Державний торговельно-економічний університет

Кафедра комп'ютерних наук та інформаційних систем

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Методи та засоби побудови комп'ютеризованих діалогових систем торговельного центру»

Студента 2 курсу, 4м групи
спеціальності
122 «Комп'ютерні науки» _____ *підпис студента* Устименко Назар Ігорович

Науковий керівник
кандидат педагогічних наук, доцент _____ *підпис керівника* Дивак Володимир Валерійович

Гарант освітньої програми
доктор фізико-математичних наук, професор _____ *підпис керівника* Пурський Олег Іванович

Київ 2023

Державний торговельно-економічний університет

Факультет інформаційних технологій
Кафедра комп'ютерних наук та інформаційних систем
Спеціальність 122 «Комп'ютерні науки»
Освітня програма «Комп'ютерні науки»

Затверджую
Зав. кафедри _____Пурський О.І.
«9» грудня 2022р.

Завдання
на випускну кваліфікаційну роботу студенту

Устименку Назару Ігоровичу
(прізвище, ім'я, по батькові)

1. Тема випускної кваліфікаційної роботи
«Методи та засоби побудови комп'ютеризованих діалогових систем торговельного центру»
Затверджена наказом ректора від *«06» грудня 2022 р. № 3284*

2. Строк здачі студентом закінченої роботи *24 листопада 2023 року*

3. Цільова установка та вихідні дані до роботи
Мета роботи: дослідити методи та засоби побудови комп'ютеризованих діалогових систем торговельного центру та навести наочний приклад.
Об'єкт дослідження: об'єктом дослідження є самий процес створення, а також методи та засоби побудови комп'ютеризованих діалогових систем торговельного центру.
Предмет дослідження: предметом дослідження є реалізація комп'ютеризованої діалогової системи.

4. Перелік графічного матеріалу_____
_____
_____

5. Консультанти по роботі із зазначенням розділів, за якими здійснюється консультування:

Розділ	Консультант (прізвище, ініціали)	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Пурський О.І.	09.12.2022 р.	09.12.2022 р.
2	Пурський О.І..	09.12.2022 р.	09.12.2022 р.
3	Пурський О.І.	09.12.2021 р.	09.12.2022 р.

6. Зміст випускного кваліфікаційної роботи (перелік питань за кожним розділом)

ВСТУП

РОЗДІЛ 1. ТЕОРЕТИЧНІ АСПЕКТИ ОЦІНКИ КОНКУРЕНТОСПРОМОЖНОСТІ ПІДПРИЄМСТВА

1.1. Аналіз проблематики та існуючих методів управління конкурентоспроможністю

1.2. Особливості оцінки та управління конкурентоспроможністю підприємств електронної комерції

1.3. Концептуальна модель оцінки та управління конкурентоспроможністю підприємств електронної комерції

РОЗДІЛ 2. МАТЕМАТИЧНІ МОДЕЛІ ОЦІНКИ ТА УПРАВЛІННЯ КОНКУРЕНТОСПРОМОЖНІСТЮ ПІДПРИЄМСТВА

2.1. Система показників та модель оцінки конкурентоспроможності підприємств електронної комерції

2.2. Модель управління конкурентоспроможністю підприємства

2.3. Моделювання процесу оцінки конкурентоспроможності підприємства

РОЗДІЛ 3. ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ОЦІНКИ КОНКУРЕНТОСПРОМОЖНОСТІ ПІДПРИЄМСТВ ЕЛЕКТРОННОЇ ТОРГІВЛІ

3.1. Інформаційно-логічна модель системи оцінки конкурентоспроможності підприємств

3.2. Специфіка програмно-апаратної реалізації інформаційної системи оцінки конкурентоспроможності підприємств

3.3.Технологія використання інформаційної системи оцінки конкурентоспроможності підприємств електронної торгівлі

ВИСНОВКИ

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

7. Календарний план виконання роботи

№ Пор.	Назва етапів випускної кваліфікаційної роботи	Строк виконання етапів роботи	
		За планом	фактично
1	2	3	4
1	*Вибір теми випускної кваліфікаційної роботи*	*01.11.2022*	*01.11.2022*
2	*Розробка та затвердження завдання на випускну кваліфікаційну роботу*	*09.12.2022*	*09.12.2022*
3	*Вступ*	*01.05.2023*	*01.05.2023*
4	*РОЗДІЛ 1. Теоретичні аспекти оцінки конкурентоспроможності підприємства*	*14.06.2023*	*14.06.2023*
5	*Підготовка статті у збірник наукових статей магістрів*	*20.06.2023*	*20.06.2023*
6	*РОЗДІЛ 2. Математичні моделі оцінки та управління конкурентоспроможністю підприємства*	*08.09.2023*	*08.09.2023*
7	*РОЗДІЛ 3. Інформаційна технологія оцінки конкурентоспроможності підприємств електронної торгівлі*	*20.10.2023*	*20.10.2023*
8	*Висновки*	*02.11.2023*	*02.11.2023*
9	*Здача випускної кваліфікаційної роботи на кафедру науковому керівнику*	*22.11.2023*	*22.11.2023*
10	*Попередній захист випускної кваліфікаційної роботи*	*29.11.2023*	*29.11.2023*
11	*Виправлення зауважень, зовнішнє рецензування випускної кваліфікаційної роботи*	*04.12.2023*	*04.12.2023*
12	*Представлення готової зшитої випускної кваліфікаційної роботи на кафедру*	*06.12.2023*	*06.12.2023*
13	*Публічний захист випускної кваліфікаційної роботи*	*За розкладом роботи ЕК*	

8. Дата видачі завдання «9» грудня 2022 р.

9. Керівник випускного кваліфікаційного проєкту Дивак В.В.
(прізвище, ініціали, підпис)

10. Гарант освітньої програми Пурський О.І.
(прізвище, ініціали, підпис)

11. Завдання прийняв до виконання студент УстименкоН.І.
(прізвище, ініціали, підпис)

12. Відгук керівника випускної кваліфікаційної роботи

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

Керівник випускної кваліфікаційної роботи

_____

(підпис, дата)

13. Висновок про випускну кваліфікаційну роботу

Випускна кваліфікаційна робота студента _____

(прізвище, ініціали)

може бути допущена до захисту в екзаменаційній комісії.

Гарант освітньої програми_____ Пурський О.І.

(підпис, прізвище, ініціали)

Завідувач кафедри _____ Пурський О.І.

(підпис, прізвище, ініціали)

«_____»_____2023 р.

Анотація

Випускна кваліфікаційна робота присвячена дослідженню та розробці методів і засобів створення комп'ютеризованих діалогових систем для торговельних центрів. Основна мета дослідження полягає у вивченні сучасних підходів до побудови ефективних та зручних систем взаємодії між клієнтами та торговельними приміщеннями.

Робота розглядає та порівнює різноманітні методи реалізації діалогових систем, зокрема застосування штучного інтелекту, обробки природної мови та технологій голосового управління. Проведено аналіз попередніх досліджень у цьому напрямку та визначено переваги та недоліки різних підходів.

В рамках роботи розроблено прототип комп'ютеризованої діалогової системи для торговельного центру, яка використовує передові технології для покращення взаємодії з відвідувачами. Здійснено тестування та оцінку роботи прототипу з використанням реальних вхідних даних та зіставлення результатів з очікуваними показниками.

Отримані результати свідчать про потенційну ефективність використання комп'ютеризованих діалогових систем у торговельних центрах для поліпшення обслуговування клієнтів та забезпечення їм більш зручного та інтерактивного досвіду покупок. Дана дипломна робота може бути використана як вихідний пункт для подальших досліджень у галузі розвитку та оптимізації діалогових систем у торговельних центрах.

Anotation

The graduation qualification work is devoted to the research and development of methods and means of creating computerized dialog systems for shopping centers. The main goal of the study is to study modern approaches to building effective and convenient systems of interaction between customers and retail premises.

The work examines and compares various methods of implementing dialogue systems, in particular, the use of artificial intelligence, natural language processing, and voice control technologies. An analysis of previous research in this direction was carried out and the advantages and disadvantages of various approaches were determined.

As part of the work, a prototype of a computerized dialog system for a shopping center was developed, which uses advanced technologies to improve interaction with visitors. The prototype was tested and evaluated using real input data and comparing the results with the expected indicators.

The obtained results indicate the potential effectiveness of using computerized dialog systems in shopping centers to improve customer service and provide them with a more convenient and interactive shopping experience. This thesis can be used as a starting point for further research in the field of development and optimization of dialog systems in shopping centers.

ЗМІСТ

ВСТУП

Актуальність теми дослідження: Зростання ролі технологій та їх широкий доступ призвели до стрімкого розвитку інтернет-комерції. Приблизно 63% населення України користується інтернетом, а 33% з них здійснюють покупки принаймні один раз на рік. Однак не все можна обмежити онлайн-продажами. Згідно з аудитом компанії Deloitte, з 50 найшвидше зростаючих магазинів чи маркетплейсів у світі, лише 8% належать магазинам без (або з обмеженою кількістю) фізичних точок. Великі торгові центри стикаються з проблемою розсіяння магазинів по всій території без системності. Це призводить до того, що покупці часто витрачають години на пошук потрібного товару, але так і не знаходять його, що зменшує їхнє задоволення від покупок і призводить до меншої ймовірності повернення в торговий центр.

Мета і завдання дослідження. Метою даної роботи є створення комп'ютеризованого діалогового вікна, яке сприятиме покупцям у знаходженні потрібних товарів у торгових центрах та надаватиме всю необхідну інформацію про товари та магазини. Основні завдання включають аналіз методів машинного навчання та розробку алгоритмів для ефективного визначення потреб покупців. Для досягнення поставленої мети необхідно було вирішити наступні **завдання:**

- провести комплексне аналітичне дослідження методів та засобів побудови комп’ютеризованих діалогових систем;
- дослідити методи які використовують торговельні центри для приваблення своїх покупців;
- визначити принципи формування системи діалогових вікон для спілкування з покупцем;
- Розробити комп’ютеризовану діалогову систему торговельного

центру, опираючись на результати досліджень і аналізу ринку та засобів для побудови діалогових систем.

Об'єкт дослідження: Об'єктом дослідження є процеси розробки комп'ютеризованих жіалогових систем торговельного центру.

Предмет дослідження: методи та засоби які використовуються для створення діалогових систем.

Методи дослідження: Для досягнення поставлених завдань використовуються аналітичний, економіко-статистичний та теоретико-емпіричний методи дослідження.

Наукова новизна одержаних результатів полягає в розробці рекомендаційного діалогового вікна для торговаог центру для комунікації з покупцем використовуючи новітні методи та засоби .

Практичне значення. Отримані результати, можуть бути використані керівниками торгових центрів для побудови власної двалогової системи дял комунікації з покупцями. Розроблене програмне забезпечення дозволяє покупцям швидко оцінити та аналізувати пропозиції продавців, знаходячи потрібний товар. Для продавців це є додатковим інструментом для просування товарів та виділення серед конкурентів, а загалом, має збільшити трафік у торгових центрах та оптимізувати роботу окремих магазинів.

