандартизований засіб для роботи з даними, що значно спрощує аналіз та взаємодію з реляційними базами даних.

2.4 Розробка UML-діаграми класів інформаційної технології прогнозування кількості порушень правил дорожнього руху

Моделювання структури системи за допомогою UML-діаграми класів є ефективним інструментом для візуалізації компонентів системи, їх властивостей та взаємодій. Діаграми класів є фундаментальними для процесу моделювання об’єктів і моделюють статичну структуру системи. UML-діаграми класів − це графічні позначення, які використовуються для візуалізації та побудови об’єктно-орієнтованих систем [27]. Точніше кажучи, діаграма класів в уніфікованій мові моделювання (UML) − це статична структурна діаграма, яка описує структуру системи, представляючи класи системи, їхні атрибути, операції та зв’язки між об’єктами.

Діаграми класів корисні на багатьох етапах проектування системи. На етапі аналізу діаграма класів може допомогти зрозуміти вимоги проблемної галузі та визначити її компоненти. У проекті об’єктно-орієнтованого програмного забезпечення діаграми класів, які створюються на ранніх стадіях проекту, містять класи, які часто перетворюються на фактичні класи та об’єкти програмного забезпечення під час написання коду.

Діаграма класів надає зручний спосіб візуалізації класів системи та їх зв’язків, а також дає змогу розглядати систему на високому рівні, зосереджуючись на класах та їх взаємодії.

На етапах аналізу та проектування циклу розробки, створення діаграми класів має на меті виконання таких функцій:

- визначити структуру класів та інших класифікаторів;
- визначити зв’язки між класами та класифікаторами;
- проілюструвати структуру моделі за допомогою атрибутів, операцій і сигналів;

- продемонструвати загальні ролі та обов’язки класифікатора, які визначають поведінку системи;
- показати класи реалізації в пакеті;
- відобразити структуру та поведінку одного чи кількох класів;
- продемонструвати ієрархію успадкування серед класів і класифікаторів.

На етапі впровадження циклу розробки програмного забезпечення використання діаграми класів відбувається для перетворення моделей у код і для перетворення коду в моделі.

На рисунку 2.2 зображено UML-діаграму класів інформаційної технології прогнозування кількості порушень правил дорожнього руху.

Вона включає такі розроблені класи Main, DBManager, GraphicsPanel, ResultCache, Main, ManuPage, DataProcessing, Line, DAO, Point, EstService, PrListener, GraphForFirst.

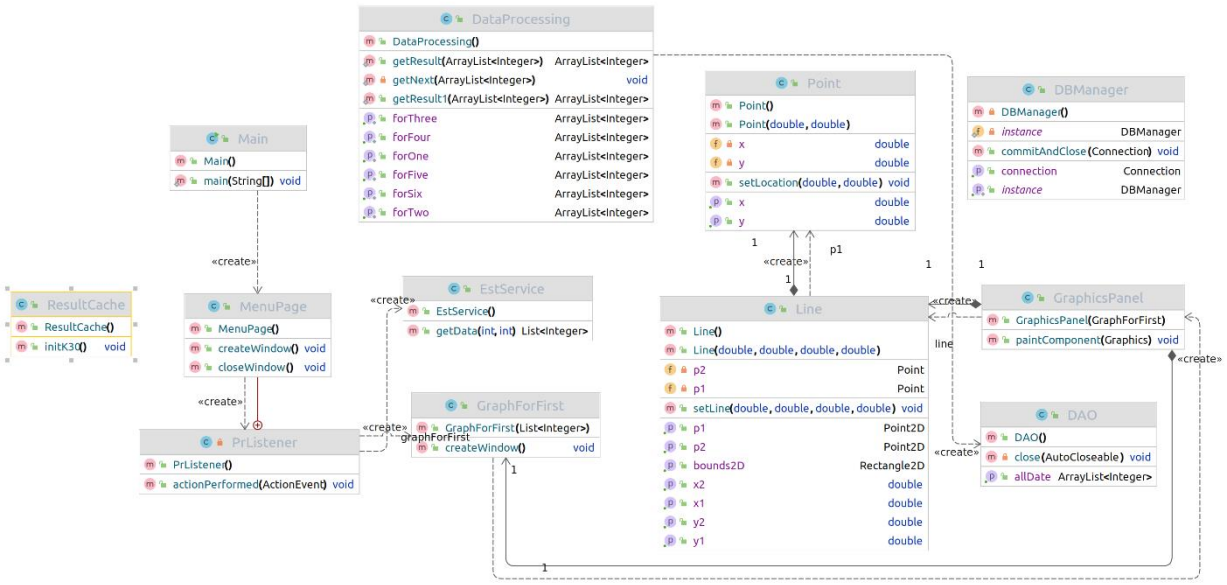

Рисунок 2.2 – UML-діаграма класів інформаційної технології прогнозування кількості порушень правил дорожнього руху

Створення основних вікон програм відбувається в класах EstService, PrListener. Ці класи доцільно розробити з використанням інструменту для

створення графічного інтерфейсу користувача Swing. Цю ж бібліотеку використовуємо при розробці класів Line, Point та GraphicsPanel, створених для забезпечення побудови графіка, що міститиме результати прогнозування кількості порушень правил дорожнього руху.

В класі DataProcessing міститиметься основна логіка програми, де буде використана модель ARIMA, доповнена алгоритмом на основі аналізу геопросторових даних для прогнозування кількості порушень правил дорожнього руху.

Клас DBManager реалізує підключення до бази даних, а клас DAO створений для підключення до бази даних та взаємодії з нею. Цей клас містить запити мовою SQL для отримання інформації, що міститься в базі даних.

Отже, в цьому підрозділі було здійснено моделювання UML-діаграми класів, проведено огляд розроблених класів та інструментів для їх реалізації.

2.5 Висновок до розділу 2

У другому розділі було обґрунтовано вибір методу прогнозування для розробки інформаційної технології прогнозування кількості порушень правил дорожнього руху. Метод ARIMA визнаний як один із найбільш ефективних для вирішення цієї задачі, і його доцільно поєднувати з алгоритмом, заснованим на аналізі геопросторових даних. За використанням цих методів створено математичну модель, спрямовану на підвищення точності прогнозування кількості порушень правил дорожнього руху. В розділі також аргументовано вибір реляційної бази даних та мови SQL для маніпулювання даними, а також викладено ER-діаграму порушень правил дорожнього руху та UML-діаграму класів інформаційної технології прогнозування кількості порушень правил дорожнього руху.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ПРОГНОЗУВАННЯ КІЛЬКОСТІ ПОРУШЕНЬ ПРАВИЛ ДОРОЖНЬОГО РУХУ

3.1 Проектування структури інформаційної технології прогнозування кількості порушень правил дорожнього руху

Проектування структури інформаційної технології має на меті вирішенням конкретної задачі або проблеми. Однією з ключових цілей проектування інформаційної технології є підвищення продуктивності та ефективності операцій, процесів та завдань. Технологія допомагає збирати, зберігати та забезпечувати доступ до інформації. Інформаційні технології можуть допомагати покращити якість продукту або послуги, що надається. Впровадження технологій може призвести до зменшення витрат на операційні процеси, обслуговування та інфраструктуру.

Загалом, проектування структури інформаційної технології має на меті досягнення ряду конкретних цілей, включаючи покращення продуктивності, забезпечення доступу до інформації, підвищення якості продукту або послуги, підвищення інноваційності та підвищення конкурентоспроможності.

Робота з методами прогнозування кількості порушень правил дорожнього руху вимагає великої обчислювальної потужності. Вхід інформаційної технології містить період, на який потрібно зробити прогноз та камеру, на локації якої можуть бути здійснені прогнозовані порушення. Після отримання даних проводиться перевірка з даними, що зберігаються в базі даних та формується прогнозована кількість порушень. Тобто, на виході ми отримуємо результати прогнозування, які задовільняють потреби користувача. Структура інформаційної технології прогнозування кількості порушень правил дорожнього руху наведена на рисунку 3.1.

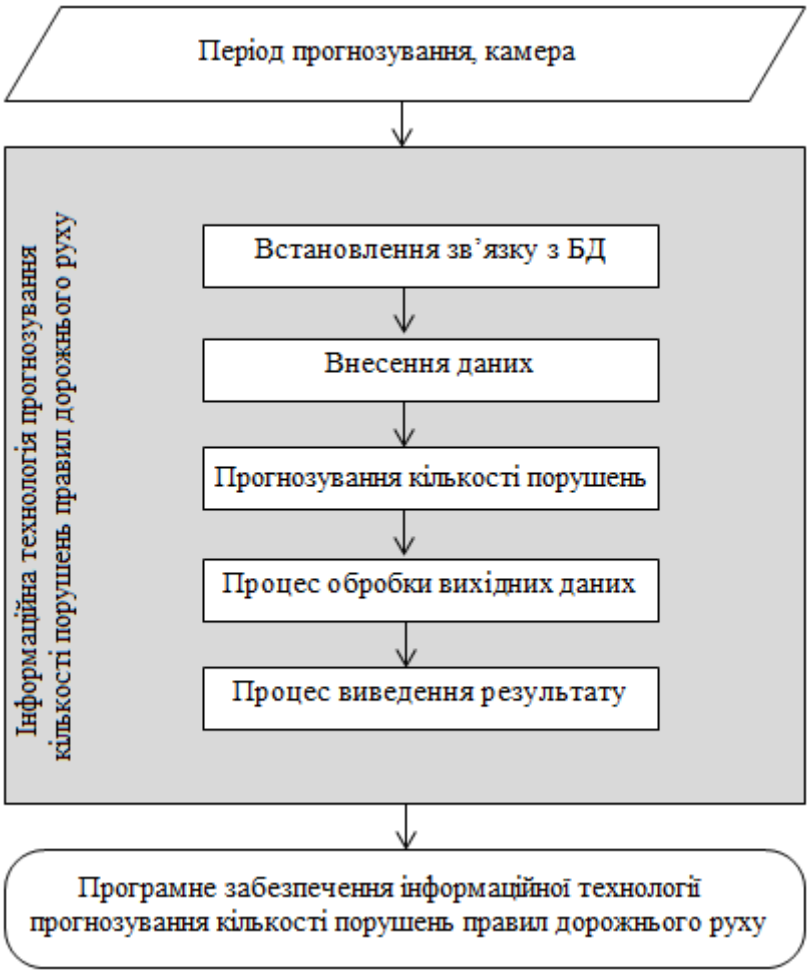

Рисунок 3.1 – Структура інформаційної технології прогнозування кількості порушень правил дорожнього руху

Модульний підхід є ефективним підходом до проектування структури інформаційної технології прогнозування кількості порушень правил дорожнього руху, оскільки він полегшує організацію проекту на менші самостійні модулі або компоненти [28]. Кожен модуль відповідає за певну функцію або завдання, що спрощує розробку, тестування та обслуговування інформаційних технологій. Це полегшує розробку, підтримку та розширення системи, оскільки кожен модуль можна досліджувати окремо та оптимізувати незалежно.

В програмі будуть представлені модуль збору даних, модуль обробки даних, модуль аналізу та прогнозування, модуль візуалізації та модуль інтеграції. Структурна схема прогнозування кількості порушень правил дорожнього руху наведена на рисунку 3.2.

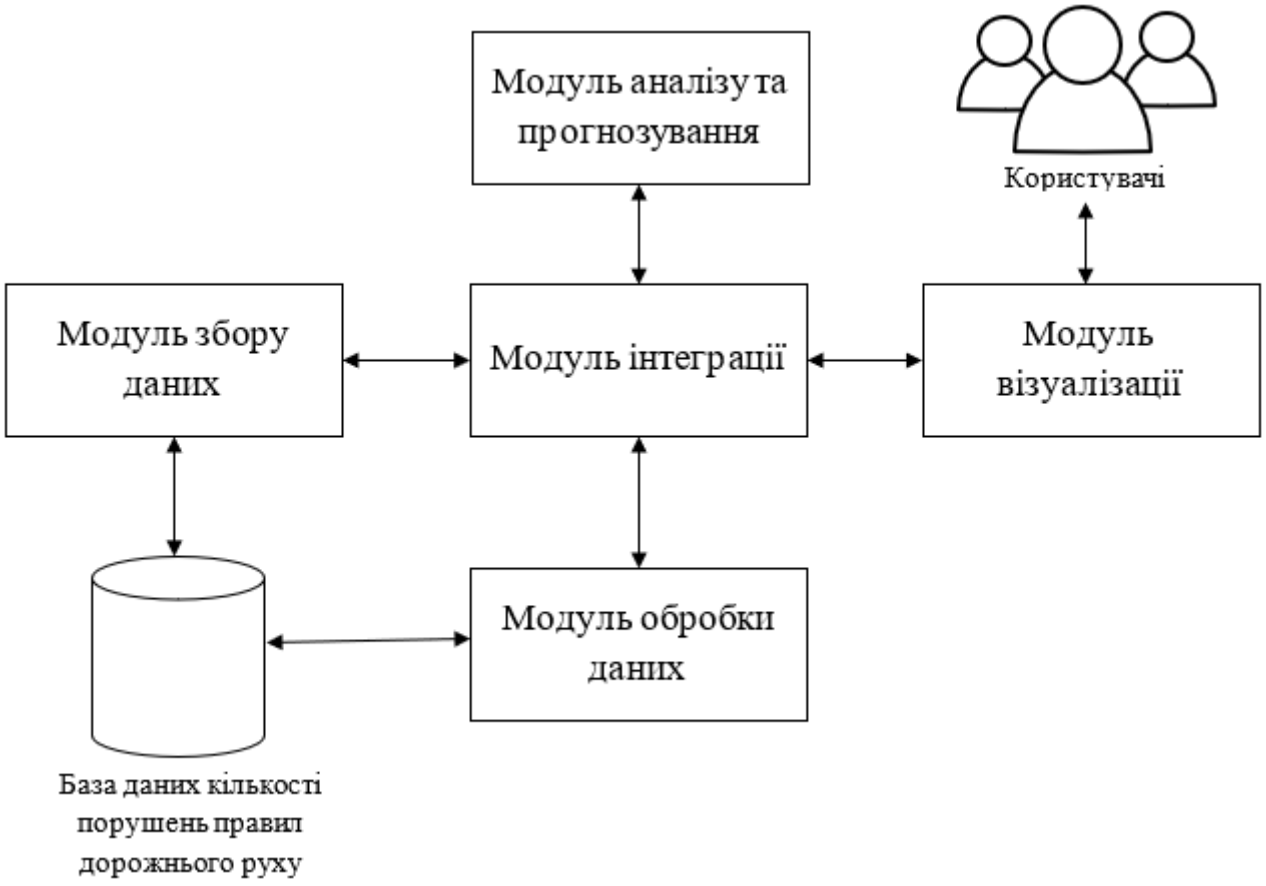

Рисунок 3.2 – Загальна структурна схема

Спочатку розглянемо модуль збору даних. Цей модуль може включати отримання інформації про дорожній рух, такої як швидкість руху, погодні умови, геопросторові дані та інші важливі параметри.

Модуль для обробки даних може включати очищення, агрегування та попередню обробку даних для підготовки їх до аналізу та моделювання. Ці два модулі призначені для взаємодії з базою даних.

База даних має містити інформацію про кількість порушень правил дорожнього руху, зафіксованих камерами фіксації, бути синхронізованою з інформаційною технологією прогнозування кількості порушень правил дорожнього руху та забезпечувати швидкий імпорт та автоматичний експорт усієї інформації, необхідної для прогнозування.

Модуль аналізу та прогнозування містить інтелектуальний алгоритм прогнозування кількості порушень правил дорожнього руху. Прогнозування відбується при застосуванні методу ARIMA.

Модуль для візуалізації результатів і створення звітів дозволить користувачам системи легко розуміти прогнози та аналізи. Завдяки цьому модулю програма може побудувати графік, що відображає прогнозовану кількість порушень правил дорожнього руху. Користувач взаємодіє з інформаційною технологією безпосередньо через модуль візуалізації.

