ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ І. І. МЕЧНИКОВА

(повне найменування закладу вищої освіти)

Факультет математики, фізики та інформаційних технологій

(повне найменування факультету)

Кафедра інформаційних технологій

(повна назва кафедри)

Кваліфікаційна робота

на здобуття ступеня вищої освіти «Магістр»

«Програмна реалізація інтерактивного інформаційного застосунку для системи контролю розповсюдження вірусної інфекції**»**

(тема кваліфікаційної роботи українською мовою)

«Software implementation of an interactive information application for the system of controlling the spread of a viral infection»

(тема кваліфікаційної роботи англійською мовою)

Виконала: здобувачка заочної форми навчання

спеціальності 122 Комп'ютерні науки

(код, назва спеціальності)

Освітня програма Комп'ютерні науки

(назва)

Стафік Ірина Олександрівна

(прізвище, ім'я, по-батькові здобувача)

Керівник д.т.н., доцент Великодний С.С. _____

(науковий ступінь, вчене звання, прізвище, ініціали) (підпис)

Рецензент д.т.н., доцент, професор кафедри кібербезпеки НУ "Одеська юридична академія", Соколов Артем Валерійович

(науковий ступінь, вчене звання, прізвище, ініціали)

Рекомендовано до захисту:
Протокол засідання кафедри
Інформаційних технологій
№ ___ від ___ _____ 2024 р.

Завідувачка кафедри

_____ _____
(підпис) (прізвище,ім'я)

Захищено на засіданні ЕК № _____
протокол № ___ від _____ 2024 р.

Оцінка _____ / _____ / _____
(за національною шкалою/шкалою ECTS/ бали)

Голова ЕК

_____ _____
(підпис) (прізвище,ім'я)

Одеса 2024

АНОТАЦІЯ

У кваліфікаційній роботі розробляється тема «Програмна реалізація інтерактивного інформаційного застосунку для системи контролю розповсюдження вірусної інфекції».

Мета роботи. Розробити діючий прототип інформаційної системи у вигляді програмної реалізації інтерактивного інформаційного застосунку, в якій основним джерелом інформації є сама людина. Така система дозволяє в реальному часі відстежувати епідеміологічну ситуацію, швидко ідентифікувати нові спалахи, оцінювати ефективність карантинних заходів та виявляти осіб, які можуть бути потенційними носіями інфекції.

Для того щоб ефективно протидіяти поширенню вірусу, важливо забезпечити оперативний збір, обробку та аналіз даних. Це забезпечено у розробленому інтерактивному інформаційному застосунку, який включає інформацію про нові випадки інфікування чи підозру на захворювання.

На основі зібраних даних було побудовано інформаційну систему у вигляді веб-інтерфейсу з Django та використано бібліотеки на основі HTML5, CSS та JS для фундаментального дизайну сторінок інтерфейсу користувача з метою максимальної сумісності із браузером.

Вирішено використовувати Visual Studio Code як IDE, оскільки він доступний як безкоштовне програмне забезпечення та має широкий спектр вбудованої підтримки бібліотек. Завдяки підтримці мови Python, вбудованій у Visual Studio Code, при розробці ця мова використовується без потреби у сторонніх бібліотеках.

У результаті як наукові дослідження, так і теоретичне розуміння були посилені та розширені. Для усвідомлення особливостей тематичної області створено прототип інформаційної системи, що підкріплює концепцію запропонованих рішень.

ABSTRACT

In the qualification work, the topic «Software implementation of an interactive information application for the system of controlling the spread of a viral infection» is developed.

The purpose of the work. Develop a working prototype of an information system in the form of a software implementation of an interactive information application, in which the main source of information is the person himself. Such a system allows monitoring the epidemiological situation in real time, quickly identifying new outbreaks, evaluating the effectiveness of quarantine measures, and identifying persons who may be potential carriers of the infection.

In order to effectively counter the spread of the virus, it is important to ensure prompt data collection, processing and analysis. This is provided in the developed interactive information application, which includes information about new cases of infection or suspicion of the disease.

Based on the collected data, an information system was built in the form of a web interface with Django and libraries based on HTML5, CSS and JS were used for the fundamental design of the user interface pages with the aim of maximum compatibility with the browser.

It was decided to use Visual Studio Code as the IDE because it is available as free software and has a wide range of built-in library support. Thanks to the support for the Python language built into Visual Studio Code, development uses this language without the need for third-party libraries.

As a result, both scientific research and theoretical understanding have been strengthened and expanded. To understand the peculiarities of the thematic area, a prototype of the information system was created, which reinforces the concept of the proposed solutions.

ЗМІСТ

Стор.

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ І ТЕРМІНІВ

ВООЗ – Всесвітня організація охорони здоров’я

ДНК – дезоксирибонуклеїнова кислота

ІС – інформаційні системи

ІТ – інформаційні технології

ООН – Організація Об’єднаних Націй

ПЛР – полімеразна ланцюгова реакція

ADF – Augmented Dickey-Fuller test

ARIMA – Autoregressive Integrated Moving Average

CDC – Centers for Disease Control

HQS – Home Quarantine System

MSEIR – Maternally derived immunity – Susceptible – contact – Infected – Recovered

NSSN – NSW Smart Sensing Network

RT-PCR – Real-time polymerase chain reaction

WHO – World Health Organization

ВСТУП

Пандемія коронавірусу стала безпрецедентним викликом для сучасної світової спільноти. Вона не тільки загрожувала здоров’ю мільярдів людей, але й призвела до масштабних змін у соціальному, економічному та політичному житті. В умовах глобальної кризи, спричиненої стрімким поширенням вірусу, стало очевидно, що однією з основних проблем, з якою зіткнулися уряди країн, є ефективний моніторинг та контроль інфекції. Ключовим елементом у подоланні пандемії є швидка та надійна інформація про стан захворюваності, джерела інфікування та поширення вірусу. У цьому контексті виникає необхідність створення комплексної інформаційної системи (ІС) для контролю інфікування коронавірусом.

Інформаційні технології (ІТ) відіграють критичну роль у боротьбі з пандемією. Без швидкого доступу до точної інформації про поточний стан захворюваності та динаміку поширення вірусу уряди та медичні установи позбавлені можливості приймати обґрунтовані рішення щодо заходів контролю та обмеження. За допомогою таких систем можливо координувати зусилля не лише на національному, але й на міжнародному рівні, що особливо важливо в умовах глобалізації та високої мобільності населення.

Без систематичного та централізованого контролю даних про інфекцію неможливо побудувати точну картину поширення вірусу. Уряди країн, які стикаються з обмеженими ресурсами охорони здоров'я, можуть вчасно не вжити необхідних заходів або, навпаки, впроваджувати занадто жорсткі обмеження, які завдають шкоди економіці та соціальному життю. Тому, створення інформаційної системи для контролю інфікування коронавірусом є не лише технологічним завданням, а й стратегічною необхідністю для забезпечення безпеки населення та стабільності суспільства.

Актуальність теми. Для того щоб ефективно протидіяти поширенню вірусу, важливо забезпечити оперативний збір, обробку та аналіз даних. Це буде забезпечено у розроблюваному інтерактивному інформаційному застосун-

ку, який включає інформацію про нові випадки інфікування чи підозру на захворювання.

Мета роботи. Розробити діючий прототип інформаційної системи у вигляді програмної реалізації інтерактивного інформаційного застосунку, в якій основним джерелом інформації є сама людина. Така система дозволить в реальному часі відстежувати епідеміологічну ситуацію, швидко ідентифікувати нові спалахи, оцінювати ефективність карантинних заходів та виявляти осіб, які можуть бути потенційними носіями інфекції.

Для досягнення мети роботи, така ІС контролю інфікування повинна забезпечувати виконання низки завдань, серед яких:

– збір даних у реальному часі: автоматичний або напівавтоматичний збір інформації з різних джерел, таких як медичні установи, тестові лабораторії, пункти вакцинації та державні установи;

– аналітика та прогнозування: використання алгоритмів машинного навчання та штучного інтелекту для прогнозування подальшого поширення вірусу, оцінки ризиків нових спалахів та моделювання сценаріїв для прийняття рішень;

– взаємодія з населенням: через мобільні додатки або веб-портали громадяни можуть отримувати інформацію про можливі контакти з хворими, рекомендації щодо дій та своєчасні сповіщення про необхідність тестування або самоізоляції;

– міжнародна співпраця: ІС може об'єднувати зусилля різних країн для обміну інформацією про інфікування, що допоможе координувати дії на глобальному рівні.

1 АНАЛІЗ ОБРАНОЇ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Аналіз методів поширення та досвіду розповсюдження

Коронавірус отримав свою назву від білків-шипів, які виступають із них і мають вигляд корони. Ці спайкові білки мають вирішальне значення для біології вірусу [1]. Спайковий білок – це компонент вірусу, який з'єднується з клітиною людини, щоб інфікувати її, дозволяючи вірусу розмножуватися всередині клітини та поширюватися на інші клітини. Націлюючись на ці спайкові білки, деякі антитіла можуть захистити від SARS-CoV-2. Через важливість цієї специфічної частини вірусу вчені, які секвенують вірус для повторного пошуку, використовують процедуру, відому як геномний моніторинг, щоб постійно стежити за мутаціями, які викликають зміни спайкового білка [2].

Вірус SARS-CoV-2 починає створювати генетичні лінії, оскільки з часом у вірусі відбуваються генетичні зміни. Вірус SARS-CoV-2, як і будь-який інший, має генеалогічне дерево, яке можна простежити. Іноді різні гілки цього дерева мають різні характеристики, які впливають на швидкість поширення вірусу, тяжкість хвороби, яку він викликає, або ефективність противірусних ліків. Вчені називають ці віруси «варіантами». Вони все ще SARS-CoV-2, хоча їх поведінка може змінитися [3].

Подивимося, як розвивалася пандемія COVID-19, які заходи запроваджували країни та до якого результату прийшли.

31 грудня 2019 р. Всесвітня організація охорони здоров'я (ВООЗ) отримала повідомлення про випадки пневмонії в Ухані (Китай) без відомої причини. Влада Китаю визначила новий коронавірус 2019-nCoV як причину цих інфекцій. 7 січня. Через кілька тижнів 30 січня 2020 р. ВООЗ визнала спалах COVID-19 надзвичайною ситуацією в галузі охорони здоров'я, що має міжнародне занепокоєння. Однак наступного місяця, 11 лютого, новий коронавірус отримав своє офіційне позначення – COVID-19. Центри з контролю

та профілактики захворювань США (Centers for Disease Control – CDC) підтвердили першу смерть від COVID-19 в країні через дев'ять днів. Чоловікові було за п'ятдесят і він проживав у штаті Вашингтон.

У перші місяці COVID-19 офіційні особи з охорони здоров'я, урядові організації та широка громадськість не знали, як хвороба поширюється та порушить повсякденне життя. 1 березня 2020 р. Організація Об'єднаних Націй (ООН) виділила 15 мільйонів доларів США на допомогу у боротьбі з COVID-19 у всьому світі. Через тиждень, 7 березня, кількість випадків COVID-19 перевищила 100 000. Через кілька днів, 11 березня, ВООЗ оголосила COVID пандемією. COVID-19 практично миттєво перетворився з важкої хвороби, яка, здавалося, була поширена лише в Китаї, до всесвітньої катастрофи в галузі охорони здоров'я (рис. 1.1).

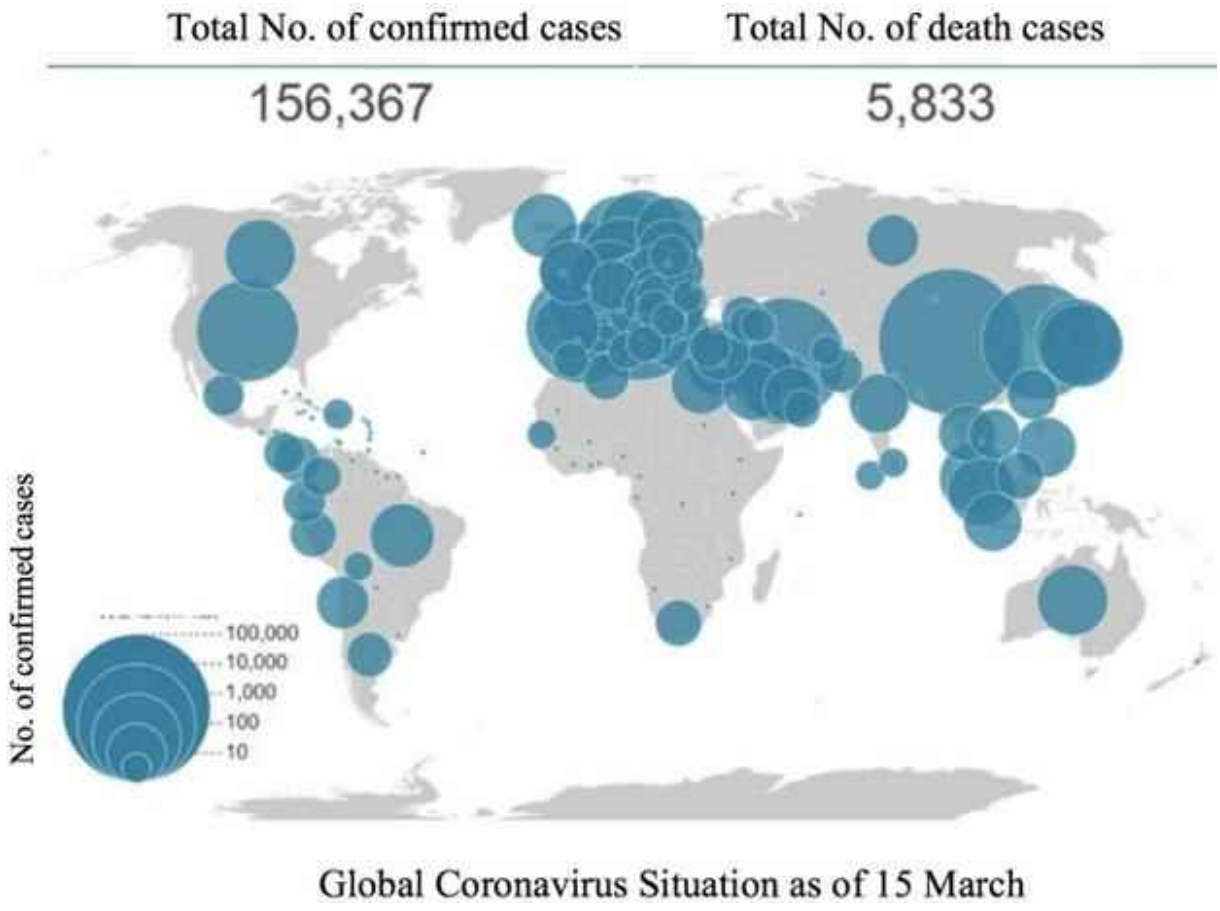

Рисунок 1.1 – Діаграма розповсюдження випадків COVID-19 станом на момент оголошення пандемії

До цього часу ситуація в Ухані була розсіяною після введення безпрецедентних заходів для стримування вірусу. На початку спалаху Китай повідомляв про тисячі нових випадків на день, які скоротилися до десятків до березня. З іншого боку, в Європі кількість випадків стрімко зростала з кожним днем, а в Італії було зафіксовано безпрецедентні 250 смертей за 24 години з 12 по 13 березня. У результаті 13 березня ВООЗ оголосила, що епіцентром пандемії стала Європа. Того ж дня у США було оголошено надзвичайний стан (рис. 1.2).

Рисунок 1.2 – Правила Техасу щодо COVID-19

Для боротьби з епідемією в усьому світі були введені суворі запобіжні заходи. У березні набули чинності соціальне дистанціювання та обмеження на поїздки у громадському транспорті, а також вказівки щодо правильного миття рук (рис. 1.3).

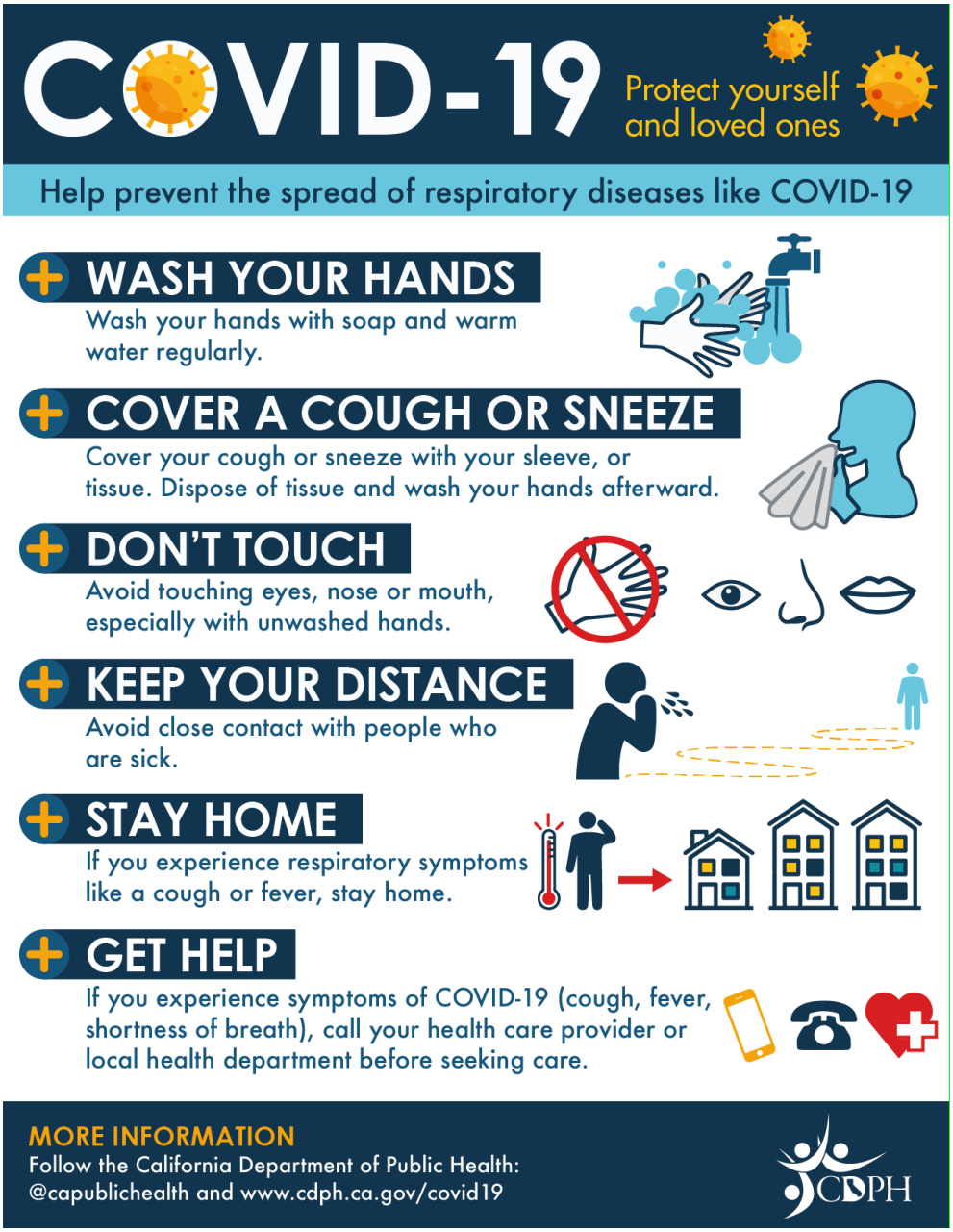

Рисунок 1.3 – Рекомендації щодо запобігання поширенню захворювань

Однак експерти прогнозували, що ці зусилля лише затримають поширення вірусу [4], і для того, щоб перемогти пандемію, потрібно буде створити

вакцину. Перші випробування імунізації людей проти COVID-19 з використанням мРНК-вакцини Moderna розпочнуться 17 березня 2020 р.

Початкових заходів було явно недостатньо, щоб зупинити поширення COVID-19. Більшість регіонів швидко посилили обмеження, а Велика Британія прийняла наказ залишатися вдома 26 березня. Приблизно у цей період багато європейських країн запровадили власний національний карантин. Станом на 2 квітня загальна кількість випадків COVID-19 у світі зросла до 1 мільйона.

Завдяки цій статистиці стала зрозумілою справжня серйозність епідемії, і уряди зробили все можливе, щоб уповільнити поширення вірусу, доки вакцину не буде визнано безпечною для використання. 6 квітня ВООЗ опублікувала рекомендації щодо носіння масок, оскільки нові дослідження вказали на важливість аерозольного шляху у поширенні хвороби.

Багато країн спостерігали зниження кількості інфекцій, госпіталізацій і летальних випадків протягом літа 2020 р. в результаті обмежень, накладених на їхніх жителів, щоб запобігти поширенню вірусу. Однак різновид Lambda була ідентифікована в Перу наприкінці літа, у серпні 2020 р. За даними ВООЗ, цей різновид поширився щонайменше у 29 країнах.

Альфа-версію виявили у Великобританії у вересні 2020 р., через місяць. Виявлення цих варіацій було важливим, оскільки воно демонструвало, що вірус змінювався (рис. 1.4).

В результаті змінилися симптоми і наслідки захворювання. Наприклад, дані свідчать про те, що варіація «Альфа» може збільшити ймовірність поганих результатів щодо COVID-19. З появою цих нових різновидів кількість випадків COVID-19 знову почала зростати в багатьох країнах, і станом на 29 вересня 2020 р. було зафіксовано один мільйон летальних випадків від COVID-19.