Публікації. Результати дослідження опубліковано у збірнику наукових статей студентів, які здобувають освітній ступінь магістра за спеціалізацією «Комп'ютерні науки» КНТЕУ на тему: «Методи та засоби побудови комп'ютеризованих діалогових систем торговельного центру», 2023 р.

Структура та обсяг випускної кваліфікаційної роботи. Випускна кваліфікаційна робота складається із вступу, трьох розділів, висновків, списку використаних джерел із 38 найменувань, додатків і містить 43 сторінки основного тексту, 5 рисунків і 4 таблиці.

РОЗДІЛ 1.
ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ СТВОРЕННЯ ДІАЛОГОВОЇ СИСТЕМИ

1.1. Огляд предметної області

Проблематика, пов'язана з вибором товару в торговому центрі, глибоко корінитьсяся в сучасному споживачівському оточенні. Велика кількість різноманітних товарів ускладнює процес вибору для покупців, а той факт, що це може зайняти весь день, робить цю проблему особливо актуальною. Торгові центри, розкидані на великих територіях, нерідко стають справжнім лабіринтом, де покупці втрачаються серед безлічі магазинів.

Недостатньою інформаційною підтримкою також стає фактор великої кількості покупців. Продавці-консультанти не завжди в змозі забезпечити вичерпну інформацію всім відвідувачам. Це призводить до заторів та невдоволення покупців, які витрачають значний час на очікування обслуговування.

Іншою стороною медалі стає загальний розвиток технологій та популярність онлайн-торгівлі. Інтернет став невід'ємною частиною життя, а 63% населення України користується ним. Особливо це стосується покупок: 33% вже здійснюють їх принаймні один раз на рік. Вибір інтернет-магазинів обумовлений такими факторами, як швидкість обслуговування, широкий асортимент, можливість порівняти ціни та якість, зручність в користуванні, якість обслуговування та конкурентоспроможні ціни.

Незважаючи на це, фізична присутність у торгових точках лишається невід'ємною для успішного бізнесу. За даними Deloitte, тільки 8% з 50 найшвидше зростаючих магазинів світу відсутні фізично. Таким чином, сучасна торгівля вимагає збалансованого підходу, де фізична та онлайн-присутність взаємодіють для забезпечення повного та задовільного обслуговування споживачів.

1.2. Мета розробки та перспективи використання

В ході розробки комп’ютеризованої діалогової системи, яка спрямована на торгові центри, основна мета полягає в створенні не просто зручного, але й винятково ефективного інструменту для покупців. Ця інноваційна система прагне надати не лише швидкий, але й інтуїтивно зрозумілий доступ до широкого спектру інформації. Ця інформація охоплює не лише характеристики товарів, а й повні дані про магазини, які реалізують ці товари, а також загальні відомості про сам торговий центр.

Система, яку ми розробляємо, має стати не просто посередником між покупцем, що завітав до торгового центру, і магазинами, розташованими в його межах, але й справжньою помічницею. Завдяки системі організованого виведення товарів покупці зможуть швидко оцінити різноманітні варіації, що пропонуються в різних магазинах. Після вибору оптимального товару покупець з легкістю та ефективністю може знайти не лише магазин, а й конкретний прилавок для здійснення покупки.

Зокрема, великий акцент приділяється системі відгуків, яка спрямована на створення прозорого середовища для покупців. Тут споживачі можуть ділитися враженнями не тільки про вибрані товари, але і про якість обслуговування в конкретних магазинах. Більше того, система надає можливість споживачам використовувати різноманітні фільтри та сортування для товарів за різними параметрами, включаючи ціну та рейтинг.

Такий комплексний підхід до впровадження інноваційної системи ставить за мету не лише стимулювання конкуренції між магазинами, але і активізацію продавців, щоб вони були уважніші до конкурентоспроможності цін, якості товарів та рівня обслуговування. В такому конкурентному середовищі покупцеві буде приємніше та зручніше робити свої покупки в межах торгового центру. Усі аспекти цієї інноваційної системи націлені на подальше поліпшення загального досвіду покупців та оптимізацію процесу вибору товарів у торговому центрі.

Цей підхід забезпечить не лише інноваційні можливості для покупців та магазинів у торгових центрах, але й стане кроком до покращення загального рівня обслуговування та задоволення від покупок в цих просторах.

В розробці даного продукту важливо враховувати сучасні тенденції розвитку інтернет-технологій та зміни у споживчих уподобаннях. Зростання використання Інтернету для покупок створює нові виклики та можливості для торгових центрів. Швидкий розвиток технологій змінює спосіб, яким покупці шукають та обирають товари. Тому розробка комп'ютеризованої діалогової системи, спрямованої на полегшення цього процесу, є належним відповіддю на виклики сучасного ринку.

Поширення Інтернету в Україні та інших країнах світу робить його важливим каналом для здійснення покупок. За даними статистики, більше половини населення України використовує Інтернет, а третина з них здійснює покупки хоча б раз на рік. Проте, несмотря на зростання інтернет-торгівлі, фізичні торгові центри залишаються актуальними. Важливо враховувати це у розробці діалогової системи, яка сприятиме не лише онлайн-покупкам, але й полегшить навігацію та пошук у традиційних торгових просторах.

Однією з ключових проблем для покупців у фізичних торгових центрах є складність вибору серед великого асортименту товарів та розміщення

магазинів у великих просторах. Розробка діалогової системи, яка допомагатиме покупцям швидко знаходити необхідний товар та отримувати інформацію про нього, має великий потенціал у поліпшенні користувацького досвіду.

Мета нашого дослідження полягає у створенні діалогової системи, яка буде функціонувати в торгових центрах, допомагаючи покупцям знаходити потрібний товар та надавати інформацію про магазини та продукцію. Для досягнення цієї мети, ми використовуємо алгоритми машинного навчання, зокрема алгоритм k-найближчих сусідів, для визначення потреб покупців та підбору оптимальних варіантів товарів.

Аналітичні, економіко-статистичні та теоретико-емпіричні методи дослідження дозволяють нам отримати глибоке розуміння та аналіз сучасних тенденцій у споживчому поведінці та економіці торгівлі. Результати нашої роботи були представлені та апробовані на міжнародних конференціях, що свідчить про важливість та актуальність наших досліджень у галузі розвитку технологій.

Практичне значення отриманих результатів полягає в тому, що наша діалогова система дозволяє покупцям ефективно взаємодіяти з торговим центром, швидко знаходячи необхідний магазин та отримуючи докладну інформацію про нього. Для продавців це створює нові можливості для просування товарів та підвищення конкурентоспроможності. Загалом, наше програмне забезпечення має сприяти підвищенню трафіку в торгових центрах та оптимізації роботи окремих магазинів.

У структурі роботи важливо виділити розрахунково-пояснювальну записку та графічну частину. Розрахунково-пояснювальна записка включає в себе вступ, шість частин, висновки, перелік посилань та додатків.

Наш підхід до розробки комп'ютеризованої діалогової системи для торгових центрів має значущий потенціал у покращенні ефективності покупок та оптимізації досвіду від відвідування торгових центрів. Враховуючи різноманітні виклики, що стоять перед споживачами та продавцями у торговельній сфері, наша розробка може виявитися ключовим інструментом для досягнення успіху в умовах швидко змінюючогося ринку та технологічного прогресу.

1.3. Експлорація алгоритмів впровадження системи тестування для клієнтів.

Останній десятиліток характеризувалося стрімким розвитком технологій, що вплинули на всі сфери людського життя. Відокремившись від традиційних форм покупок, інтернет-комерція завойовує світ. В Україні цей тренд виявився особливо виразним, де приблизно 63% населення користуються мережею Інтернет, а 33% роблять покупки принаймні раз на рік. Це є індикатором того, що інтернет-простір швидко стає основним каналом для задоволення потреб споживачів.

Проте, навіть при такому стрімкому розвитку онлайн-торгівлі, не всі сфери можуть повністю замінити фізичний взаємодію. За даними Deloitte, тільки 8% з 50 найшвидше зростаючих торгових точок в світі існують виключно в онлайн-середовищі без фізичних представництв.

Однією з проблем, з якою стикаються великі торгові центри, є безлад у розміщенні магазинів, який призводить до втрат часу покупців, що блукають безсистемно в пошуку необхідних товарів. Це стимулює їх вдачу в онлайн-шопінг, де вони можуть ефективніше та зручніше знаходити потрібне.

З метою вирішення цих труднощів було проведено дослідження та розроблено комп'ютеризовану діалогову систему. Її мета — надавати покупцям швидкий та легкий доступ до інформації про товари та магазини у торговому центрі. Застосовуючи алгоритм k-найближчих сусідів, система допомагає класифікувати товари та спрощує пошук для споживачів.

Загальний внесок цієї роботи у сферу роздрібної торгівлі полягає в створенні додаткового інструменту для ефективного вибору товарів покупцями та покращенні маркетингових можливостей для продавців. Розроблене програмне забезпечення сприятиме збільшенню трафіку та оптимізації роботи торгових центрів.

Тим не менш, варто відзначити, що в теперішньому економічному контексті важливо наводити новаторські рішення для забезпечення конкурентоспроможності, і розробка таких систем відіграє важливу роль у модернізації торговельного взаємодії.

РОЗДІЛ 2.
ВИБІР ОПТИМАЛЬНИХ ПІДХОДІВ ДО ВПРОВАДЖЕННЯ КОМП'ЮТЕРИЗОВАНИХ ДІАЛОГОВИХ СИСТЕМ У ТОРГОВЕЛЬНОМУ ЦЕНТРІ

2.1. Оцінка та порівняння сучасних технологій в контексті їх застосування в торговельних центрах

У світі стрімкого технологічного розвитку, торговельні центри виявляють значний інтерес до інновацій, щоб удосконалити свій бізнес та відповідати змінюваним уявленням споживачів. Розгляд порівняння сучасних технологій у контексті їх впровадження в торговельні центри може допомогти виявити переваги та особливості кожного напрямку.

1. Інтерактивні додатки та Інтернет речей (IoT)

Переваги:

Зручність для покупців: Інтерактивні додатки надають інформацію про акції та розташування товарів, полегшуючи вибір споживачів.

Оптимізація управління: Використання IoT для зв'язку елементів торгового центру дозволяє ефективно управляти ресурсами.

Обмеження:

Безпека: Збір та обробка великої кількості даних може вивести на новий рівень питання щодо кібербезпеки.

2. Розпізнавання обличчя та аналіз даних

Переваги:

Персоналізовані послуги: Розпізнавання обличчя дозволяє надавати персоналізовані рекомендації та обслуговування.

Аналітика: За допомогою аналізу даних можливо прогнозувати попит та удосконалювати стратегії управління.

Обмеження:

Приватність: Використання таких технологій може порушити особисту приватність покупців, створюючи етичні та правові аспекти.