Модуль інтеграції передбачає інтеграцію всіх попередніх модулів в єдину систему.

Модульний підхід полегшує роботу над проектом і дозволяє розробникам і аналітикам працювати над окремими частинами системи паралельно. Це також полегшує керування та обслуговування системи в майбутньому.

Спроектована інформаційна технологія сприяє розвитку дорожньої безпеки, поліпшує моніторинг дорожнього руху та допомагає органам влади та правоохоронним службам вчасно реагувати на можливі ризики. Крім того, вона може бути корисною для розробки стратегій та дій, спрямованих на зменшення кількості порушень правил дорожнього руху та аварій.

3.2 Розробка схеми алгоритму роботи інформаційної технології прогнозування кількості порушень правил дорожнього руху

Графічне представлення алгоритму функціонування інформаційної технології прогнозування кількості порушень правил дорожнього руху представлене на рисунку 3.3.

Схема алгоритму роботи інформаційної технології стане ключовим інструментом для розуміння процесу прогнозування порушень правил дорожнього руху. Вона розкриє послідовність дій, які використовуються для аналізу даних, навчання моделей і отримання прогнозів. Ця схема є необхідною для подальшого успішного впровадження і використання технології в реальних умовах. Завдяки розробленому алгоритму технологія створює потужний інструмент для прогнозування та зменшення кількості порушень, зменшення ризику аварій та підвищення безпеки дорожнього руху в цілому.

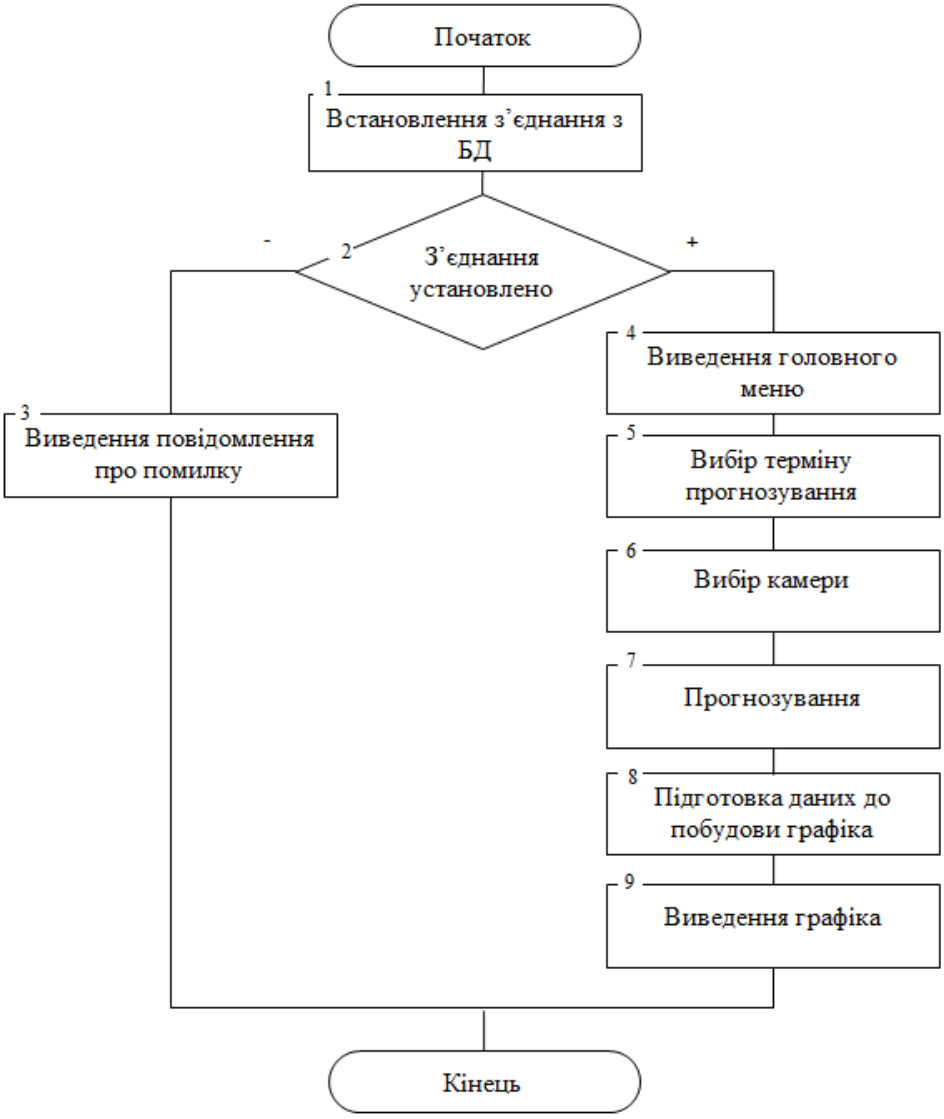

Рисунок 3.3 – Схема алгоритму функціонування інформаційної технології прогнозування кількості порушень правил дорожнього руху

Опишемо наведену схему алгоритму функціонування інформаційної технології прогнозування кількості порушень правил дорожнього руху, який подано на рисунку 3.3:

1. Спочатку встановлюється зв'язок з базою даних.
2. Після цього відбувається перевірка встановленного з'єднання. Якщо з'єднання встановлення – перехід до пункту 4.
3. Якщо на попередньому кроці було визначено, що з'єднання з базою даних відсутнє – виводиться повідомлення про помилку встановлення зв'язку. Після цього відбувається перехід до кроку 10.

4. Виведення за допомогою графічного інтерфейсу головного меню програми.
5. Користувачеві пропонується вибрати термін прогнозування з випадаючого списку.
6. Користувачеві пропонується вибрати камеру фіксацій порушень правил дорожнього руху.
7. На цьому етапі відбувається прогнозування кількості порушень правил дорожнього руху відповідно до введених параметрів.
8. Підготовка даних до побудови та відображення графіка.
9. Відображення графіка з прогнозованою кількістю порушень правил дорожнього руху на період, вказаний користувачем та за вибраною ним локацією.
10. Закінчення діалогу користувача з системою.

3.3 Обґрунтування вибору мови та середовища програмування

Для розробки інформаційної технології прогнозування кількості порушень правил дорожнього руху слід обрати мову програмування, що базується на об'єктно-орієнтованому підході. Цей вибір обумовлений необхідністю використання мови, яка надає значні переваги для досягнення максимальної ефективності вирішення даної завдання. Об'єктно-орієнтовані мови надають зручний спосіб для створення модульних та структурованих програм. Тому у проекті, де потрібно прогнозувати кількість порушень правил дорожнього руху, це дозволить легше організувати та керувати кодом. Об'єктно-орієнтована мова дозволяє представити дані та функції як об'єкти та класи. Це забезпечує можливість моделювати різні аспекти порушень правил дорожнього руху як об'єкти, що спрощує аналіз та обробку даних. Об'єктно-орієнтовані мови дозволяють абстрагувати дані та функції, що полегшує роботу з складними системами та підтримує роз'ємність між компонентами програми. Вибір об'єктно-орієнтованої мови сприяє покращенню структури коду та підтримує розробку великих проектів з численними модулями та функціями.

Для вибору мови програмування проведемо аналіз найчастіше використовуваних для реалізації подібних завдань мов: C # та Java.

Обидві мови мають велику спільноту розробників та багато ресурсів для навчання, підтримують об'єктно-орієнтовану парадигму, що допомагає структурувати код.

Java і C# мають як сильні, так і слабкі посилання на об’єкти. Обидві мови підтримують методи завершення. Через невизначеність того, коли видаляти об’єкт, фіналізатори не можна використовувати для звільнення системних ресурсів, зайнятих об’єктом, що вимагає створення додаткових методів для «очищення» об’єкта та явного виклику.

Спочатку розглянемо переваги C #.

C# – це об'єктно-орієнтована мова програмування, розроблена компанією Microsoft. Вона була вперше представлена у 2000 році та стала однією з основних мов розробки для платформи Microsoft .NET. C# набула широкого застосування для створення різноманітних програм та додатків. C# використовується для створення Windows-додатків, таких як текстові редактори, графічні програми, обробники баз даних та інші. Її можна використовувати для розробки веб-додатків з використанням ASP.NET, ASP.NET Core та інших технологій, які базуються на цій мові. За допомогою платформи Xamarin, яка базується на C#, розробники можуть створювати крос-платформені мобільні додатки для Android та iOS. C# використовується для створення відеоігор з використанням платформи Unity, що є однією з найпопулярніших у галузі геймдеву. Мову C# можна використовувати для розробки програм, які взаємодіють з реляційними базами даних, такими як Microsoft SQL Server, або іншими системами управління базами даних. C# може бути використана для створення програмного забезпечення для вбудованих систем, таких як системи керування промисловими процесами, медичні пристрої, автомобільні системи і багато інших. Також C# може бути використана для розробки хмарних додатків і послуг, що використовують різні хмарні платформи, такі як Microsoft Azure.

Мова програмування C# зазвичай використовується в поєднанні з різноманітними структурами, які впливають на її поведінку. Це дозволяє використовувати C# в різних галузях [29]. Ось деякі переваги C#:

- Мова знаходиться серед десяти найбільш затребуваних на ринку.
- C# підтримує об'єктно-орієнтований підхід, що сприяє структуруванню коду та полегшує розробку та обслуговування додатків.
- Використання автоматичного збирача сміття допомагає управляти пам'яттю та уникнути витоків пам'яті, що є проблемою в інших мовах.
- З появою .NET 5 і подальших версій C#, можна використовувати для розробки додатків на різних операційних системах, включаючи Windows, Linux та macOS.
- Мова включає механізми типізації та обробки виключень, що сприяють покращенню безпеки додатків.
- Багато синтаксичних доповнень та готових конструкцій.
- Спільнота розробників C# є великою та активною, а Microsoft регулярно надає підтримку та оновлення мови та інструментів.
- За допомогою платформи Unity, яка використовує C#, можна розробляти відеоігри для різних платформ.
- Мова має зрозумілий синтаксис, що робить її привабливим вибором для початківців у програмуванні.
- Активний розвиток мови C# призводить до швидкого впровадження нових покращень, що перевищує темпи інших мов програмування.

Далі розглянемо мову програмування Java.

Java – одна з найпопулярніших мов програмування, займає вражаючу кількість приблизно 9 мільйонів користувачів, і програми, написані на Java, функціонують на семи мільярдах різноманітних пристроїв [30].

Java – це високорівнева, об'єктно-орієнтована мова програмування, створена компанією Sun Microsystems (пізніше придбаною Oracle Corporation). Вона відома своєю платформонезалежністю, надійністю та широким спектром застосувань. Найчастіше використовується для написання мобільних додатків,

ігор, застосовується в сфері наукової розробки, мобільних телефонів, ігрових консолей та суперкомп’ютерів.

Java є основою майже всіх типів мережевих додатків. У багатьох дослідженнях серед розробників її називають мовою програмування номер 1.

Основні характеристики Java:

Кросплатформенність. Програми Java компілюються в байт-код, який виконується на Java Virtual Machine (JVM). Це робить код Java універсальним і працездатним на будь-якій платформі, де існує відповідна реалізація JVM.

Спільнота. Java – мова, яку визнають як номер один у світі програмування. Велика та активна спільнота розробників надає безліч ресурсів, включаючи документацію, форуми та бібліотеки.

Надійність. Мова має вбудовані механізми безпеки, включаючи контроль доступу та цифрові підписи. Це робить її популярною для розробки безпечних додатків.

Об'єктна орієнтація. Java базується на об'єктно-орієнтованому підході, що полегшує створення модульного та розширюваного коду.

Багатопоточність. Мова підтримує багатопоточність, що дозволяє створювати програми з паралельними потоками для оптимізації продуктивності.

Відносна простота. ava є мовою, яка, можливо, не така проста для освоєння, як Python, але значно легше, ніж C або C++. Можливість легко оновлювати до нових версій полегшує процес навчання.

Гнучкість. Java застосовується в різних галузях, включаючи веб-розробку (Java EE), мобільну розробку (Android), вбудовані системи, робототехніку, ігри та інші області. Це робить її привабливою для початківців.

Розширені інструменти розробки. Для розробки на Java доступні різні інтегровані середовища розробки (IDE), такі як Eclipse, IntelliJ IDEA та NetBeans, які спрощують процес програмування.

У результаті, в даний момент Java визнається більш популярною, порівняно з C#, і з огляду на тенденції розвитку, ця ситуація не буде змінюватися у найближчому майбутньому.

Синтаксис C# виявляється більш конкретним, адже він використовує довгі слова замість довгих фраз. Особливою перевагою є можливість розділення класів на частини та розміщення їх в окремих файлах за допомогою «часткового» оператора. З точки зору швидкості розробки програмного забезпечення, C# дозволяє швидше створювати програми з графічним інтерфейсом порівняно з Java.

Якщо програма повинна бути кросплатформенною, вибір між C# і Java стає очевидним – у цьому випадку вигідніше використовувати Java. Зазначимо, що для забезпечення стабільності, кросплатформенної сумісності та оптимізації швидкодії програми, розробка на Java виявляється більш перспективною. Однак, якщо вам необхідно створити програму, яка працюватиме тільки на платформі Windows, використання C# може бути більш вигідним варіантом.

Отже, вибір між C# та Java залежить від таких ключових факторів:

- платформа, на якій буде працювати технологія (Windows, Android, інше);
- досвід і знання в області однієї з мов;
- інструменти розробки та бібліотеки, що найкраще підходять до конкретного проекту;
- вимоги до продуктивності та масштабованості.

Оцінюючи всі позитивні сторони Java, а також її вільну доступність та наявність обширного спектру безкоштовних бібліотек, вирішено використовувати цю мову для розробки програмного забезпечення. Для створення інформаційної технології на Java обрано середовище розробки IntelliJ IDEA 2021. Розглянемо особливості та переваги цього інструменту.

IntelliJ IDEA – це інтегроване середовище розробки (IDE) для мови програмування Java та інших пов'язаних технологій [31]. Розроблена компанією JetBrains, ця IDE є однією з найпопулярніших середовищ розробки для роботи з Java-програмами та іншими мовами на базі JVM (Java Virtual Machine). Воно відоме своєю потужністю та набором інструментів, які полегшують розробку програм на Java та інших мовах. Вона широко використовується розробниками для створення високоякісних програм та додатків. IntelliJ IDEA може бути

корисним для розробки інформаційної технології прогнозування кількості порушень правил дорожнього руху з наступних причин:

Підтримка Java. Як вже було вказано, Java була вибрана як мова програмування для проекту. IntelliJ IDEA є однією з найкращих IDE для розробки на Java, оскільки вона надає потужні інструменти та середовище для розробки на цій мові.

Аналіз коду. IntelliJ IDEA має вбудовані інструменти для аналізу та оптимізації коду. Вони можуть допомогти виявити можливі проблеми та виправити їх, що важливо для забезпечення якості та надійності інформаційної технології. Також IntelliJ IDEA надає зручне автодоповнення коду, що робить процес програмування швидшим і менш помилковим. Вона також надає підсвічування синтаксису, що полегшує виявлення помилок у коді.

Підтримка інших мов та технологій. Крім Java, IntelliJ IDEA підтримує інші мови та технології, такі як SQL, HTML, JavaScript тощо. Це може бути корисно, якщо ваш проект включає в себе компоненти, написані на різних мовах та робить її відмінним інструментом для розробки різноманітних додатків.

Підтримка інструментів для аналізу даних. При використанні машинного навчання або інших методів аналізу даних для прогнозування порушень правил дорожнього руху, IntelliJ IDEA може бути використана для розробки та відлагодження таких інструментів.

Підтримка роботи з базами даних. Інформаційні технології для прогнозування можуть вимагати роботи з базами даних. IntelliJ IDEA має плагіни та підтримку для роботи з різними Системами керування базами даних (СКБД).

Підтримка Git та інших систем контролю версій. IntelliJ IDEA вбудовується з системами контролю версій, такими як Git. Це полегшує спільну роботу над проектом та відслідковування змін в коді.