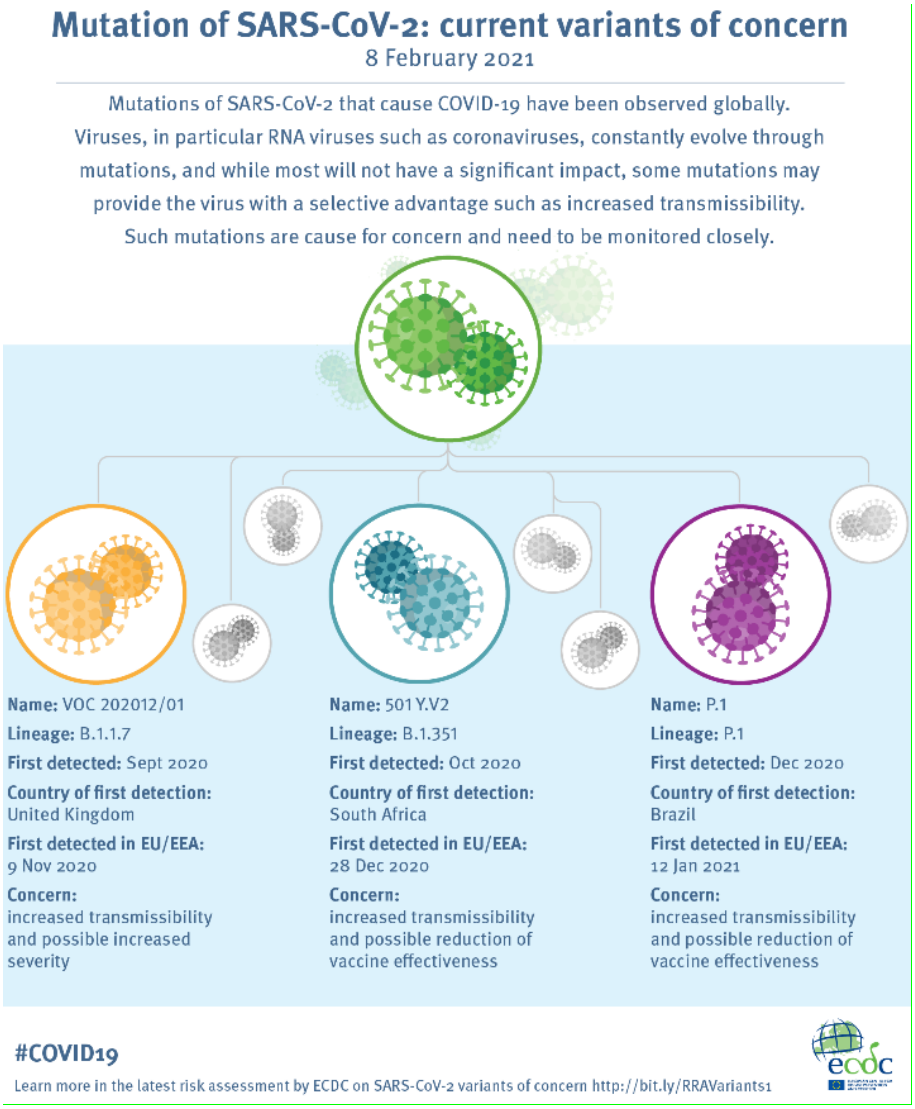

Рисунок 1.4 – Варіанти мутацій COVID

1.2 Огляд існуючих методів стримування інфікування

Під час пандемії важливо залишатися в безпеці та не наражати на небезпеку інших. Щоб запобігти поширенню захворювання, країни вводять обмеження та заходи особистої безпеки, за допомогою яких є шанс знизити рівень зараження та поширення інфекції. Деякі заходи передбачають лише особисті рекомендації, інші охоплюють всю державу та змінюють спосіб життя людей.

Загальні рекомендації:

– соціальне дистанціювання;

– носіння масок;

– карантин;

– щеплення;

– моніторинг стану здоров'я.

Як наслідок: запропоновані варіанти ґрунтуються на прагматичності та враховують необхідність збереження важливих соціальних функцій. Додатковий ризик передачі, місцеві епідеміологічні умови, можливість тестування та відстеження контактів, а також соціально-економічні наслідки пандемії в конкретному середовищі – усе це слід брати до уваги при прийнятті рішення про зміну рекомендацій щодо карантину та ізоляції.

Всесвітня організація охорони здоров’я (ВООЗ) – World Health Organization (WHO) рекомендує три типи масок для населення:

– багаторазові немедичні маски, які відповідають стандарту ASTM F3502 або робочій угоді CEN 17553, або немедичні маски, які відповідають ключовим вимогам ВООЗ (рис. 1.5);

– одноразові маски, які відповідають стандартам медичних масок EN 14683, тип I, ASTM F2100, рівень 1, YY/T 0969, YY 0469 (або аналогічні);

– якщо немає інших рішень, прийнятною альтернативою є різні форми добре підігнаних немедичних масок, включаючи багатошарові маски ручної роботи.

Рисунок 1.5 – Маска за стандартом ASTM F3502-21

Одноразові медичні маски також рекомендують використовувати наступним категоріям людей, оскільки вони з більшою ймовірністю захворіють на COVID-19 у критичному стані та помруть у разі зараження:

– особи старше 60 років;

– люди будь-якого віку, які мають основні проблеми зі здоров'ям, такі як хронічні респіраторні захворювання, серцево-судинні захворювання, рак, ожиріння;

– люди з ослабленим імунітетом і цукровим діабетом 2-го типу.

Щоб запобігти подальшій передачі, карантин – це метод ізоляції тих, хто мав тісний контакт із хворим на COVID-19, від інших осіб. На час зараження пацієнти з підтвердженою чи підозрою на COVID-19 мають бути ізольовані від інших людей.

Незважаючи на те, що COVID-19 зустрічається рідше, ніж особистий контакт з людиною, CDC стверджує, що COVID-19 може поширюватися аерозольним шляхом (як було зазначено раніше). Відповідно до CDC, певні захворювання можуть поширюватися через контакт з вірусом повітряно-крапельним шляхом і частинками, які тривають хвилину або годину [1]. Ці віруси можуть поширюватися на інших людей, які перебувають на відстані більше 2-х м від ураженої людини або після того, як вони покинули територію. Ці трансляції відбувалися в тісних приміщеннях з невеликою вентиляцією.

Таким чином, важливо, щоб використання масок було частиною всеохоплюючого плану дій, щоб зупинити поширення хвороби та врятувати життя. Ефективний ступінь захисту від COVID-19 неможливо забезпечити лише носінням маски. Оскільки тісний контакт від людини до людини є одним із основних джерел передачі, соціальне дистанціювання залишається ключовим способом пом’якшення поширення. CDC рекомендує дотримуватися відстані приблизно 2 м від інших у громадських місцях [2]. Ця відстань допоможе вам уникнути прямого контакту з дихальними краплями, які утворюються під час кашлю чи чхання.

Важливо часто мити руки з милом і водою протягом принаймні 20 секунд, особливо після перебування в громадському місці або після сморкання, кашлю чи чхання. Це не тільки зменшує можливість поширення вірусу через предмети та поверхні, але й допомагає захистити нашу імунну систему від інших захворювань, які можуть зробити уразливими до COVID [3].

Всі перераховані заходи більшою мірою є рекомендаціями для особистої безпеки. Контроль таких заходів надзвичайно складний, тому немає високої впевненості, що ці заходи будуть виконані. З цієї та багатьох інших причин країни вводять додаткові обмеження, які хоч і ускладнюють життя та пересування людей, але дозволяють ефективніше запобігати поширенню вірусу, застосовуючи захисні заходи завчасно, коли симптоми хвороби виявлено.

Одним із основних заходів, які запровадили багато країн, є так звана «домашня ізоляція». Цей запобіжний захід передбачав двотижневу ізоляцію особи, яка прибула в країну. Два тижні карантину за тривалістю відповідають інкубаційному періоду коронавірусної інфекції – часу, коли хвороба може проявити перші симптоми. Особам, які контактували з хворими, а також тим, хто повернувся з відрядження або перебував у відрядженні, необхідно повідомити про повернення до штабу з контролю за коронавірусною інфекцією та дотримуватися карантину вдома чи у визначеному місці на 14 днів. Якщо людина перебувала в неблагополучних за COVID-19 країнах з друзями чи сім’єю, можна дотримуватися спільного двотижневого карантину в одній кімнаті чи квартирі.

Для контролю за карантином можуть використовуватися електронні та технічні засоби контролю. Усі, хто перебуває на карантині, перебувають під медичним наглядом і щодня міряють температуру. На 10-й день карантину лікарі відбирають матеріал для дослідження на COVID-19 (мазок з носа або носоглотки). Під час карантину людина може спілкуватися зі своїми друзями та рідними за допомогою:

– відеозв’язку;

– аудіозв’язку;

– Інтернету.

Головна вимога – не залишати місце ізоляції до закінчення карантину.

Під час карантину також дуже важливо обробляти шкіру антисептиками – перед їжею, перед контактом зі слизовими оболонками очей, рота, носа, після відвідування туалету. Крім того, варто регулярно провітрювати приміщення і проводити вологе прибирання з використанням побутової хімії з миючим або мийно-дезінфікуючим ефектом.

1.3 Методи тестування та ідентифікації

Крім того, різні країни також запроваджують вимоги щодо наявності ПЛР-тесту з негативним результатом, експрес-тесту на антиген або довідки про щеплення перед в'їздом до країни. Це допомагає заздалегідь виявити вірусну інфекцію і вжити відповідних заходів.

Полімеразна ланцюгова реакція (або ПЛР) – експериментальний метод молекулярної біології, метод значного збільшення низьких концентрацій певних фрагментів дезоксирибонуклеїнової кислоти (ДНК) у біологічному матеріалі (пробі). Кері Малліс розробив цей метод тестування ще у 1983 р. Згодом він отримав за цей винахід Нобелівську премію [4]. На сьогоднішній день ПЛР-діагностика є одним з найбільш точних і чутливих методів діагностики інфекційних захворювань. Метод ПЛР заснований на багаторазовому подвоєнні певної ділянки ДНК за допомогою ферментів у штучних умовах. У результаті утворюється достатня кількість ДНК для візуального виявлення. У цьому випадку копіюється тільки та область, яка задовольняє заданим умовам, і тільки якщо вона присутня в досліджуваному зразку.

Крім простого збільшення кількості копій ДНК (цей процес називається ампліфікацією), ПЛР дозволяє проводити багато інших маніпуляцій з генетичним матеріалом (введення мутацій, сплайсінг фрагментів ДНК), і широко використовується в біологічній і медичній практиці, наприклад, діагно-

стувати захворювання (спадкові, інфекційні), встановлювати батьківство, клонувати гени, вносити мутації, виділяти нові гени.

Цей метод молекулярної діагностики, який став «золотим стандартом» для ряду інфекцій, перевірений часом і ретельно клінічно перевірений. Метод ПЛР дозволяє визначити наявність збудника захворювання, навіть якщо в зразку присутні лише кілька молекул ДНК збудника [5]. ПЛР дозволяє діагностувати наявність тривало зростаючих збудників, не вдаючись до трудомістких мікробіологічних методів, що особливо важливо в гінекології та урології при діагностиці урогенітальних інфекцій, що передаються статевим шляхом. Також цей метод використовується для діагностики вірусних інфекцій, таких як гепатити, ВІЛ, коронавірус COVID-19 та ін. Чутливість методу значно перевищує імунохімічні та мікробіологічні методи, а принцип методу дозволяє діагностувати наявність інфекцій зі значною антигенною варіабельністю.

Специфічність ПЛР при використанні технології ПЛР навіть для всіх вірусних, хламідійних, мікоплазмених, уреаплазмових і більшості інших бактеріальних інфекцій досягає 100%. Метод ПЛР дозволяє виявити навіть поодинокі клітини бактерій або вірусів. ПЛР-діагностика виявляє наявність збудників інфекційних захворювань у випадках, коли це неможливо зробити іншими методами (імунологічним, бактеріологічним, мікроскопічним).

Метод ПЛР особливо ефективний для діагностики важко культивованих, некультивованих і латентних форм мікроорганізмів, які часто зустрічаються при латентних і хронічних інфекціях, оскільки цей метод дозволяє уникнути труднощів, пов'язаних з вирощуванням таких мікроорганізмів в лабораторії.

Тест на антиген є одним із тестів, який використовують для діагностики коронавірусу. Він дає змогу швидко та точно виявити антиген SARS-CoV-2 у зразках мазка з носа чи носоглотки (залежно від типу дослідження) та встановити попередній діагноз COVID-19. Негативний результат не може вважатися винятком щодо інфекції SARS-CoV-2. Тест на антиген використо-

вується завдяки можливості отримати результат швидше, ніж у випадку класичного ПЛР-тесту (приблизно 10 – 30 хвилин проти, зазвичай, 24 годин).

Тест на антиген в назофарингеальному матеріалі виявляє специфічні для вірусу білки, що утворюються під час його реплікації, на відміну від ПЛР-тестів, які виявляють у зразку генетичний матеріал вірусу. Тепер ВООЗ дозволила проводити тести на антигени, коли генетичні тести (молекулярні, наприклад, RT-PCR (Real-time polymerase chain reaction) та FRANKD) обмежені в доступності або коли час, необхідний для їх виконання, обмежує їхню клінічну корисність [6].

Схвалені ВООЗ тести на антигени повинні мати чутливість ≥ 80% і специфічність ≥ 97%. На даний момент рекомендується перевіряти як позитивні, так і негативні результати за допомогою молекулярного тесту (RT-PCR, ПЛР). Тест проводиться аналогічно ПЛР-тестам на основі мазку з носа або носоглотки (залежно від тесту). Після взяття мазка зразок матеріалу поміщають в лунку на експериментальній пластині і через певний час зчитують. Час виконання залежить від постачальника тестів, але результати зазвичай доступні через 10 – 30 хвилин.

Однак варто пам’ятати, що тест на антиген обмежується початковою стадією захворювання – до 5 – 7 днів після появи симптомів, що свідчать про COVID-19, оскільки саме тоді відбувається інтенсивне розмноження вірусу (тобто розмножується). Немає даних для оцінки використання цього тесту в безсимптомних популяціях. На пізніх стадіях COVID-19 концентрація вірусу може зменшуватися, і тести можуть не виявити його присутності.

Паспорт вакцинації – це міжнародне свідоцтво, яке засвідчує, що ви пройшли вакцинацію двома дозами прийнятної вакцини для профілактики або негативний тест на COVID-19. Цей паспорт називається Excelsior Pass і відрізняється від паперової картки вакцинації, яка видається після щеплень. Подібно до електронного квитка, кожен Excelsior Pass має унікальний QR-код безпеки, який можуть просканувати державні установи чи організації,

щоб підтвердити, що особа була вакцинована проти COVID-19, перш ніж її допустять.

Це «електронний паспорт», який є безкоштовним і безпечним способом підтвердити вакцинацію або негативний тест на COVID-19.

Існує три види паспортів:

– проти COVID-19 – для осіб, які пройшли повний курс щеплень проти COVID-19;

– на ПЛР-тест на COVID-19 – для людей з негативним результатом тесту на COVID-19;

– на тест до антитіл до COVID-19 – для осіб з негативним результатом експрес-тесту на COVID-19.

Крім того, багато країн можуть запроваджувати додаткові локальні обмеження на:

– відвідування громадських місць;

– скупчення великої кількості людей тощо.

Тому в джерелах новин часто можна зустріти подібну інформацію:

– Данія першою у ЄС зняла всі обмеження щодо COVID-19, але потім повернула їх знову. У січні знову було знято низку обмежень: кінотеатри, музеї та театри знову зможуть приймати відвідувачів. Вхід можливий за наявності довідки про вакцинацію від COVID-19, перенесеної хвороби або негативного результату тесту на коронавірус;

– У Чехії відмовилися від ідеї обов'язкової вакцинації, що дозволить уникнути «поглиблення розколу» в суспільстві;

– бари та ресторани Норвегії знову зможуть продавати алкогольні напої. Щоправда, до 23:00 за місцевим часом. Також молодь зможе брати участь у спортивних змаганнях, а карантин у багатьох випадках буде замінено тестуванням;

– з 15 січня Нідерланди послабили заходи, запроваджені для боротьби з поширенням COVID-19. Магазинам, спортзалам, культурним закладам і контактним службам дозволяється працювати до 17.00 щодня.

1.4 Застосування інформаційних систем для забезпечення режиму карантину

При запровадженні особистого карантину важливо переконатися, що вимоги повністю дотримані, для цього використовувалися інформаційні системи (ІС). З їх допомогою створено системи відстеження та контролю за місцезнаходженням потенційно інфікованих людей у місцях ізоляції. Такі ІС називаються «системами моніторингу» і часто застосовуються не лише до осіб, які прибули в країну з-за кордону, а й до осіб, у яких виявлено симптоми потенційного зараження хворобою. Однією з таких систем є «Дій вдома». Ця система моніторингу за допомогою мобільного додатку контролює людину, її місцезнаходження та підтверджує факт перебування на карантині [7].

«Дій вдома» – додаток для тих людей, які перебувають у групі ризику через коронавірус (наприклад, нещодавно повернулися з країни, де є спалах, або контактували з хворою людиною). За його допомогою влада може перевірити, як така особа дотримується самоізоляції чи обсервації. Суть його роботи полягає в тому, що користувач отримує на смартфон push-повідомлення з нагадуванням зробити та відправити селфі. Сервіс перевіряє геолокацію і таким чином стежить, чи сидить людина вдома (рис. 1.6).

Додаток розроблено Мінцифрою у березні 2020 р. для контролю за дотриманням самоізоляції людьми, які приїжджають в Україну з-за кордону. Ті, хто перетинає кордон України, повинні встановити додаток і позначати своє місцезнаходження щодня протягом 14 днів. Згодом у застосунку додали можливість дострокового виходу з самоізоляції за наявності негативного результату ПЛР-тесту на коронавірус. Зокрема, цей додаток мав використовуватися для встановлення контактних осіб. Проте законним шляхом змусити громадян щось встановити на телефон неможливо. Тому, згодом, авторизуватися у додатку потрібно було лише тим, хто приїжджає в Україну з-за кордону.

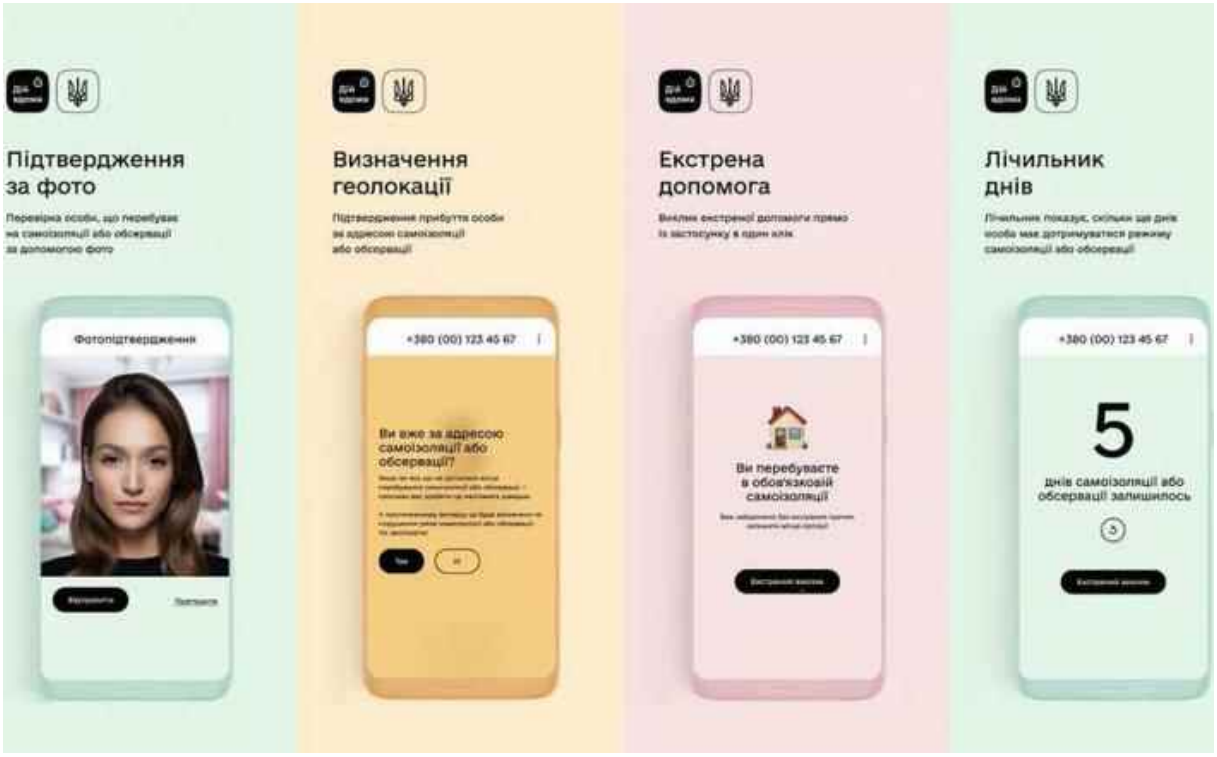

Рисунок 1.6 – Інтерфейс інформаційної системи «Дій вдома»

Офіційно можна уникнути необхідності встановлення додатку, якщо у людини є негативний результат тесту на коронавірус, але він має бути отриманий не пізніше 48 годин після здачі тесту. Зокрема, багато туристичних компаній допомагають з пошуком лабораторій і надають ваучери на знижки на тести на коронавірус.

1.5 Аналіз недоліків систем забезпечення карантину

Але, на жаль, як і будь-яка технологія, ця система не позбавлена багів і недоліків. Деякі з них не впливають на роботу програми; інші можуть ускладнити життя людини на карантині.

Іноземці та українці перед перетином українського кордону мають повідомити прикордонній службі код із SMS-повідомлення, необхідний для авторизації. Це незручно, оскільки Ви повинні чекати на кордоні, поки не буде отримано SMS, оскільки повідомлення не приходить вчасно.

Крім того, ІС могла помилково визначити місцезнаходження абонента, а потім вказати межу як точку самоізоляції. Необхідно буде зареєструватися за новим номером телефону, тому що неможливо самостійно оновити адресу. Інше питання, що коли український громадянин почав самоізоляцію, то самостійно його зупинити важко. Іншими словами, неможливо розпочати відстеження знову після іншої подорожі. У результаті прикордонники не мають іншого вибору, окрім як пропустити особу або запропонувати звернутися до технічної допомоги. Її робота також є предметом нарікань.

Іншою проблемою була вимога для селфі для підтвердження геолокації. Знімок, який буде перевірено додатком, практично неможливо зробити, якщо камера телефону поганої якості та недостатньо світла. Програма також пропонує користувачеві робити фото багато разів щодня; однак, якщо цього не зробити негайно, поліція може прибути, щоб перевірити, чи користувач все ще вдома.

Крім того, потрібно було зв’язатися зі службою технічної підтримки, якщо ви хочете достроково припинити самоізоляцію та провести тест на коронавірус у лабораторії, яка не входить до списку затверджених лабораторій Міністерства охорони здоров’я. Дані цих лабораторій не включаються автоматично в систему. Крім того, клієнти повідомляли про випадки, коли передчасне завершення самоізоляції було неможливим, незважаючи на виклик техпідтримки.

1.6 Аналіз закордонних аналогів

Крім вітчизняних систем, у світі також розробляються й інші варіанти контролю самоізоляції, наприклад, розглянути такі іноземні системи, як HQS або PAIMCOS.

Система домашнього карантину (HQS – Home Quarantine System) – це мобільний додаток для Android та iOS, розроблений ексклюзивно компанією Asura Technologies [8], який дає змогу органам влади автоматично перевіря-

ти, чи дотримуються люди, які перебувають на домашньому карантині, карантинних правил, і постійно контролювати стан їхнього здоров'я. Метою проекту було звільнити місцевий відділ поліції від трудомістких карантинних перевірок (рис. 1.7).

Рисунок 1.7 – Інтерфейс IC HQS, розроблений Asura Technologies

Система складається з адміністративного інтерфейсу та програми для смартфонів для кінцевих користувачів. Щоб продемонструвати, що вони не залишають адресу карантину, особам, які перебувають на карантині, потрібно завантажити додаток і віддалено зареєструватися. Для того, щоб медичний персонал міг стежити за станом здоров'я пацієнта, користувачі також можуть заповнювати щоденні анкети медичного огляду.