3. Мобільні платежі та електронні гаманці

Переваги:

Швидкість та зручність: Мобільні платежі прискорюють оплату та надають додаткову зручність споживачам.

Програми лояльності: Електронні гаманці забезпечують реалізацію програм лояльності та персоналізованих знижок.

Обмеження:

Залежність від технології: Проблеми з мережею чи відмови в програмному забезпеченні можуть вплинути на доступність цих послуг.

4. Віртуальна та Розширена Реальність

Переваги:

Ефективне тестування товарів: Віртуальна реальність дозволяє споживачам випробувати товари в онлайн-середовищі.

Захопливий досвід: Застосування AR/VR створює унікальний та захопливий досвід для покупок.

Обмеження:

Вартість інфраструктури: Впровадження AR/VR вимагає суттєвих витрат на обладнання та підтримку.

5. Аналітика та Штучний Інтелект

Переваги:

Прогнозування попиту: Аналітика та штучний інтелект допомагають визначити попит та ефективно вирішувати стратегічні завдання.

Обмеження:

Складність впровадження: Впровадження штучного інтелекту може потребувати спеціалізованих знань та інтеграції з існуючими системами.

Загалом, вибір сучасних технологій для торгових центрів залежить від конкретних завдань та стратегічних цілей бізнесу. Оптимальне поєднання різних технологій може забезпечити успіх у змінному економічному середовищі, підвищуючи конкурентоспроможність та задоволення споживачів.

Але оскільки Штучний Інтелект набирає великих обертів з виходом різних чат ботів побудованих на штучному інтелекті, я вирішив інтегрувати саме цю модель при побудові компютеризованої діалогової системи торгового центу . Ця інтеграція Штучного Інтелекту в комп'ютеризовану діалогову систему торгового центру має на меті революціонізувати взаємодію між покупцями та торговими центрами.

Штучний Інтелект у чат-ботах може значно полегшити процес вибору товарів для покупців. Завдяки алгоритмам машинного навчання, бот може аналізувати попередні покупки, вподобання та запитання користувачів для надання персоналізованих рекомендацій. Це спростить і прискорить пошук необхідного товару, забезпечуючи при цьому задоволення від покупок.

Використання Штучного Інтелекту також дозволяє розширити можливості системи взаємодії. Чат-бот може здійснювати більше функцій, таких як планування маршруту в торговому центрі, надання інформації про акції та знижки, а також спрощення процесів оплати та доставки.

Окрім того, інтеграція Штучного Інтелекту дозволяє системі навчатися від кожної взаємодії з користувачем. Чим більше вона працює, тим точніше стають її рекомендації та відповіді. Це робить систему гнучкою та адаптованою до змін у попиті та поведінці покупців.

Однак важливо враховувати етичні та конфіденційність питання при використанні Штучного Інтелекту. Необхідно забезпечити адекватний рівень захисту особистих даних користувачів та гарантувати, що система працює в межах визначених етичних стандартів.

В цілому, інтеграція Штучного Інтелекту в комп'ютеризовану діалогову систему торгового центру відкриває шлях до нових можливостей у покращенні якості обслуговування та зручності для покупців.

На сучасний момент існує розмаїття варіантів для розробки веб-додатків, включаючи різноманітні технології та інструменти. Один із таких варіантів – MERN-стек, який є високоцільовим вибором для повноцінного веб-додатка, охоплюючи весь процес розробки від бекенду до фронтенду.

Веб-програми використовують комбінацію різних технологій, яку називають "стеком". Першим популярним стеком був LAMP, що включав Linux, Apache, MySQL, PHP – всі компоненти з відкритим вихідним кодом. Однак із зростанням популярності односторінкових додатків (SPA), виникла необхідність в більш інтерактивних компонентів, що призвело до виникнення нових стеків.

SPA – це парадигма веб-додатків, що дозволяє уникати повного перезавантаження сторінки, використовуючи легкі виклики на сервер для отримання даних або фрагментів для оновлення веб-сторінки. Це призвело до зростання популярності фронтенд-фреймворків, оскільки більшість логіки перейшла на сторону клієнта.

MEAN-стек (MongoDB, Express.js, AngularJS, Node.js) став одним з перших відкритих стеків зі схильністю до SPA та використання NoSQL. Замість AngularJS у MERN-стеку використовується React – фронтенд технологія від Facebook. Таким чином, MEAN став MERN, що вказує на заміну "A" на "R".

Хоча вибір основних технологій визначає стек розробки, для повноцінної веб-програми потрібні допоміжні інструменти та бібліотеки, які полегшують процес розробки та розширюють можливості основних технологій, зокрема React.

2.2 Засоби, моделі і розрахунки необхідні для побудови компютеризованої діалогові систем торгового центру

На даний момент у сфері розробки веб-додатків існує велика різноманітність підходів, що включають як широкий спектр технологій та

рівнів, так і різноманітні інструменти, які спрощують процес розробки. Одним з таких методів є використання MERN стеку, який став популярним вибором для створення комплексних веб-додатків, охоплюючи весь спектр розробки від серверної (back-end) до клієнтської (front-end) частини.

Будь-який веб-додаток базується на комбінації технологій, які у сукупності формують так званий "стек". Одним з перших широко поширених був стек LAMP, що складається з Linux, Apache, MySQL та PHP. Зростання популярності односторінкових додатків (SPA) зумовило зміни в підходах до розробки. SPA дозволяють уникнути повного перезавантаження сторінки для оновлення контенту, замість цього використовуючи легкі запити до сервера для завантаження нових даних або фрагментів сторінки, що значно підвищує ефективність.

MEAN стек, який включає MongoDB, Express.js, AngularJS та Node.js, став одним з перших відкритих стеків, що акцентував увагу на SPA та використанні NoSQL. AngularJS як фронтенд фреймворк забезпечує модель MVC, в той час як MongoDB використовується як гнучка NoSQL база даних. Node.js та Express.js відіграють роль серверної платформи та веб-сервера відповідно.

2.2.1 React

З появою React, який розробляється Facebook, стек MEAN почав трансформуватися. React, замінюючи AngularJS у MEAN, утворює MERN стек. Ця зміна відбулася через зростаючу популярність React, який хоч і не є повноцінною MVC-технологією, пропонує гнучкість і потужність для розробки інтерфейсів.

Однак, лише основні технології стеку недостатні для створення повноцінного веб-додатку. Інструменти розробки та допоміжні бібліотеки є необхідними для підтримки процесу розробки, а також для розширення функціональності і можливостей React. Це включає в себе інструменти для управління залежностями, тестування, побудови інтерфейсів, а також різноманітні плагіни та розширення, які дозволяють створювати більш гнучкі та масштабовані рішення для веб-додатків.

Таким чином, MERN стек, як і його попередник MEAN, представляє собою сучасний, гнучкий підхід до розробки веб-додатків, що дозволяє розробникам ефективно створювати високопродуктивні і інтерактивні веб-додатки, використовуючи відкриті та масштабовані технології.

2.2.2 Детальний огляд MERN стеку

MERN стек представляє собою комплексний набір технологій, який використовується для створення повноцінних веб-додатків, реалізованих на JavaScript. Цей стек включає в себе розробку як клієнтських, так і серверних аспектів веб-сервісу, об'єднуючи чотири ключові технології для створення інтегрованого веб-рішення.

Для клієнтської частини веб-додатку в MERN стеку використовується бібліотека React.js, яка забезпечує гнучкий інтерфейс і дозволяє створювати реактивні користувацькі інтерфейси. React.js є відомим і широко використовуваним рішенням, що дозволяє легко створювати інтерактивні елементи веб-сторінок.

На серверній стороні MERN стек використовує Node.js - потужне середовище виконання JavaScript, що дозволяє запускати JS-код поза браузером. Це надає можливість реалізації серверної логіки веб-додатків із використанням однієї мови програмування як для клієнтської, так і для серверної частини. Express.js, що також є частиною стеку, виступає в ролі гнучкого веб-фреймворку, який спрощує розробку веб-серверів, забезпечуючи більш ефективну обробку запитів та відповідей.

Для зберігання даних у MERN стеку використовується MongoDB - сучасна NoSQL база даних, яка пропонує гнучку і масштабовану систему зберігання. MongoDB використовує формат документів подібний до JSON, що забезпечує легке зберігання та маніпулювання даними, а також підтримку складних запитів та агрегації даних.

Також варто розглянути тезнологію MERN, оскільки він доволі популярний за рахунок своїх переваг.

Завдяки цій комбінації технологій, MERN стек став популярним вибором серед розробників, оскільки він дозволяє створювати масштабовані

та високопродуктивні веб-додатки, які легко підтримувати та розвивати. Така уніфікація технологій на всіх етапах розробки забезпечує більшу синергію та ефективність у процесі створення сучасних веб-додатків.

2.2.3 React

React, що є основним компонентом у структурі MERN стеку, представляє собою бібліотеку JavaScript з відкритим кодом, підтримувану Facebook. Ця бібліотека відіграє ключову роль у створенні користувацького інтерфейсу веб-сайтів, використовуючи HTML для візуального представлення. На відміну від AngularJS, React не є повнофункціональним фреймворком; він зосереджений на рендерингу інтерфейсу (V у MVC) і дає розробникам свободу у виборі архітектури інших компонентів програми.

React відрізняється від інших бібліотек і фреймворків для розробки клієнтської частини веб-додатків декількома ключовими особливостями:

- Декларативність: React дозволяє розробникам створювати інтерфейси, які автоматично оновлюються при зміні станів або даних, мінімізуючи необхідність безпосереднього втручання у DOM. Завдяки декларативному підходу, інтерфейси стають більш передбачуваними та легшими у підтримці.

- Компонентна структура: Основою React є компоненти, кожен з яких має власний стан і відповідає за своє відображення. Створення компонентів і їх подальше об'єднання для формування повного інтерфейсу веб-сторінки спрощує процес розробки та забезпечує більшу модульність.

- Використання JSX: React використовує JSX, проміжну мову, яка дозволяє інтегрувати HTML-подібну синтаксичну структуру безпосередньо в JavaScript. Це робить код більш зрозумілим та спрощує процес створення віртуального DOM, хоча JSX не є обов'язковим для використання.

- Ізоморфність: React може виконуватися як на сервері, так і в браузері, що робить його гнучким рішенням для веб-розробки. Це забезпечує можливість попередньої рендерації сторінок на сервері, що може бути

корисним для пошукової оптимізації (SEO) та підвищення швидкості завантаження сторінок.

Завдяки цим характеристикам, React став популярним вибором серед розробників, які шукають ефективні та гнучкі рішення для створення інтерактивних користувацьких інтерфейсів для веб-додатків. Ця технологія забезпечує потужні засоби для реалізації динамічних і відповідних веб-інтерфейсів, використовуючи сучасні підходи до розробки програмного забезпечення.