IntelliJ IDEA може бути потужним інструментом для розробки інформаційної технології прогнозування порушень правил дорожнього руху, оскільки воно надає широкий спектр функцій та інструментів, які допоможуть впоратися з різними аспектами проекту.

3.4 Опис програмних рішень

В ході розробки програмного застосунку було використано стандартні бібліотеки Java Standard Library (Java SE), Java Collections Framework, JavaFX, Java Persistence API (JPA).

Java Standard Library (Java SE): це основна бібліотека, яка входить до складу Java Development Kit (JDK). Вона містить класи і методи для роботи з основними завданнями, такими як робота з рядками, масивами, введенням/виведенням, мережевими операціями, роботою з файлами, і багато інших [32].

Java Collections Framework: ця бібліотека надає інтерфейси і реалізації для роботи з різними структурами даних, такими як списки (List), множини (Set), мапи (Map), черги (Queue) і інші. Вона дозволяє ефективно зберігати і обробляти дані [33].

JavaFX: ця бібліотека використовується для розробки графічних інтерфейсів користувача (GUI) для десктопних застосунків. Вона надає інструменти для створення вікон, кнопок, меню та інших елементів інтерфейсу [34].

Java Persistence API (JPA): для роботи з базами даних Java надає JPA, який дозволяє легко взаємодіяти з реляційними базами даних. Для збереження та управління даними про порушення правил дорожнього руху знадобиться база даних. Java має підтримку різних DBMS через JDBC (Java Database Connectivity) [35].

Розглянемо більш детально інструмент для роботи з базами даних JDBC.

JDBC (Java Database Connectivity) − це Java-технологія, яка дозволяє взаємодіяти з базами даних, використовуючи мову програмування Java. Ця технологія є дуже корисною для реалізації проектів, що передбачають роботу з базами даних.

Наведемо основні можливості технології;

Підключення до бази даних. JDBC дозволяє підключитися до різних систем керування базами даних (СКБД), таких як MySQL, PostgreSQL, Oracle, і багатьох

інших. Для цього ви використовуєте JDBC-драйвер, який підтримується конкретною базою даних.

Створення SQL-запитів. Ви можете використовувати JDBC для виконання SQL-запитів до бази даних. Це включає в себе запити на вибірку, оновлення, вставку і видалення даних.

Обробка результатів запитів. JDBC дозволяє вам отримувати результати SQL-запитів у вашому Java-коді. Ви можете обробляти ці результати, виводити їх на екран, або використовувати для подальшої обробки в програмі.

Захист від SQL-ін'єкцій. При роботі з базами даних, важливо дотримуватися найкращих практик щодо безпеки, включаючи захист від SQL-ін'єкцій. JDBC має засоби для безпечного виконання SQL-запитів.

Транзакції. Ви можете використовувати JDBC для керування транзакціями в базі даних. Це важливо, якщо вам потрібно гарантувати цілісність даних під час виконання декількох операцій в базі.

Підготовлені заявки. JDBC підтримує використання підготовлених заявок, що може полегшити виконання одного і того ж SQL-запиту з декількох місць у вашому коді.

Закриття ресурсів. Важливо закривати з'єднання, заявки і результати запитів, коли вони більше не потрібні, щоб уникнути витоку ресурсів.

Обробка помилок. Важливо налагодити обробку помилок під час роботи з JDBC, оскільки помилки можуть виникнути через різні фактори, такі як втрата з'єднання з базою даних або неправильний SQL-запит.

Загальний підхід до використання JDBC включає такі етапи:

1. Підключення до бази даних за допомогою JDBC-драйвера.
2. Виконання SQL-запитів та обробка результатів.
3. Керування транзакціями, якщо це необхідно.
4. Закриття ресурсів після завершення операцій з базою даних.

Для того, щоб описати реалізацію встановлення з’єднання з базою даних, наведемо діаграму діяльності системи при встановленні з’єднання, зображену на рисунку 3.4.

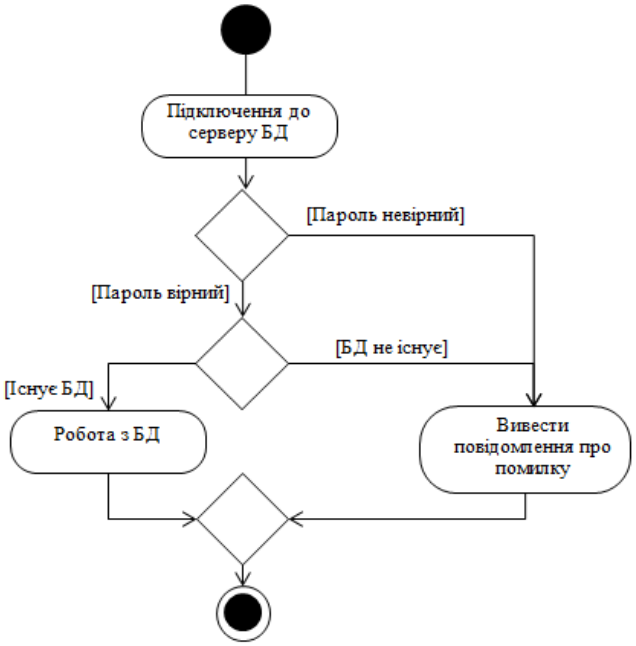

Рисунок 3.4 – Діаграма діяльності системи при встановленні з’єднання з базою даних

Зображений вище алгоритм (рис. 3.4) можна записати у вигляді коду:

```
package db;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
public class DBManager {

   private static DBManager instance;
   private static final String URL = "jdbc:mysql://localhost:3306/pdr";
   private static final String USERNAME = "root";
   private static final String PASSWORD = "root";

   public static synchronized DBManager getInstance() {
      if (instance == null)
         instance = new DBManager();
      return instance;

   }
   private DBManager() {
   }
  public Connection getConnection() throws SQLException {
      Connection con = null;
```

```
        try {
            con = DriverManager.getConnection(URL, USERNAME, PASSWORD);
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return con;
    }
    public void commitAndClose(Connection con) {
        try {
            if (con != null) {
                con.commit();
                con.close();
            }
        } catch (SQLException throwables) {
            throwables.printStackTrace();
        }
    }
}
```

Діаграма розпочинається з початкової точки, яка позначає початок встановлення з'єднання з базою даних.

Спершу, система запускає сервер бази даних (наприклад, сервер MySQL або PostgreSQL), який готовий обробляти запити на з'єднання. Система починає процес встановлення з'єднання з базою даних, використовуючи відповідний JDBC-драйвер для конкретної СКБД.

Потім система передає ідентифікаційні дані для аутентифікації на сервері бази даних. Це включає в себе ім'я користувача та пароль для доступу до бази даних. Після аутентифікації, система відправляє запит на з'єднання до сервера бази даних, вказуючи інші параметри підключення, такі як URL бази даних. Якщо з'єднання встановлено успішно, система переходить до виконання інших операцій з базою даних, таких як виконання SQL-запитів або отримання даних.

Діаграма завершується, показуючи завершення процесу встановлення з'єднання з базою даних.

Для побудови графіка з прогнозованою кількістю порушень правил дорожнього руху розроблено клас Line, що використовує абстрактні класи Point2D (визначення точки, яка представляє місце в координатному просторі (x,

y)), Line2D (представлення відрізку лінії в просторі координат (x, y)) та Rectangle2D (опис прямокутника, визначеного розташуванням (x, y) і розмірністю (w x h)) [36].

Ці класи є лише абстрактними суперкласами усіх об’єктів, які зберігають 2D-координату. Фактичне представлення зберігання координат залишається для підкласу.

Діаграма діяльності сервера для побудови графіка зображена на рисунку 3.5.

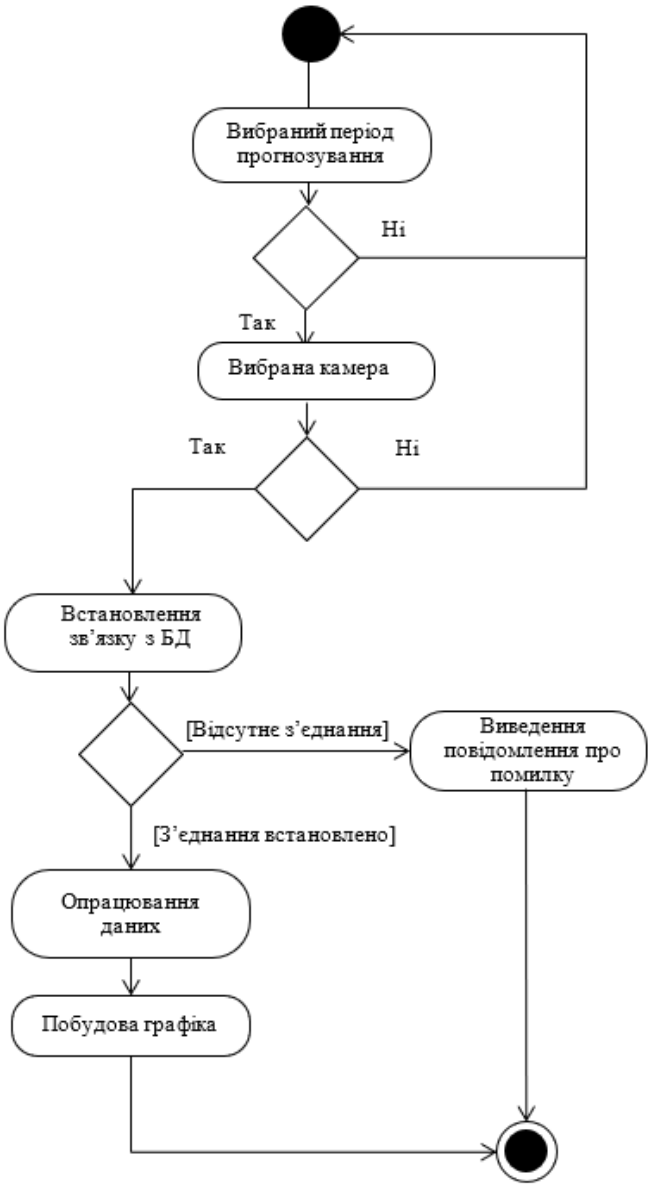

Рисунок 3.5 – Діаграма діяльності сервера для побудови графіка

Зображений вище алгоритм (рис. 3.5) можна записати у вигляді коду:

```
package painting;

import java.awt.geom.Line2D;
import java.awt.geom.Point2D;
import java.awt.geom.Rectangle2D;

public class Line extends Line2D{

    private Point p1;
    private Point p2;

    public Line(){
        p1 = new Point();
        p2 = new Point();
    }

    public Line(double x1, double y1, double x2, double y2){
        p1 = new Point(x1, y1);
        p2 = new Point(x2, y2);
    }

    @Override
    public Rectangle2D getBounds2D() {
        return null;
    }

    @Override
    public Point2D getP1() {
        return p1;
    }

    @Override
    public Point2D getP2() {
        return p2;
    }

    @Override
    public double getX1() {
        return p1.getX();
    }

    @Override
    public double getX2() {
        return p2.getX();
    }

    @Override
    public double getY1() {
        return p1.getY();
```

```
}

@Override
public double getY2() {
   return p2.getY();
}

@Override
public void setLine(double x1, double y1, double x2, double y2) {
   p1.setLocation(x1, y1);
   p2.setLocation(x2, y2);

}
      }
```

Діаграма діяльності сервера − це графічне зображення послідовності дій або процесів, які виконує сервер (програма на сервері) для побудови графіка або обробки запитів, пов'язаних з графіком [37]. Діаграма діяльності може бути створена за допомогою стандартів моделювання, таких як UML (Unified Modeling Language).

Діаграма діяльності сервера починається з початкової точки, яка вказує на початок виконання процесу. Спершу сервер очікує отримання запиту від клієнта або іншого джерела. Цей запит може містити інформацію про те, який графік потрібно побудувати, його параметри або дані для побудови. В такому випадку потрібно отримати початкові дані, такі як період прогнозу та камеру.

Сервер перевіряє отримані дані для визначення, чи є вони дійсними та безпечними для обробки. Далі сервер розпочинає обробку отриманого запиту. Це включає в себе визначення алгоритму та методів побудови графіка на основі отриманих даних. Для побудови графіка потрібні дані з бази даних, тому сервер може виконувати запити до бази даних для отримання потрібної інформації. Далі сервер виконує побудову графіка.

Після побудови графіка сервер генерує відповідь для клієнта або іншого джерела запиту. Ця відповідь може бути в форматі, придатному для відображення графіка, наприклад, як зображення або векторний графік. Сервер надсилає готовий графік або відповідь клієнту чи джерелу запиту, яке робило

запит. Діаграма завершується, показуючи завершення процесу побудови графіка та надсилання відповіді.

3.5 Тестування та аналіз результатів роботи інформаційної технології прогнозування кількості порушень правил дорожнього руху

Процес тестування та аналізу результатів роботи допомагає впевнитися в правильності роботи системи та оцінити її продуктивність. Важливо виконати кілька видів тестування та провести аналіз результатів.

Функціональне тестування дозволить переконайтися, що програма виконує всі функції, передбачені у специфікації проекту. Необхідно спробувати різні сценарії введення даних та переконатися, що програма генерує очікувані прогнози [38].

Також важливо перевірити, чи програма правильно обробляє та валідує вхідні дані. Це важливо для забезпечення коректної роботи та захисту від можливих атак. Важливим етапом є вимірювання швидкості роботи програми при обробці великої кількості даних. Потрібно переконатися, що вона працює ефективно та з відповідною швидкістю. Крім того, треба провести тести на стійкість програми до помилок та переконатися, що вона може відновити свою роботу після аварійного завершення.

Одним з найважливіших етапів тестування є порівняння результатів прогнозів програми з реальними даними та відповідності їх очікуванням. Якщо є відхилення, потрібно проаналізувати їх і виправити програму.

Після проведення тестів відбувається аналіз результатів. Необхідно порівняти результати прогнозів з реальними даними і визначити точність та ефективність системи. Якщо є недоліки, потрібно виправити їх і провести повторне тестування. Наслідком аналізу результатів тестування має бути підвищення точності прогнозування кількості порушень правил дорожнього руху.

Після успішного завершення тестування і виправлення виявлених проблем, можна бути впевненим в правильності та ефективності інформаційної технології прогнозування кількості порушень правил дорожнього руху.

Після запуску програми відкривається вікно початкової активності, яке включає два випадаючі меню. Перше призначене для введення терміну, на який користувач бажає здійснити прогноз, а друге – для вибору камери, розташованої на конкретній локації, яка цікавить користувача. Крім того, на вікні початкової активності розташована кнопка «Прогнозувати», яку необхідно натискати після визначення періоду та камери (рис. 3.6).

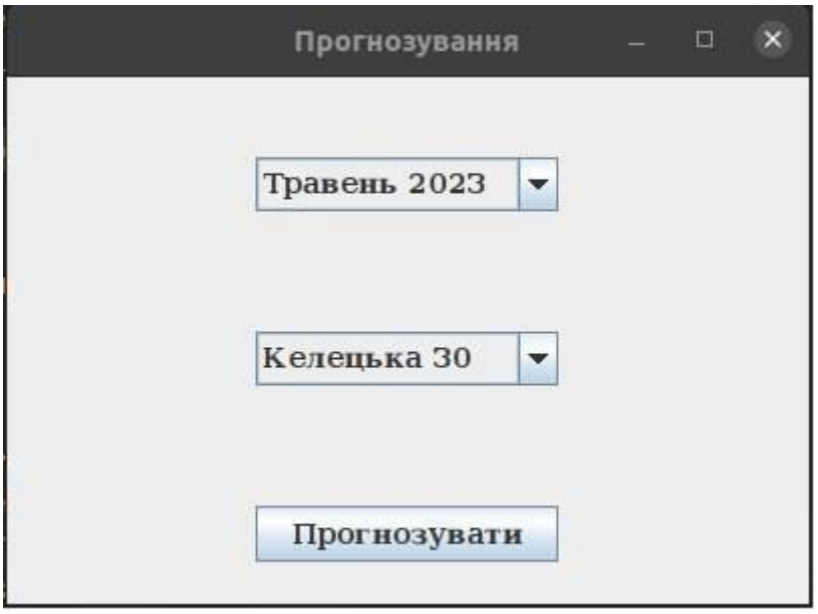

Рисунок 3.6 – Вікно початкової активності

Після введення користувачем періоду прогнозу та вибору камери, він натискає кнопку «Прогнозувати» (рис. 3.7). У результаті цієї дії відбувається перехід до вікна побудови прогнозу. Тут здійснюється відображення графіку прогнозованої кількості порушень правил дорожнього руху (рис. 3.8, 3.9).