Органи влади відстежують користувачів за допомогою інтерфейсу адміністратора, який містить списки осіб, які перебувають на карантині, автоматично сповіщає оператора про пропущені реєстрації та допомагає прогнозувати навантаження, яке пацієнти покладуть на місцеві ресурси охорони здоров'я. Під час епідемії COVID-19 система функціонує як офіційна карантинна програма Угорщини.

Вчені з Австралії створили систему контролю впливу пандемії під назвою PAIMCOS. Це легкодоступна програма для моніторингу карантину, яка потребує лише смартфона та підключення до Інтернету [9]. Система не вимагає від користувачів завантажувати програму. Вона була розроблена для збору мінімальної кількості конфіденційних даних, які будуть знищені після закінчення періоду карантину. Користувачі повинні надіслати текстове повідомлення на безпечний номер, який потім попросить їх увійти на веб-сайт. Тоді веб-сайт отримає доступ до місцезнаходження користувача, але не зберігатиме цю інформацію після періоду моніторингу.

На додаток до моніторингу домашнього карантину, PAIMCOS оснащений опціями управління кордонами для ефективного управління гарячими точками (територіями з високим рівнем зараження та обмеженнями на пересування) і приманками (зонами, куди користувачі можуть захотіти потрапити нелегально, наприклад, спортивні заходи) [10]. Система використовує геозонування та голос користувача для перевірки особи (рис. 1.8).

Щоб забезпечити систему моніторингу карантину, яка підтримує мету штату повернутися до нормального життя та може використовуватися новоприбулими після відновлення державних і національних кордонів, PAIMCOS співпрацює з NSW Smart Sensing Network (NSSN) [11], Сіднейським технологічним університетом (UTS) та Сіднейським університетом.

Завдяки спілкуванню з користувачами через їхні смартфони та без збереження даних, які можуть бути використані для масового стеження, ця технологія усуває занепокоєння щодо конфіденційності, які виникають у відповідь на інші програми моніторингу карантину, які зараз використовуються в

Австралії. За словами Адріана Іордакеску, генерального директора PAIMCOS, ця ініціатива має на меті зміцнити ІС шляхом поєднання ML та штучного інтелекту для посилення запобігання шахрайству, високої масштабованості та захисту від кібератак.

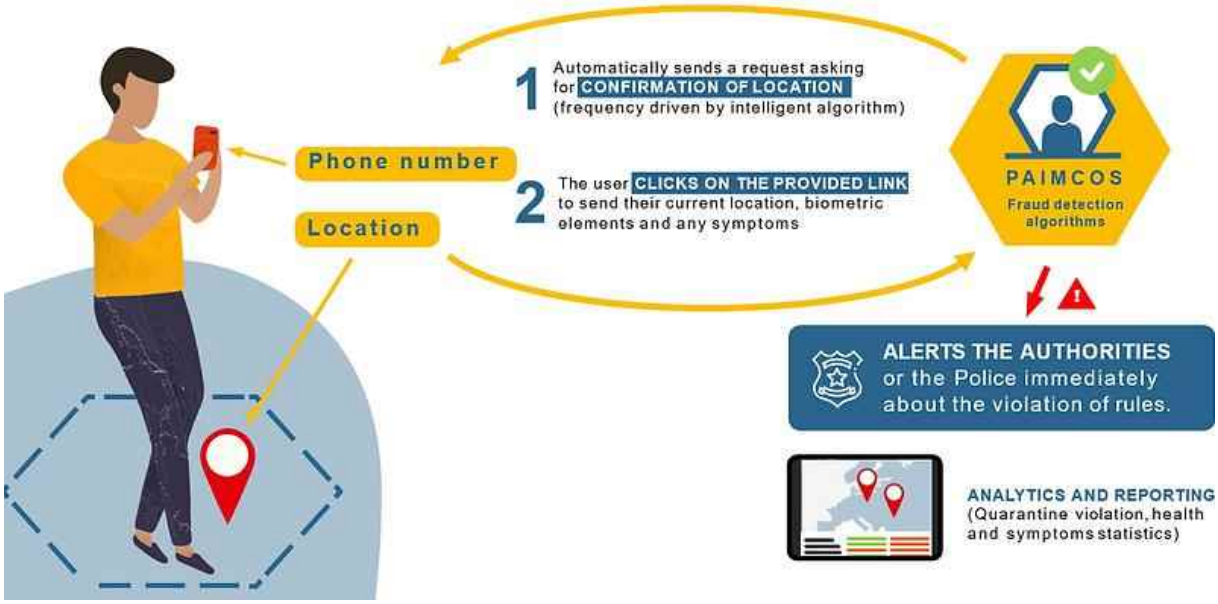

Рисунок 1.8 – Алгоритм роботи системи моніторингу карантину PAIMCOS

1.7 Висновки за розділом

Розглянуті загрози, а також запобіжні заходи, які використовуються для їх усунення, показали, що найсильнішою стороною є соціалізація та соціальна відповідальність людини. Застосування профілактичних заходів у вигляді маскового режиму, дистанціювання та карантину хоч і дає хороший результат, але не може повністю виключити людський фактор із факторів впливу. Причиною цього може бути відсутність повного контролю за виконанням таких заходів.

Багато хто, на жаль, не усвідомлюють ступеня важливості та відповідальності, яку вони можуть нести, порушуючи встановлені запобіжні заходи та наражаючи на небезпеку інших. Запровадження автоматичних систем конт-

ролю, сертифікації тощо хоч і дають бажаний ефект і допомагають досягти бажаного результату, але не змінюють ставлення людей до таких заходів. Багато хто шукає способи обійти такі обмеження та обдурити систему. Крім того, такі системи не позбавлені багів і недоліків, які можна помітити при уважному розгляді.

Наприклад, практика роботи застосунку «Дій вдома» показала, що, в деяких випадках, він може працювати некоректно, унеможливлюючи відправку звіту або взагалі, створює труднощі при зміні режиму ізоляції або зміні місця під час реєстрації, через складність і ступінь навантаження на систему. Основною проблемою таких систем може бути неоднорідність методів аналізу анамнезу, а також аспектів, які можуть бути ключовими при роботі з пацієнтом. Крім того, важливу роль відіграє адаптація до місцевих стандартів.

Виходячи з вищесказаного, слід зробити висновки, що інформаційна система (ІС) має працювати, в першу чергу, з людиною, даючи розуміння реальної загрози та її близькості. Крім того, система не повинна бути прив’язана до якихось стандартів окремої країни чи регіону, а має базуватися лише на загальних рекомендаціях ВООЗ. Простота, гнучкість і швидкість обробки також є ключовими факторами для такої системи, що робить її універсальним інструментом для роботи не тільки під час пандемії, але й в інших ситуаціях, тому розробка такої ІС, яка задовольняла би висунутим практичним вимогам та вирішувала би окреслені проблеми – є метою нашої КРМ.

2 МЕТОДИ ПРОГНОЗУВАННЯ ТА АНАЛІЗУ ДАНИХ

Сучасні математичні моделі дозволяють дуже добре врахувати найважливіші параметри, які впливають на поширення та інтенсивність епідемій – щільність населення, наявність інкубаційного періоду захворювання, частоту контактів, карантин, вакцинацію та ін. Результати моделювання добре узгоджуються з експериментальними даними. Усе разом це дає надію, що пандемії такого ж масштабу нам більше не загрожуватимуть.

Завдяки впровадженню математичного прогнозування в аналітичні системи ми зможемо не тільки прогнозувати розвиток інфекцій і пандемій у майбутньому, а й коригувати заходи відповідно до очікуваних результатів. Розглянута методика прогнозування, хоч і досить проста і не враховує велику кількість факторів, які можуть вплинути на розвиток того чи іншого захворювання, все ж може дати детальну картину і розуміння того, як діяти, щоб запобігти серйозним наслідки.

Обсяг даних, необхідних для аналізу, залишається під великим питанням, оскільки симптоми та умови для спалаху в окремому регіоні можуть сильно відрізнятися, для цієї моделі краще використовувати усереднені дані, щоб отримати правильний прогноз. У випадку, коли нам потрібно використовувати більше параметрів, а усереднювати дані досить складно або недоцільно, слід використовувати інші моделі, які будуть враховувати необхідні параметри та коригувати свої результати на основі цих даних.

2.1 Метод прогнозування MSEIR

Одним з найновіших методів аналізу розповсюдження вірусної інфекції став метод MSEIR (Maternally derived immunity – Susceptible – contact – Infected – Recovered), який працює за принципом: імунітет матері, яка враховує імунітет дітей, отриманий внутрішньоутробно [12].

Модель MSEIR, побудована для захворювання з інкубаційним періодом і враховує внутрішньоутробний імунітет дітей, є однією з найскладніших для аналізу через наявність великої кількості незалежних параметрів. Система рівнянь для цієї моделі має такий вигляд:

$$\frac{dM}{dT} = \beta - \delta M - \mu M,$$
$$\frac{dS}{dT} = \delta M - \beta SI - \mu S,$$
$$\frac{dE}{dT} = \beta SI - (\varepsilon + \mu)E,$$
$$\frac{dI}{dT} = \varepsilon E - (\gamma + \mu)I,$$
$$\frac{dR}{dT} = \gamma I - \mu R.$$

Ця система рівнянь відрізняється від розглянутих раніше моделей тим, що вона враховує народження дітей, ймовірність зараження яких зростає з часом, оскільки вони втрачають внутрішньоутробно набутий імунітет. Ці залежності описані в перших двох рівняннях системи.

Внутрішньоутробний імунітет може бути не у всіх народжених дітей, але сто відсотків немовлят може бути охоплено щепленням. Введення цього параметра в математичну модель призводить до якісної зміни картини розвитку епідемій.

Система рівнянь для цієї моделі буде наступною:

$$\frac{dS}{dt} = \mu N(1 - P) - \mu S - \beta \frac{I}{N} S,$$
$$\frac{dI}{dt} = \beta \frac{I}{N} S - (\mu + \gamma)I,$$
$$\frac{dV}{dt} = \mu NP - \mu V$$

де P – частка вакцинованих немовлят, $0 < P < 1$.

Перші два рівняння повторюють модель SIR, враховуючи, що ймовірність інфікування вакцинованих дітей дорівнює нулю, що означає, що ймовірність інфікування дорівнює ймовірності того, що дитина не вакцинована, і, у свою чергу, дорівнює $(1 – P)$. Останнє рівняння враховує смертність від інших причин і дозволяє розрахувати загальну чисельність популяції.

2.2 Метод ковзаючої середньої ARIMA

ARIMA (Autoregressive Integrated Moving Average) є акронімом від «авторегресійної інтегрованої ковзаючої середньої». Модель ARIMA є узагальненням у статистиці та економетриці, зокрема в аналізі часових рядів [13]. Науковці з аналізу даних дедалі частіше використовують модель ARIMA для прогнозування майбутнього попиту, наприклад прогнозування продажів, виробничих планів або цін на акції. Наприклад, при прогнозуванні цін на акції модель не вимірює фактичні значення, а враховує відхилення між значеннями в ряді.

Ця модель застосовуються до даних часових рядів, щоб покращити їх розуміння та / або прогнозувати наступні моменти в ряді (прогнозування). У деяких ситуаціях, коли дані вказують на нестаціонарність середнього значення (але не дисперсію чи автоковаріацію), використовуються моделі ARIMA, і початковий крок розрізнення (еквівалентний «інтегрованому» елементу моделі) може бути застосований один або більше разів, щоб усунути нестаціонарність середньої функції тобто тренду (рис. 2.1).

Коли часовий ряд демонструє сезонність, можна використати сезонні відмінності, щоб перемістити сезонний компонент. В цьому випадку виникає вимога зробити стаціонарним нестаціонарний часовий ряд, наприклад, застосовуючи розрізнення, перш ніж ми зможемо використовувати модель ARIMA, тому що, згідно з теоремою декомпозиції Уолда, модель ARIMA є

теоретично адекватною для представлення регулярного (він же повністю недетермінований) стаціонарний часовий ряд широкого сенсу [14].

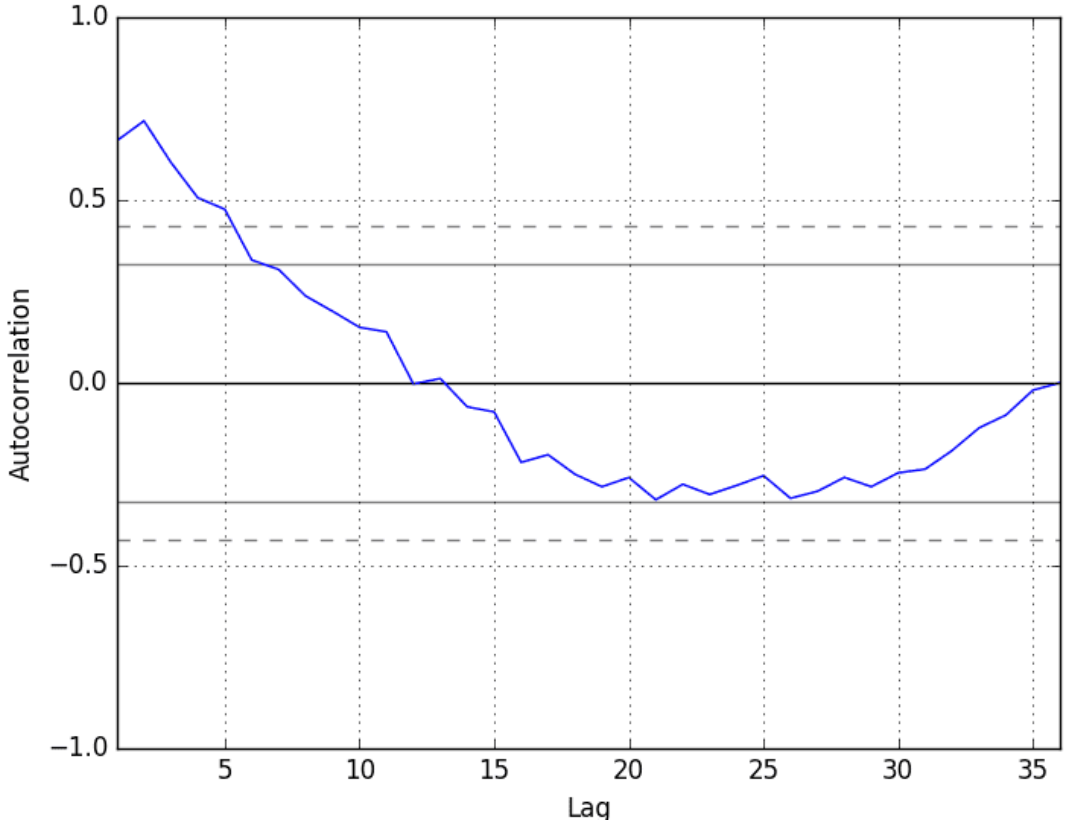

Рисунок 2.1 – Приклад побудови автокореляційної функції в рамках методу ARIMA

Зауважимо, що за структурою ARIMA передбачуваний компонент розглядається як ненульове середнє, але періодичний компонент, який видаляється сезонною різницею, якщо часовий ряд містить передбачуваний підпроцес (також відомий як чиста синусоїда або комплекснозначний експоненціальний процес).

Розглянемо структуру ARIMA:

– «AR» розшифровується як авторегресія, яка описує модель, у якій змінна змінюється та регресує на основі своїх власних сталих або минулих значень тобто підкреслює залежний зв’язок між спостереженням і його попе-

редніми спостереженнями. Іншими словами: виконуються прогнози щодо майбутнього на основі минулого;

– «I» означає інтегрований, що означає, що ця складова шукає відмінності між поточними та історичними значеннями в статичних даних тобто демонструє тенденції чи сезонності, а також застосовується розрізнення зазвичай це включає віднімання спостереження від його попереднього спостереження. Мета полягає в тому, щоб отримати стабільні дані. Це вказує на те, що статистичні характеристики ряду даних, такі як середнє значення, дисперсія та автокореляція, залишаються незмінними протягом часу. Щоб переконатися, що дані стабільні, дослідники використовують розширений тест Дікі-Фуллера (ADF – Augmented Dickey-Fuller test) [15];

– «MA» означає ковзаюче середнє, яке є співвідношенням між спостережуваним значенням і залишковою помилкою, що є результатом моделі ковзаючого середнього, застосованої до попередніх даних на основі спостережень із запізненням.

Три змінні складають модель ARIMA:

– AR (p), яка представляє кількість спостережень із запізненням або компонентів авторегресії в моделі;

– I (d), яка представляє різницю між несезонними спостереженнями;

– MA (q), що представляє розмір рухомого середнього.

Під час виконання моделі ARIMA порядок відображається як (p, d, q) зі значеннями для порядку або частоти кожної функції. Нульові значення допустимі. Кожен із цих компонентів явно вказується в моделі як параметр. Для ARIMA (p, d, q) використовується стандартна нотація, де параметри замінюються цілими значеннями, щоб швидко вказати конкретну модель ARIMA, що використовується.

Параметри моделі ARIMA визначаються наступним чином:

– p: порядок відставання, що представляє кількість спостережень відставання, включених у модель;

– d: ступінь розходження, що позначає кількість разів, коли необроблені спостереження піддаються розрізненню;

– q: порядок ковзного середнього, що вказує на розмір вікна ковзного середнього.

Оскільки модель ARIMA використовує відмінні дані, щоб зберегти дані стаціонарними, дані є послідовними протягом усього часу. За допомогою цієї функції тренди або сезонність більше не можуть впливати на статистику.

Лінійна регресійна модель будується, включаючи вказану кількість і тип термінів, а дані готуються за ступенем різниці, щоб зробити їх стаціонарними, тобто видалити тенденцію та сезонні структури, які негативно впливають на регресійну модель.

Прийняття моделі ARIMA для часового ряду передбачає, що основний процес, який генерує спостереження, є процесом ARIMA. Це може здатися очевидним, але допомагає вмотивувати необхідність підтвердити припущення моделі в необроблених спостереженнях і залишкові помилки прогнозів моделі.

Далі розглянемо, як ми можемо використовувати модель ARIMA в Python.

Для створення моделей ARIMA можна використовувати різноманітні програмні засоби, включаючи Python. Перш ніж вибрати модель ARIMA, фахівець із обробки даних повинен переконатися, що розглядуваний процес відповідає моделі.

Фахівець із обробки даних створює модель і навчає її на наборі даних перед тим, як вводити дані в реальному часі для створення та побудови прогнозу, якщо дані відповідним чином відповідають моделі ARIMA.

Цікаво, що для будь-якого з цих параметрів можна встановити значення 0. Якщо дві з трьох змінних дорівнюють нулю, абревіатуру моделі можна скоротити шляхом вилучення літер «AR», «I» або «MA» на користь ненульового параметра. Такі конфігурації дозволяють моделі ARIMA імітувати функції простіших моделей, таких як ARMA, AR, I або MA.

Для ілюстрації:

– ARIMA (1, 0, 0) дорівнює AR (1);

– ARIMA (0, 1, 0) дорівнює I (1);

– ARIMA (0, 0, 1) дорівнює MA (1).

Двома найпопулярнішими методами прогнозування часових рядів є одновимірний і багатовимірний. Щоб спрогнозувати майбутні значення, однофакторний враховує лише минулі значення часового ряду. До послідовності чисел, багатоваріантність додатково використовує зовнішні фактори для створення прогнозу.

Виконаємо конфігурацію моделі ARIMA (5, 1, 0) (рис. 2.2):

– 5 проходів для авторегресії (AR);

– різниця 1-го порядку (I);

– без ковзаючого середнього (MA).

```
# fit an ARIMA model and plot residual errors
from pandas import datetime
from pandas import read_csv
from pandas import DataFrame
from statsmodels.tsa.arima.model import ARIMA
from matplotlib import pyplot
# load dataset
def parser(x):
 return datetime.strptime('190'+x, '%Y-%m')
series = read_csv('shampoo-sales.csv', header=0, index_col=0, parse_dates=True, squeeze=True, date_parser=p
arser)
series.index = series.index.to_period('M')
# fit model
model = ARIMA(series, order=(5,1,0))
model_fit = model.fit()
# summary of fit model
print(model_fit.summary())
# line plot of residuals
residuals = DataFrame(model_fit.resid)
residuals.plot()
pyplot.show()
# density plot of residuals
residuals.plot(kind='kde')
pyplot.show()
# summary stats of residuals
print(residuals.describe())
```

Рисунок 2.2 – Приклад реалізації визначеної конфігурації моделі ARIMA за допомогою мови Python

Базуючись на власних попередніх значеннях, модель ARIMA робить прогнози для даного часового ряду. Будь-яка несезонна послідовність чисел,

яка відображає шаблони і не є набором випадкових випадків, може використовувати її. Наприклад, оскільки інформація була зібрана протягом певного часу, вона буде часовим рядом. Той факт, що дані збираються протягом послідовних регулярних інтервалів, є однією з його основних особливостей. Для моделювання прогнозів, що охоплюють кілька сезонів, можна зробити модифіковану версію.

Дані повинні бути скориговані на сезонні зміни протягом періоду з великою кількістю сезонів (рис. 2.3). Свята, наприклад, припадають на різні дні протягом року, що надає даним сезонний відтінок. Залежно від того, де в календарі випадає свято, показники реєстрації захворювань можуть бути штучно збільшені або зменшені. Щоб забезпечити точний прогноз слід мати можливість сезонно змінювати дані, але на графіку можна побачити, що значення демонструють певну тенденцію та знаходяться в правильному масштабі.