2.2.4 Node.js

Node.js є важливою частиною сучасної веб-розробки, діючи як середовище виконання JavaScript, яке функціонує поза браузером. Це середовище, розроблене на базі V8 Engine від Google Chrome, дає змогу використовувати JavaScript незалежно, створюючи можливості для більш гнучкої серверної розробки.

Основна перевага Node.js полягає у його асинхронній, подієво-орієнтованій моделі, що дозволяє обробляти велику кількість паралельних запитів без блокування. Це відкриває можливості для високопродуктивних та масштабованих додатків.

Переваги Node.js:

Модульна структура: Node.js використовує систему модулів CommonJS, що дозволяє інтегрувати різноманітні бібліотеки та інструменти через механізм "require". Ця модульність сприяє кращій організації коду і розподілу функціональності між окремими частинами додатку.

Велика екосистема модулів: Завдяки npm (Node Package Manager), Node.js має одну з найбільших екосистем пакетів, які можна легко інтегрувати у проекти. npm спрощує управління залежностями та встановлення додаткових бібліотек.

Використання подій та асинхронного програмування: Відмінною особливістю Node.js є його асинхронна природа, яка дозволяє виконувати неблокуючі операції введення/виведення. Це забезпечує високу продуктивність, особливо

в умовах великої кількості одночасних запитів, що є критичним для веб-додатків у реальному часі.

Гнучкість у розробці: Node.js дає розробникам свободу у створенні структури додатку, виходячи з потреб проекту. Відсутність строгих правил чи шаблонів дозволяє адаптувати додаток під конкретні вимоги та інтегрувати різні інструменти та бібліотеки.

Node.js ефективно поєднує в собі простоту JavaScript із потужністю серверного програмування, роблячи його ідеальним вибором для розробки високопродуктивних веб-додатків. Його універсальність і масштабованість роблять Node.js одним з найпопулярніших інструментів у сучасній веб-розробці.

2.2.5 Express.js

Express.js є ключовим фреймворком для розробки серверних додатків у середовищі Node.js. Його основне призначення - спрощення процесу створення веб-серверів, що робить розробку на Node.js більш доступною і ефективною.

Express.js вирізняється своєю здатністю легко налаштовувати маршрути та обробляти HTTP-запити. Це дає розробникам гнучкість у визначенні, як повинні бути оброблені вхідні запити та які відповіді слід надсилати клієнтам. Специфікації маршрутів, що використовують регулярні вирази, надають значну гнучкість у визначенні шаблонів URL.

Особливості Express.js:

- Обробка запитів: Express забезпечує зручні методи для роботи з HTTP-запитами, включаючи стягування URL, заголовків і параметрів.
- Відправка відповідей: Фреймворк дозволяє налаштовувати коди відповідей HTTP, встановлювати кукі, надсилати спеціальні заголовки та інше.
- Підтримка шаблонізаторів: Хоча Express.js сам по собі не має вбудованого шаблонізатора, він підтримує використання різних

шаблонних движків, таких як pug, mustache тощо, дозволяючи розробникам обирати найзручніший інструмент для генерації HTML.

У контексті односторінкових додатків (SPA), які зазвичай розробляються у стеку MERN, Express.js відіграє роль веб-сервера, що обслуговує статичні файли та API-запити. Це означає, що всі динамічні оновлення контенту виконуються на клієнтській стороні, в той час як сервер відповідає за обробку запитів до API та доставку необхідних ресурсів.

Загалом, Express.js є надійним і гнучким вибором для розробки серверної частини веб-додатків у Node.js. Його лаконічність та ефективність роблять його одним з найпопулярніших фреймворків для розробки сучасних веб-додатків, особливо тих, які використовують архітектуру MERN.

2.2.6 MongoDB

MongoDB є невід'ємною складовою стеку MERN, виступаючи як гнучка та масштабована NoSQL база даних. Ця система управління базами даних зосереджується на зберіганні даних у формі документів, які мають вигляд подібний до JSON, забезпечуючи гнучкість у структурі даних та легкість їх обробки.

Важливими характеристиками MongoDB є:

Документно-орієнтований підхід: Відмінною рисою MongoDB є зберігання даних у формі документів, що відрізняється від традиційної реляційної моделі з таблицями та рядками. Це дає можливість більш гнучкого представлення даних, а також спрощує роботу з складними даними і глибоко вкладеними структурами.

Горизонтальне масштабування: MongoDB підтримує горизонтальне масштабування, яке дозволяє розподілити навантаження на кілька серверів. Це забезпечує високу доступність та ефективне використання ресурсів, особливо в умовах обробки великої кількості даних.

Гнучкі схеми: В MongoDB відсутній жорсткий набір схем, що дає розробникам свободу у проектуванні структури даних. Це забезпечує легкість внесення змін і адаптації бази даних під поточні потреби проекту.

Підтримка ODM (Object Document Mapping) бібліотек: Для роботи з MongoDB часто використовуються ODM бібліотеки, такі як Mongoose, які надають додаткові функції для моделювання даних та валідації.

Гнучка мова запитів: MongoDB використовує мову запитів, засновану на JSON, що робить процес створення та виконання запитів інтуїтивно зрозумілим та зручним для розробників, оскільки вона тісно інтегрована з JavaScript.

Вбудований JavaScript-шел: MongoDB має вбудований JavaScript-шел, що дозволяє виконувати складні операції та запити безпосередньо з командного рядка, використовуючи JavaScript.

Завдяки цим особливостям, MongoDB відіграє ключову роль у створенні гнучких, масштабованих та високопродуктивних веб-додатків у рамках MERN стеку, забезпечуючи ефективне зберігання та обробку даних.

2.2.6 Новітні технології

В 2023 році штучний інтелект (ШІ) набув популярності завдяки його здатності трансформувати різноманітні сектори, включаючи медицину, освіту, виробництво та, звичайно, роздрібну торгівлю. Штучний інтелект відкриває нові можливості для підвищення ефективності, персоналізації досвіду клієнтів та впровадження інноваційних продуктів та послуг. Тому використання штучного інтелекту є одним з обов’язкових амспектів для побудови проивабливого і актуального прєку. Тому при побудові діалогової системи буде враховуватись данний аспект.

Середовище для комунікації з клієнтами торгового центру теж відіграє важливу роль в комунікації з клієнтами. Епоха сайтів і док станцій минула і більшість клієнтів перейшли в месенджери. Лідером серед месенджерів в 2023 році являється Telegram, хоч це і не найбільш популярний месенджер в світі, але після 2022 року за опитуванням його вважають найкращим мессенджером в Україні 50,6% респондентів. З 2020 року кожен популяний сагазин або бренд веде свою сторінку в телеграмі де ділиться інформацією про новинки та анкії компанії. Тому для легкого старту та популярності діалогова система буде розроблена під Telegram.

Провівши повний аналіз побудови комп'ютеризованої діалогової системи для торгового центру під Telegram з використанням Open AI , було визначено основний перелік ключових елементів:

1. Створення Телеграм-бота:

Для створення телеграм бота можна скористатись двома засобами:

- Зареєструвати бота в Телеграмі через @BotFather.
- Створити бота за допомогою мови програмвання Python або Java.

Оскільки до телеграм потрібно інтегрувати OpenAI GPT, то доведеться об'єднати обидва засоби. За допомогою @BotFather ми створимо бота, і налаштовуємо його за допомогою Python.

2. Використання OpenAI GPT:

Для інтеграції чат бота з OpenAI потрібно зареєструватись і отримати ключ для звернень до OpenAI API та отримання відповідей від GPT моделі.

3. Створення Бази Даних:

Оскільки потрібно щоб OpenAI використовував лише інформацію, для відповіді клієнтам тогргового центру, потрібно створити інформаційну базу даних підходящого формату. База даних повинна мати дані про торговий центр і магазини які там знаходяться. Так, створення інформаційної бази даних є ключовим етапом для забезпечення OpenAI GPT необхідною інформацією для відповідей на запитання клієнтів торгового центру.

4. Налаштування Взаємодії між Телеграм-ботом та OpenAI GPT:

Для успішної інтеграції OpenAI GPT у телеграм-бот для торгового центру, спочатку потрібно налаштувати зв'язок між телеграм-ботом і API OpenAI, використовуючи ключі API, які надасть OpenAI та BotFather. Для взаємодії з API OpenAI потрібно використовувати бібліотеку requests, передаючи текстові запити та обробляючи отримані відповіді у телеграм-боті. Також потрібно додати логіку обробки відповідей, інтегруючи їх у відповіді бота, та забезпечити коректну обробку текстових даних, що будуть надходити від OpenAI GPT.

5. Інтеграція з Базою Даних:

Оскільки Open AI потрібно навчити правильно відповідати і інтерпритувати запити які будуть надавати клієнти, потрібно задати приклади запитів клієнтів і адаптувати інформацію з бази даних для GPT щоб він брав за приклад ці запити і рівнявся на них при написанні відповідей клієнтам. Тому важливо враховівати тип та структуру бази данх яку буде використовувати Open AI. Також потрібно створити базу даних яка місститть

6. Безпека та Аутентифікація:

При розробці та впровадженні телеграм-бота для торгового центру важливо приділити належну увагу питанням безпеки та аутентифікації. Забезпечити захист від несанкціонованого доступу до системи та бази даних, використовуючи надійні методи аутентифікації, такі як двофакторна аутентифікація та використання сильних паролів.

Враховувати можливість шифрування комунікації між телеграм-ботом та іншими складовими системи, зокрема, сервісом OpenAI GPT та базою даних. Застосовувати відповідні практики шифрування для захисту конфіденційності інформації, яку обмінює ваш бот.

Також, регулярно оновлювати та моніторити безпекові налаштування телеграм-бота та всіх залучених сервісів, щоб уникнути вразливостей та запобігти можливим атакам. Важливо стежити за оновленнями програмного забезпечення та вчасно виправляти виявлені безпекові проблеми.

Для безпеки варто використовувати токени для аутентифікації взаємодії між компонентами. Такі токени використовує сам OpenAI при роботі з зовнішніми джерелами отримання інформації та комунікації.

7. Тестування та Оптимізація:

Після розробки та інтеграції телеграм-бота з системами OpenAI GPT та базою даних, наступним кроком буде проведення тестування та оптимізація системи. Завдання полягає в тому, щоб забезпечити комплексне тестування всіх функціональних можливостей бота, переконатися в правильності відповідей на запитання та ефективності взаємодії з користувачами.

Важливо виявити та виправити всі можливі помилки, а також оптимізувати роботу бота для максимальної продуктивності. Потрібно забезпечити розгляд можливостей кешування деяких запитань для швидкого доступу до відповідей та використання кешування для оптимізації часу відповіді бота.

Також, не зайвим буде введення механізмів збору та аналізу даних про використання бота, щоб забезпечити отримання зворотного зв'язку від користувачів та вдосконалювати функціонал у майбутньому. Збирати дані про типові запитання, реакції користувачів та можливість вдосконалення алгоритмів відповідей.

Після успішного тестування та оптимізації завданням є забезпечити готовність бота до впровадження та масштабування, забезпечуючи його стабільну та ефективну роботу в реальних умовах.