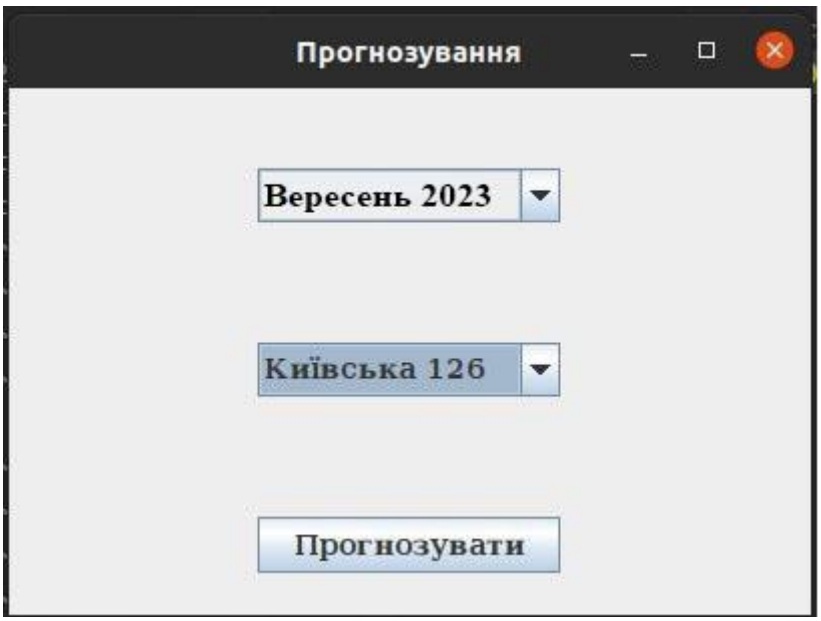

Рисунок 3.7 – Введення параметрів прогнозування

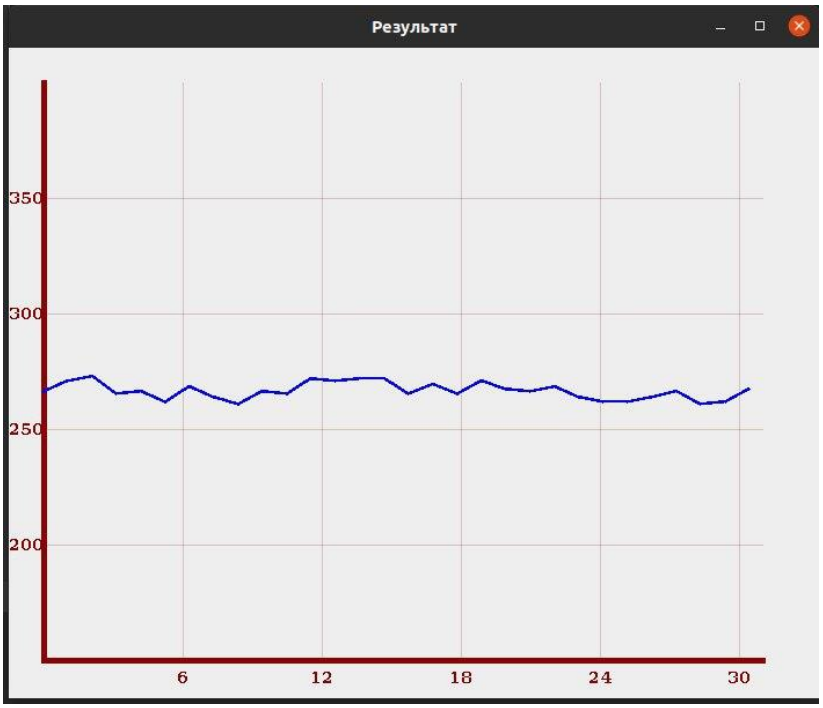

Рисунок 3.8 – Прогнозування кількості порушень

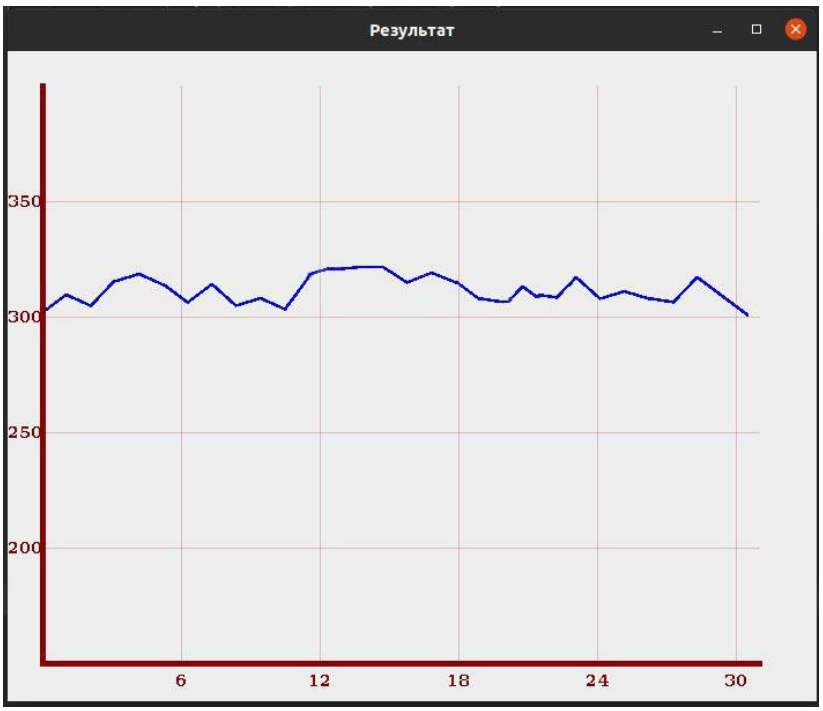

Рисунок 3.9 – Прогнозування кількості порушень

Розроблена інформаційна технологія прогнозування кількості порушень правил дорожнього руху пройшла етап випробувань, що дозволило переконатися в її ефективності та надійності. Під час цього тестування було проведено 150 запусків додатку, докладно досліджено всі аспекти його функціоналу та перевірено його можливості. Цей процес тестування включав в себе систематичне випробування різних сценаріїв використання, а також аналіз відповідності результатів очікуваним показникам. Проведені експерименти не лише підтвердили коректність роботи технології, але й надали можливість отримати глибше розуміння її робочих можливостей.

Проведемо аналіз результатів прогнозування, отриманих розробленою інформаційною технологією, порівнявши їх з двома найближчими аналогами та фактичними значеннями порушень за період з 1 по 30 вересня 2023 року на локації Київська 126. У таблиці 3.1 представлені отримані результати, які відображають прогнозовану кількість порушень правил дорожнього руху протягом місяця та вказують на точність прогнозу порівняно з фактичною кількістю порушень, яка відображена у статистиці. З використанням цих даних

буде можливість розрахувати точність прогнозування розробленого програмного продукту та його аналогів.

Таблиця 3.1 – Результати тестування розробленого програмного продукту порівняно з аналогом

Дата	Результат прогнозування розробленим програмним продуктом	Результат прогнозування аналогом SAS Visual Analytics	Результат прогнозування аналогом PTV Visum	Кількість порушень
01.09.2023	304	273	280	323
02.09.2023	310	271	279	323
03.09.2023	305	332	311	290
04.09.2023	324	275	306	336
05.09.2023	332	298	306	336
06.09.2023	314	295	301	334
07.09.2023	310	263	255	310
08.06.2023	314	276	276	325
09.09.2023	308	284	295	328
10.09.2023	312	286	283	333
11.09.2023	304	248	262	285
12.09.2023	330	351	343	319
13.09.2023	332	342	281	304
14.09.2023	333	346	331	306
15.09.2023	334	341	340	308
16.09.2023	324	253	252	292
17.09.2023	332	269	271	300
18.09.2023	324	274	279	310
19.09.2023	312	254	319	296
20.09.2023	322	280	280	329
21.09.2023	313	313	280	327
22.09.2023	311	277	261	314
23.09.2023	317	300	302	335
24.09.2023	311	297	300	321
25.09.2023	315	253	249	298
26.09.2023	317	319	313	286
27.09.2023	309	264	254	274
28.09.2023	338	298	301	349
29.09.2023	320	297	294	339
30.09.2023	309	246	258	280

Щоб визначити точність прогнозування та виразити її у відсотках, можна скористатися формулою для розрахунку відсотку похибки прогнозу:

$$\delta = \left|\frac{\Delta}{x}\right| 100\%, \quad (3.1)$$

де Δ - відхилення прогнозованого результату від фактичного значення;

x – фактичне значення.

Ця формула визначає абсолютне значення відсоткової різниці між фактичною та прогнозованою кількістю порушень, а множення на 100 перетворює це значення у відсотки. Такий підхід дозволяє отримати величину, яка представляє собою відсоток правильних прогнозів

Переведемо результати, наведені в таблиці 3.1 у значення відсотку похибки прогнозування для визначення точності прогнозування (табл. 3.2).

Таблиця 3.2 – Порівняння похибки

Назва	% похибки
Розроблений додаток	5,8
SAS Visual Analytics	12,4
PTV Visum	11,1

Отже, з таблиці 3.2 видно, що прогнозування за допомогою розробленого програмного продукту дає точніші результати порівняно з аналогом «SAS Visual Analytics» на 6,6%, а порівняно з аналогом «PTV Visum» на 5,3 %, що означає досягнення мети.

3.6 Висновок до розділу 3

Розроблено структуру інформаційної технології прогнозування кількості порушень правил дорожнього руху, загальну структурну схему та схему

алгоритмів функціонування програмного модуля прогнозування кількості порушень правил дорожнього руху.

Здійснено обґрунтуваня вибору мови програмування для реалізації інформаційної технології прогнозування кількості порушень правил дорожнього руху. Інформаційна технологія розроблена об’єктно-орієнтованою мовою програмування Java, враховуючи її переваги у вигляді кросплатформності, об’єктно-орієнтованості та простоти розробки, в середовищі розробки IntelliJ IDEA, завдяки його глибокому розумінні коду, задоволенні усіх вимог розробника, розумному доповненню коду.

Описано програмні рішення для програмної реалізації інформаційної технології. Завдяки використанню JDBC відбувається зручне підключення до бази даних, а розробка графічного інтерфейсу користувача відбувається за допомогою зручного інструменту – бібліотеки Swing. Програмно реалізовано інформаційну технологію прогнозування кількості порушень правил дорожнього руху.

Проведено тестування роботи інформаційної технології прогнозування кількості порушень правил дорожнього руху та проаналізовано його результати. Тестування програми підтвердило очікувані результати та правильність роботи програми, довівши підвищення точності прогнозування на 6,6% порівняно з аналогом «SAS Visual Analytics» та на 5,3% порівняно з аналогом «PTV Visum», що підтвердило доцільність розробки інформаційної технології.

4 ЕКОНОМІЧНА ЧАСТИНА

4.1 Проведення комерційного та технологічного аудиту інформаційної технології прогнозування кількості порушень правил дорожнього руху

Метою проведення комерційного і технологічного аудиту є оцінювання науково-технічного рівня та рівня комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності, тобто під час виконання магістерської кваліфікаційної роботи.

Для проведення комерційного та технологічного аудиту залучаємо 3-х незалежних експертів, якими є провідні викладачі випускової або спорідненої кафедри.

Оцінювання науково-технічного рівня розробки та її комерційного потенціалу здійснюємо із застосуванням п’ятибальної системи оцінювання за 12-ма критеріями, а результати зводимо до таблиці 4.1 [39].

Таблиця 4.1 – Результати оцінювання науково-технічного рівня і комерційного потенціалу інформаційної технології прогнозування кількості порушень правил дорожнього руху

Критерії	Експерти		
	Експерт 1	Експерт 2	Експерт 3
	Бали, виставлені експертами		
Технічна здійсненність концепції	3	4	4
Ринкові переваги (наявність аналогів)	3	4	4
Ринкові переваги (ціна продукту)	3	3	3
Ринкові переваги (технічні властивості)	4	3	3
Ринкові переваги (експлуатаційні витрати)	3	3	3
Ринкові перспективи (розмір ринку)	4	4	4
Ринкові перспективи (конкуренція)	3	4	3
Практична здійсненність (наявність фахівців)	3	3	3
Практична здійсненність (наявність фінансів)	4	4	4

Продовження таблиці 4.1

Критерії	Експерти		
	Експерт 1	Експерт 2	Експерт 3
	Бали, виставлені експертами		
Практична здійсненність (необхідність нових матеріалів)	4	4	4
Практична здійсненність (термін реалізації)	3	4	3
Практична здійсненність (розробка документів)	4	4	3
Сума балів	41	40	41
Середньоарифметична сума балів, СБ	41		

За результатами розрахунків, наведених в таблиці 4.1 робимо висновок про те, що науково-технічний рівень та комерційний потенціал інформаційної технології прогнозування кількості порушень правил дорожнього руху – вище середнього.

4.2 Розрахунок витрат на здійснення науково-дослідної роботи

Витрати на оплату праці. Належать витрати на виплату основної та додаткової заробітної плати керівникам відділів, лабораторій, секторів і груп, науковим, інженерно-технічним працівникам, конструкторам, технологам, креслярам, копіювальникам, лаборантам, робітникам, студентам, аспірантам та іншим працівникам, безпосередньо зайнятим виконанням конкретної теми, обчисленої за посадовими окладами, відрядними розцінками, тарифними ставками згідно з чинними в організаціях системами оплати праці, також будь-які види грошових і матеріальних доплат, які належать до елемента «Витрати на оплату праці» [40].

Основна заробітна плата дослідників. Витрати на основну заробітну плату дослідників ($З_о$) розраховують відповідно до посадових окладів працівників, за формулою:

$$З_о = \sum_{i=1}^{к} \frac{М_{ni} \cdot t_i}{Т_р}, \quad (4.1)$$

де к – кількість посад дослідників, залучених до процесу дослідження;

$М_{ni}$ – місячний посадовий оклад конкретного розробника (інженера, дослідника, науковця тощо), грн.;

$Т_р$ – число робочих днів в місяці; приблизно $Т_р$ = (21…23) дні;

t_i – число робочих днів роботи розробника (дослідника).

Зроблені розрахунки зводимо до таблиці 4.2.

Таблиця 4.2 – Витрати на заробітну плату дослідників

Посада	Місячний посадовий оклад, грн.	Оплата за робочий день, грн.	Число днів роботи	Витрати на заробітну плату, грн.
Керівник	37000	1762	35	61635
Розробник	29000	1381	35	48335
Всього:	109970			

Основна заробітна плата робітників. Витрати на основну заробітну плату робітників ($З_р$) за відповідними найменуваннями робіт розраховують за формулою:

$$З_р = \sum_{i=1}^{n} C_i \cdot t_i, \quad (4.2)$$

де $С_i$ – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год;

t_i – час роботи робітника на виконання певної роботи, год.

Погодинну тарифну ставку робітника відповідного розряду С і можна визначити за формулою:

$$С_i = \frac{М_м \cdot К_i \cdot К_с}{Т_р \cdot t_{зм}}, \quad (4.3)$$

де $М_м$ – розмір прожиткового мінімуму працездатної особи або мінімальної місячної заробітної плати (залежно від діючого законодавства), у 2023 році $М_м$=6700 грн;

$К_і$ – коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду;

$К_с$ – мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих об’єднань і підприємств до законодавчо встановленого розміру мінімальної заробітної плати;

$Т_р$ – середня кількість робочих днів в місяці, приблизно $Т_р$ = 21…23 дні; $t_{зм}$ – тривалість зміни, год.