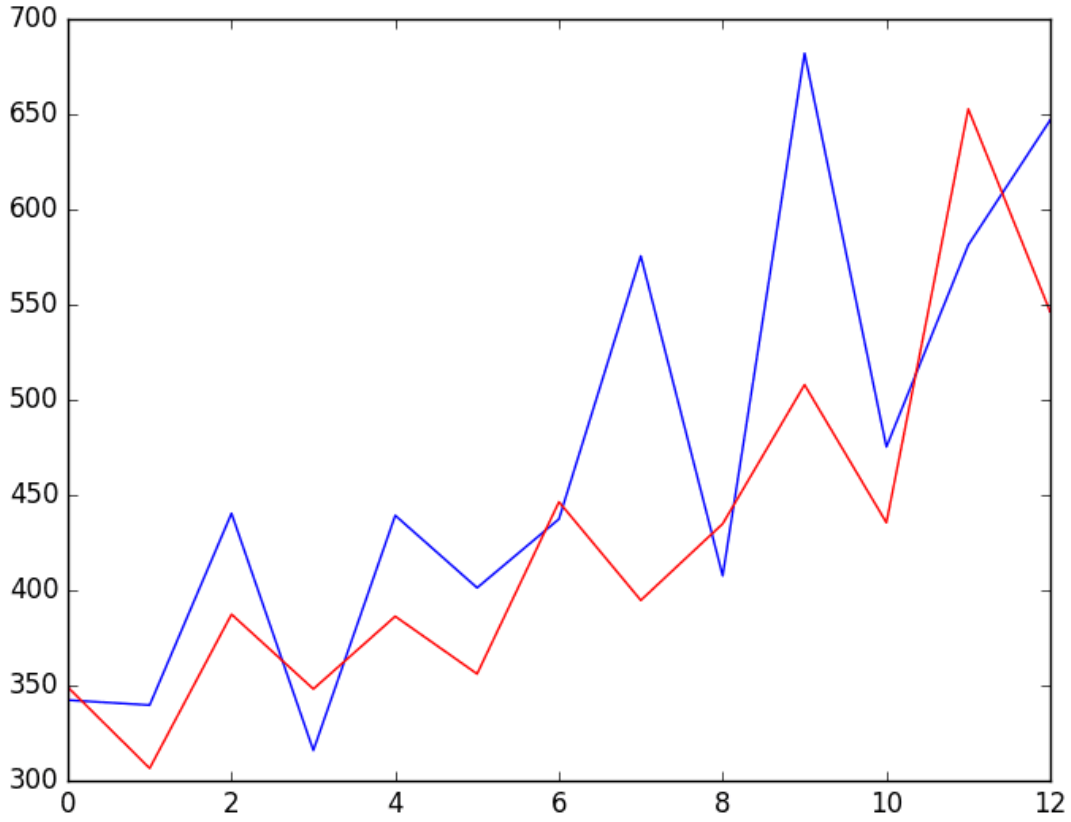

Рисунок 2.3 – Графік з відображеннями очікуваних значень (синій колір) у порівнянні із поточними прогнозами (червоний колір)

2.3 Висновки за розділом

У випадку, коли дані відображають впізнавані, повторювані моделі, можна робити висновок, що присутня сезонність. Необхідно враховувати сезонність, оскільки вона може вплинути на точність результатів. Для створення моделей ARIMA можна використовувати як сезонні, так і несезонні формати. Необхідно враховувати кількість випадків у кожному сезоні на додаток до авторегресійних, диференційних і середніх термінів для кожного сезону в сезонній моделі.

Для порівняння моделей часових рядів використовується ковзний прогноз. Щоразу поточний прогноз оновлюється. Наприклад, поточна оцінка інфікування за 10 днів оновлюється на основі інфікування за попередній день. Для обробки нестаціонарних даних слід використовувати модель ARIMA з ковзним прогнозом. Спеціаліст з аналізу даних може побудувати графік ковзного середнього значення та ковзного стандартного відхилення, щоб оцінити, чи є ряд даних стаціонарним, а тест ADF можна використати, щоб виявити, чи дані не є стаціонарними. Довірчий інтервал до даних з більш ранніх періодів часу для виконання різницевих обчислень враховується вже у ковзному прогнозі.

Таким чином, модель ARIMA – це важливий інструмент для спеціалістів із обробки даних для надання точних прогнозів у різноманітних сферах. Виробничі корпорації використовують моделі ARIMA, щоб керувати бізнес-плануванням, закупівлями та виробничими цілями. У нашому ж випадку, модель ARIMA буде застосовуватись для обробки та накопичення даних щодо інфекційного захворювання та прогнозування розповсюдження вірусної інфекції.

3 РЕАЛІЗАЦІЯ ПРОТОТИПУ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1 Принцип роботи інформаційної системи

Формальна соціотехнічна організаційна структура, яка називається інформаційною системою (ІС) і створена для збору, обробки, зберігання та розповсюдження інформації. З соціотехнічної точки зору інформаційні системи складаються з чотирьох елементів: роботи, людей, структури (або ролей) і технології. Інформаційні системи складаються з компонентів, які працюють разом для збору, зберігання та аналізу даних. Потім ці дані використовуються для створення цифрових продуктів, які допомагають у прийнятті рішень і пропонують інформацію.

Система, яка складається з людей і комп'ютерів і обробляє або інтерпретує інформацію, відома як комп'ютерна інформаційна система. Ця фраза також іноді використовується для опису комп'ютерної системи, у якій є програмне забезпечення. Друга академічна область, заснована на системах, називається «інформаційні системи», яка конкретно відноситься до інформаційних систем і комплементарних мереж комп'ютерного обладнання та програмного забезпечення, які окремі особи та організації використовують для збору, фільтрації, аналізу, створення та розповсюдження даних. Важливість інформаційної системи з чіткими кордонами, користувачами, процесорами, сховищем, входами, виходами та комунікаційними мережами, зазначеними вище.

Підрозділ або підрозділ, відповідальний за обробку даних та інформаційні системи, іноді називають «інформаційними службами» в корпораціях.

Будь-яка інформаційна система прагне допомогти в управлінні, операціях і прийнятті рішень. Інформаційно-комунікаційні технології (ІКТ), які використовує компанія, а також те, як працівники використовують цю технологію для підтримки ділової діяльності, є компонентами інформаційної системи.

Деякі автори чітко розрізняють інформаційні системи, комп'ютерні системи та бізнес-процеси. ІКТ часто є частиною інформаційних систем, але вони не зосереджені лише на ІКТ; скоріше вони зосереджені на тому, як інформаційні технології використовуються на практиці. Бізнес-процеси та інформаційні системи відрізняються один від одного. Ефективністю бізнес-процесів можна керувати за допомогою інформаційних систем.

Згідно з Альтером, є переваги розгляду інформаційної системи як окремого виду робочої системи. Робоча система – це система, в якій люди або машини виконують процедури та завдання, використовуючи ресурси для надання певних товарів або послуг споживачам. Дії інформаційної системи зосереджені на зборі, надсиланні, зберіганні, пошуку, зміні та представленні інформації.

У результаті інформаційні системи мають стосунки як з системами даних, так і з системами діяльності. Комунікаційна система, відома як інформаційна система, яка використовує дані для представлення та аналізу соціальної пам'яті. Інформаційну систему також можна вважати напівофіційною мовою, яка полегшує людські дії та прийняття рішень. Організаційна інформатика Основним напрямком досліджень є інформаційні системи.

Інформаційні системи мають багато різних форм, включаючи системи керування БД, системи обробки транзакцій, системи підтримки прийняття рішень, системи управління знаннями, системи управління навчанням та офісні ІС (рис. 3.1). Інформаційні технології, створені для того, щоб дозволити людям виконувати завдання, для яких людський мозок погано пристосований, наприклад, працювати з великими обсягами інформації, виконувати складні обчислення та контролювати багато одночасних процесів, є важливими для більшості ІС.

Усі наступні шість елементів повинні бути присутніми для створення ІС:

– обладнання та машини, що називаються технікою. У сучасній ІС комп'ютер і все його допоміжне обладнання входять до цієї категорії. При-

строї введення / виведення, зберігання та комунікаційні пристрої входять до складу допоміжного обладнання. Головні книги та чорнило були звичайними компонентами докомп'ютерних ІС;

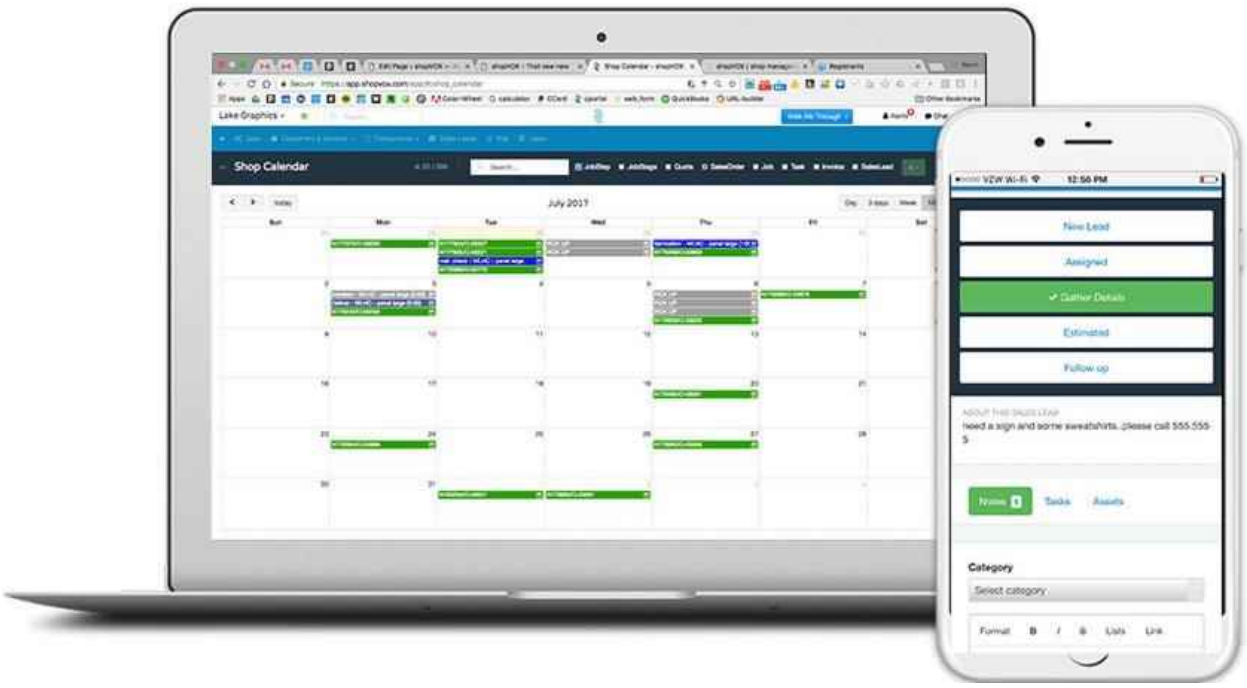

Рисунок 3.1 – Приклади інформаційних систем

– програмне забезпечення або програми. Комп'ютерні програми та будь-які супровідні посібники, які далі називають програмним забезпеченням. Комп'ютерні програми – це машинозчитувані інструкції, які повідомляють апаратним компонентам схем системи, як працювати, щоб вони могли витягувати значущу інформацію з даних. Як правило, програми зберігаються на носії введення / виведення, яким часто є диск або стрічка. Заголовки стовпців у книзі головної книги, наприклад, використовувалися як «програмне забезпечення» для докомп'ютерних ІС для підготовки обладнання до використання (довідник для карткового каталогу);

– самі дані. Це реальні факти, які системи використовують для створення відповідної інформації. Поки вони не знадобляться комп'ютеру, дані в сучасних ІС часто зберігаються в машиночитаній формі на диску чи стрічці;

– процедури. Правила, які визначають, як працює ІС, називаються процедурами. Ілюстрація функції процедур у системі іноді дається так: «Процедури для людей є тим же, що програмне забезпечення для апаратних засобів»;

– люди, тому що кожна система потребує людей для функціонування. Люди – це, мабуть, та частина системи, якою найчастіше нехтують і яка найбільше впливає на успіх чи невдачу ІС. Це стосується «не лише користувачів, а й людей, які керують і обслуговують комп'ютери, тих, хто підтримує дані, і тих, хто підтримує комп'ютерну мережу»;

– Інтернет, поєднує людей і дані, хоча його використання не обов'язкове для роботи компонента.

Таким чином, можна сказати, що основою будь-якої ІС є можливість обробки, зберігання та надання даних кінцевому користувачеві системи.

3.2 Вибір засобів розробки

3.2.1 Загальна характеристика можливостей Python

Для зручності розробки була обрана мова програмування Python, а за основу веб-додатку обрано фреймворк Django. Причиною цього стала простота і гнучкість розробки, а також можливість використовувати модульність, для зручної реалізації потрібних функцій і вирішення завдань за допомогою готових рішень.

Python – популярна мова програмування для створення веб-сайтів і програм, автоматизації процесів і аналізу даних. Оскільки Python є мовою загального призначення, його можна використовувати для розробки широкого спектру програм і не призначено для вирішення будь-яких конкретних проблем. Вона стала однією з найпопулярніших мов програмування, яка використовується сьогодні, завдяки своїй універсальності та зручності для початківців. Це була друга за популярністю мова програмування серед розроб-

ників у 2021 році, згідно з опитуванням дослідницької компанії RedMonk [16].

Python використовується багатьма непрограмістами, зокрема бухгалтерами та науковцями, для ряду рутинних дій, зокрема управління фінансами, оскільки його легко вивчити. Python зарекомендував себе як стандарт у науці про дані, дозволяючи аналітикам даних та іншим експертам використовувати його для виконання складних статистичних обчислень, розробки алгоритмів машинного навчання, обробки та аналізу даних, серед інших видів діяльності [17].

Це дає нам змогу створювати широкий спектр візуалізацій даних, включаючи лінійні та гістограми, секторні діаграми, гістограми та тривимірні графіки. Крім того, Python надає різноманітні бібліотеки, такі як TensorFlow і Keras, які допомагають програмістам швидше й ефективніше створювати програми для аналізу даних і машинного навчання.

Компоненти веб-сайту чи програми, які користувачі не бачать, – серверна частина – часто розробляються за допомогою Python. Надсилання даних на сервери та з серверів, обробка даних і взаємодія з базами даних, маршрутизація URL-адрес і підтримка безпеки – це всі можливі варіанти використання Python у веб-розробці. Python надає ряд фреймворків веб-розробки. Django і Flask є прикладами часто використовуваних. Інженери бек-енду, інженери повного стеку, розробники Python, інженери програмного забезпечення та інженери DevOps – це лише деякі посади веб-розробників, які використовують Python.

Python можна використовувати для автоматизації завдань, які ви часто виконуєте, щоб підвищити продуктивність. Сценарії – це процес створення комп’ютерного коду для цих автоматичних процедур. У сфері кодування автоматизація може бути використана для виконання простих математичних обчислень, перетворення файлів, перевірки наявності проблем у багатьох файлах і усунення дублікатів даних.

Навіть абсолютні початківці можуть використовувати Python для автоматизації простих комп’ютерних завдань, таких як перейменування файлів, пошук і завантаження інформації в Інтернеті або надсилання електронних листів чи SMS за розкладом. Це може допомогти в таких діях, як контроль збірки, відстеження проблем і тестування в розробці програмного забезпечення. Розробники програмного забезпечення можуть автоматизувати тестування нових функцій або продуктів за допомогою Python. Green і Requestium є двома прикладами інструментів тестування Python.

Для людей, які працюють у сферах з меншим обсягом даних, таких як журналістика, власники невеликих компаній або маркетинг у соціальних мережах, вивчення Python може розширити їхні кар’єрні можливості. Python також може допомогти непрограмістам оптимізувати деякі їхні щоденні обов’язки. Використовуючи Python, можна автоматизувати такі завдання, наприклад:

– спостерігати за курсами криптовалюти або фондового ринку;

– надіслати собі текстове повідомлення, щоб нагадати вам завжди брати з собою парасольку під час дощу;

– переглянути список покупок;

– перейменовувати величезну кількість файлів;

– конвертація текстових файлів в електронні таблиці;

– роздавати членам сім'ї обов'язки навмання;

– автоматично заповнювати веб-форми;

Є кілька причин, чому Python так люблять. Ось більш детальний аналіз того, що робить його таким адаптивним і простим для використання програмістами:

– легше читати та розуміти завдяки простій граматиці, яка імітує природну англійську – в результаті прискорюється розробка та вдосконалення проекту;

– адаптований – Python можна використовувати для широкого кола проектів, включаючи машинне навчання та веб-розробку;

– він подобається програмістам-початківцям, оскільки він зручний;

– будучи відкритим вихідним кодом, його можна використовувати та поширювати безкоштовно, навіть з метою отримання прибутку;

– Python має значну бібліотеку та репозиторій модулів, що розширюється, тобто колекції коду, написаного сторонніми розробниками для покращення функціональності Python;

– велика й активна спільнота підтримує Python, додає до своєї колекції модулів і бібліотек та служить цінним ресурсом для інших програмістів, крім того: завдяки великій мережі підтримки знайти рішення легко.

3.2.2 Побудова ІС на веб-фреймворку Django

Dajngo – це високорівневий веб-фреймворк Python, який сприяє швидкому розвитку та спрощеному практичному дизайну. Він був створений досвідченими програмістами та обробляє багато зусиль, пов’язаних із веб-розробкою, звільняючи від можливості зосередитися на створенні свого додатка без необхідності винаходити велосипед. Він відкритий і безкоштовний.

Двом програмістам із Lawrence Journal-World у Канзасі приписують розробку Django. Для публікації новин в Інтернеті, газеті потрібен був веб-додаток. Зрештою розробники Django прийшли до висновку, що їх рішення перетворилося на функціональну структуру та стало доступним для широкої публіки. Як тільки Django створив спільноту, вона швидко зросла, і кількість веб-сайтів, які використовують Django, зросла [18]. The Washington Post, Dropbox, Spotify і Pinterest є одними з найпопулярніших програм Django.

Цей фреймворк економить час, оскільки Django дотримується концепції Don't Repeat Yourself (DRY). Іншими словами, Django дозволяє створити свій веб-сайт як набір Lego, усуваючи необхідність переробляти поточний код. Фреймворк дозволяє економити час розробки за рахунок великої кількості допоміжних об'єктів і підходить для систем з великим навантаженням – це викликано архітектурою, яка описана більш детально нижче.

Django має охайну «пітонічну» структуру та повністю створено на Python. Model-View-Controller (MVC) була оригінальною структурою, і вона все ще існує в останній версії.

Дизайн MVC дозволяє розробникам незалежно змінювати бізнес-логіку та естетичні компоненти програми, не впливаючи на інші. Але насправді розробники зазвичай називають дизайн Django (MVT) [19]. Модель, вигляд і шаблон – це три рівні (рис. 3.2). Кожен шар має окрему функцію і може використовуватися окремо.

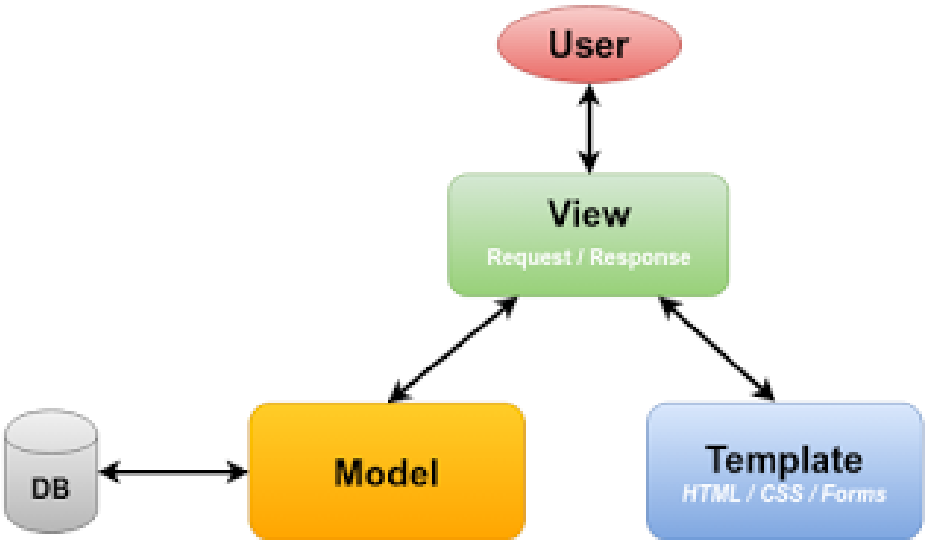

Рисунок 3.2 – Модель MVT

Відповідно до посібника Django, модель є «єдиним авторитетним джерелом інформації про ваші дані». У ньому містяться ключові поля та поведінка даних, які ви зберігаєте. Кожна модель зазвичай відповідає одній таблиці БД. Програми часто використовують чотири БД:

– PostgreSQL;
– MySQL;
– SQLite;
– Oracle, які Django офіційно підтримує.

Атрибути використовуються для представлення моделей, які містять інформацію про дані (поля). Модель не знає інших рівнів Django, оскільки це просто простий клас Python. Лише інтерфейс прикладного програмування забезпечує зв’язок між рівнями (API). Моделі зберігають елементи, пов’язані з маніпулюванням даними, такі як бізнес-логіка, унікальні методи, атрибути тощо. Об'єкти (набори даних) у вихідній базі даних також можна створювати, читати, оновлювати та видаляти за допомогою моделей.

Представлення виконує операції з деревом, зокрема приймає запити HTTP, обробляє їх за допомогою класів і функцій Python і відповідає на ці запити за допомогою HTTP. Іншими словами, представлення отримує дані з моделі та або надає доступ до певних даних для кожного шаблону, який буде представлено, або готує дані для представлення.

Django містить надійну систему шаблонів і багатий набір інструментів для власної мови розмітки. Шаблони – це файли з кодом HTML, які використовуються для відображення даних. Вміст цих файлів може бути статичним або динамічним. Шаблон використовується виключно для передачі даних, оскільки йому бракує бізнес-логіки.

Люди, які не знайомі з Django, помилково вважають, що це лише система керування контентом. Насправді це частина програмного забезпечення, яка використовується для створення онлайн-додатків і керування ними.

Розуміння складності фреймворку вимагає знання того, звідки походить його назва: джазовий музикант Джанго Рейнхардт, який грав мерехтливі рядки на своїй гітарі, незважаючи на паралізовані два пальці після нещасного випадку, надихнув назву фреймворку Django [18]. Фреймворк Django так само здатний обробляти широкий спектр завдань. Django можна використовувати для створення:

– системи управління взаємовідносинами з клієнтами (CRM – Client Relationship Management);

– системи управління контентом (CMS – Content Management Systems) для внутрішнього та комерційного використання;

– комунікаційні платформи;

– системи бронювання;

– платформи адміністрування документів.

Django виділяється, серед іншого, генераторами на основі алгоритмів, рішеннями електронної пошти, системами перевірки та системами фільтрації зі складними параметрами та правилами, які динамічно змінюються. Він може вирішити проблеми аналізу даних і складні обчислення за допомогою методів машинного навчання.