8. Моніторинг та Аналіз:

На етапі моніторингу та аналізу реалізованої діалогової системи для торгового центру важливо забезпечити постійний контроль за функціональністю бота. Налаштовано систему моніторингу, яка включає в себе перевірку доступності та відповідності ключових сервісів. Запроваджено систему сповіщень для оперативного реагування на можливі аварійні ситуації.

Окрім цього, проводиться аналіз використання бота, вивчаючи дані щодо активності користувачів, популярних запитань та інших метрик. Цей підхід дозволяє зрозуміти потреби користувачів, вчасно виявляти та виправляти можливі недоліки, а також постійно вдосконалювати функціонал бота.

Система моніторингу та аналізу допомагає забезпечити стабільну та надійну роботу телеграм-бота, а також адаптувати його до змінних потреб користувачів та ринкових умов.

9. Документація:

Щоб забезпечити ефективне управління, необхідно підготувати докладну та зрозумілу документацію для розгортання, налаштування та експлуатації розробленої діалогової системи торгового центру. Документація включає в

себе опис архітектури системи, інструкції з встановлення та конфігурації, а також рекомендації щодо подальшого розвитку та підтримки.

У документації детально розглядаються всі компоненти системи, їх функціональні можливості та взаємодія між ними. Окремий розділ присвячено інтеграції з існуючою базою даних та взаємодії з OpenAI GPT. Крім того, надаються рекомендації щодо забезпечення безпеки та захисту персональних даних користувачів.

Документація є ключовим інструментом для команди підтримки, розробників та адміністраторів, допомагаючи їм розуміти, налаштовувати та вдосконалювати систему. Забезпечення зрозумілої та структурованої документації сприяє ефективному функціонуванню та розвитку діалогової системи торгового центру.

РОЗДІЛ 3.
РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ КОМП'ЮТЕРИЗОВАНОЇ ДІАЛОГОВОЇ СИСТЕМИ ТОРГОВОГО ЦЕНТРУ

3.1 Технічне завдання на створення системи

Технічне завдання для реалізації системи включає в себе деталізований опис функціональності, вимог до архітектури, інтерфейсів та взаємодії компонентів системи. Процес виконання технічного завдання передбачає взаємодію розробників, аналітиків та замовника для забезпечення успішної реалізації всіх визначених вимог і завдань. Схема розробленої роботи компонентів комп'ютеризованої діалогової системи торгового центру представлена на рис. 3.1.

Рис. 3.1. Інформаційно-логічна модель роботи компонентів комп'ютеризованої діалогової системи

Telegram bot – відправляє запит клієнта до Open AI.

Open AI – аналізує запит і шукає відповідь на запит в базі данних.

База даних – база даних містить структровану і зрозумілу для AI інформацію.

Оскільки схема передбачає мінімалізацію кількості блоків, це сприяє оптимізації часу відповіді на запити користувачів. Такий підхід може покращити загальну продуктивність системи та забезпечити більш ефективну взаємодію між користувачами та торговим центром. Крім того, ефективна структура бази даних, орієнтована на обробку AI, сприяє більш точному виведенню відповідей, що підвищує якість обслуговування клієнтів.

Така система може бути особливо корисною для торгових центрів, що прагнуть надати своїм відвідувачам персоналізовану інформацію, рекомендації продуктів або допомогу в навігації простором торгового центру. Вона може також інтегрувати функції машинного навчання для поліпшення взаємодії з користувачами на основі їх попередніх запитів і поведінки, тим самим підвищуючи користувацький досвід.

Задля досягнення максимальної ефективності, розробники повинні також враховувати безпеку даних, шкальованість системи та її здатність до інтеграції з іншими платформами та сервісами, які можуть бути використані в торговому центрі.

3.2 Специфіка програмно-апаратної реалізації

Кожна розробка програмного продукту починається з архітектури, а саме: архітектура Бази Даних (БД); архітектура програмного засобу (ПЗ).

Архітектура бази даних включає в себе вибір самої бази даних і саму структуру даних. Оскільки Open AI може аналізувати лише дані представлені в файлах формату txt та jsonl, то дані для аналізу мають специфічний вигляд. На рис 3.2 зображено текстові дані.

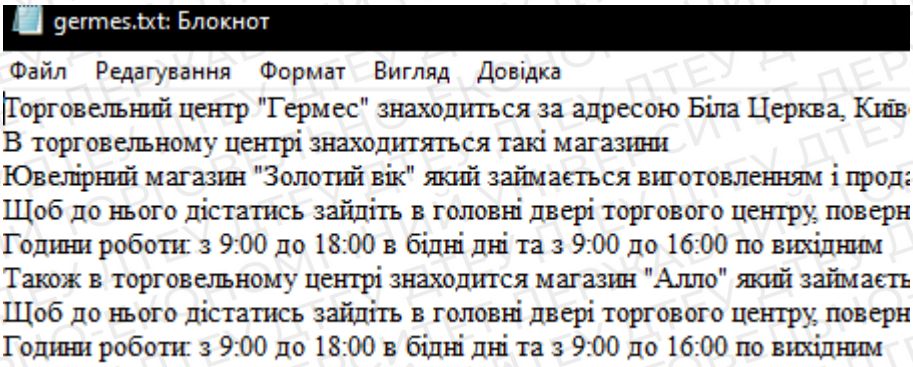


Рис. 3. 2. Дані для Open AI

Дані формату jsonl використовуються для тонкого налаштування Open Ai, вони мають відповідати певному формату представленому на **Рис. 3. 3**

Рис. 3. 3. Дані для тонкого налаштування Open AI

Дані тонког налаштування потрібні для створення власної моделі AI доступної через API.

Тонке налаштування дозволяє досягти від моделей, доступних через API, таких переваг, як:

- Отримання результатів вищої якості, ніж від стандартних підказок.

- Можливість тренувати модель на більшому обсязі прикладів, ніж може вмістити звичайна підказка.
- Заощадження жетонів завдяки скороченим підказкам.
- Зменшення затримок при отриманні відповідей на запити

Простими словами Open AI аналізує дані тонкого налаштування і запам’ятовує їх, якщо клієнт відправляє запит який співпадає з прикладом даних для тонкого налаштування Open AI не звертається до бази даних, а викоистовує дані з файла тонкого налаштування для відповіді.

Для створення бота використовується бот від телеграму під назвою BotFather, він створює лише оболонку і генерує токен який використовується як логін до бота, за допомогою токена можна підключатись до оболонки і вностит зміни в неї, в інтернеті є багато інформації як підключитись та налаштувати бот за допомогою Python, тому тут не виникає проблем (рис. 3.4).

C: > chatgpt-telegram-bot-main > bot > telegram_bot.py > ...

```
1   from __future__ import annotations
2
3   import asyncio
4   import logging
5   import os
6
7   from uuid import uuid4
8   from telegram import BotCommandScopeAllGroupChats, Update, constants
9   from telegram import InlineKeyboardMarkup, InlineKeyboardButton, InlineQueryResultArticle
10  from telegram import InputTextMessageContent, BotCommand
11  from telegram.error import RetryAfter, TimedOut
12  from telegram.ext import ApplicationBuilder, CommandHandler, MessageHandler, \
13      filters, InlineQueryHandler, CallbackQueryHandler, Application, ContextTypes, CallbackContext
14
```

Рис.3.4 Частина коду чат бота

Після налаштування базового вигляду чат боту потрібно підключити Open AI до бота, для цього потрібно перейти на офіційний сайт Open AI та зареєструватись. Після реєстрації відкриється меню, де можна згенерувати

ключ API. Ключ API (або API keys) - це унікальні ідентифікатори, які використовуються для аутентифікації та авторизації користувачів чи програм, що звертаються до API. Ключі API є частиною механізму безпеки, який дозволяє контролювати доступ та використання API.

Після отримання ключа потрібно підключити його до телеграм бота (рис. 3.5).

```
# Your OpenAI API key
OPENAI_API_KEY="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"

# Your Telegram bot token obtained using @BotFather
TELEGRAM_BOT_TOKEN="6834140463:AAGbthz3Nkr47SLHTzSX4BxFcrGS4Ci7aV8"

# Telegram user ID of admins, or - to assign no admin
ADMIN_USER_IDS=ADMIN_1_USER_ID,ADMIN_2_USER_ID

# Comma separated list of telegram user IDs, or * to allow all
ALLOWED_TELEGRAM_USER_IDS="*"
```

Рис.3.5. Код підключення Open AI до бота

Тепер потрібно створити свою модель Open AI, яку потрібно навчити під власні потреби, для цього потрібно використати базу даних та дані для тонкого налаштування. В цьому допомагає середовище на офіційному сайті Open AI, на вкладці Fine-tuning можна створити та навчити на основі своє бази даних модель (рис 3.6).

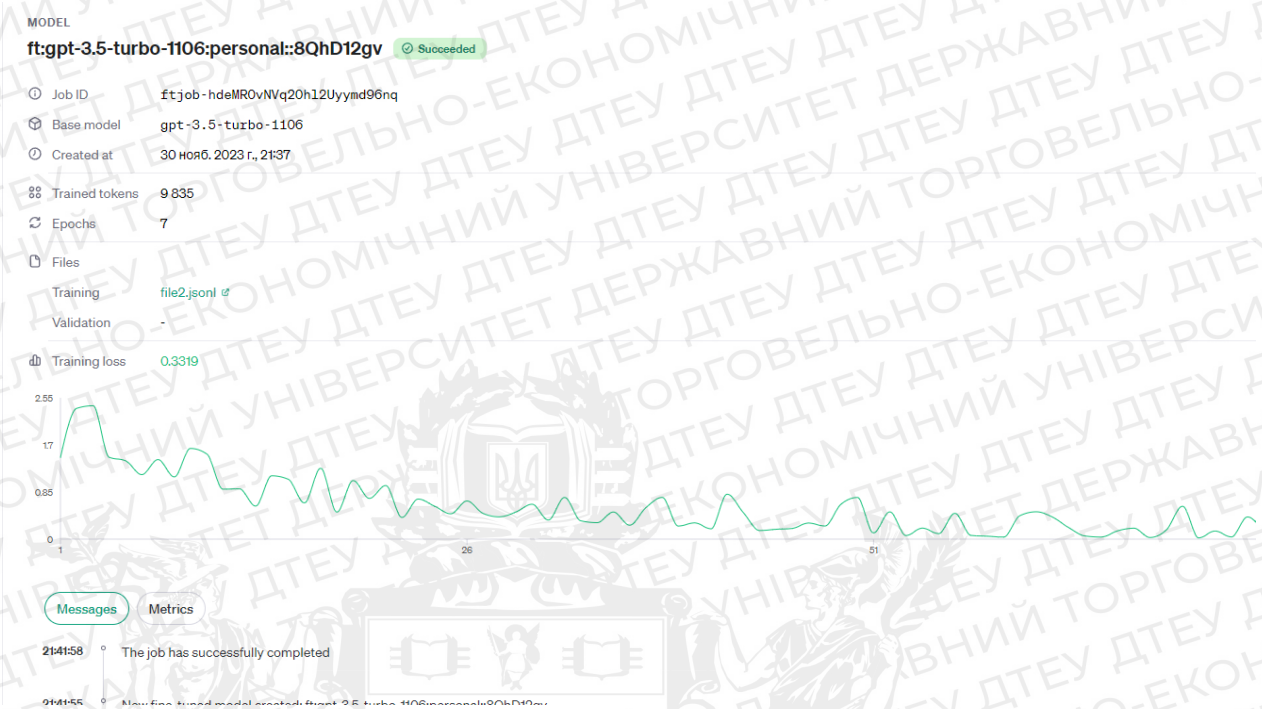


Рис.3.6.Вікно власної моделі Open AI

Так як модель пройшла навчання успішно і проаналізувавши дані не знайшла орфографішних помилок подання тексту, її можна встановити як основну модель яка буде використовуватись при звертанні до Open AI, в вкладці Playground на офіційному сайті Open AI, також потрібно створити назву моделі та написати інструкцію для неї. Інструкцію ще називають Prompt, Prompt – це фрагмент текста , який задає моделі стиль спілкування і описує його роль, яку він має грати для клієнтів коли дає відповідь. Оскільки модель не має власного стилю, то копіює його виходячи з даних які використовує для відповіді, тому при відсутності Prompt, інтонація і стиль відповідей на різні питання може відрізнятись. Тому потрібно задавати Prompt, в якому модель буде відігравати роль незалежно від стилю даних які вон буде використовувати для відповіді.