Результати обчислень подано в таблиці 4.3.

Таблиця 4.3 – Витрати на заробітну плату робітників

Найменування робіт	Трудомісткість, н-год.	Розряд роботи	Погодинна тарифна ставка	Тариф. коеф.	Величина, грн.
Аналіз предметної області	20	5	54,2	1,36	1084
Моделювання інформаційної технології	100	5	54,2	1,36	5420
Створення основної структури проєкту	16	6	57,8	1,45	925
Створення та наповнення бази даних	74	6	57,8	1,45	4277
Тестування інформаційної технології	70	5	54,2	1,36	3794
Всього					15500

Додаткова заробітна плата. Додаткова заробітна плата $З_д$ всіх розробників та робітників, які брали участь у виконанні даного етапу роботи, розраховується як (10…12)% від суми основної заробітної плати всіх розробників та робітників, тобто:

$$З_д = 0{,}1 \cdot (З_о + З_р) = 0{,}1 \cdot (109970 + 15500) = 12547 \text{ (грн.)}.$$

Відрахування на соціальні заходи. Нарахування на заробітну плату $Н_{зп}$ розробників та робітників, які брали участь у виконанні даного етапу роботи, розраховуються за формулою:

$$Н_{зп} = \beta \cdot (З_о + З_р + З_д) =$$
$$= 0{,}22 \cdot (109970 + 15500 + 12547) = 30364 \text{ (грн.)},$$

де $З_о$ – основна заробітна плата розробників, грн.;

$З_р$ – основна заробітна плата робітників, грн.;

$З_д$ – додаткова заробітна плата всіх розробників та робітників, грн.;

β – ставка єдиного внеску на загальнообов'язкове державне соціальне страхування, % (приймаємо для 1-го класу професійності ризику 22%).

Програмне забезпечення. До балансової вартості програмного забезпечення входять витрати на його інсталяцію, тому ці витрати беруться додатково в розмірі 10…12% від вартості програмного забезпечення. Балансову вартість програмного забезпечення розраховують за формулою:

$$В_{прг} = \sum_{1}^{к} Ц_{іпрг} \cdot С_{прг.і} \cdot К_і, \qquad (4.4)$$

де $Ц_{іпрг}$ – ціна придбання програмного забезпечення і-го виду, грн.;

$С_{прг.і}$ – кількість одиниць програмного забезпечення відповідного виду, шт.;

$К_і$ – коефіцієнт, що враховує інсталяцію, налагодження програмного забезпечення;

$К_і$ = (1,1…1,12); к – кількість видів програмного забезпечення.

Результати обчислень подано в таблиці 4.4.

Таблиця 4.4 – Витрати на придбання програмного забезпечення

Найменування програмного забезпечення	Ціна за одиницю, грн.	Витрачено	Вартість програмного забезпечення, грн.
Intellij Idea	23361	1	23361
MySQL Workbench	36147	1	36147
Всього, з врахуванням коефіцієнта інсталяції та налагодження			65459

Амортизація обладнання. Амортизація обладнання, комп'ютерів та приміщень, які використовувались під час (чи для) виконання даного етапу роботи.

У спрощеному вигляді амортизаційні відрахування А в цілому бути розраховані за формулою:

$$\text{А} = \frac{\text{Ц}_\text{б}}{\text{Т}_\text{в}} \cdot \frac{t}{12}, \tag{4.5}$$

де $\text{Ц}_\text{б}$ – загальна балансова вартість всього обладнання, комп'ютерів, приміщень тощо, що використовувались для виконання даного етапу роботи, грн.;

t – термін використання основного фонду, місяці;

$\text{Т}_\text{в}$ – термін корисного використання основного фонду, роки.

В таблиці 4.5 подано результати обчислень.

Таблиця 4.5 – Амортизаційні відрахування за видами основних фондів

Найменування	Балансова вартість, грн.	Строк корисного використання, років	Термін використання, місяців	Сума амортизації, грн.
Ноутбук	43999	3	1,75	2139
Монітор	7199	3	1,75	350
Стіл	1855	5	1,75	54
Стілець	1799	5	1,75	53
Настільна лампа	489	5	1,75	14
Всього	2610			

Витрати на електроенергію для науково-виробничих цілей. Витрати на силову електроенергію $В_е$, якщо ця стаття має суттєве значення для виконання даного етапу роботи, розраховуються за формулою (4.6) і занесемо розрахунки до таблиці 4.6:

Таблиця 4.6 – Витрати на електроенергію

Найменування обладнання	Потужність, кВт	Тривалість годин роботи
Ноутбук	0,096	280
Монітор	0,036	280
Освітлення	0,024	100

$$В_е = \sum \frac{W_i \cdot t_i \cdot Ц_е \cdot К_{впі}}{ККД}, \quad (4.6)$$

$$В_е = \frac{0{,}096 \cdot 280 \cdot 7{,}5 \cdot 0{,}95}{0{,}98} + \frac{0{,}036 \cdot 280 \cdot 7{,}5 \cdot 0{,}95}{0{,}98} + \frac{0{,}024 \cdot 100 \cdot 7{,}5 \cdot 0{,}95}{0{,}98} = 286 \text{ (грн.)}.$$

де W_i – встановлена потужність обладнання, кВт;

t_i – тривалість роботи обладнання на етапі дослідження, год.;

Це – вартість 1 кВт електроенергії, грн.;

$К_{впі}$ – коефіцієнт використання потужності; ККД – коефіцієнт корисної дії обладнання.

Інші витрати. До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками.

Витрати за статтею «Інші витрати» розраховуються як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$Ів = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \quad (4.7)$$

де $Н_{ів}$ – норма нарахування за статтею «Інші витрати».

$$Ів = (109970 + 15500) \cdot \frac{80}{100} = 100376 \text{ (грн)}.$$

Накладні (загальновиробничі) витрати. До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов'язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов'язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов'язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін.

Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуються як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$В_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (4.8)$$

$$В_{нзв} = (109970 + 15500) \cdot \frac{150}{100} = 188205 \text{ (грн.)},$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати».

Витрати на проведення науково-дослідної роботи. Витрати на проведення науково-дослідної роботи розраховуються як сума всіх попередніх статей витрат за формулою:

$$В_{заг} = З_о + З_р + З_{дод} + З_н + К_в + В_{спец} + В_{прг} + А_{обл} + В_е + +І_в + В_{нзв}. \quad (4.9)$$

$В_{заг} = 109970 + 15500 + 12547 + 30364 + 65459 + 2610 + 100376 + 286 + 188205 = 525317$ (грн).

Загальні витрати. Загальні витрати ЗВ на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховуються за формулою:

$$ЗВ = \frac{В_{заг}}{\eta}, \quad (4.10)$$

$$ЗВ = \frac{525317}{0,9} = 583686 \text{ (грн.)},$$

де η – коефіцієнт, що характеризує етап виконання науково-дослідної роботи. Оскільки, якщо науково-технічна розробка знаходиться на стадії впровадження, то η=0,9.

4.3 Розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором

В ринкових умовах узагальнюючим позитивним результатом, що його може отримати потенційний інвестор від можливого впровадження результатів тієї чи іншої науково-технічної розробки, є збільшення у потенційного інвестора величини чистого прибутку.

В даному випадку відбувається розробка засобу, тому основу майбутнього економічного ефекту буде формувати: ΔN – збільшення кількості споживачів, яким надається відповідна інформаційна послуга в аналізовані періоди часу; N – кількість споживачів, яким надавалась відповідна інформаційна послуга у році до впровадження результатів нової науково-технічної розробки; $Ц_б$ – вартість послуги у році до впровадження інформаційної системи; $\pm\Delta Ц_о$ – зміна вартості

послуги (зростання чи зниження) від впровадження результатів науково-технічної розробки в аналізовані періоди часу.

Можливе збільшення чистого прибутку у потенційного інвестора ΔПі для кожного із років, протягом яких очікується отримання позитивних результатів від можливого впровадження та комерціалізації науково-технічної розробки, розраховується за формулою:

$$\Delta П = (\pm \Delta Ц_о \cdot N + Ц_о \cdot \Delta N_i)_i \cdot \lambda \cdot \rho \cdot \left(1 - \frac{\vartheta}{100}\right), \qquad (4.11)$$

де ±ΔЦ – зміна основного якісного показника від впровадження результатів науково-технічної розробки в аналізованому році. Зазвичай, таким показником може бути зміна ціни реалізації одиниці нової розробки в аналізованому році (відносно року до впровадження цієї розробки); $\pm\Delta Ц_о$ може мати як додатне, так і від'ємне значення (від'ємне – при зниженні ціни відносно року до впровадження цієї розробки, додатне – при зростанні ціни); N – основний кількісний показник, який визначає величину попиту на аналогічні чи подібні розробки у році до впровадження результатів нової науково-технічної розробки; $Ц_о$ – основний якісний показник, який визначає ціну реалізації нової науково-технічної розробки в аналізованому році; $Ц_б$ – основний якісний показник, який визначає ціну реалізації існуючої (базової) науково-технічної розробки у році до впровадження результатів; ΔN – зміна основного кількісного показника від впровадження результатів науково-технічної розробки в аналізованому році. Зазвичай таким показником може бути зростання попиту на науково-технічну розробку в аналізованому році (відносно року до впровадження цієї розробки); λ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість. У 2023 році ставка податку на додану вартість становить 20%, а коефіцієнт λ =0,8333; ρ – коефіцієнт, який враховує рентабельність інноваційного продукту (послуги). Рекомендується брати ρ =0,2...0,5; ϑ – ставка

податку на прибуток, який має сплачувати потенційний інвестор, у 2023 році ϑ =18%.

Очікуваний термін життєвого циклу розробки 1 рік, тому:

$$\Delta П = ((750 - 210) \cdot 12 \cdot 1000 - (2000 - 1000) \cdot 210 \cdot 12) \cdot 0{,}8333 \cdot 0{,}3 \cdot \left(1 - \frac{18}{100}\right) = 867715 \text{ (грн)}.$$

Далі розраховують приведену вартість збільшення всіх чистих прибутків ПП, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \qquad (4.12)$$

де ΔП – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн.;

Т – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки (приймаємо Т=1 рік);

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ =0,05…0,15;

t – період часу (в роках) від моменту початку впровадження науково-технічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

$$ПП = \frac{867715}{(1+0{,}1)^1} = 788832 (\text{грн}).$$

Далі розраховують величину початкових інвестицій PV , які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної

розробки. Для цього можна використати формулу:

$$PV = k_{\text{інв}} \cdot \text{ЗВ}, \quad (4.13)$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію. Це можуть бути витрати на підготовку приміщень, розробку технологій, навчання персоналу, маркетингові заходи тощо; зазвичай $k_{інв}$=1...5, але може бути і більшим;

ЗВ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, грн.

$$PV = 1 \cdot 583686 = 583686 \text{ (грн.)}.$$

Тоді абсолютний економічний ефект $E_{абс}$ або чистий приведений дохід для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$\text{Е}_{\text{абс}} = \text{ПП} - PV, \quad (4.14)$$

де ПП – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, грн.;

PV – теперішня вартість початкових інвестицій, грн.

$$\text{Е}_{\text{абс}} = 788832 - 583686 = 205146(\text{ грн.}).$$

Оскільки $E_{абс}$>0, то можемо припустити про потенційну зацікавленість інвесторів у розробці.

Для остаточного прийняття рішення з цього питання необхідно розрахувати внутрішню економічну дохідність Ев або показник внутрішньої норми дохідності вкладених інвестицій та порівняти її з так званою бар’єрною ставкою дисконтування, яка визначає ту мінімальну внутрішню економічну

дохідність, нижче якої інвестиції в будь-яку науково-технічну розробку вкладати буде економічно недоцільно.

Внутрішня економічна дохідність інвестицій Ев, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки, розраховується за формулою:

$$E_в = \sqrt[Т_ж]{1 + \frac{Еабс}{PV}}, \quad (4.15)$$

де $Т_ж$ – життєвий цикл розробки, роки.

$$E_в = \sqrt[1]{1 + \frac{205146}{583686}} = 1{,}16.$$

Визначимо бар'єрну ставку дисконтування $\tau_{мін}$, тобто мінімальну внутрішню економічну дохідність інвестицій, нижче якої кошти у впровадження науково-технічної розробки та її комерціалізацію вкладатися не будуть.

Мінімальна внутрішня економічна дохідність вкладених інвестицій $\tau_{мін}$ визначається за формулою:

$$\tau_{мін} = d + f, \quad (4.16)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2023 році в Україні d =0,9...0,12;

f – показник, що характеризує ризикованість вкладення інвестицій; зазвичай величина f =0,05...0,5, але може бути і значно вищою.

$$\tau_{мін} = 0{,}9 + 0{,}05 = 0{,}95.$$

Оскільки $E_в$=1,16>$\tau_{мін}$=0,95, то потенційний інвестор може бути зацікавлений у фінансуванні впровадження науково-технічної розробки та виведенні її на ринок, тобто в її комерціалізації.

Далі розраховуємо період окупності інвестицій $T_о$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$T_о = \frac{1}{E_в}, \quad (4.17)$$

$$T_о = \frac{1}{1{,}16} = 0{,}86 \text{ (року.)}.$$

Оскільки $T_о$=0,86<1…3-х років, то це свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження цієї розробки та виведення її на ринок.

4.4 Висновок до розділу 4

Згідно з проведеними дослідженнями рівень комерційного потенціалу розробки за темою «Інформаційна технологія прогнозування кількості порушень правил дорожнього руху» становить 41 бал, що свідчить про комерційну важливість проведення таких досліджень (рівень комерційного потенціалу розробки вище середнього).

Економічна частина цієї роботи містить розрахунок витрат на розробку інформаційної технології прогнозування кількості порушень правил дорожнього руху, сума яких складає 583686 гривень. Було спрогнозовано орієнтовану величину витрат по кожній з статей витрат. Також розраховано чистий прибуток, який може отримати виробник від реалізації нового технічного рішення та економічний ефект при використанні цієї розробки. Також термін окупності становить 0,86 р., що менше 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження розробки та виведення її на ринок.

Отже, можна зробити висновок про доцільність проведення науково-дослідної роботи за темою «Інформаційна технологія прогнозування кількості порушень правил дорожнього руху».

ВИСНОВКИ

Всі завдання, поставлені для реалізації інформаційної технології прогнозування кількості порушень правил дорожнього руху виконані в повному обсязі, а саме: проведено аналіз програм-аналогів та обґрунтувано доцільність розробки інформаційної технології прогнозування кількості порушень правил дорожнього руху; розроблено структуру інформаційної технології прогнозування кількості порушень правил дорожнього руху; розроблено математичну модель прогнозування кількості порушень правил дорожнього руху; програмно реалізовано інформаційну технологію прогнозування кількості порушень правил дорожнього руху; проведено тестування програми та проаналізовано отримані результати; економічно обґрунтувано доцільність розробки інформаційної технології прогнозування кількості порушень правил дорожнього руху.

Під час виконання магістерської кваліфікаційної роботи було реалізовано інформаційну технологію прогнозування кількості порушень правил дорожнього руху.

Обґрунтовано доцільність розробки інформаційної технології прогнозування кількості порушень правил дорожнього руху. Проведено аналіз сучасних аналогів, які використовуються для прогнозування ситуацій на дорогах. Виявлено проблему відсутності інтуїтивно зрозумілого інтерфейсу та недостатню точність прогнозування, що довело актуальність розробки інформаційної технології прогнозування кількості порушень правил дорожнього руху.

Було розроблено структуру інформаційної технології прогнозування кількості порушень правил дорожнього руху та спроектовано схему алгоритму роботи розроблюваної програми.