3.2.3 Аналіз можливостей веб-фреймворку Django

Переваги Django:

– екосистема. Розробники радять читати Django як систему. Вони мають на увазі, що Django поставляється з великою кількістю сторонніх програм. Залежно від потреб проекту, ці програми можуть бути включені. Подумайте про Лего, щоб допомогти вам уявити це. Існує кілька варіантів кубиків Лего. «Блок» для надсилання електронних листів або авторизації майже завжди присутній у проектах розробки додатків. У Django є різноманітні програми Plug-And-Play, у тому числі програми для авторизації та надсилання електронної пошти;

– вбудована адмін панель. Ви можете керувати своєю програмою за допомогою панелей адміністратора. Хоча розробка панелі адміністратора вручну займе багато часу та буде абсолютно безглуздим, панель адміністратора Django створюється автоматично з коду Python. Завдяки програмам сторонніх розробників панель адміністратора Django має масу гнучкості для налаштування. Django також дозволяє створювати інформаційні панелі відповідно до ваших вимог і змінювати інтерфейс за допомогою сторонніх обгорток;

– добре для SEO. Якщо ви хочете, щоб ваш веб-сайт мав високі позиції в результатах пошуку, Python відомий своїм кодом, який легко зрозуміти людям. Використовуючи найдоречніші ключові слова та найкращі практики

пошукової оптимізації (SEO – Search Engine Optimization), Django дозволяє створювати читабельні URL-адреси веб-сайтів і посилання. Зрештою, доменне ім'я – це лише «читабельний» текст, який відповідає IP-адресі, набору «зручних для комп'ютера» цифр. Люди, як правило, зосереджуються на виборі ідеального доменного імені, нехтуючи URL-адресою. Django може це змінити;

– підключається (тобто розширюється). Django можна розширити за допомогою плагінів і природно підключати. Модулі – це комп'ютерні програми, які надають програмістам безліч варіантів налаштування, дозволяючи їм додавати певні функції до програми. Ви можете зв'язатися зі Stripe, щоб обробляти платежі, інтегрувати карти Google або створювати складні дозволи за допомогою будь-якого із сотень пакетів. Крім того, ви можете видалити певні компоненти та замінити їх на інші, які краще відповідають вашим потребам зараз, якщо вам колись знадобиться розширити свій проект;

– кількість бібліотек. У кожній мові програмування є бібліотеки для виконання типових завдань. Попередньо написаний код, класи, сценарії, процедури, інформація про конфігурацію тощо можна знайти в бібліотеці програмного забезпечення. Бібліотека часто додається до програмного забезпечення, щоб збільшити функціональність або автоматизувати процес без необхідності вручну писати додатковий код. Час виходу на ринок скорочується. При створенні будь-якого проекту Django дозволяє розробникам використовувати бібліотеки. Фреймворк Django REST, який відповідає за створення інтерфейсів прикладного програмування (API – Application Programming Interfaces), Django CMS, який призначений для керування вмістом веб-сайту, та Django-allauth, інтегрована колекція програм Django для автентифікації, реєстрації, керування обліковими записами і автентифікація сторонніх (соціальних) облікових записів – це деякі приклади добре відомих бібліотек;

– об'єктно-реляційний маппер (ORM – Object-Relational Mapper) у Django, який забезпечує взаємодію з БД для розробників, високо цінується. Бібліотека під назвою ORM перетворює дані з БД, таких як PostgreSQL і

MySQL, в об’єкти, які часто використовуються в коді програми. Здатність Django ORM витягувати інформацію прискорює створення веб-додатків і допомагає програмістам швидко створювати функціональні прототипи. Щоб маніпулювати даними, розробникам не обов’язково бути знайомими з мовою, яка використовується для підключення до БД. ORM Django також дозволяє розробникам переходити між реляційними базами даних з невеликою зміною коду. Це може дозволити, наприклад, працювати локально за допомогою SQLite і конвертувати в MySQL для використання у виробництві. Щоб запобігти будь-яким проблемам під час переходу, часто рекомендується використовувати одну БД;

Недоліки Django:

– не підходить для невеликих завдань. Хоча Django іноді може бути надмірним, Python дозволяє використовувати інші фреймворки для створення простих рішень. Наприклад, Django може бути занадто великим фреймворком, якщо вам потрібно лише розробити невеликий чат. Замість цього використовуйте фреймворк мікросервісу Flask;

– веб-сокети не підтримуються за замовчуванням. Інформація в режимі реального часу або оновлення подій можливе за допомогою WebSockets. Веб-додатки в реальному часі наразі не підтримуються Django. У результаті ви повинні використовувати різні фреймворки, такі як «aiohttp»;

– монолітність. Від ваших інженерів знадобиться багато роботи, щоб змінити структуру, що лежить в основі кількох внутрішніх модулів Django, таких як ORM і форми, які важко замінити;

– може бути важко налаштувати поведінку Django. Через філософію Django кілька внутрішніх модулів, як-от інтерфейс адміністратора, складно налаштувати. Це може зайняти години, якщо ви хочете додати посилання, динамічну статистику чи щось особливе, що не є частиною середовища Django. Тому не впадайте в шок, коли отримаєте рахунок;

Без фреймворків розробка веб-додатків тривала б довше. У середовищі Python є й інші фреймворки, крім Django. Pyramid, Flask і Tornado є іншими

фреймворками Python. Усі вони призначені для використання як у простих програмах, так і в масштабних підприємствах.

3.2.4 Інші технології реалізації інформаційних систем

Мова програмування PHP має власні фреймворки для швидкої розробки програм, наприклад CakePHP, Symphony і Laravel. Laravel, наприклад, можна використовувати як для великих, так і для малих програм. Як і Django, він пропонує потужний механізм створення шаблонів. Розгляньте Laravel для свого проекту, якщо необхідний акцент на безпеку через його потужні пакети шифрування. З Laravel можна використовувати БД MySQL, PostgreSQL, SQL Server і SQLite.

Мова програмування Ruby використовується для створення фреймворку веб-додатків, відомого як Ruby on Rails. Крім того, Ruby on Rails пропонує швидку розробку додатків і велику кількість вбудованих можливостей, які дозволяють програмістам зосередитися на бізнес-логіці, а не на кодуванні. Веб-програми, які використовують БД, ідеально підходять для Ruby on Rails.

Хоча Rails є гнучким, це може бути проблемою. Інженерам потрібно більше часу, щоб зрозуміти роботу коду, тому що одна й та сама функція може бути реалізована різними способами. Тому, якщо ви вирішите передати свій проект іншій команді, навчання може бути важким.

Середовищем розробки було обрано Microsoft Visual Studio Code.

Microsoft створила Visual Studio Code, відомий як VS Code, редактор вихідного коду для Windows, Linux і macOS [20]. Підтримка налагодження, підсвічування синтаксису, інтелектуальне завершення коду, фрагменти, рефакторинг коду та інтегрований Git є серед функцій (рис. 3.3). Тема, комбінації клавіш, параметри та розширення, які пропонують більше функціональних можливостей, можуть бути змінені користувачами.

Серед 82 000 розробників, які взяли участь в дослідженні Stack Overflow 2021 Development Survey, 70 відсотків сказали, що вони використо-

вують Visual Studio Code, що робить його найпоширенішим інструментом середовища розробника.

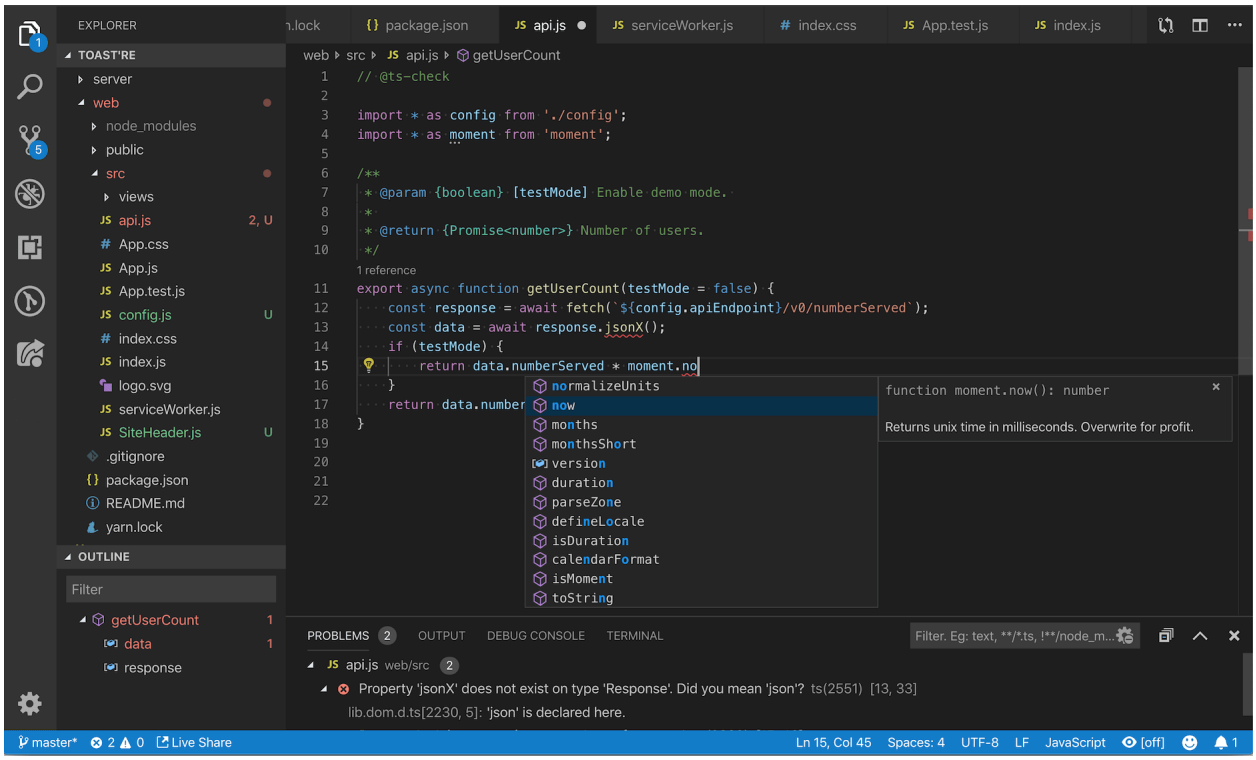

Рисунок 3.3 – Інтерфейс VSCode

На конференції Build 2015 Microsoft спочатку представила Visual Studio Code 29 квітня 2015 р. [21]. Невдовзі з'явилася попередня збірка. Ліцензія MIT була застосована до випуску вихідного коду Visual Studio Code 18 листопада 2015 р., і він був доступний на GitHub. Також була заявлена підтримка розширень. Код Visual Studio був опублікований онлайн 14 квітня 2016 р. після завершення етапу публічного попереднього перегляду. Хоча випуски Microsoft є приватним програмним забезпеченням, більша частина вихідного коду Visual Studio Code була доступна на GitHub під дозвільною ліцензією MIT.

Java, JavaScript, Go, Node.js, Python, C++ і Fortran – це лише деякі з мов програмування, які можна використовувати з Visual Studio Code, редактором вихідного коду. Фреймворк Electron, який використовується для створення

веб-додатків Node.js, які використовують механізм компонування Blink, служить його основою. Компонент редактора (під кодовою назвою «Monaco»), який використовується в Azure DevOps, також використовується в Visual Studio Code (раніше називався Visual Studio Online і Visual Studio Team Services).

Більшість популярних мов програмування мають мінімальну підтримку у Visual Studio Code. Ця фундаментальна підтримка складається з настроюваних фрагментів, згортання коду, зіставлення дужок і підсвічування синтаксису. Разом із підтримкою налагодження для Node.js Visual Studio Code також поставляється з IntelliSense для JavaScript, TypeScript, JSON, CSS і HTML. Безкоштовні розширення на VS Code Marketplace можуть запропонувати підтримку для нових мов.

Це дозволяє користувачам відкривати одну або кілька папок, які згодом можуть бути збережені в робочих областях для подальшого використання замість системи проекту. Як результат, він може функціонувати як мовно-нейтральний редактор коду для будь-якої мови. Він підтримує широкий спектр мов програмування, кожна з яких має власний набір можливостей. За допомогою налаштувань непотрібні файли та папки можна виключити з дерева проекту. Багато аспектів Visual Studio Code можна отримати за допомогою палітри команд, а не через меню чи інтерфейс користувача.

Розширення – це надбудови для Visual Studio Code, доступ до яких здійснюється через загальне сховище. Це покращує редактор і підтримує нові мови. Використовуючи протокол мовного сервера, можна створювати розширення, які додають підтримку нових мов, тем, налагоджувачів, налагоджувачів подорожей у часі, статичного аналізу коду та лінтерів коду.

Visual Studio Code постачається з вбудованим інструментом для керування джерелами. Ви можете отримати доступ до налаштувань контролю версій і переглянути зміни, внесені до поточного проекту, за допомогою спеціальної вкладки на панелі меню. Ви повинні підключити Visual Studio Code до сумісної системи контролю версій, щоб використовувати функції (Git,

Apache Subversion, Perforce тощо). Це дає змогу надсилати запити «push» і «pull» із програми Visual Studio Code, а також створювати репозиторії.

Завдяки численним FTP-розширенням Visual Studio Code можна використовувати як безкоштовну заміну для веб-розробки. Без завантаження додаткового програмного забезпечення код можна синхронізувати між редактором і сервером. Символ нового рядка, мова програмування та кодова сторінка, на якій зберігається активний документ, можуть бути налаштовані користувачами Visual Studio Code. Це дозволяє використовувати його з будь-якою мовою програмування, у будь-якому місці та на будь-якій платформі.

3.3 Реалізація основних функцій системи

Розглянемо основні функції системи, а також елементи інтерфейсу для управління системою.

Після натискання на адресу URL користувач бачить головну сторінку системи з картою та полем введення для пошуку (рис. 3.4). З огляду на те, що користувач не авторизований в системі, він може лише переглянути карту, інформацію про точки та перейти до обраної адреси через поле введення.

При наведенні вказівника на маркер буде відображена інформація про цю точку, а клацання перемістить і збільшить карту та покаже радіус потенційного зараження у вибраній точці (рис. 3.5).

Під час внесення адреси в поле введення користувачеві будуть запропоновані варіанти вибору адреси, яку використовувати (рис. 3.6). Після вибору адреси карта автоматично переміститься до вказаної точки.

Крім того, на карті можна побачити своє місцезнаходження, щоб перевірити, чи є точки поблизу. Натиснувши на значок стрілки у верхньому лівому куті, браузер запитає дозвіл на доступ до геолокації (якщо вона не включена за замовчуванням) і покаже поточне місце розташування на карті (рис. 3.7).

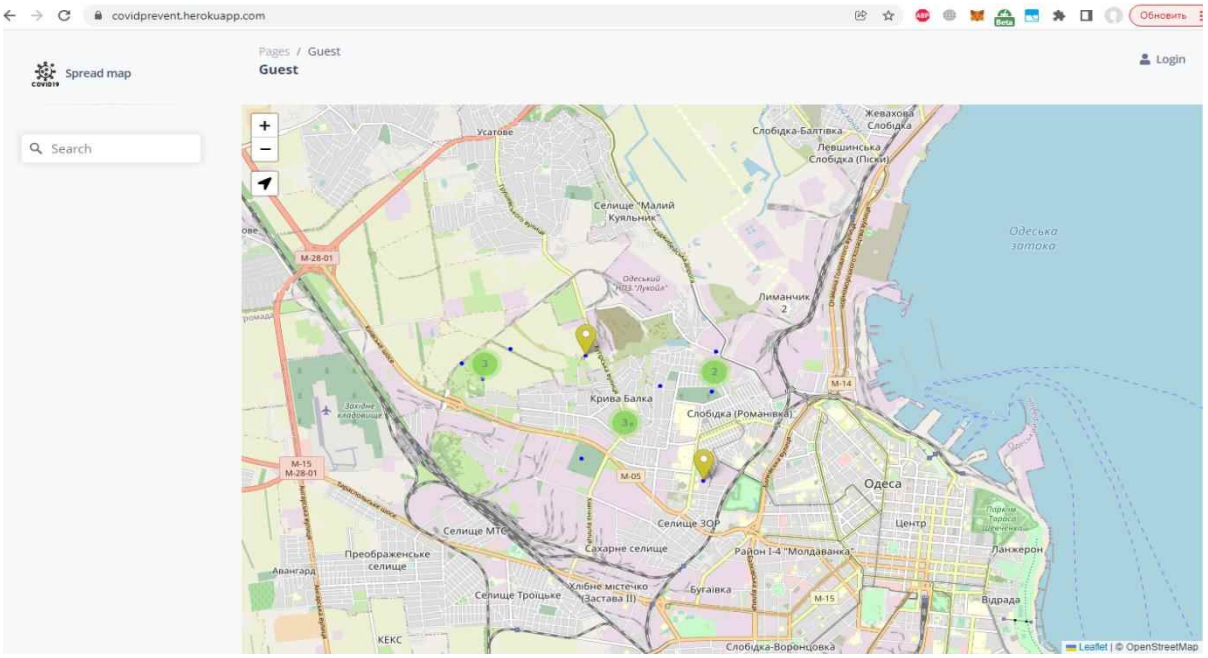

Рисунок 3.4 – Головна сторінка

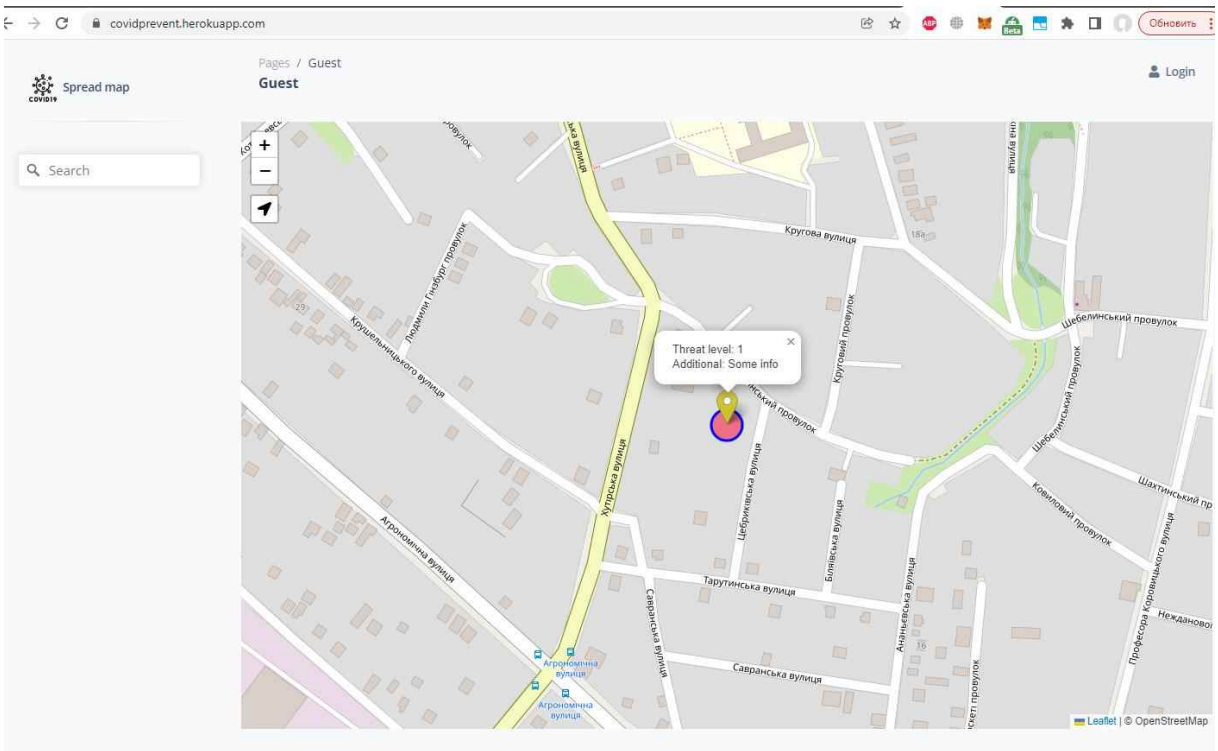

Рисунок 3.5 – Інформація про точку

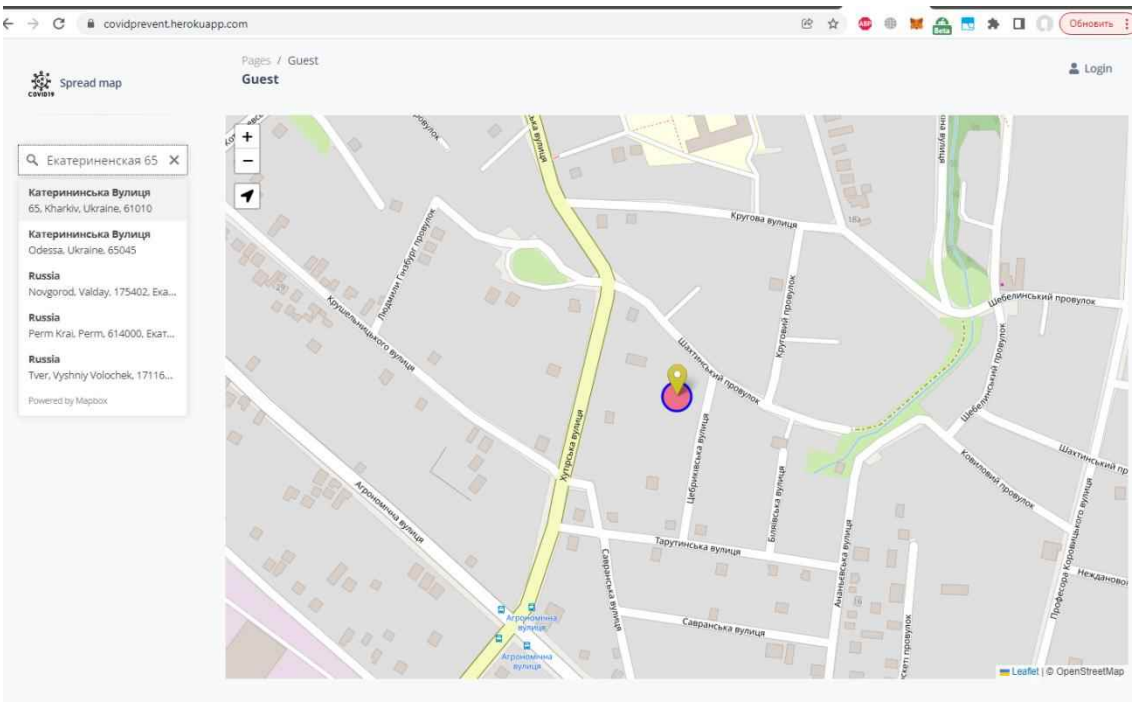

Рисунок 3.6 – Поле введення

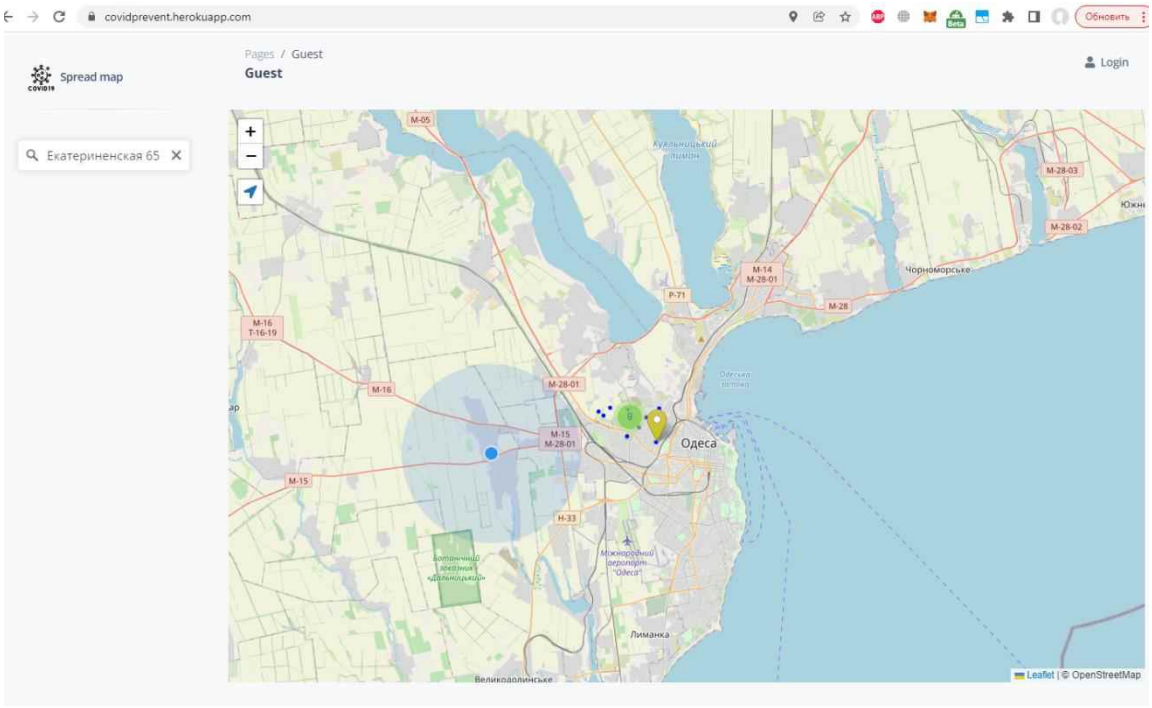

Рисунок 3.7 – Розташування користувача

Також існує кластеризація точок на карті. Це означає, що кілька сусідніх точок можна об’єднати в групи та показати кількість точок в одному міс-

ці за допомогою кольорового кодування. При наведенні курсора на такий кластер ви можете побачити область, охоплену точками, що входять до кластера (рис. 3.8). Коли ви клацнете на ньому, карта збільшиться та покаже точки, які там містяться.