Рис.3.7. Налаштувань моделі Open AI

Після навчання і налаштування власної моделі потрібно перевірити коректність її роботи, для цього потрібно телеграм бота запитати інформацію яка міститься в базі даних txt.

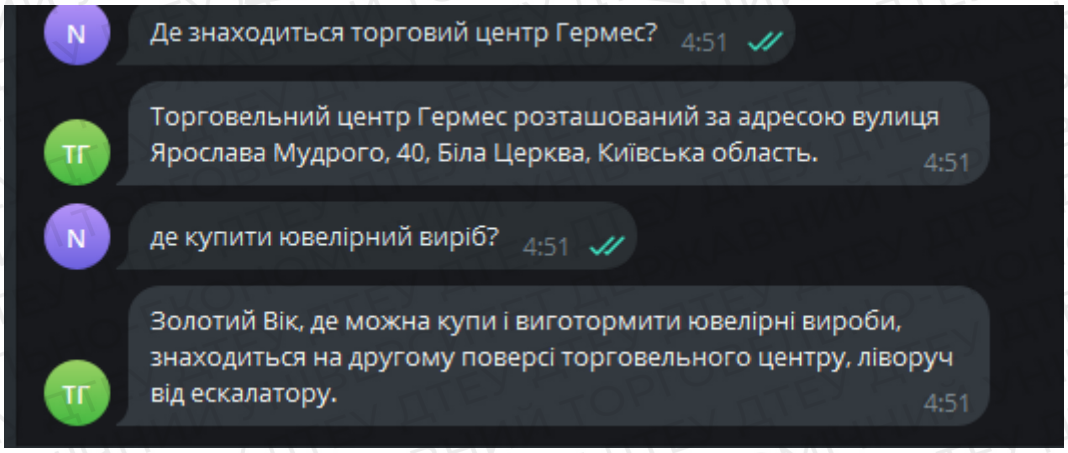


Рис.3.8. Перевірка коректності роботи Open AI

По результатам перевірки можна дійти висновку що модель працює коректно, дані використовуються вцірно.

3.3 Технологія використання комп'ютеризованої діалогової системи торгового центру

Для отримання доступу до діалогової системи достатньо знайти торгового бота в телеграм по назві (рис. 3.9).

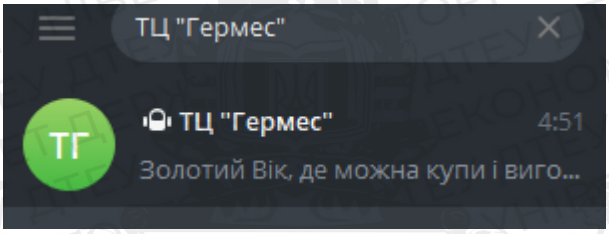


Рис.3.9 Вікно пошуку Telegram

Для зручності клієнтів булло додано меню з швидкими командами (рис. 3.10).

Рис.3.10. Меню бота телеграм

Користувач відразу може скористатись ботом просто написавши свій запит або відправивши голосове повідомлення. (рис 3.11).

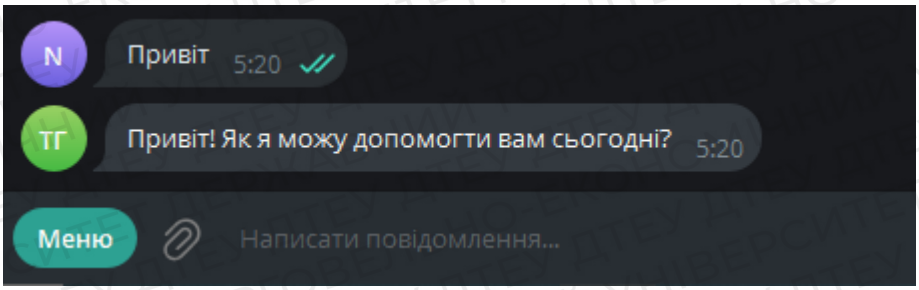
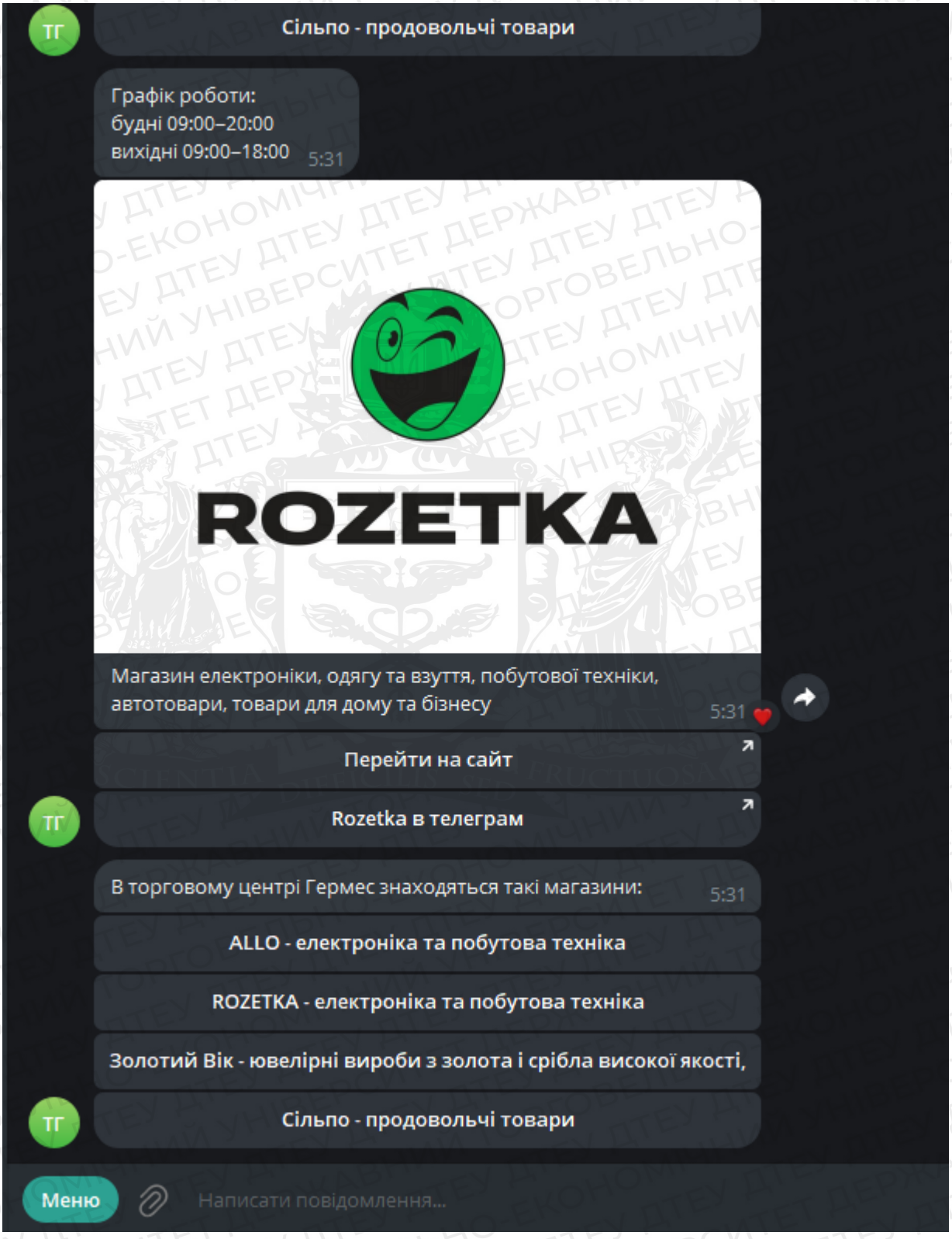


Рис.3.11.Вікно Комунікації з ботом

Оскільки Open AI це відносно молода технологія, інформації про неї доволі мало, також через новизну це доволі сирий продукт, і тому він має обмежений доступ. Open AI поки не дозволяэ використовувати свої моделі для переходу на зовнішні посилання і аналізу сторонньої інформації з різних джерел, оскільки відповідальність за результат лежить на Open AI. Тому, окрім додавання моделі, було встановлено декілька блоків меню для переходу на сторонні телеграм канали магазинів які знаходяться в торговому центрі.



.

Рис.3.12. Блок який відкривається за допомогою Меню

Меню потрібно виключно для переходу на сторінку магазину щоб зменшити навантаження на комп'ютеризовану діалогову систему торгового центру. Оскільки магазини мають власні бази даних великого обсягу, для аналізу яких моделі від Open AI доведеться витрачати багато часу для аналізу інформації про товари і послуги які оновлюються постійно, це може призвести до збільшенню часу відповіді моделі і надання помилкової інформації.

Таким чином було реалізовано повноцінну діалогову систему яка задовольняє запити користувачів торговельного центру без участі людського фактору. За допомогою новітніх методів і засобів було досягнуто максимальнальної результативності у побудові діалогової системи для торгового центру, система доволі проста і має інтуїтивно зрозумілий інтерфейс і дозволяє задовольнити запити на інформацію всіх кратегорій клієнтів торговельного центру.