На основі аналізу існуючих моделей та методів, що застосовуються для вирішення поставленої задачі було обґрунтовано використання в інформаційній технології алгоритмів прогнозування часових рядів, використання алгоритмів на

основі аналізу геопросторових даних, які увійшли в розроблену математичну модель прогнозування кількості порушень правил дорожнього руху, поєднання яких дає змогу підвищити точність прогнозування.

Здійснено програмну реалізацію інформаційної технології прогнозування кількості порушень правил дорожнього руху. Обґрунтовано вибір мови програмування. Розробка інформаційної технології велася на об'єктно-орієнтованій мові програмування Java в середовищі розробки IntelliJ IDEA.

Проведено тестування розробленої програми та підтверджено її працездатність. Порівняно з прототипом точність прогнозування зросла на 5,3%, що означає доведення поставленої мети.

Загальна похибка точності прогнозування кількості порушень правил дорожнього руху розробленого програмного продукту – 5,8, що на 5,3% краще, ніж результат аналогу «PTV Visum» та на 6,6% краще, ніж результат аналогу «SAS Visual Analytics».

Здійснено економічне обґрунтування доцільності розробки інформаційної технології прогнозування кількості порушень правил дорожнього руху. Проведено оцінювання комерційного потенціалу розробки. Проведено технологічний аудит із залученням експертів. Згідно висновків експертів, рівень комерційного потенціалу розробки є вищим середнього. Здійснено прогнозування витрат на виконання науково-дослідної роботи. Загальні витрати становлять 583686 грн.

Мета магістерської кваліфікаційної роботи, яка полягала в підвищенні точності прогнозування кількості порушень правил дорожнього руху, досягнута за рахунок використання математичної моделі на основі методу часових рядів ARIMA та аналізу геопросторових даних. Це дозволяє максимально точно визначити кількість порушень правил дорожнього руху, аналізуючи просторові закономірності та передбачаючи ризиковані зони.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Львовський О. О. Крилик Л. В. Дослідження функціональних характеристик методів та засобів для прогнозування кількості порушень правил дорожнього руху. *Матеріали LI науково-технічної конференції підрозділів Вінницького національного технічного університету (НТКП ВНТУ–2022) : збірник доповідей.* [Електронний ресурс]. Вінниця : ВНТУ, 2022. С. 808−810. URL: https://conferences.vntu.edu.ua/public/files/1/vntu_2022_netpub.pdf (дата звернення: 26.09.2023).

2. Львовський О. О., Крилик Л.В. Перспективи розробки інформаційної технології прогнозування кількості порушень правил дорожнього руху. *Матеріали LII науково-технічної конференції підрозділів Вінницького національного технічного університету (НТКП ВНТУ–2023) : збірник доповідей* [Електронний ресурс]. –Вінниця : ВНТУ, 2023. С. 408 – 410 (PDF, 3030 с.). URL: https://press.vntu.edu.ua/index.php/vntu/catalog/view/788/1373/2632-1 (дата звернення: 26.09.2023).

3. Безпека дорожнього руху по-новому: переваги та ризики [Електронний ресурс]. URL: https://uifuture.org/publications/bezpeka-dorozhnogoruhu-po-novomu-perevagy-ta-ryzyky/ (дата звернення: 24.09.2023).

4. Інформаційний портал Національної поліції України [Електронний ресурс]. URL: https://zakon.rada.gov.ua/laws/show/z1059-17#Text (дата звернення: 25.09.2023).

5. Патрульна поліція [Електронний ресурс]. URL: https://patrolpolice.gov.ua/ (дата звернення: 25.09.2023).

6. Road traffic injuries. World Health Organization [Електронний ресурс]. URL: https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries (дата звернення: 25.09.2023).

7. Безпека на дорозі: правила та застороги. Міністерство охорони здоров’я України [Електронний ресурс]. URL: https://moz.gov.ua/article/news/bezpeka-na-dorozi-pravila-ta-zastorogi (дата звернення: 25.09.2023).

8. Opendatabot [Електронний ресурс]. URL: https://opendatabot.ua/ (дата звернення: 26.09.2023).

9. Hunt E. B., Marin J., Stone P. Experiments in Induction by E.B. *The American Journal of Psychology*. N.Y.: Academic Press, 1966. Vol. 80, № 4. P. 651−653.

10. Основні причини ДТП в Україні: нетверезе водіння та перевищення швидкості [Електронний ресурс]. URL: https://zn.ua/ukr/UKRAINE/osnovni-prichini-dtp-v-ukrayini-netvereze-vodinnya-ta-perevischennya-shvidkosti-326851_.html (дата звернення: 26.09.2023).

11. Широкорад Д. В. Машинне навчання: навч. посіб. Запоріжжя : Національний університет «Запорізька політехніка», 2021. 320 с.

12. Бідюк П. І., Романенко В., Тимощук О. Аналіз часових рядів: навч. посіб. Київ: Політехніка, 2010. 317 с.

13. Grunwald P. D., Myung I. J. (eds.) Advances In Minimum Description Length: Theory And Applications. Springer. 2005.

14. Єріна А. М. Статистичне моделювання та прогнозування: навч. посіб. К.: КНЕУ, 2001. 170 с.

15. Ghebreyesus TA., Swaminathan S. Scientists are sprinting to outpace the novel coronavirus. Lancet. 2020; 395: 762−764.

16. Zhou Y.; Chen L. Twenty-Year Span of Global Coronavirus Research Trends: A Bibliometric Analysis. Int. J. Environ. Res. Public Health 2020, 17, 3082.

17. Прогнозування соціально-економічних процесів : навчальний посібник для студентів напряму підготовки 6.030502 "Економічна кібернетика" денної форми навчання / Т. С. Клебанова, В. А. Курзенев, В. М. Наумов та ін. Х. : ХНЕУ ім. С. Кузнеця, 2015. 656 с.

18. Додаток Aimsun [Електронний ресурс]. URL: https://www.aimsun.com/ (дата звернення: 15.09.2023).

19. Додаток VISSIM [Електронний ресурс]. URL: https://your.visum.ptvgroup.com/vision-traffic-suite-students-en (дата звернення: 15.09.2023).

20. Додаток TransCAD [Електронний ресурс]. URL: https://play.google.com/store/apps/details?id=com.caberset.transcard&hl=uk&gl=US&pli=1 (дата звернення: 15.09.2023).

21. Додаток SAS Visual Analytics [Електронний ресурс]. URL: https://www.sas.com/en_us/software/visual-analytics.html (дата звернення: 15.09.2023).

22. Додаток PTV Visum [Електронний ресурс]. URL: https://pro-mobility.org/vprovadzhennja-programnogo-zabezpechennja/programne-zabezpechennja -ptv-visum/ (дата звернення: 24.09.2023).

23. Додаток HERE Traffic Analytics [Електронний ресурс]. URL: https://app.mobito.io/data-product/here-traffic-analytics (дата звернення: 24.09.2023).

24. Поліщук В. П., Дзюба О. П. Теорія транспортного потоку: методи та моделі організації дорожнього руху. К.: Знання України, 2008. 175 с.

25. Романюк О. Н., Савчук Т. О. Організація баз даних і знань : навч. посіб. Вінниця: УНІВЕРСУМ-Вінниця, 2003. 217с.

26. Гайна Г. А. Основи проектування баз даних : навч. посіб. К: КНУБА, 2005. 204 с.

27. Алхімов С. М. Об’єктно-орієнтоване програмування : підручник. Київ : КПІ ім. Ігоря Сікорського, 2019. 192 с.

28. Коваленко О. С., Добровська Л. М. Проектування інформаційних систем: загальні питання теорії проектування ІС : конспект лекцій. Київ : КПІ ім. Ігоря Сікорського, 2020. 192 с.

29. Коноваленко І.В., Марущак П. О., Савків В. Б. Програмування мовою C# 7.0 : навч. посіб. Тернопіль : Тернопільський національний технічний унверситет імені Івана Пулюя, 2017. 300 с.

30. Копитко М. Ф., Іванків К. С. Основи програмування мовою Java: тексти лекцій. Львів: Видавничий центр ЛНУ ім. Івана Франка, 2002. 83с.

31. Intellij Idea − середовище розробки для Java [Електронний ресурс]. URL: https://ualinux.com/uk/ubuntu-apps-programming/idea-ic (дата звернення: 18.09.2023).

32. Standard libraries [Електронний ресурс]. URL: https://introcs.cs.princeton.edu/java/stdlib/ (дата звернення: 26.09.2023).

33. Collections Framework Overview [Електронний ресурс]. URL: https://docs.oracle.com/javase/8/docs/technotes/guides/collections/overview.html (дата звернення: 26.09.2023).

34. JavaFX [Електронний ресурс]. URL: https://openjfx.io/ (дата звернення: 26.09.2023).

35. Java JDBC [Електронний ресурс]. URL: https://www. javatpoint.com/java-jdbc (дата звернення: 26.09.2023).

36. Побудова графіків з використанням Java [Електронний ресурс]. URL: http://do-want-to-know.blogspot.com/2010/03/java.html (дата звернення: 27.09.2023).

37. Добровська Л. М., Аверьянова О. В. Проєктування інформаційних систем : комп’ютерний практикум. Київ : КПІ ім. Ігоря Сікорського, 2021. 202 с.

38. Авраменко А. С., Авраменко В. С., Костенюк Г. В. Тестування програмного забезпечення : навч. посіб. Черкаси : ЧНУ ім. Богдана Хмельницького, 2017. 284 с.

39. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. Вінниця : ВНТУ, 2021. 42 с.

40. Кавецький В. В. Економічне обґрунтування інноваційних рішень: практикум / В. В. Кавецький, В. О. Козловський, І. В. Причепа Вінниця : ВНТУ, 2016. 113 с.

Додаток А (обов'язковий)
Протокол перевірки кваліфікаційної роботи на наявність текстових запозичень

ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Інформаційна технологія прогнозування кількості порушень правил дорожнього руху

Тип роботи: магістерська кваліфікаційна робота
(БДР, МКР)

Підрозділ кафедра комп'ютерних наук, ФІТА
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 85,7% Схожість 14,3%

Аналіз звіту подібності (відмітити потрібне):

✓ Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

□ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

□ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи Львовський О. О.

Керівник роботи Крилик Л. В.

Опис прийнятого рішення

Магістерську кваліфікаційну роботу допущено до захисту

Особа, відповідальна за перевірку Озеранський В. С.

Додаток Б (обов'язковий)

Лістинг програми

Клас DataProcessing

```
package procesing;

import db.DAO;

import java.math.BigInteger;
import java.util.ArrayList;
import java.util.Random;

public class DataProcessing {
    public static ArrayList<Integer> getForOne() {
        ArrayList<Integer> result = new DAO().getAllDate();
        getNext(result);
        getNext(result);
        getNext(result);
        getNext(result);
        getNext(result);
        getNext(result);
        return getResult(result);
    }

    public static ArrayList<Integer> getForTwo() {
        ArrayList<Integer> result = getForOne();
        getNext(result);
        getNext(result);
        getNext(result);
        getNext(result);
        getNext(result);
        getNext(result);
        return getResult(result);
    }
```

```
public static ArrayList<Integer> getForThree() {
    ArrayList<Integer> result = getForTwo();
    getNext(result);
    getNext(result);
    getNext(result);
    getNext(result);
    getNext(result);
    getNext(result);
    return getResult1(result);
}

public static ArrayList<Integer> getForFour() {
    ArrayList<Integer> result = getForThree();
    getNext(result);
    getNext(result);
    getNext(result);
    getNext(result);
    getNext(result);
    getNext(result);
    return getResult(result);
}

public static ArrayList<Integer> getForFive() {
    ArrayList<Integer> result = getForFour();
    getNext(result);
    getNext(result);
    getNext(result);
    getNext(result);
    getNext(result);
    getNext(result);
    return getResult(result);
}

public static ArrayList<Integer> getForSix() {
```

```
    ArrayList<Integer> result = getForFive();
    getNext(result);
    getNext(result);
    getNext(result);
    getNext(result);
    getNext(result);
    getNext(result);
    return getResult(result);
}

public static ArrayList<Integer> getResult(ArrayList<Integer> arg){
    ArrayList<Integer> rez = new ArrayList<>();
    for (int j=6; j>2; j--){
        rez.add(arg.get(arg.size()-j));
        int start = arg.get(arg.size()-j);
        int fin = arg.get(arg.size()-j+1);
        for (int i = 1; i < 7; i++) {
            int rand = new Random().nextInt(400);
            int rizn = (fin-start)/7;
            rizn = rizn*i;
            rizn = rizn+rand-200;
            rez.add(start+rizn);
        }
    }
    int start = arg.get(arg.size()-2);
    int fin = arg.get(arg.size()-1);
    for (int i = 1; i < 3; i++) {
        int rand = new Random().nextInt(400);
        int rizn = (fin-start)/7;
        rizn = rizn*i;
        rizn = rizn+rand-200;
        rez.add(start+rizn);
    }
    rez.add(arg.get(arg.size()-1));
    return rez;
```

```
}

public static ArrayList<Integer> getResult1(ArrayList<Integer> arg){
    ArrayList<Integer> rez = new ArrayList<>();
    int start = arg.get(arg.size()-6);
    for (int j=6; j>2; j--){
        start = start + 700;
        for (int i = 1; i < 8; i++) {
            int rand = new Random().nextInt(400);
            int rizn = 700/7;
            rizn = rizn*i;
            rizn = rizn+rand-200;
            rez.add(start+rizn);
        }
    }
    for (int i = 1; i < 5; i++) {
        int rand = new Random().nextInt(400);
        int rizn = 400/7;
        rizn = rizn*i;
        rizn = rizn+rand-200;
        rez.add(start+rizn);
    }
    return rez;
}

private static void getNext(ArrayList<Integer> args) {

    ArrayList<Integer> listY = args;

    int sumY = 0;
    int sumX = 0;
    int sumX2 = 0;
    int sumXY = 0;
    for (int j = 0; j < listY.size(); j++) {
        int x = j+1;
```

```
            sumY = sumY + listY.get(j);
            sumXY = sumXY + (listY.get(j) * x);
            sumX = sumX + x;
            sumX2 = sumX2 + (x * x);
        }
        int n = listY.size();

        BigInteger a11 = BigInteger.valueOf(sumY).multiply(BigInteger.valueOf(sumX2));
        BigInteger a12 = BigInteger.valueOf(sumX).multiply(BigInteger.valueOf(sumXY));
        BigInteger a1 = a11.subtract(a12);
        BigInteger a21 = BigInteger.valueOf(n).multiply(BigInteger.valueOf(sumX2));
        BigInteger a22 = BigInteger.valueOf(sumX).multiply(BigInteger.valueOf(sumX));
        BigInteger a2 = a21.subtract(a22);
        long a = a1.longValue() / a2.longValue();

        BigInteger b11 = BigInteger.valueOf(n).multiply(BigInteger.valueOf(sumXY));
        BigInteger b12 = BigInteger.valueOf(sumX).multiply(BigInteger.valueOf(sumY));
        BigInteger b1 = b11.subtract(b12);

        BigInteger b21 = BigInteger.valueOf(n).multiply(BigInteger.valueOf(sumX2));
        BigInteger b22 = BigInteger.valueOf(sumX).multiply(BigInteger.valueOf(sumX));
        BigInteger b2 =  b21.subtract(b22);

        long b = b1.longValue() / b2.longValue();
        int yr = (int) (a + b* (n + 1));
        args.add(yr);
    }
}
```

Клас GraphForFirst

```
package procesing;


import painting.GraphicsPanel;
```

```
import javax.swing.*;
import java.awt.*;
import java.util.List;

public class GraphForFirst {
    public static ImageIcon image;
    public static JFrame frame;
    public List<Integer> list;


    public GraphForFirst(List<Integer> list) {
        this.list = list;
//        image = new ImageIcon(getClass().getResource("1.jpg"));
    }


    public void createWindow() {
        frame = new JFrame();
        frame.setTitle("Результат");
        frame.setSize(700, 600);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLocationRelativeTo(null);
        frame.setLayout(new GridBagLayout());

        frame.setVisible(true);



        GraphicsPanel graphicsPanel = new GraphicsPanel(this);

        frame.add(graphicsPanel, new GridBagConstraints(0, 0, 1, 1, 1, 1,
GridBagConstraints.CENTER,
                GridBagConstraints.BOTH, new Insets(0, 0, 0, 0), 0, 0));
        frame.setVisible(true);
```

```
    }

}
```