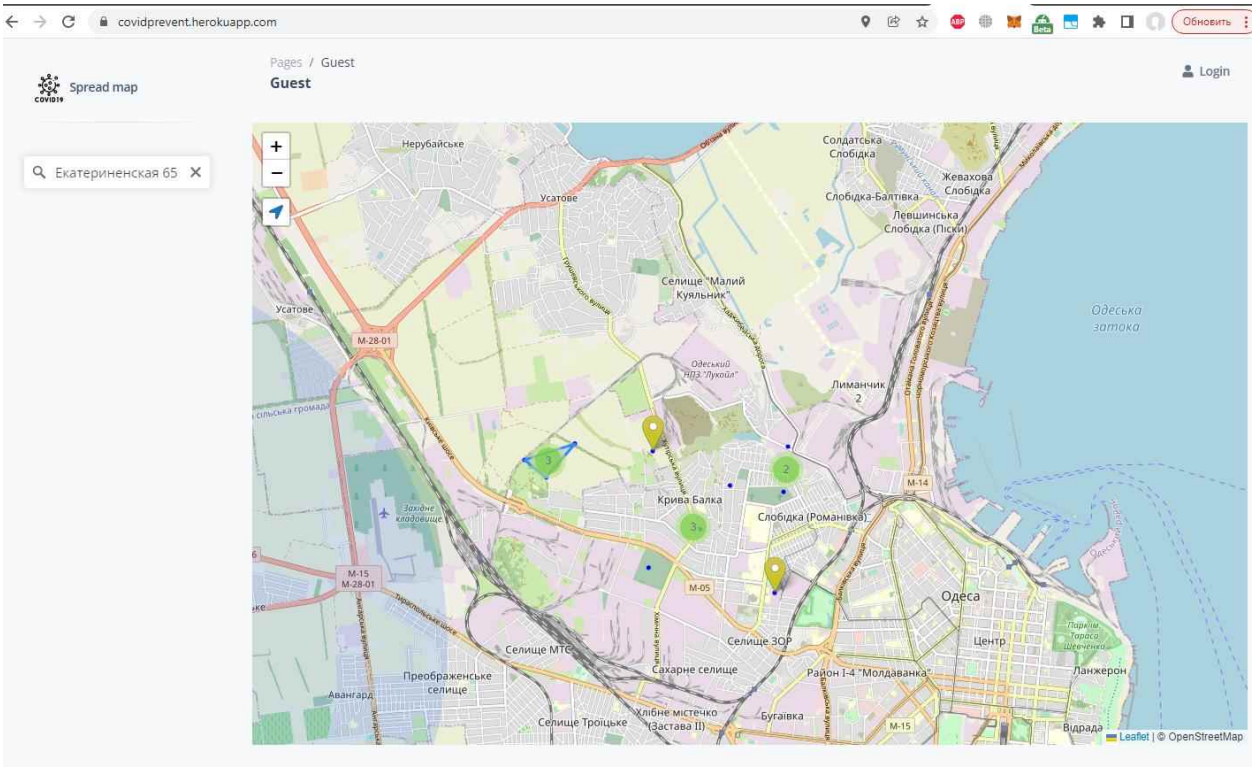

Рисунок 3.8 – Кластеризація

Для повноцінної роботи з системою необхідно скористатися даними авторизації та натиснувши на кнопку «Login» у верхньому правому куті, заповнити дані для входу та авторизуватися (рис. 3.9). Якщо дані невірні, система попередить про це.

На сторінці входу до системи (рис. 3.9) можна побачити кнопку для переходу в гостьовий режим, а також можливість реєстрації у системі. Окрім сторінки входу, на сторінці реєстрації також є соціальні мережі автора, наведені нижче, за допомогою яких можна переглянути його активність (рис. 3.10).

Після входу в систему (як і у разі успішної реєстрації користувача), користувач буде перенаправлений на сторінку карти з усіма необхідними елементами керування. Для кожного типу користувачів у системі елементи будуть різними. Ми вже бачили, які параметри є в профілі гостя, тепер давайте подивимося на профіль зареєстрованого користувача (рис. 3.11).

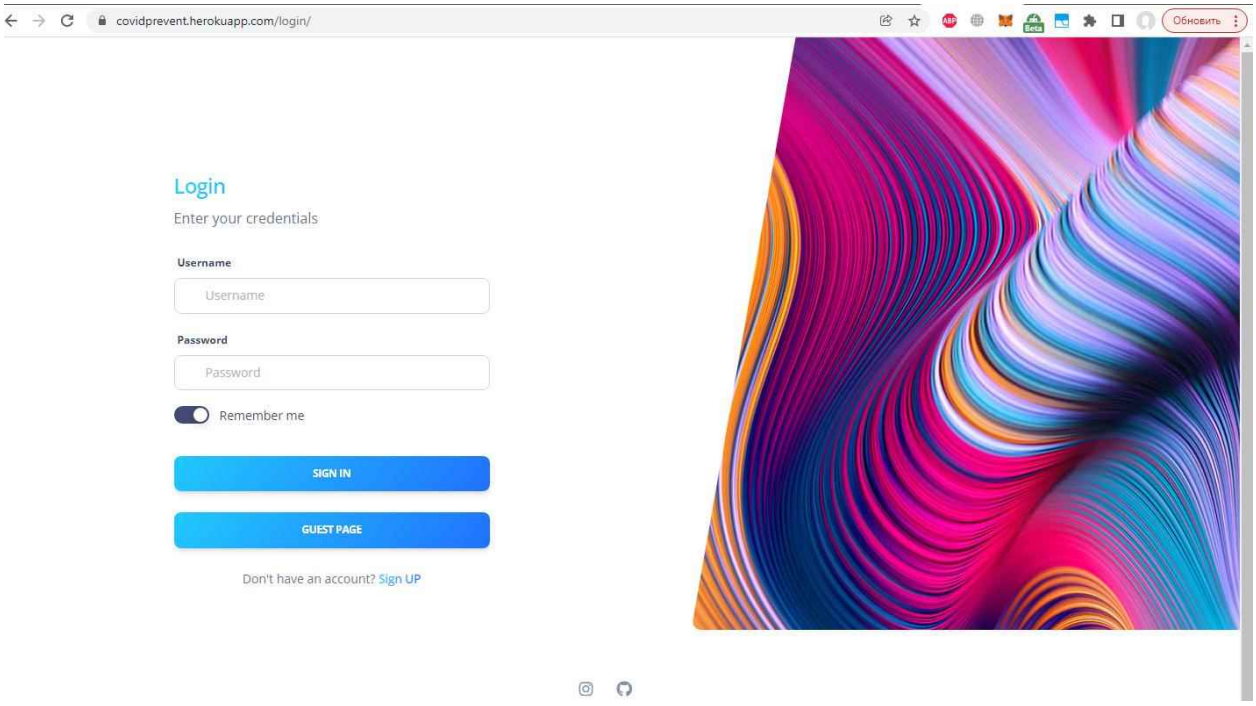

Рисунок 3.9 – Вхід до системи

Зареєстрований користувач може додати будь-яку точку на карті та описати до неї всю необхідну інформацію, таку як:

– ім’я пацієнта;

– контактний телефон;

– основні симптоми;

– додаткова інформація тощо.

Основні симптоми, доступні для вибору, не є остаточними і можуть бути розширені у будь-який час.

Після додавання точки на карту користувач побачить повідомлення і точку буде додано до черги погашення (рис. 3.12).

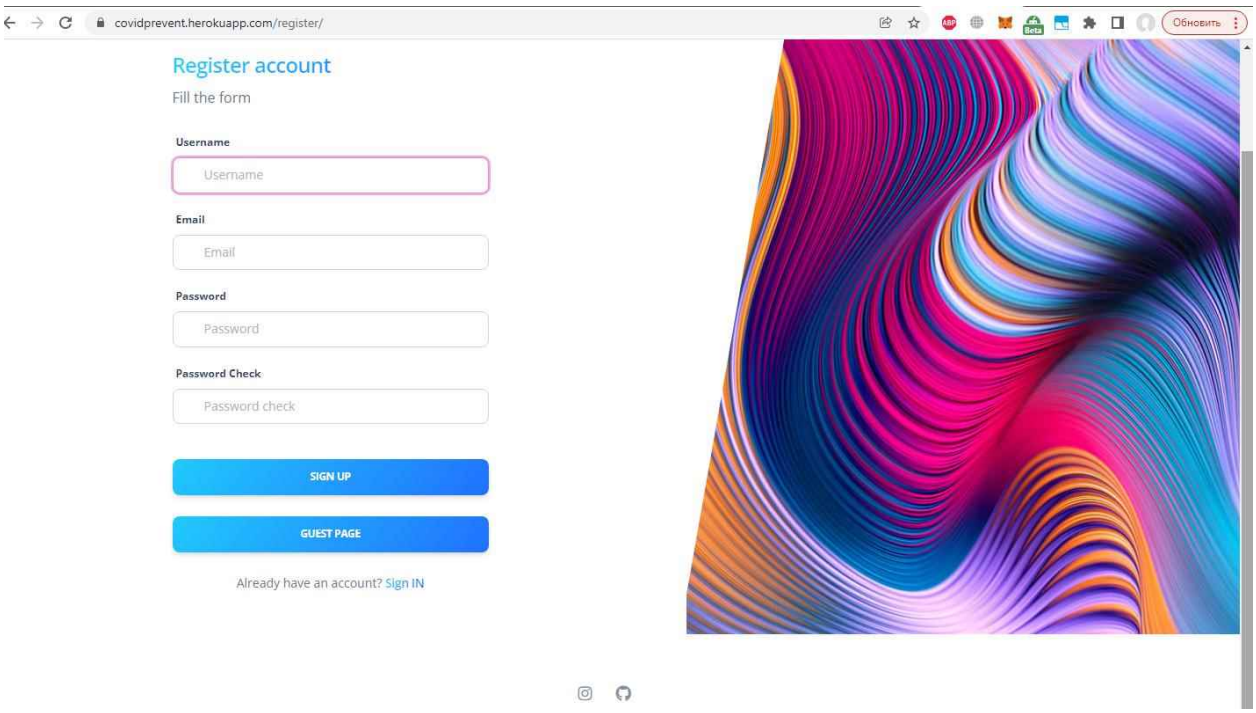

Рисунок 3.10 – Сторінка реєстрації

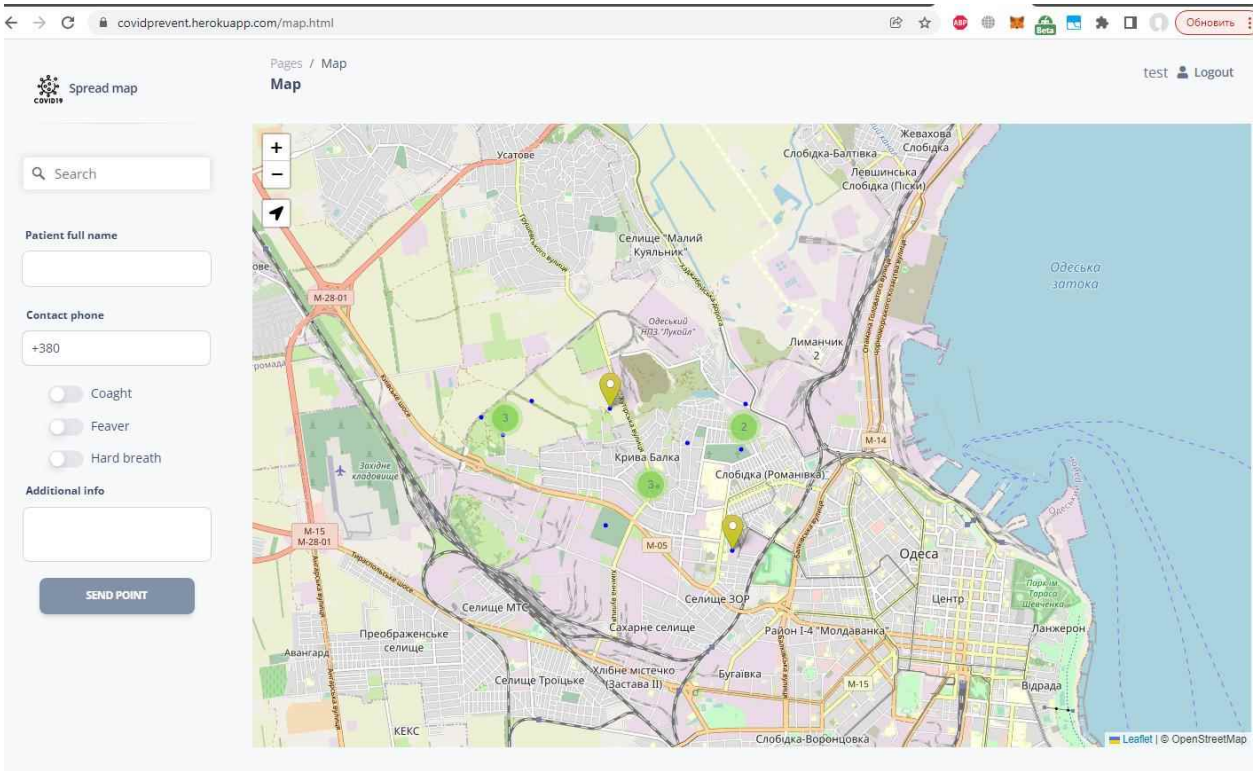

Рисунок 3.11 – Головна сторінка зареєстрованого користувача

Користувач системи має можливість виставляти лише 4 бали за раз, поки вони не будуть затверджені. Система працює таким чином, що кількість спроб обмежена до 4, але якщо один з балів пройшов модерацію або був видалений – користувач зможе поставити кількість балів, які були виділені саме для його облікового запису (рис. 3.13). Цей механізм є базовим фільтром спам-атак і допомагає захистити систему, що розроблюється, від спроб її дестабілізації.

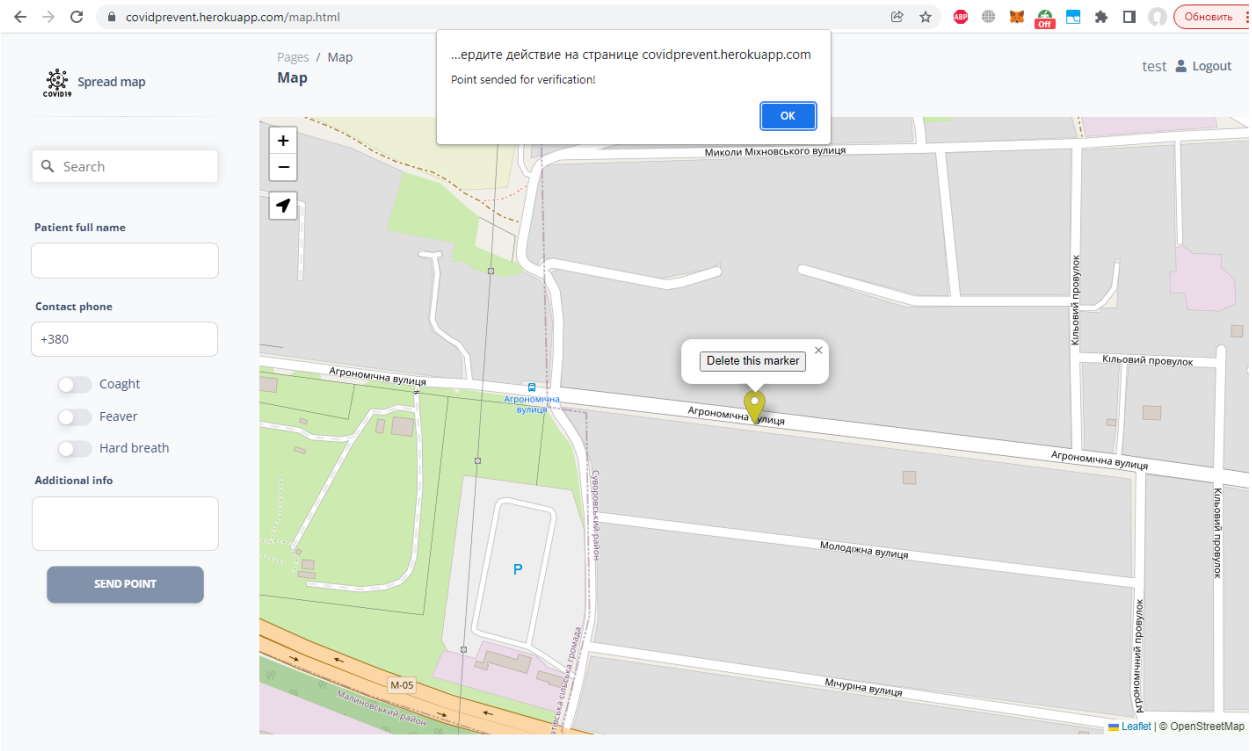

Рисунок 3.12 – Встановлення точки

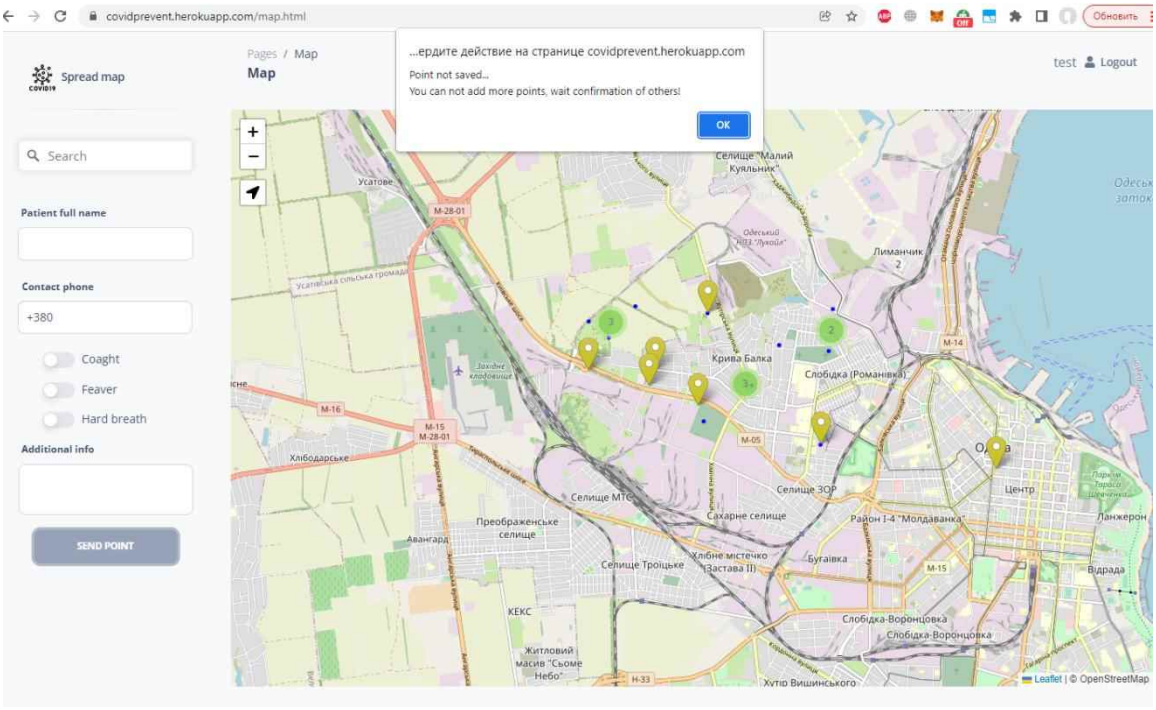

Рисунок 3.13 – Повідомлення про обмеження точки

Як бачимо, шпильки показують іншу інформацію для користувача: щойно поставлена шпилька відображатиме більше інформації, ніж шпильки від інших користувачів (рис. 3.14).

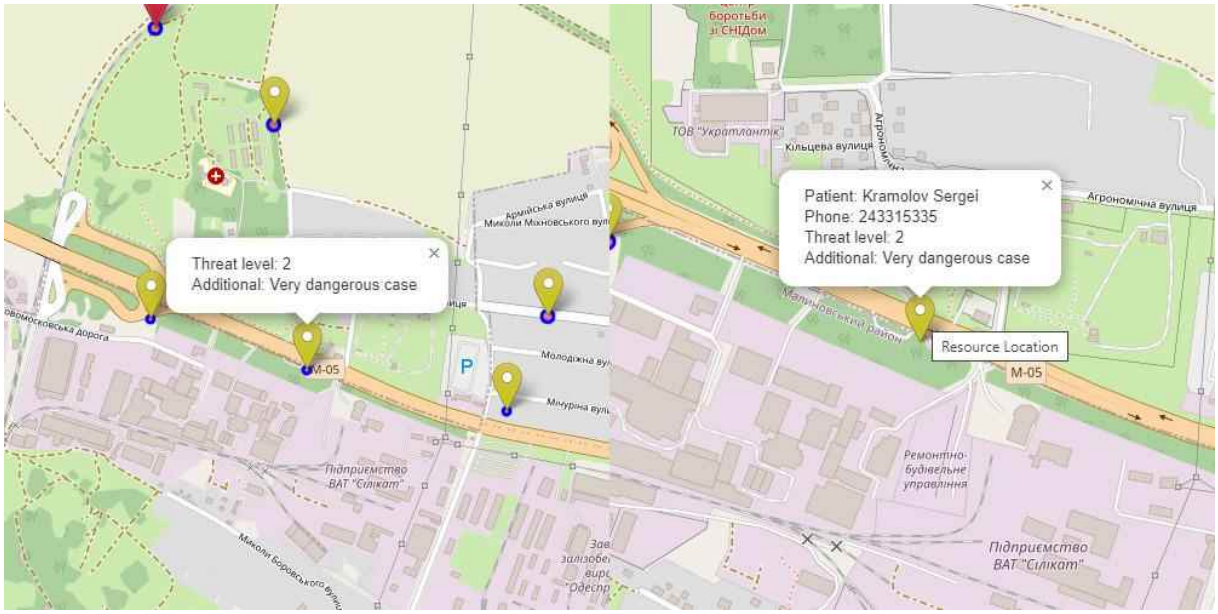

Рисунок 3.14 – Різниця в інформації про бали

Причина цього – конфіденційність інформації. Користувачеві системи не потрібно знати:

– ім'я;

– номер телефону;

– точну адресу пацієнта.