ВИСНОВКИ

У випускній кваліфікаційній роботі представлено результати теоретичних і прикладних досліджень, що полягають у аналізі методів та засобів побудови комп'ютеризованих діалогових систем торгового центру. Результати прикладних досліджень стали основою для створення комп'ютеризованої діалогової системи торгового центру. В результаті проведених досліджень були отримані такі висновки:

1. Комп'ютеризовані діалогові системи можуть привабити нових клієнтів до торгових центрів і завоювати іх прихильність доступністю інформації. Відносна дешевизна обслуговування і побудови комп'ютеризованої діалогової системи може заощадити кошти на маркетинг і просування торгового центру в медіа просторі, а також комунікації з клієнтами.

2. При проведенні соціально-економічних досліджень важливо використовувати передові інформаційні технології для забезпечення високоякісних даних. Використання сучасних інформаційних і діалогових систем є ключовим для здійснення аналізу та прийняття адекватних управлінських рішень, спрямованих на врахування актуальних соціально-економічних умов у конкретному регіоні. Ці технології дозволяють збирати та обробляти дані, взаємодіяти з користувачами та адаптуватися до змін у соціально-економічному середовищі, сприяючи ефективному управлінню та прийняттю обгрунтованих стратегічних рішень.

3. Вивчені різноманітні методи створення комп'ютеризованих діалогових систем, приділяючи особливу увагу їх ефективності та застосовності в контексті побудови діалогової системи для торгового центру. Основний акцент робився на ідентифікації найбільш оптимальних методів,

які враховують специфіку та потреби торговельного середовища. Досліджено різні підходи та визначено кращі практики для впровадження в діалогові системи, спрямовані на поліпшення взаємодії з клієнтами в торгових центрах.

4. Виконано програмне втілення методу розробки комп'ютеризованої діалогової системи, спеціально адаптованого для потреб торгового центру. Процес реалізації включав в себе впровадження програмних алгоритмів та технологічних рішень, спрямованих на оптимізацію взаємодії з клієнтами та полегшення їхнього досвіду в торговому середовищі. Результати програмної реалізації відображають високий ступінь адаптації до специфічних вимог та особливостей торгового центру.

5. Розроблена та впроваджена за допомогою передових програмних рішень комп'ютеризована діалогова система представляє собою ефективний інструмент взаємодії з клієнтами торгового центру. Її функціонал дозволяє не лише надавати вичерпні відповіді на запитання клієнтів, але і взаємодіяти з ними в динамічних діалогових сценаріях. Використання сучасних програмних засобів підвищує швидкість та точність відповідей, а автоматизований характер системи дозволяє обслуговувати клієнтів без прямої участі людського персоналу. Такий підхід сприяє покращенню обслуговування та підвищенню задоволеності клієнтів у торговому центрі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

6. Allen, James F., and C. Raymond Perrault. "Analyzing intention in utterances." Artificial Intelligence 15.3 (1980): 143-178.
7. Allen, James F., et al. "The TRAINS project: A case study in building a conversational planning agent." Journal of Experimental & Theoretical Artificial Intelligence 7.1 (1995): 7-48.
8. Amir, Ofra, and Yakov Kobi Gal. "Plan recognition and visualization in exploratory learning environments" (2013).
9. Raskin Jef. The Humane Interface. — Addison Wesley. 2000. 233 p.
10. Day J. Patterns in Network Architecture: A Return to Fundamentals. 2007. 429 p.
11. Bates R. J., D. W. Gregory. Voice & data communications handbook. 2011. 650 p.
12. Єсаулов С. М., Бабічева О. Ф. Автоматизація технологічних процесів та установок. Конспект лекцій. Харків, 2009. 78 с.
13. Lammale T., S. Odom., K. Wallase. CCNP: Routing Study Guide. Alameda, 2015. 444 p.
14. Machine learning ARTIFICIAL INTELLIGENCE. 2009. URL: http://www.britannica.com/EBchecked/topic/1116194/machine-learning (дата звернення: 06.11.2019).
15. Russell Stuart, Norvig Peter. Artificial Intelligence: A Modern Approach (вид. 2nd). Prentice Hall. 2009. 1152 p.
16. Grady Butch., Jim Conallen., Michael Engle. Object Oriented Analysis and Design with Application Examples. 2008.
17. Добровольський В. В. Екологічна експертиза. Навчальний посібник. Миколаїв, 2013. 220 с.
18. Radio Frequency Interference - And What to Do About It. 2011. URL: http://www.radiosky.com/journal0901.html (дата звернення: 17.10.2019).

19. Шевченко Л. С. Основи економічної теорії. Харків, 2008. 448 с.
20. Altman, N. S. An introduction to kernel and nearest-neighbor nonparametric regression. 1992. 185 p.
21. D. Coomans; D.L. Massart. Alternative k-nearest neighbour rules in supervised pattern recognition : Part 1. k-Nearest neighbour classification by using alternative voting rules. 2002. 137 p.
22. Michael L. Johnson, Ludwig Brand. Computer Methods, Part C. 2011. 637 p. 17.
23. More deeply, the framework exists to separate the representation of information from user interaction. The DCI Architecture: A New Vision of Object-Oriented Programming URL: https://www.cnblogs.com/happyframework/p/3302238.html (дата звернення 26.10.2019)
24. Flanagan D. JavaScript - The Definitive Guide. 2011. 1096 p.
25. Дзякевич Ю. В. Економічні основи ресурсозбереження. Навчальний посібник. Тернопіль, 2015. 76 с.
26. Eckart Michaelsen, Dr. Jochen Meidow. Hierarchical Perceptual Grouping for Object Recognition. 2019. 723p/
27. Danny Goodman, Michael Morrison JavaScript Bible, 5th Edition 2006. 1184 p.
28. Peter Thiel Zero to One: Notes on Startups, or How to Build the Future 2014. 224 p.
29. Alexei White. Major JavaScript Engines 2009. 1032 p.
30. Рейсиг Д. JavaScript. Профессиональные приемы программирования 2009. 352 p.

ДОДАТОК

Програмний код реалізації Web-додатку

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Windows.Media;
namespace RadSED.Model
{  public static class MathCalc
   {  public static double StdDev(this IEnumerable<double> values)
      {   double ret = 0;
          int count = values.Count();
          if (count > 1)
          {  //Compute the Average
             double avg = values.Average();
             //Perform the Sum of (value-avg)^2
             double sum = values.Sum(d => (d - avg) * (d - avg));
             //Put it all together
             ret = Math.Sqrt(sum / count);   }
          return ret;  }
       public static double Disp(this IEnumerable<double> values)
       {   double ret = 0;
          int count = values.Count() - 1;
          if (count > 1)
          {  //Compute the Average
             double avg = values.Average();
```

```
      //Perform the Sum of (value-avg)^2
      double sum = values.Sum(d => (d - avg) * (d - avg));
      //Put it all together
      ret = (sum / count);  }
    return ret; }
  public static double Korel(double[] values, double[] values2)
  {  double ret = 0;
    int count = values.Count();
    if (count > 1)
    { //Compute the Average
      double avg = values.Average();
      double avg2 = values2.Average();
      double std = StdDev(values);
      double std2 = StdDev(values2);
      //Perform the Sum of (value-avg)^2
      double[] res = new double[values.Count()];
      for (int i = 0; i < values.Count(); i++)
        res[i] = (values[i] - avg) * (values2[i] - avg2);
      double sum = res.Sum();
      //Put it all together
      ret = (sum / (count * std * std2));  }
    return ret; }
  public static double StdSqrt(this IEnumerable<double> values)
  {  double ret = 0;
    int count = values.Count();
```

```
        if (count > 1)
        {  //Compute the Average
           double avg = values.Average();
           //Perform the Sum of (value-avg)^2
           double sum = values.Sum(d => (d - avg) * (d - avg));
           //Put it all together
           ret = Math.Sqrt(sum); }
        return ret;  }
     public static List<double[]> Multiply_matrix(List<double[]> array1,
List<double[]> arrary2)
     { int cnt = 11;
        List<double[]> res = new List<double[]>();
        for (int i = 0; i < array1.Count; i++)
        {  double[] prom_res = new double[cnt];
           for (int j = 0; j < cnt; j++)
           {  double nextVal = 0;
              for (int k = 0; k < cnt; k++)
              { nextVal += array1[i][k] * arrary2[k][j]; }
              prom_res[j] = nextVal;  }
           res.Add(prom_res);  }
        return res; }
     public static List<double[]> norm_matrixs;
     public static List<double[]> korel_matrixs;
     public static List<double[]> korel_matrixs2;
     public static List<double[]> matrixs;
```

```
    public static List<double[]> vectors;
    public static List<double[]> multiply;
    public static List<double[]> norm_fakt_matrixs;
    public static List<double[]> diperse_matrixs;
    public static List<double> diperse_district_matrixs;
    public static List<double[]> korel_factor_pokaz_matrix;
    public static List<double[]> integral_indicators;
    public static double AVG_II { get; set; } }
  public class MatrixCalc
  { int cnt = 0;
    public List<double[]> Matrix
    {  get
      { if (MathCalc.matrixs == null) MathCalc.matrixs = new List<double[]>();
        return MathCalc.matrixs;  }  }
    public List<double[]> Norm_Matrix
    { get
      { if (MathCalc.norm_matrixs != null) return MathCalc.norm_matrixs;
        MathCalc.norm_matrixs = new List<double[]>();
        double[] avg = new double[11];
        double[] std = new double[11];
        for (int i = 0; i < 11; i++)
        { IEnumerable<double> mas = from o in MathCalc.matrixs select o[i];
          avg[i] = mas.Average();
          std[i] = MathCalc.StdDev(mas); }
        foreach (double[] el in MathCalc.matrixs)
```

```
        { double[] mtr = new double[11];
          for (int j = 0; j < 11; j++)
          { mtr[j] = (el[j] - avg[j]) / std[j];  }
          MathCalc.norm_matrixs.Add(mtr);}
        return MathCalc.norm_matrixs;}}
    public List<double[]> Vector_Matrix
    { get
      { if (MathCalc.vectors != null)
          return MathCalc.vectors;
        double[,] arr = new double[11, 11];
        for (int i = 0; i < 11; i++)
          for (int j = 0; j < 11; j++)
          {arr[i, j] = Korel_Matrix[i][j];
            //arr[j, i] = MathCalc.korel_matrixs[i][j];}
        double[] wr = new double[11], wi = new double[11];
        double[,] vl = new double[11, 11], vr = new double[11, 11];
        double[,] pac = new double[11, 11];
        bool vect = alglib.smatrixevd(arr, 11, 1, true, out wr, out vl);
       //bool vect = alglib.rmatrixevd(arr, 11, 3, out wr, out wi, out vl, out vr);
        MathCalc.vectors = new List<double[]>();
        for (int i = 0; i < 11; i++)
        {double[] re = new double[11];
          for (int j = 0; j < 11; j++)
            re[j] = vl[i, j];
          MathCalc.vectors.Add(re);}
```