Клас MenuPage

```
package procesing;

import service.EstService;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.List;

public class MenuPage {
    private ImageIcon image;
    private JFrame frame;
    private JLabel label;

    private JComboBox comboBox, comboBox2;


    public MenuPage() {
//        image = new ImageIcon(getClass().getResource("1.jpg"));
    }

    public void createWindow() {
        frame = new JFrame();
        frame.setTitle("Прогнозування");
        frame.setSize(400, 300);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLocationRelativeTo(null);
```

```
        Font font1 = new Font("TimesRoman", Font.BOLD, 14);
        label = new JLabel();
        label.setLayout(new GridBagLayout());
        label.setVisible(true);
        label.setBackground(Color.BLUE);
        label.setIcon(image);
        String[] arr = new String[]{"Травень 2022", "Червень 2022", "Липень 2022", "Серпень 2022",
"Вересень 2022", "Жовтень 2022"};

        comboBox = new JComboBox(arr);
        comboBox.setFont(font1);
        label.add(comboBox, new GridBagConstraints(0, 0, 2, 1, 0.0, 0.9,
GridBagConstraints.CENTER,
                GridBagConstraints.HORIZONTAL, new Insets(30, 10, 10, 10), 0, 0));

        String[] arr2 = new String[]{"Келецька 30", "Київська 65", "Київська 126", "Юності 46"};
        comboBox2 = new JComboBox(arr2);
        comboBox2.setFont(font1);
        label.add(comboBox2, new GridBagConstraints(0, 1, 2, 1, 0.0, 0.9,
GridBagConstraints.CENTER,
                GridBagConstraints.HORIZONTAL, new Insets(30, 10, 10, 10), 0, 0));


        JButton PrognBtn = new JButton();
        PrognBtn.setText("Прогнозувати");
        PrognBtn.setFont(font1);
        label.add(PrognBtn, new GridBagConstraints(0, 2, 2, 1, 0.0, 0.9, GridBagConstraints.CENTER,
                GridBagConstraints.HORIZONTAL, new Insets(30, 10, 10, 10), 0, 0));

        PrognBtn.addActionListener(new PrListener());

        frame.add(label);
        frame.setVisible(true);

    }
```

```
    public void closeWindow() {
        frame.setVisible(false);
        frame.remove(label);

    }

    private class PrListener implements ActionListener {

        @Override
        public void actionPerformed(ActionEvent JCom) {
            int selectedIndex = comboBox.getSelectedIndex();
            int selectedIndex1 = comboBox2.getSelectedIndex();
            List<Integer> dates = new EstService().getData(selectedIndex, selectedIndex1);
                new GraphForFirst(dates).createWindow();
            }
        }
}
```

Клас GraphicsPanel

```
package painting;


import procesing.GraphForFirst;

import javax.swing.*;
import java.awt.*;

public class GraphicsPanel extends JPanel {
    public static Image image;
    private GraphForFirst graphForFirst;
```

```
private Line line;

public GraphicsPanel(GraphForFirst graphForFirst) {
    this.graphForFirst = graphForFirst;
    line = new Line(25, 100, 30, 20);
}

@Override
public void paintComponent(Graphics g) {
    Graphics2D g2 = (Graphics2D) g;
    // g2.draw(line);
    Color newColor = new Color(139, 0, 0);
    g2.setStroke(new BasicStroke(5.0f));
    g2.setColor(newColor);
    g2.drawLine(30, 530, 650, 530);
    g2.drawLine(30, 530, 30, 30);



    Font font1 = new Font("TimesRoman", Font.BOLD, 14);
    Font font = new Font("TimesRoman", Font.BOLD, 15);
    Font font2 = new Font("TimesRoman", Font.BOLD, 14);
    g2.setFont(font2);
    g2.setFont(font1);
    g2.setFont(font2);
    g2.drawString("200", 0, 435);
    g2.drawString("250", 0, 335);
    g2.drawString("300", 0, 235);
    g2.drawString("350", 0, 135);

    g2.drawString("6", 145, 550);
    g2.drawString("12", 260, 550);
    g2.drawString("18", 380, 550);
    g2.drawString("24", 500, 550);
```

```
        g2.drawString("30", 620, 550);

        g2.setStroke(new BasicStroke(1.0f));
        Color newColor1 = new Color(139, 0, 0, 50);
        g2.setColor(newColor1);
        g2.drawLine(30, 430, 650, 430);
        g2.drawLine(30, 330, 650, 330);
        g2.drawLine(30, 230, 650, 230);
        g2.drawLine(30, 130, 650, 130);

        g2.drawLine(150, 530, 150, 30);
        g2.drawLine(270, 530, 270, 30);
        g2.drawLine(390, 530, 390, 30);
        g2.drawLine(510, 530, 510, 30);
        g2.drawLine(630, 530, 630, 30);
        g2.setStroke(new BasicStroke(2.5f));
        g2.setColor(Color.BLUE);
        for (int i = 0; i < graphForFirst.list.size()-1; i++) {
            int y1 = (int) ((graphForFirst.list.get(i)));
            int y2 = (int) ((graphForFirst.list.get(i+1)));
            g2.drawLine(30+i*21, 530-y1, 50+i*21, 530-y2);

        }
//        g2.drawLine(30, 390, 750, 390);
    }
}
```

Клас Line

```
package painting;

import java.awt.geom.Line2D;
import java.awt.geom.Point2D;
import java.awt.geom.Rectangle2D;
```

```
public class Line extends Line2D{

    private Point p1;
    private Point p2;

    public Line(){
        p1 = new Point();
        p2 = new Point();
    }

    public Line(double x1, double y1, double x2, double y2){
        p1 = new Point(x1, y1);
        p2 = new Point(x2, y2);
    }

    @Override
    public Rectangle2D getBounds2D() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public Point2D getP1() {
        // TODO Auto-generated method stub
        return p1;
    }

    @Override
    public Point2D getP2() {
        // TODO Auto-generated method stub
        return p2;
    }

    @Override
    public double getX1() {
```

```
        // TODO Auto-generated method stub
        return p1.getX();
    }

    @Override
    public double getX2() {
        // TODO Auto-generated method stub
        return p2.getX();
    }

    @Override
    public double getY1() {
        // TODO Auto-generated method stub
        return p1.getY();
    }

    @Override
    public double getY2() {
        // TODO Auto-generated method stub
        return p2.getY();
    }

    @Override
    public void setLine(double x1, double y1, double x2, double y2) {
        // TODO Auto-generated method stub
        p1.setLocation(x1, y1);
        p2.setLocation(x2, y2);

    }

}
```

Клас Point

```
package painting;
```

```
import java.awt.geom.Point2D;

public class Point extends Point2D{
    private double x;
    private double y;

    public Point() {
        // TODO Auto-generated constructor stub
    }

    public Point(double x, double y) {
        this.x=x;
        this.y=y;
    }

    @Override
    public double getX() {
        // TODO Auto-generated method stub
        return x;
    }

    @Override
    public double getY() {
        // TODO Auto-generated method stub
        return y;
    }

    @Override
    public void setLocation(double arg0, double arg1) {
        // TODO Auto-generated method stub
        this.x=x;
        this.y=y;
    }

}
```

Клас DAO

```
package db;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;

public class DAO {

   private static final String mySQL = "SELECT * FROM stamp where dat > '2021-02-21' ORDER BY dat";

   public ArrayList<Integer> getAllDate() {
      ArrayList<Integer> res = new ArrayList<>();
      PreparedStatement pstmt = null;
      ResultSet rs = null;
      Connection con = null;
      try {
         con = DBManager.getInstance().getConnection();
         pstmt = con.prepareStatement(mySQL);
         rs = pstmt.executeQuery();
         while (rs.next()) {
            res.add(rs.getInt("cours"));
         }
      } catch (SQLException throwables) {
         throwables.printStackTrace();
      } finally {
         close(rs);
         close(pstmt);
         close(con);
      }
      return res;
```

```
    }



    private void close(AutoCloseable forClose) {
        if (forClose != null) {
            try {
                forClose.close();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }
}
```

Клас DBManager

```
package db;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class DBManager {

    private static DBManager instance;
    private static final String URL = "jdbc:mysql://localhost:3306/pdr";
    private static final String USERNAME = "root";
    private static final String PASSWORD = "root";

    public static synchronized DBManager getInstance() {
        if (instance == null)
            instance = new DBManager();
        return instance;
```

```
    }

    private DBManager() {
    }


    public Connection getConnection() throws SQLException {
        Connection con = null;
        try {
            con = DriverManager.getConnection(URL, USERNAME, PASSWORD);
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return con;
    }

    public void commitAndClose(Connection con) {
        try {
            if (con != null) {
                con.commit();
                con.close();
            }
        } catch (SQLException throwables) {
            throwables.printStackTrace();
        }
    }
}
```

Клас ResultCache

```
package ceche;


import java.util.ArrayList;
import java.util.List;
```

```
public class ResultCache {
    public static final List<Integer> k30 = new ArrayList<>();

    public void initK30(){

    }
}
```

Додаток В (обов'язковий)

ІЛЮСТРАТИВНА ЧАСТИНА

«ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ПРОГНОЗУВАННЯ КІЛЬКОСТІ ПОРУШЕНЬ ПРАВИЛ ДОРОЖНЬОГО РУХУ»

Виконав: студент 2-го курсу, групи 1КН-22м
спеціальності 122 – «Комп`ютерні науки»
_____ Львовський О. О.
(прізвище та ініціали)

Керівник: д.т.н., доц. каф. КН
_____ Крилик Л. В.
(прізвище та ініціали)
«07» 12 2023 р.