Тільки системний адміністратор може переглядати цю інформацію.

Тепер розглянемо сторінку адміністратора, а також дії, які він може виконувати з системою. Основна відмінність між сторінкою адміністратора та іншими користувачами полягає в кількості доступних опцій (рис. 3.15).

Як і всі користувачі, адміністратор може шукати точки за адресою за допомогою поля введення. Крім того, він може переглядати неперевірені точки на карті (рис. 3.16), координати конкретної точки до і після зміни, а також опис самої точки (рис. 3.17).

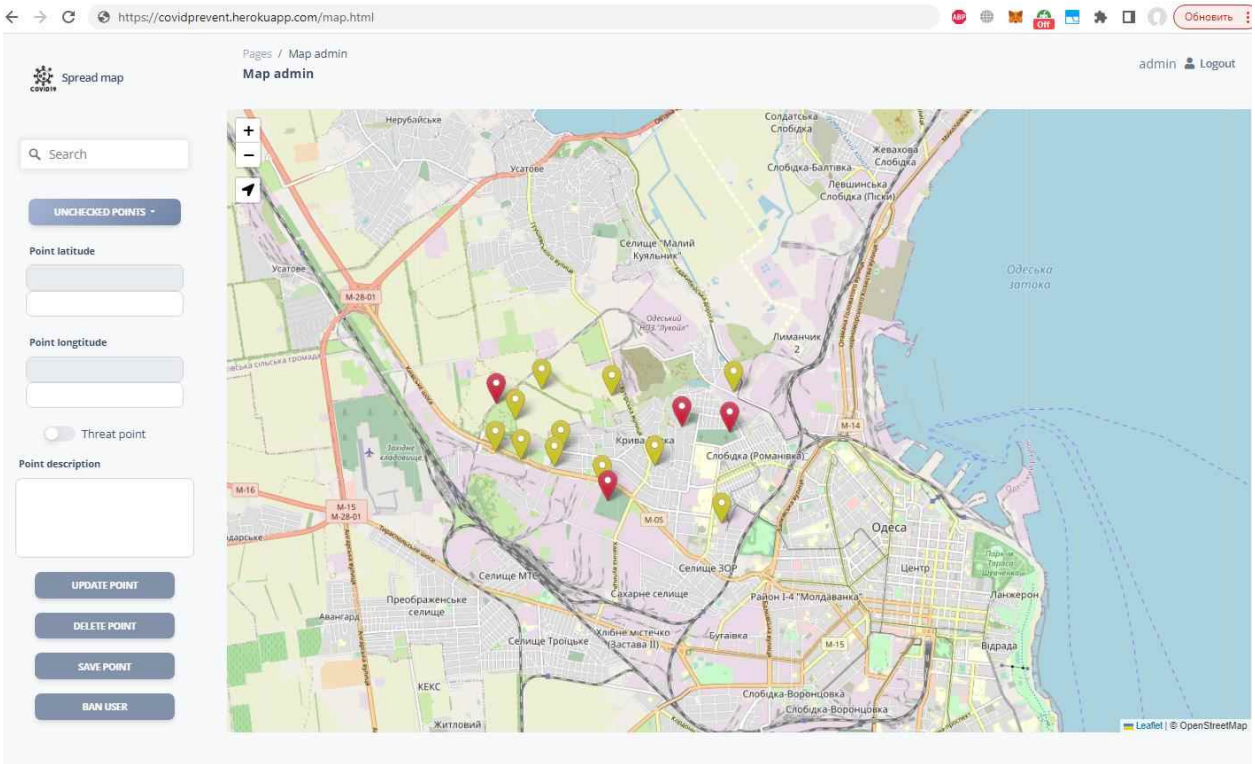

Рисунок 3.15 – Сторінка адміністратора

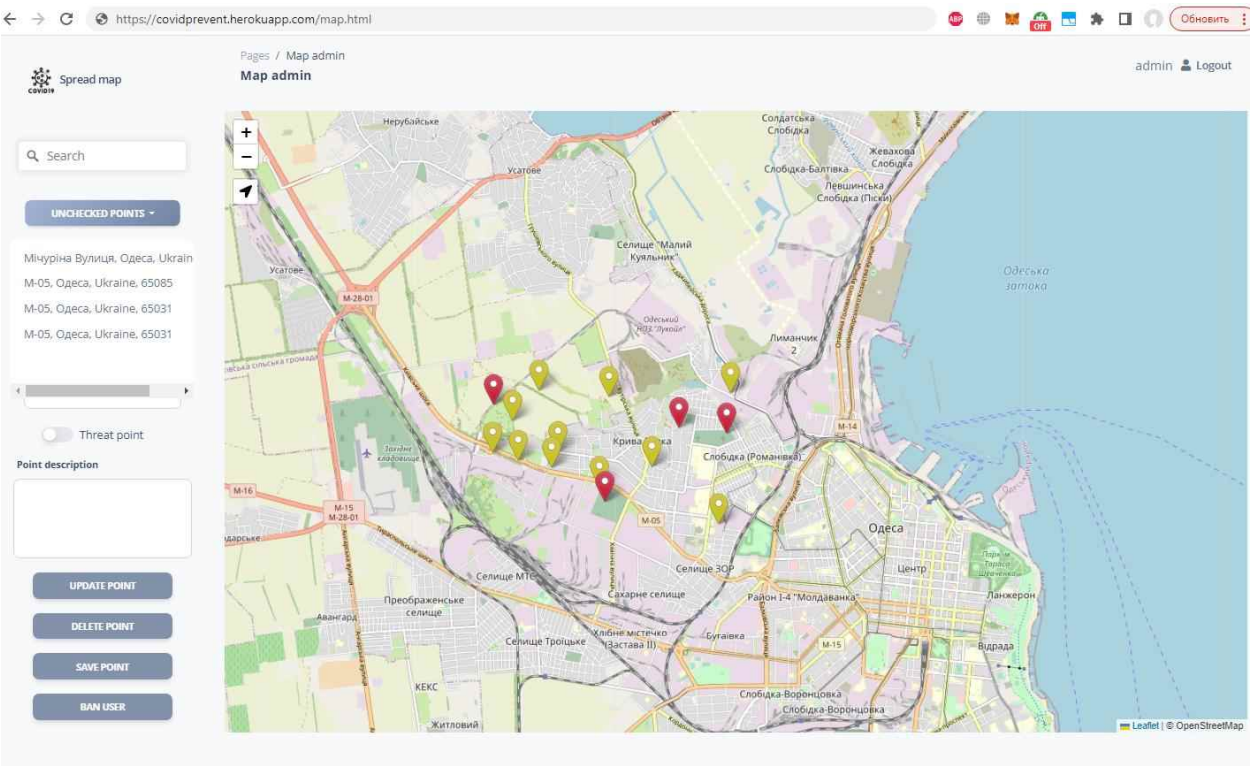

Рисунок 3.16 – Список неперевірених точок

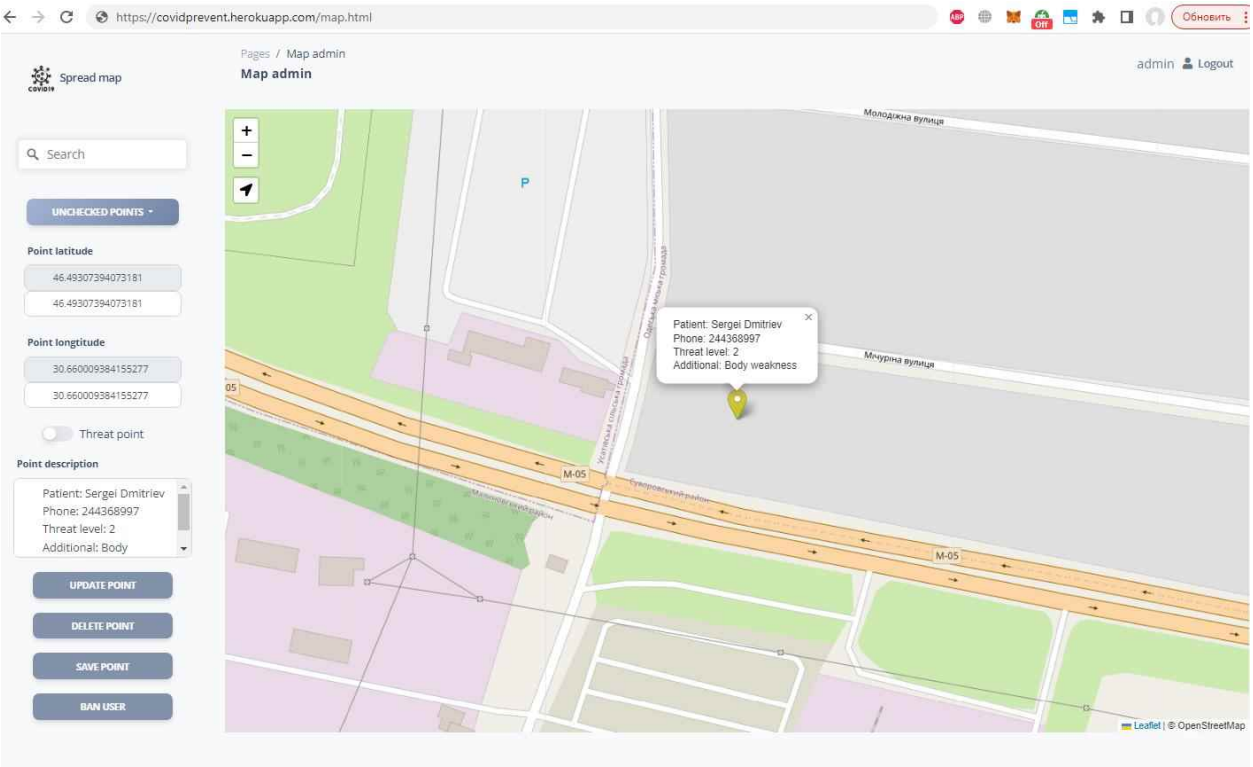

Рисунок 3.17 – Дані точки

За допомогою елемента «Точка загрози» адміністратор може позначити точку як небезпечну, присвоївши їй 3-й рівень загрози. У системі є 3 рівні загрози:

– звичайна точка без ознак небезпеки – 1;

– середній рівень: точка потенційно небезпечна, але рівень загрози все ще низький – 2;

– точка небезпечна та її слід уникати – 3.

При роботі з точками адміністратор має можливість видалення, оновлення та збереження точки. Крім того, адміністратор має можливість «забанити» користувача, який поставив точку на карті (рис. 3.18).

Після цього користувач побачить повідомлення про «бан» (заборону) і не зможе повноцінно взаємодіяти з системою (рис. 3.19).

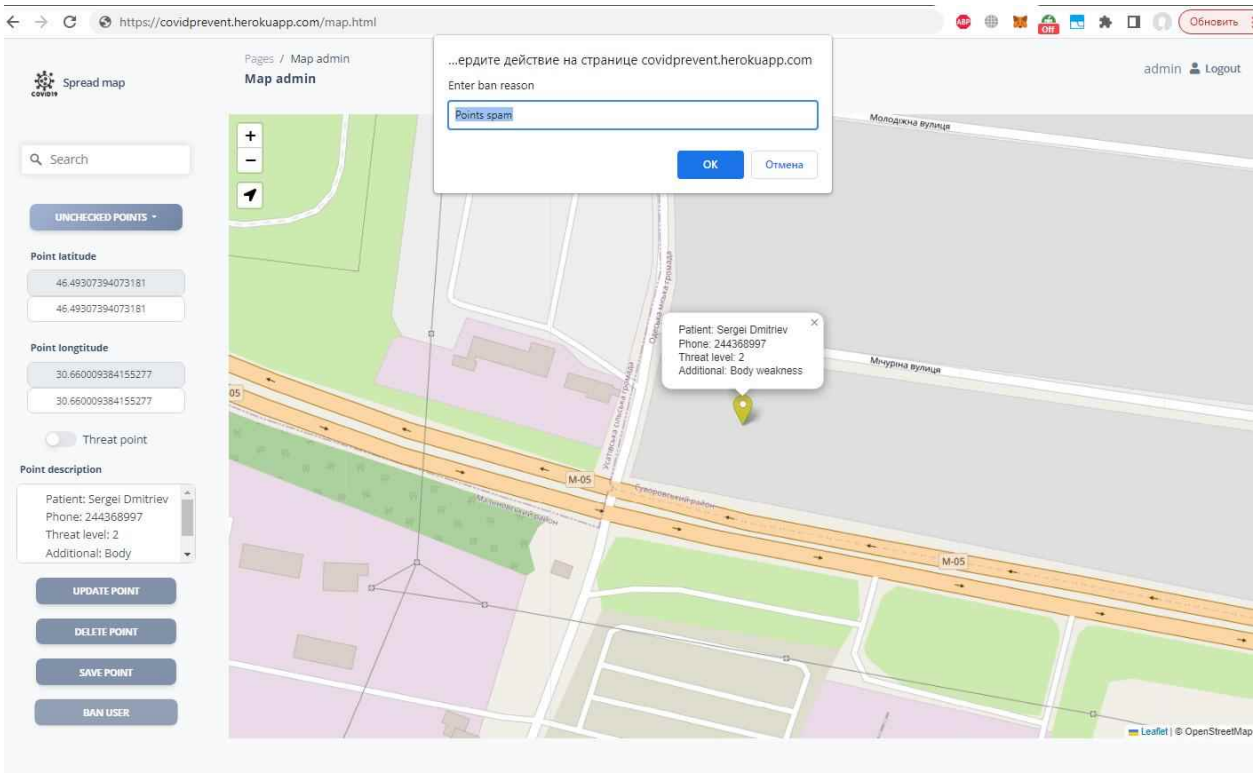

Рисунок 3.18 – «Бан» користувача за точкою

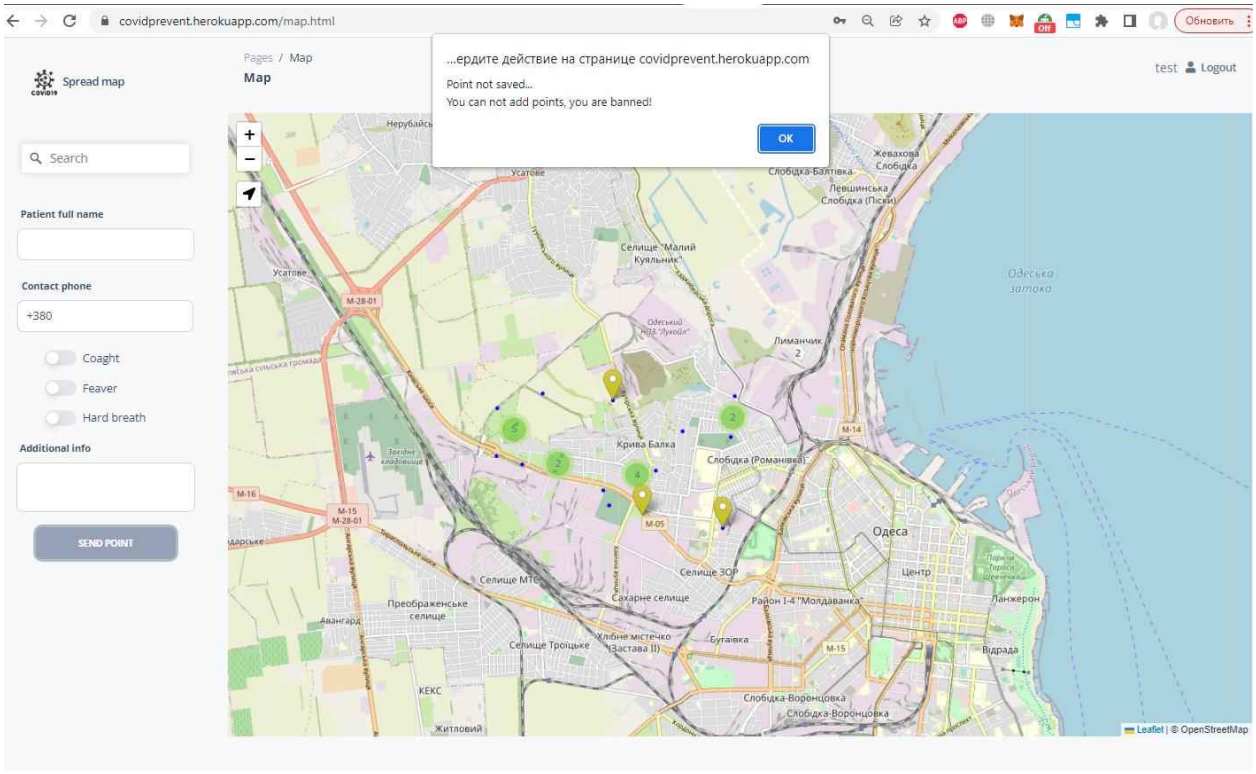

Рисунок 3.19 – Повідомлення про «забанення» користувача

Адміністратор може змінити координати точки, просто перетягнувши її в потрібне місце на карті, а також вказавши нові координати вручну (рис. 3.20).

У цьому випадку значення полів координат будуть змінюватися з кожним переміщенням точки, тобто поточні координати будуть встановлені як нові, а попередні – як старі.

Таким чином, адміністратор може змінити опис точки і додати до нього щось інше. Важливо зазначити, що оновлення точки використовується не лише для модифікації, а й для підтвердження. Якщо точку потрібно підтвердити, необхідно натиснути кнопку «Оновити точку», після чого точка буде видалена з черги на затвердження та збережена як активна.

Адміністратор також має можливість додавати точки на карту, але, на відміну від звичайних користувачів, він немає обмежень щодо ознак або опису точки (рис. 3.21).

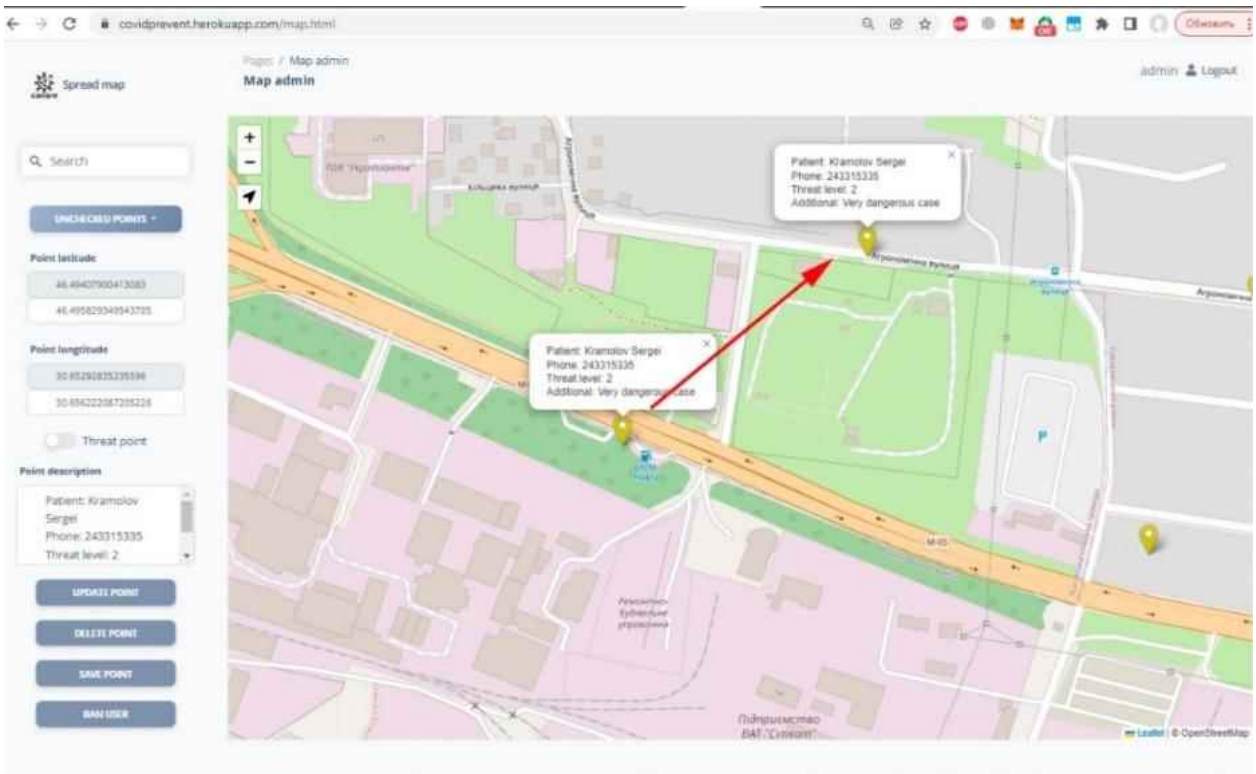

Рисунок 3.20 – Зміна координат точки

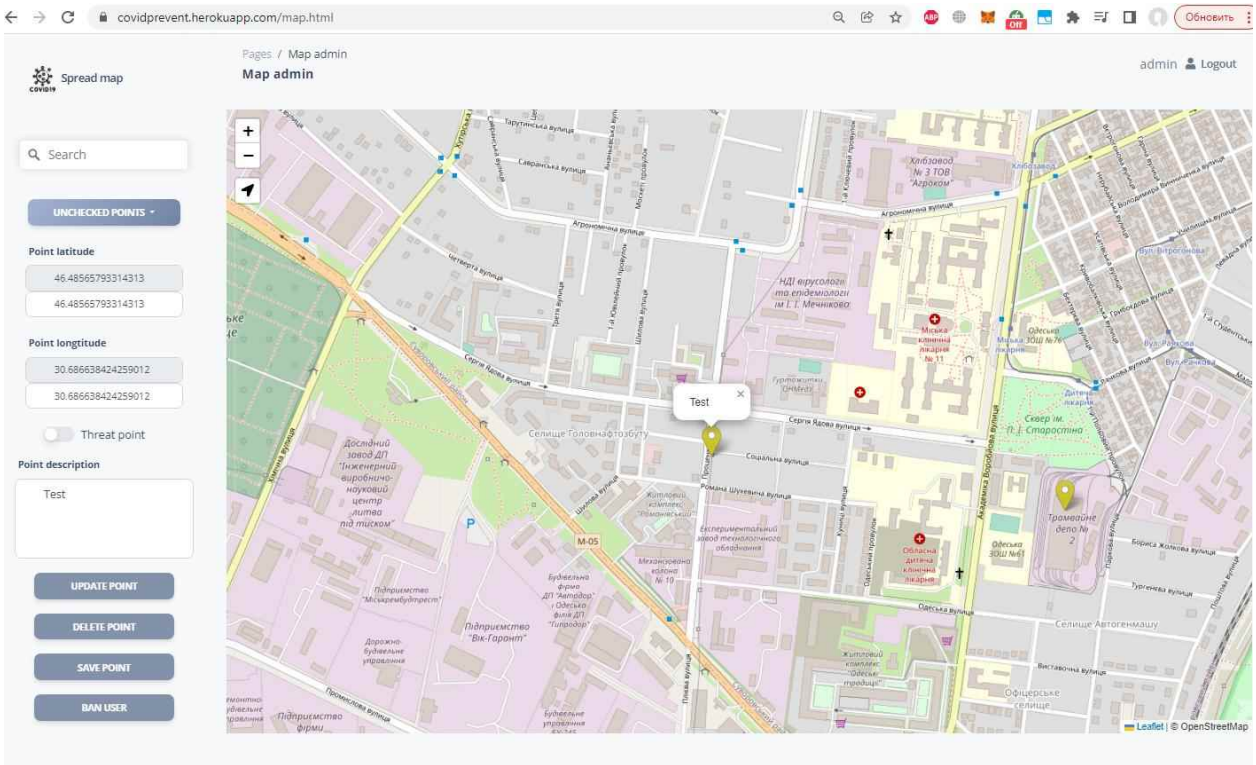

Рисунок 3.21 – Додавання нової точки адміністратором

Адміністратор може надати будь-яку інформацію, яка потрібна при додаванні точки на карту. Коли пункт демонструється звичайному користувачеві, з усієї інформації, яка була показана раніше, буде прихована лише особиста інформація (рис. 3.14).