```
        double[] re2 = new double[11];
        for (int j = 0; j < 11; j++)
          re2[10 - j] = wr[j];
        MathCalc.vectors.Add(re2);
        return MathCalc.vectors;}}
    public List<double[]> Korel_Matrix
    {get
      {if (MathCalc.korel_matrixs != null) return MathCalc.korel_matrixs;
        MathCalc.korel_matrixs = new List<double[]>();
        double[] avg = new double[11];
        for (int j = 0; j < 11; j++)
        {IEnumerable<double> mas = from o in Norm_Matrix select o[j];
          avg[j] = mas.Average();}
        List<double[]> norm_matrixs_minus = new List<double[]>();
        foreach (double[] el2 in Norm_Matrix)
        {double[] ch = new double[11];
          for (int j = 0; j < 11; j++)
          {ch[j] = (el2[j] - avg[j]);}
          norm_matrixs_minus.Add(ch);}
        for (int indx = 0; indx < 11; indx++)
        {
          List<double[]> Ch = new List<double[]>();
          double Pow = (from o in norm_matrixs_minus select
Math.Pow(o[indx], 2)).Sum();
          double[] ch = new double[11];
```

```
            //ch[indx] = 1;
            for (int j = 0; j < 11; j++)
            {IEnumerable<double> Mult = norm_matrixs_minus.Select(o => o[indx]
* o[j]);
              IEnumerable<double>  Pow2  =  norm_matrixs_minus.Select(o  =>
Math.Pow(o[j], 2));
              ch[j] = Mult.Sum() / Math.Sqrt(Pow * Pow2.Sum());}
            MathCalc.korel_matrixs.Add(ch);}
          return MathCalc.korel_matrixs;}}
    public List<double[]> Factor_Matrix
    {get
      {if (MathCalc.multiply != null) return MathCalc.multiply;
        MathCalc.multiply        =        MathCalc.Multiply_matrix(Norm_Matrix,
Vector_Matrix);
        return MathCalc.multiply;}}
    public List<double[]> Disperse_Matrix
    {get
      {if (MathCalc.diperse_matrixs != null) return MathCalc.diperse_matrixs;
        MathCalc.diperse_matrixs = new List<double[]>();
        double[] std = new double[11];
        for (int i = 0; i < 11; i++)
        {double[] mas = (from o in Factor_Matrix select o[i]).ToArray();
          std[i] = MathCalc.Disp(mas);}
        MathCalc.diperse_matrixs.Add(std);
        return MathCalc.diperse_matrixs;}}
```

```
public List<double> Disperse_District_Matrix
{get
  {if (MathCalc.diperse_district_matrixs != null) return
MathCalc.diperse_district_matrixs;
    MathCalc.diperse_district_matrixs = new List<double>();
        int i = 0;
    foreach (double[] el in Norm_Matrix)
    {//double[] std = new double[1];
      //std[0] = ;
      MathCalc.diperse_district_matrixs.Add(MathCalc.Disp(el));
      i++;}
    //MathCalc.diperse_district_matrixs.Add(std);
    return MathCalc.diperse_district_matrixs;}}
public List<double[]> Korel_Factor_Pokaz_Matrix
{get
  {if (MathCalc.korel_factor_pokaz_matrix != null) return
MathCalc.korel_factor_pokaz_matrix;
    MathCalc.korel_factor_pokaz_matrix = new List<double[]>();
    for (int i = 0; i < 11; i++)
    {double[] mtr = new double[11];
      for (int j = 0; j < 11; j++)
      {IEnumerable<double> mas = from o in Norm_Matrix select o[i];
       IEnumerable<double> mas2 = from o in Factor_Matrix select o[j];
        mtr[j] = MathCalc.Korel(mas.ToArray(), mas2.ToArray());}
      MathCalc.korel_factor_pokaz_matrix.Add(mtr);}
```

```
        return MathCalc.korel_factor_pokaz_matrix;}}
    public List<double[]> Integral_Indicators
    {get
      {if (MathCalc.integral_indicators != null) return
MathCalc.integral_indicators;
        MathCalc.integral_indicators = new List<double[]>();
        double[] avg = new double[Factor_Matrix.Count];
        foreach (double[] el in Factor_Matrix)
        {int indx = Factor_Matrix.IndexOf(el);
          double sum = 0; double[] std = new double[2];
          for (int j = 0; j < 11; j++)
            sum += el[j] * (Disperse_Matrix[0][j] +
Disperse_District_Matrix[indx]) / 2;
          std[0] = sum / 11;
          std[1] = Disperse_District_Matrix[indx];
          avg[indx] = std[0];
          MathCalc.integral_indicators.Add(std);}
        MathCalc.AVG_II = Math.Round(avg.Average(), 5);
        return MathCalc.integral_indicators;}}
    public MatrixCalc(DistrictsWithParam PID)
    {int i = 0;
      cnt = PID.Count;
      ClearMatrixs();
      foreach (NamedIdInfoParam el in PID)
      {double[] mtr = new double[11];
```

```
        for (int j = 0; j < 11; j++)
        {  switch (j)
           {case 0: mtr[j] = el.ArrP[1] != 0 ? el.ArrP[6] / (el.ArrP[1] / 1000) : 0;
break;
              case 1: mtr[j] = el.ArrP[1] != 0 ? 1000 * el.ArrP[8] / el.ArrP[1] : 0;
break;
              case 2: mtr[j] = el.ArrP[11] - el.ArrP[12]; break;
              case 3: mtr[j] = el.ArrP[14]; break;
              case 4: mtr[j] = el.ArrP[16]; break;
              case 5: mtr[j] = el.ArrP[1] != 0 ? 1000 * el.ArrP[17] / el.ArrP[1] : 0;
break;
              case 6: mtr[j] = el.ArrP[1] != 0 ? el.ArrP[4] / el.ArrP[1] : 0; break;
              case 7: mtr[j] = el.ArrP[1] != 0 ? el.ArrP[20] / el.ArrP[1] : 0; break;
              case 8: mtr[j] = el.ArrP[1] != 0 ? el.ArrP[22] / el.ArrP[1] : 0; break;
              case 9: mtr[j] = el.ArrP[25]; break;
              case 10: mtr[j] = el.ArrP[26]; break;
              default: mtr[j] = 0; break;}}
           Matrix.Add(mtr);
           i++;}
        double[] max = new double[11];
        for (i = 0; i < 11; i++)
           max[i] = (from o in Matrix select o[i]).Max();
        for (i = 0; i < PID.Count; i++)
        {Matrix[i][2] = max[2] < 0 ? Matrix[i][2] - max[2] : max[2] - Matrix[i][2];
           Matrix[i][3] = max[3] - Matrix[i][3];
```

```
            Matrix[i][9] = max[9] - Matrix[i][9];
            Matrix[i][10] = max[10] - Matrix[i][10];}}
    void ClearMatrixs()
    {MathCalc.matrixs = null;
      MathCalc.norm_matrixs = null;
      MathCalc.korel_matrixs = null;
      MathCalc.vectors = null;
      MathCalc.multiply = null;
      MathCalc.diperse_matrixs = null;
      MathCalc.korel_factor_pokaz_matrix = null;
      MathCalc.integral_indicators = null;
      MathCalc.diperse_district_matrixs = null;}}
public class StrAndDouble
{private double[] _data;
    private string _description;
    public StrAndDouble() { }
    public StrAndDouble(double[] data, string descript)
    {_data = data;
      _description = descript;}
    public StrAndDouble(double data, string descript)
    {_data = new double[1];
      _data[0] = data;
      _description = descript;}
    public double[] Data
    {get { return _data; }
```

```
    set { _data = value; }}
  public string Description
  {get { return _description; }
    set { _description = value; }}}
public class ChartSource
{public ChartSource(string name, double value)
  {Names = name;
    Value = value;
    if (value > 0) ColorBrush = this.CreateBrush("#FF8EBC00");
    else ColorBrush = this.CreateBrush("#FFE61E26");}
  public string Names { get; set; }
  public double Value { get; set; }
  public Brush ColorBrush { get; set; }
  private Brush CreateBrush(string color)
  {return new SolidColorBrush(this.GetColorFromHexString(color));}
  private Color GetColorFromHexString(string s)
  {s = s.Replace("#", string.Empty);
    byte a = System.Convert.ToByte(s.Substring(0, 2), 16);
    byte r = System.Convert.ToByte(s.Substring(2, 2), 16);
    byte g = System.Convert.ToByte(s.Substring(4, 2), 16);
    byte b = System.Convert.ToByte(s.Substring(6, 2), 16);
    return Color.FromArgb(a, r, g, b);}}
public class ArrayDouble : IEnumerable<double[]>
{private IEnumerable<double[]> _data;
  public ArrayDouble()
```

```
    public ArrayDouble(IEnumerable<double[]> data)
    {_data = data;}
    public IEnumerable<double[]> Data
    {get { return _data; }
      set { _data = value; }}
    #region IEnumerable<double[]> Members
    public IEnumerator<double[]> GetEnumerator()
    {if (_data == null)
        throw new ArgumentException("Data cannot be null.", "Data");
      int len2d = 11;
      foreach (double[] el in _data)
      {double[] arr = new double[len2d];
        for (int j = 0; j < len2d; j++)
        {arr[j] = Math.Round(el[j], 5);}
        yield return arr;}}
    #endregion
    #region IEnumerable Members
    System.Collections.IEnumerator
System.Collections.IEnumerable.GetEnumerator()
    {return this.GetEnumerator();}
    #endregion}
  public class ArrayStrDouble : IEnumerable<StrAndDouble>
  {private IEnumerable<StrAndDouble> _data;
    public ArrayStrDouble()
```

```
    public ArrayStrDouble(IEnumerable<double[]> data, IEnumerable<string>
descr)
    {List<StrAndDouble> arr = new List<StrAndDouble>();
      int indx = 0;
      foreach (double[] el in data)
      {arr.Add(new StrAndDouble(el, (descr as string[])[indx]));
        indx++;}
      _data = arr;  }
    public ArrayStrDouble(IEnumerable<double> data, IEnumerable<string>
descr)
    {List<StrAndDouble> arr = new List<StrAndDouble>();
      int indx = 0;
      foreach (double el in data)
      {arr.Add(new StrAndDouble(el, (descr as string[])[indx]));
        indx++;}
      _data = arr;}
    public IEnumerable<StrAndDouble> Data
    {get { return _data; }
      set { _data = value; }}
    #region IEnumerable<double[]> Members
    public IEnumerator<StrAndDouble> GetEnumerator()
    {  if (_data == null)
        throw new ArgumentException("Data cannot be null.", "Data");
      //int len2d = (Data as Array).Length;
      foreach (StrAndDouble el in _data)
```

```
        {int len2d = el.Data.Length;
            double[] arr = new double[len2d];
            for (int j = 0; j < len2d; j++)
            {arr[j] = Math.Round(el.Data[j], 5);}
            StrAndDouble SAD = new StrAndDouble(arr, el.Description);
            yield return SAD;}}
    #endregion
    #region IEnumerable Members
    System.Collections.IEnumerator
System.Collections.IEnumerable.GetEnumerator()
    {return this.GetEnumerator();   }
    end region  }}
```