Вінниця ВНТУ - 2023 рік

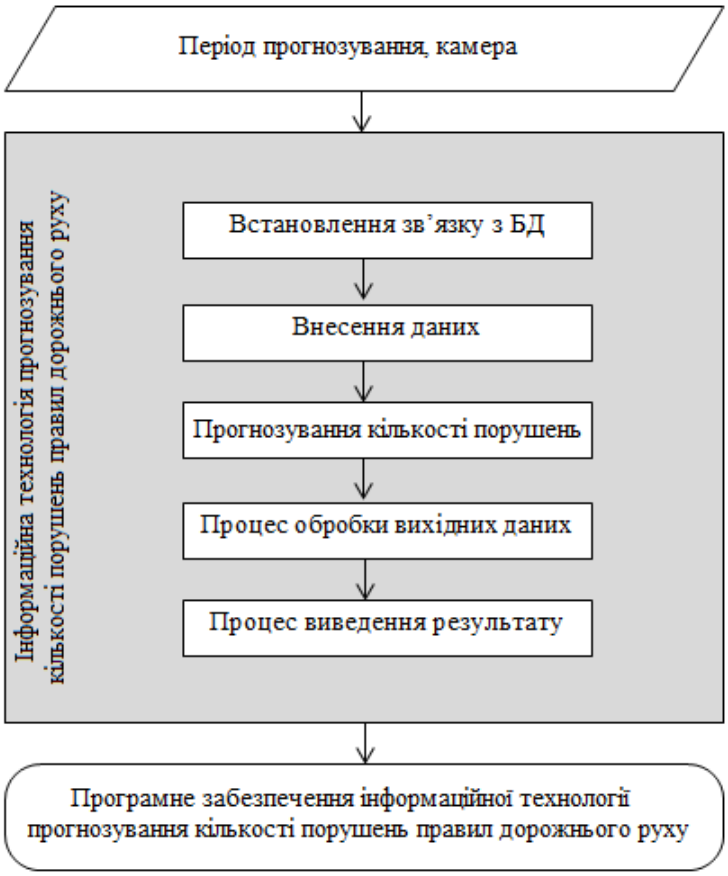

Рисунок В.1 − Структурна схема інформаційної технології

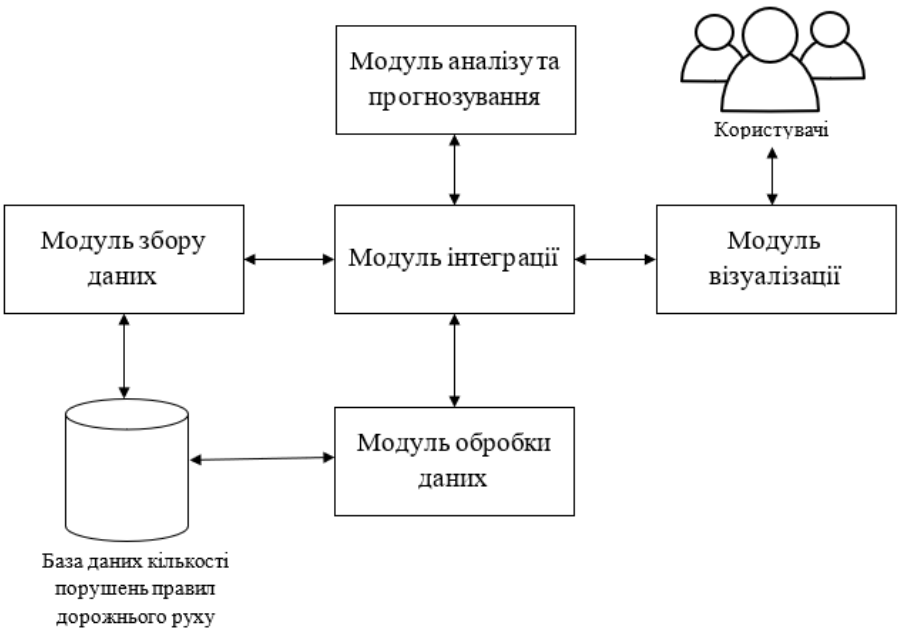

Рисунок В.2 − Загальна структурна схема

Рисунок В.3– ER-діаграма порушень правил дорожнього руху

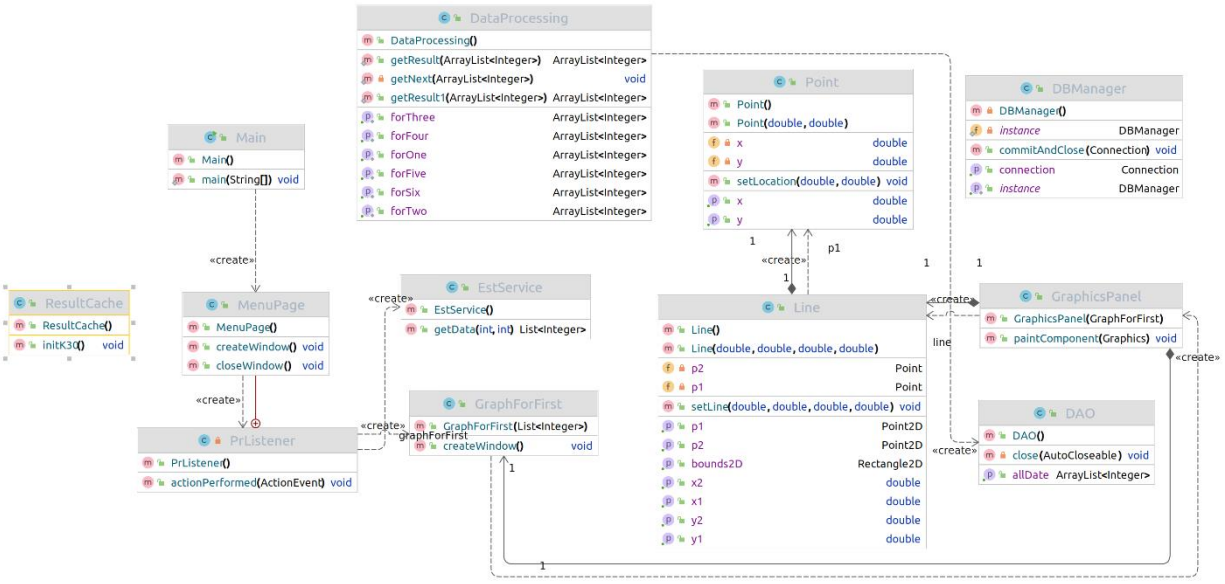

Рисунок В.4 − UML-діаграма класів

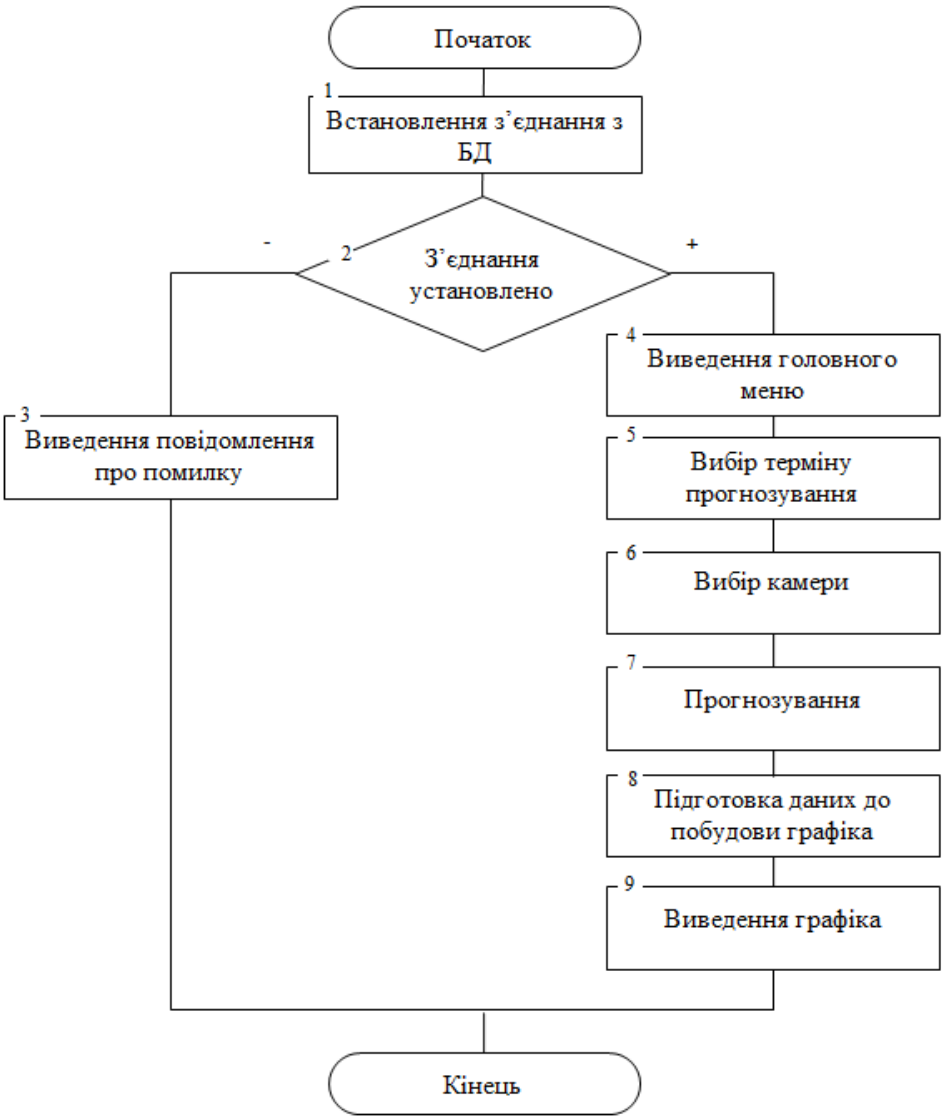

Рисунок В.5 − Схема алгоритму функціонування модуля прогнозування інформаційної технології прогнозування кількості порушень правил дорожнього руху

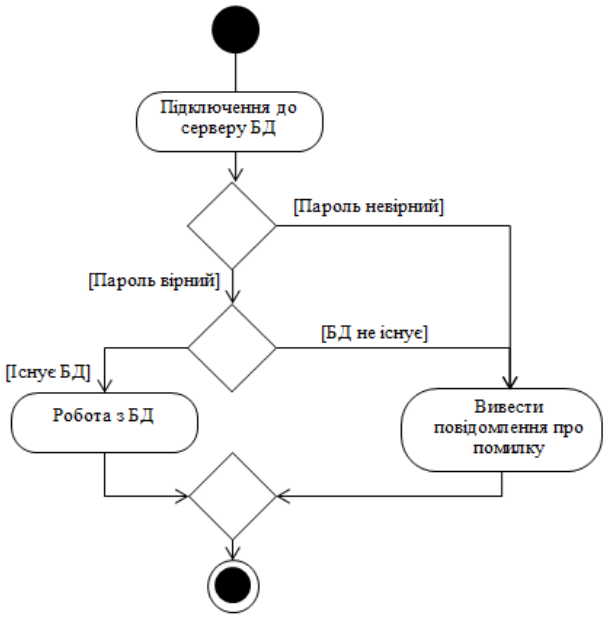

Рисунок В.6 − Діаграма діяльності системи при встановленні з’єднання з базою даних

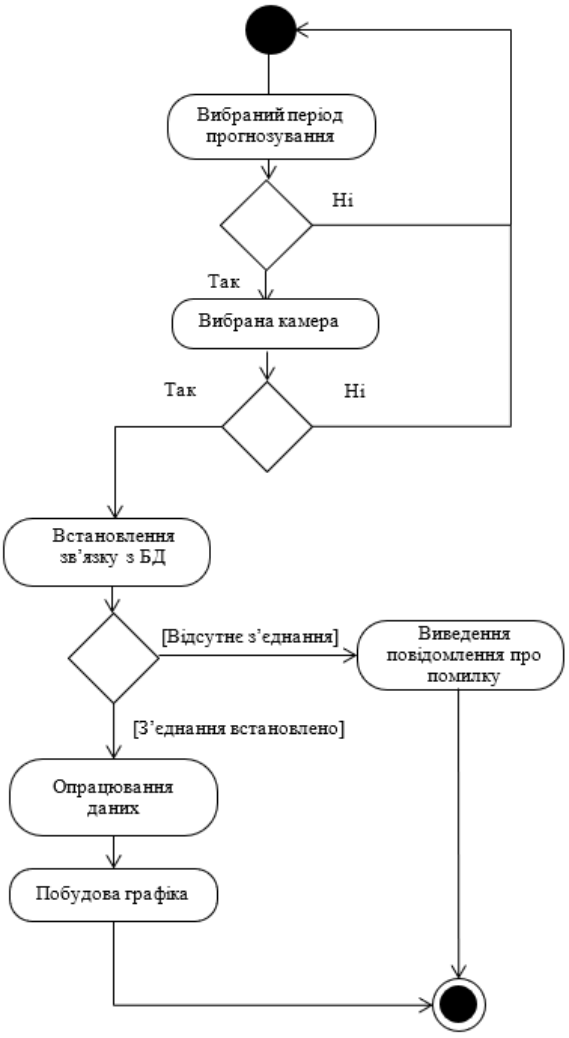

Рисунок В.7 − Діаграма діяльності сервера для побудови графіка

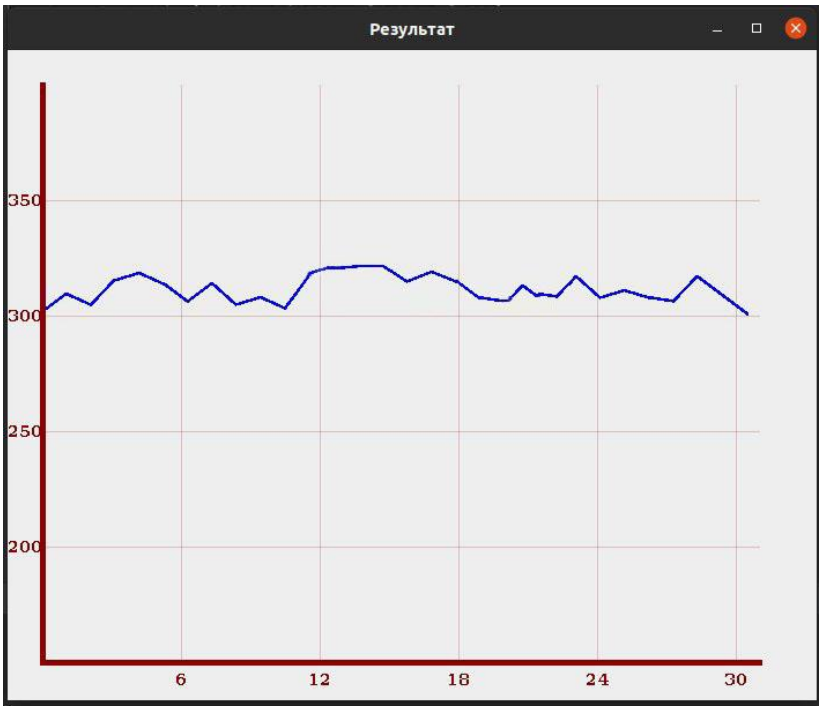

Рисунок В.8 − Приклади роботи програми

Додаток Г (довідниковий)

Інструкція користувача

Після запуску програми відкривається вікно початкової активності з двома випадаючими меню: перше для введення місяця, на який потрібно зробити прогноз, а друге для вибору камери, розташованої на певній локації. Також вікно початкової активності містить кнопку «Прогнозувати» (рис. Г.1).

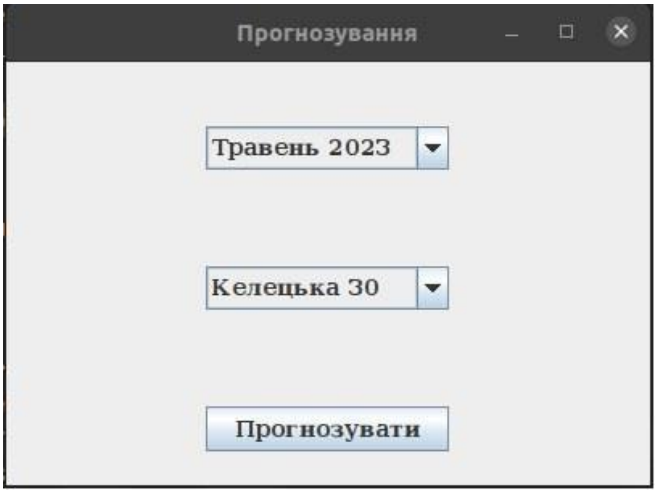

Рисунок Г.1 – Вікно початкової активності

Користувач вводить період прогнозування та камеру, після чого натискає кнопку «Прогнозувати» (рис. Г.2).

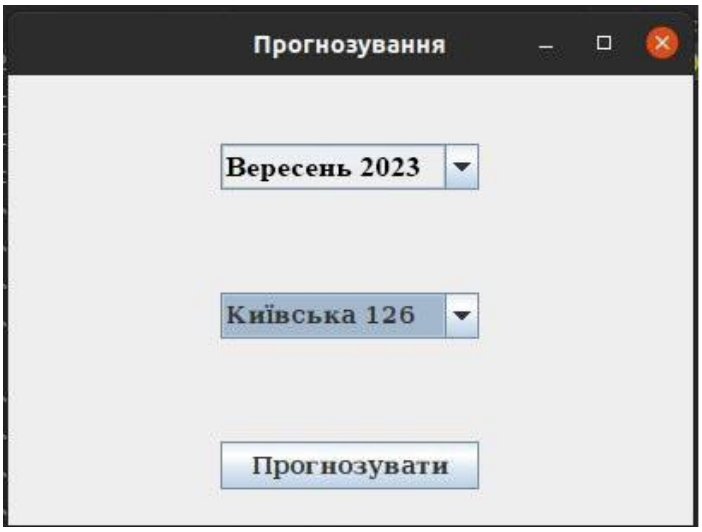

Рисунок Г.2 – Введення параметрів прогнозування

Після цього здійснюється перехід до вікна побудови прогнозу, який відбувається за допомогою виведення графіку прогнозованої кількості порушень правил дорожнього руху (рис. Г.3, Г.4).

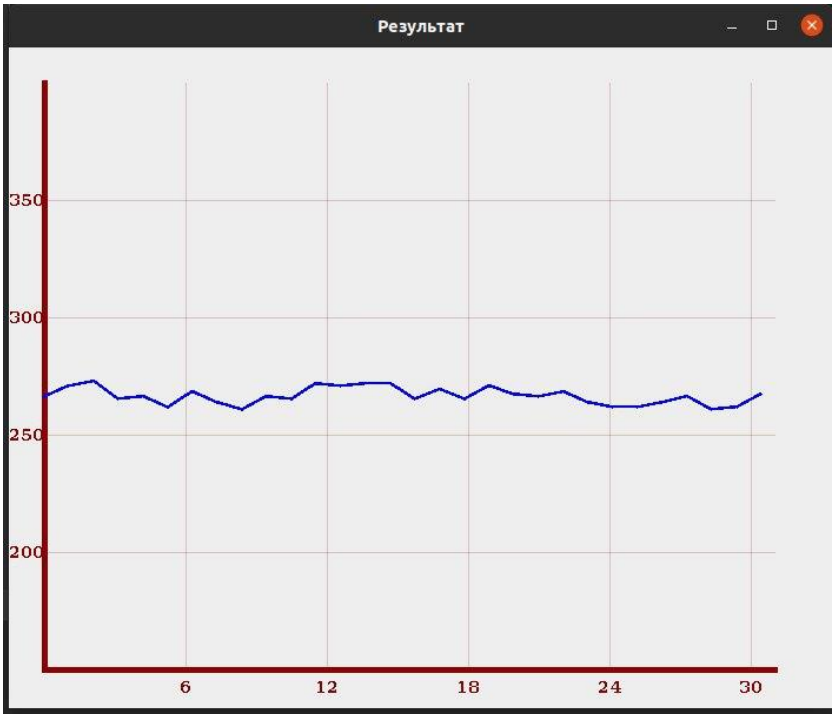

Рисунок Г.3 – Прогнозування кількості порушень

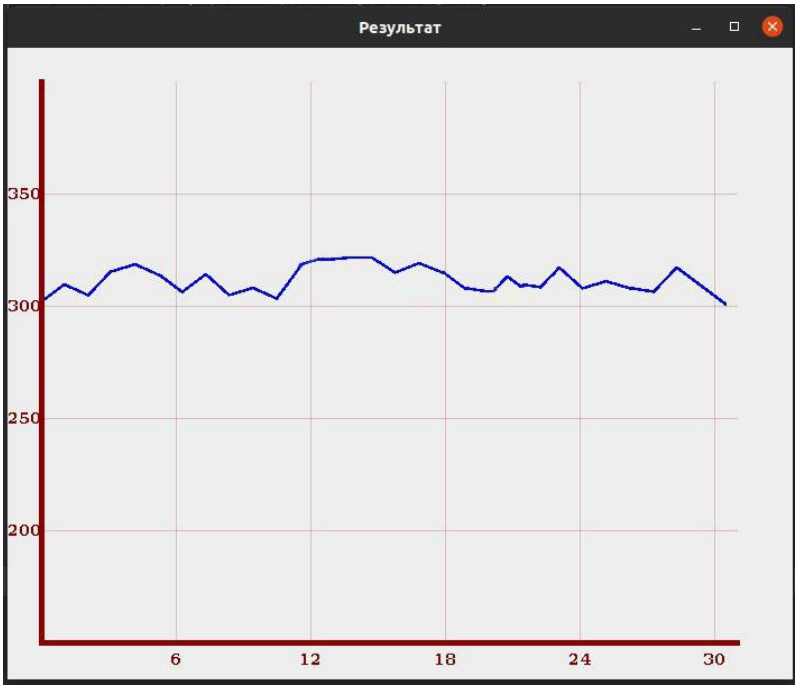

Рисунок Г.4 – Прогнозування кількості порушень