Приклади оновлення (рис. 3.22), видалення (рис. 3.23) та збереження (рис. 3.24) точки наведено на рисунках нижче.

Крім того, можна використовувати панель адміністратора Django, щоб вручну змінювати системні дані. Це доступно лише для суперкористувача та адміністратора (рис. 3.25).

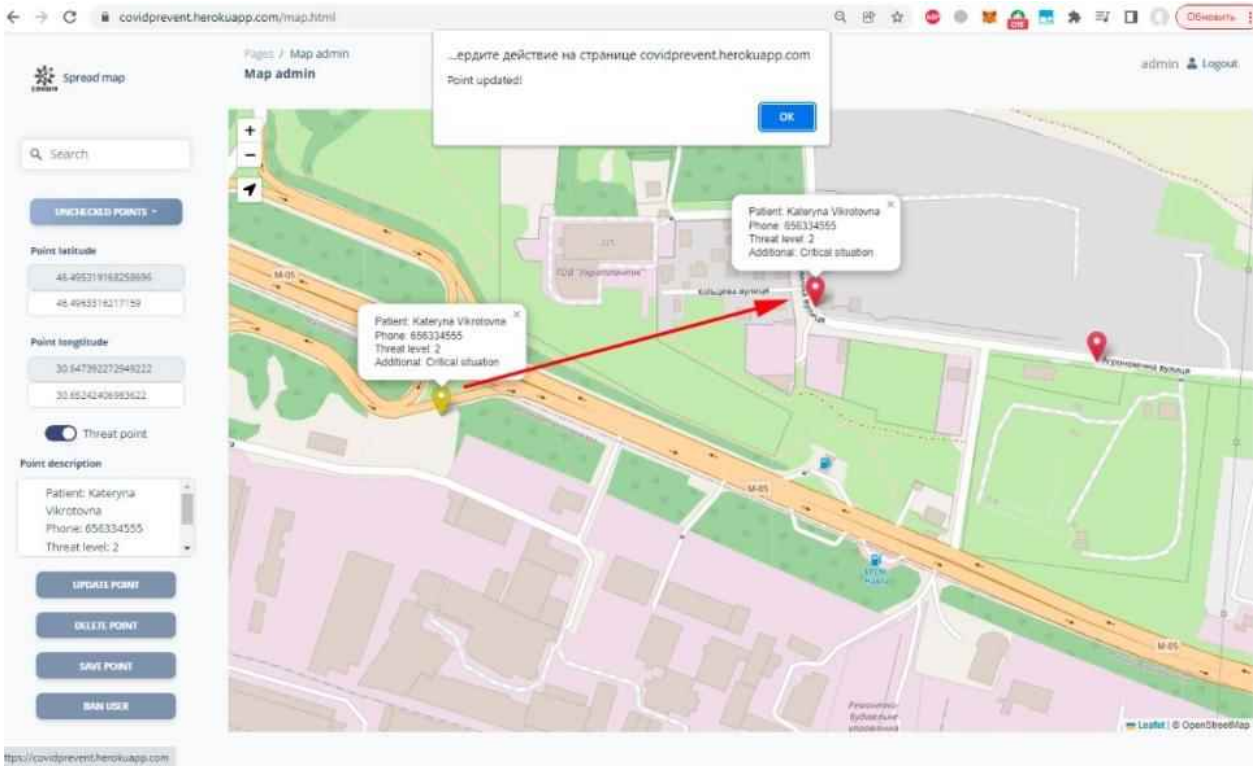

Рисунок 3.22 – Точка оновлення

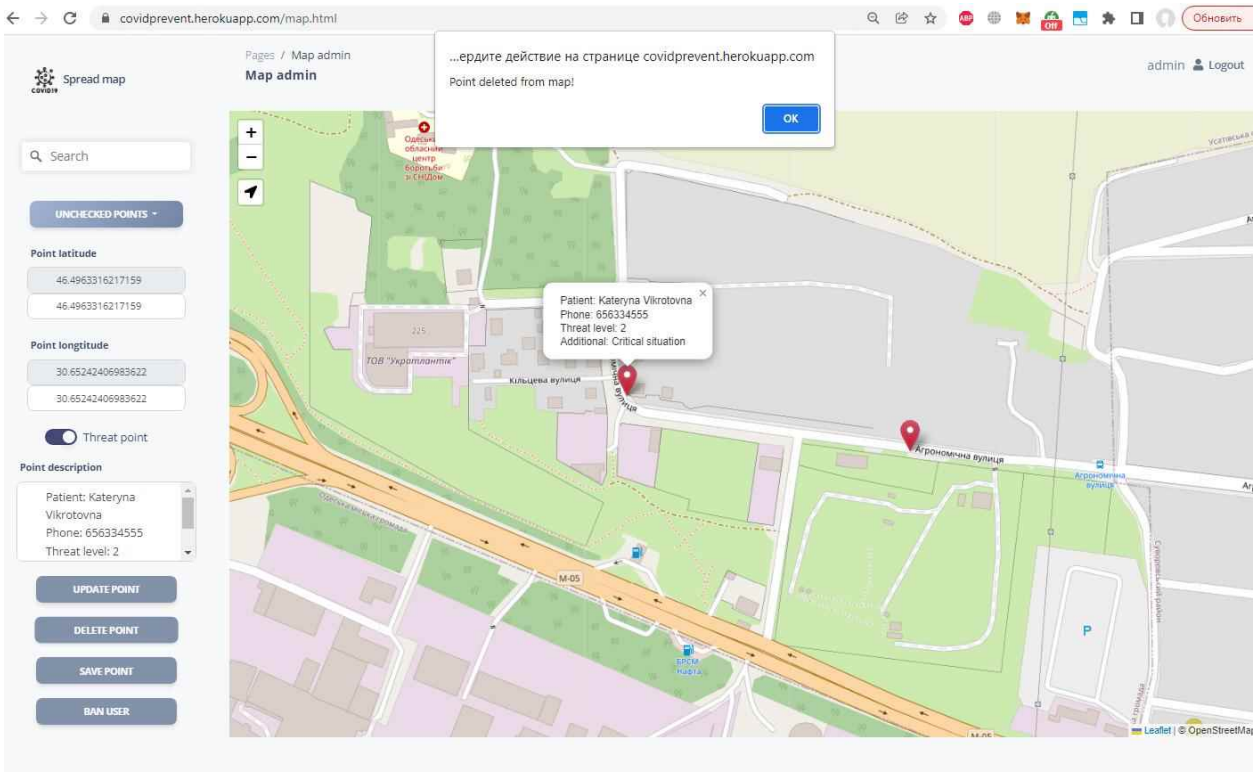

Рисунок 3.23 – Видалення точки

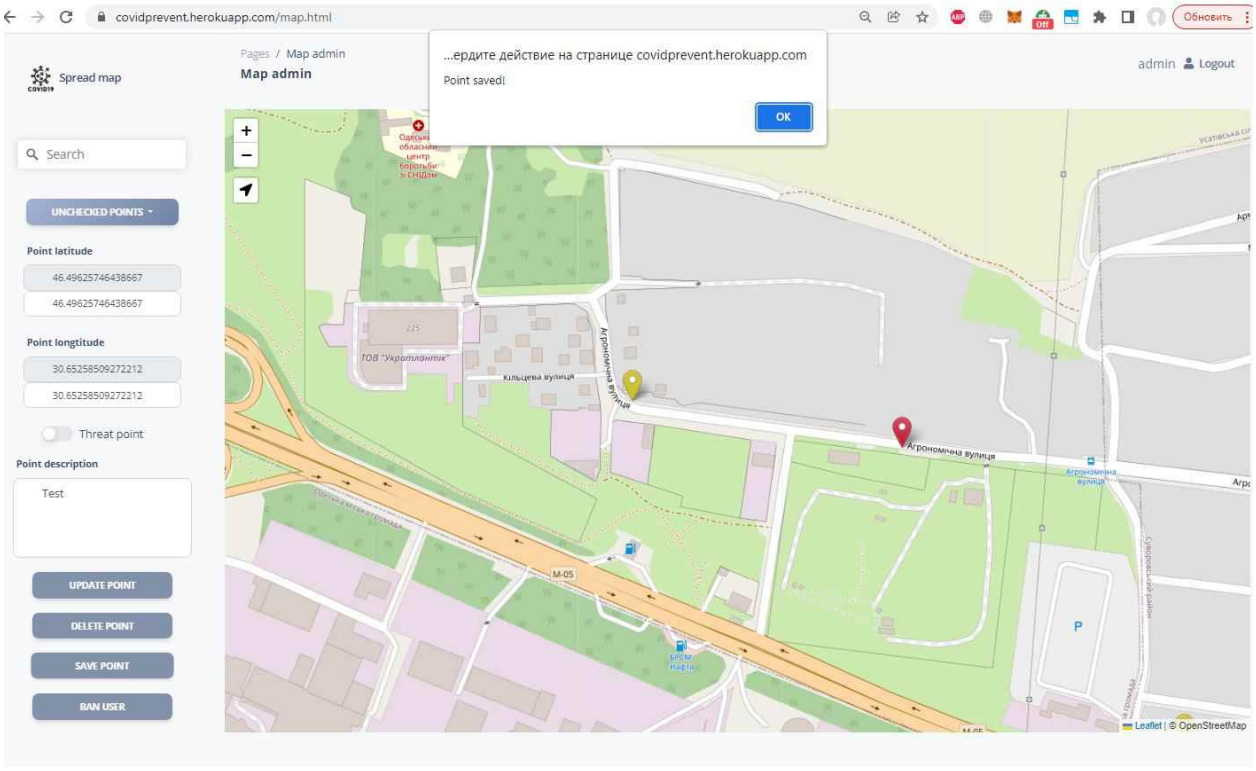

Рисунок 3.24 – Точка збереження

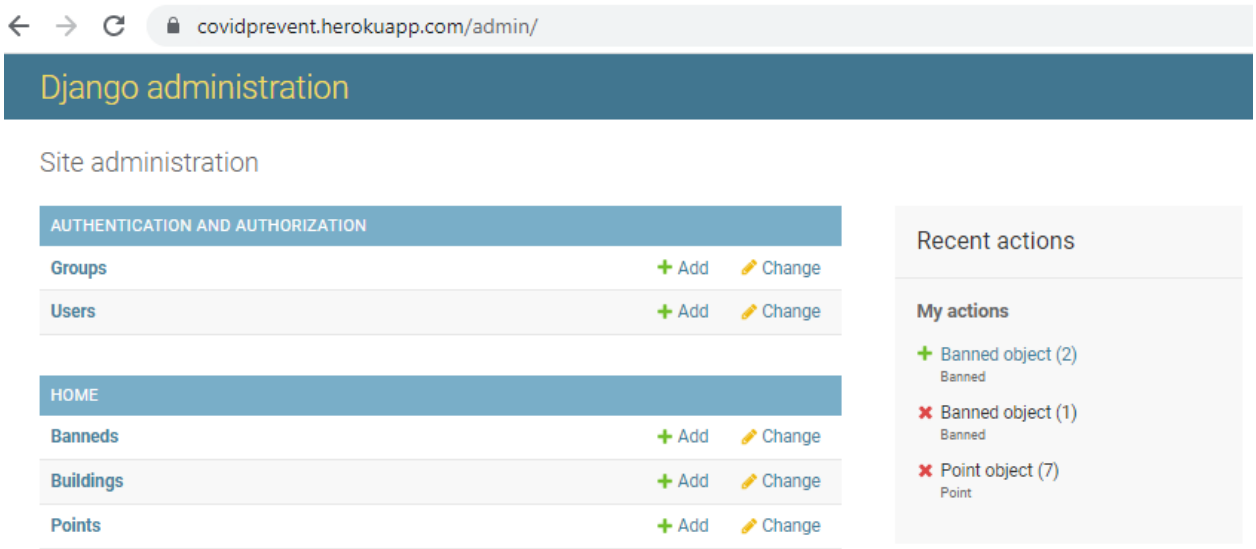

Рисунок 3.25 – Панель адміністратора Django

ВИСНОВКИ

У результаті виконання магістерської роботи було проаналізовано різні типи загроз, визначено їх класифікацію та етапи поширення. Крім того, було розглянуто заходи, які існують для запобігання їх розповсюдженню, а також запобіжні заходи в рамках держави.

На підставі отриманих результатів було визначено, що важливим фактором у разі виникнення загрози пандемії є швидкість реагування, а також наявність підготовленого плану та інструментів для ефективного стримування загрози, що виникає.

У 2020 році світ зіткнувся з пандемією COVID-19, яка показала, що медична система не готова до таких навантажень, а багато сфер життя не пристосовані швидко реагувати на такі ситуації. Багато структур були паралізовані, а інші повністю припинили свою роботу. Через великий обсяг інформації ІС не справлялися з оперативним оновленням даних, що викликало затримку.

Знаючи цю проблему, було запропоновано використання «сили спільноти» («ком'юніті») та дозволити людям ділитися інформацією про нові випадки захворювання. Як показала практика: кооперативність людей і швидкість обміну даними між ними значно вища, ніж звітність офіційних структур. Деякі громадяни можуть помітити симптоми ще до медичного підтвердження, що, у свою чергу, може бути небезпечним для інших.

Дозволяючи людям ділитися такою інформацією у візуальній формі, можна попередити деяких із них і, можливо, вберегти їх від зараження. Хоча точність інформації може бути кращою, її кількість під час пандемії важливіша за її точність. Крім того, пропоновану ІС можна модифікувати та підлаштовувати під будь-яку ситуацію, дозволяючи побудувати універсальний механізм моніторингу, який буде заздалегідь попереджати інших про загрозу, що насувається.

Таким чином, на основі зібраних даних було вирішено побудувати ІС у вигляді веб-інтерфейсу з Django з використанням бібліотеки на основі

HTML5, CSS та JS для фундаментального дизайну сторінок інтерфейсу користувача з метою максимальної сумісності із браузером.

Було також вирішено використовувати Visual Studio Code як IDE, оскільки він доступний як безкоштовне програмне забезпечення та має широкий спектр вбудованої підтримки бібліотек. Крім того, на додаток до цього, є низькі вимоги до ресурсів, що дає змогу виконувати розробку навіть на повільних комп'ютерах. Завдяки підтримці мови Python, що вбудована у Visual Studio Code, цю мову можна використовувати без потреби у сторонніх бібліотеках.

У результаті як наукові дослідження, так і теоретичне розуміння були посилені та розширені. Для усвідомлення особливостей тематичної області створено прототип ІС, що підкріплює концепцію запропонованих рішень.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Coronavirus disease (COVID-19) pandemic. URL: https://www.who.int/emergencies/diseases/novel-coronavirus-2019 (дата звернення: 03.09.2024 р.).
2. Pandemic Influenza Preparedness and Response: A WHO Guidance Document. URL: https://www.ncbi.nlm.nih.gov/books/NBK143061/ (дата звернення: 05.09.2024 р.).
3. Epidemic, Endemic, Pandemic: What are the Differences? URL: https://www.publichealth.columbia.edu/public-health-now/news/epidemic-endemic-pandemic-what-are-differences (дата звернення: 06.09.2024 р.).
4. Biochemist Kary Mullis. URL: https://www.nobelprize.org/nobel_prizes/chemistry/laureates/1993/mullis-autobio.html (дата звернення: 07.09.2024 р.).
5. Real-time polymerase chain reaction. URL: https://en.wikipedia.org/wiki/ Real-time_polymerase_chain_reaction (дата звернення: 08.09.2024 р.).
6. The Use of Real-Time Reverse Transcriptase PCR for the Quantification of Cytokine Gene Expression. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2279895/ (дата звернення: 11.09.2024 р.).
7. Як працює застосунок «Дій вдома». URL: https://www.kmu.gov.ua/news/yak-pracyuye-zastosunok-dij-vdoma (дата звернення: 12.09.2024 р.).
8. Home Quarantine System. URL: https://asuratechnologies.com/ solutions/home-quarantine-system/ (дата звернення: 13.09.2024 р.).
9. The next step in fighting the virus. URL: https://www.paimcos.com/ (дата звернення: 15.09.2024 р.).
10. Partners aim for safe quarantine monitoring system. URL: https://www.uts.edu.au/news/tech-design/partners-aim-safe-quarantine-monitoring-system (дата звернення: 16.09.2024 р.).
11. NSW Smart Sensing Network. URL: https://www.nssn.org.au/paimcos-quarantine-monitoring-system (дата звернення: 17.09.2024 р.).

12. Salaudeen A., Sathasivam S., Nam C. J., Hang, T. Y. Modelling the Early Outbreak of Covid-19 Disease in Malaysia Using SIRS Model with 4-Step Adams-Bashforth-Moulton Predictor-Corrector Method. Applied Mathematics and Computational Intelligence (AMCI). 2023. Vol. 13(2). pp. 121-136.
13. Autoregressive Integrated Moving Average (ARIMA) Prediction Model. By Adam Hayes (Updated July 31, 2024). URL: https://www.investopedia.com/terms/a/autoregressive-integrated-moving-average-arima.asp (дата звернення: 12.10.2024 р.).
14. How to Create an ARIMA Model for Time Series Forecasting in Python. URL: https://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/ (дата звернення: 13.10.2024 р.).
15. Augmented Dickey Fuller Test (ADF Test) – Must Read Guide by Selva Prabhakaran. URL: https://www.machinelearningplus.com/time-series/augmented-dickey-fuller-test/ (дата звернення: 15.10.2024 р.).
16. RedMonk. The developer-focused industry analyst firm. URL: https://redmonk.com/latest-research/ (дата звернення: 16.10.2024).
17. Python documentation [Electronic source]. URL: https://docs.python.org/ 3/ (дата звернення: 17.10.2024).
18. Django documentation [Electronic source]. URL: https://docs.djangoproject.com/en/3.0/ (дата звернення: 18.10.2024).
19. Django Project MVT Structure [Electronic source]. URL: https://www.geeksforgeeks.org/django-project-mvt-structure/ (дата звернення: 19.10.2024).
20. Documentation of VSCode [Electronic source]. URL: https://code.visualstudio.com/ docs (дата звернення: 20.10.2024).
21. Microsoft Build. URL: https://en.wikipedia.org/wiki/Microsoft_Build. (дата звернення: 22.10.2024).

ДОДАТОК А

Основні фрагменти програмного коду

Програмний код розробленого прототипу ІС містить сотні рядків коду та багато файлів, тому для демонстрації наводимо лише код компонента «view», який міститься у файлі views.py та демонструє основні функції системи.

```
// The contents of the views.py file
# -*- encoding: utf-8 -*-
import json
from django import template
from django.contrib.auth.decorators import login_required
from django.http import JsonResponse, HttpResponse
from django.shortcuts import redirect
from django.template import loader
from django.views.decorators.csrf import csrf_exempt
from .db_actions import check_ban, get_points, get_unchecked,
update_point, save_point, drop_point, ban_user, check_key,
check_spam

mapbox_access_token =
'pk.eyJ1IjoidGl6aHByb2dlciIsImEiOiJjbDN2dXB3bmYxdjYyM2lsdHZwZjRh
bmJwIn0.LrtqGvP5I0RhbKjyVpDbwA'


def home(request):
    context = {}
    if not request.user.is_authenticated:
        context['auth'] = False
        context['staff'] = False
        context['superuser'] = False
        context['page_name'] = 'Guest'
        context['mapbox_access_token'] = mapbox_access_token
        html_template = loader.get_template('home/map.html')
        return HttpResponse(html_template.render(context,
request))
    else:
        return redirect('/map.html')


@login_required(login_url="/login/")
```

```
def main(request):
    context = {}
    context['auth'] = False
    context['staff'] = False
    context['superuser'] = False
    context['segment'] = 'map'
    context['mapbox_access_token'] = mapbox_access_token
    context['user_name'] = request.user.username
    context['page_name'] = 'Map'

    if request.user.is_staff or request.user.is_superuser:
        context['staff'] = True
        context['page_name'] = 'Map admin'

    elif request.user.is_authenticated:
        context['auth'] = True

    try:
        html_template = loader.get_template('home/map.html')
        return HttpResponse(html_template.render(context,
request))
    except template.TemplateDoesNotExist:
        html_template = loader.get_template('home/page-
404.html')
        return HttpResponse(html_template.render(context,
request))

    except:
        html_template = loader.get_template('home/page-
500.html')
        return HttpResponse(html_template.render(context,
request))


def getPoints(request):
    if request.method == 'GET':
        return JsonResponse(get_points())
    else:
        response = JsonResponse({'error': 'Condition check not
satisfied, permission error'})
        response.status_code = 403
        return response


def getUnchecked(request):
    if request.method == 'GET':
```

```
        return JsonResponse(get_unchecked())
    else:
        response = JsonResponse({'error': 'Condition check not
satisfied, permission error'})
        response.status_code = 403
        return response


def savePoint(request):
    if request.method == 'POST' and (request.user.is_superuser
or request.user.is_staff or request.user.is_authenticated):
        if check_ban(request.user):
            response = JsonResponse({'error': 'You can not add
points, you are banned!'})
            response.status_code = 500
            return response

        if check_spam(request.user) and not
(request.user.is_superuser or request.user.is_staff):
            response = JsonResponse({'error': 'You can not add
more points, wait confirmation of others!'})
            response.status_code = 500
            return response

        data = json.loads(request.body)
        res = save_point(lat=data['latitude'],
lng=data['longtitude'], desc=data['description'],
adrs=data['address'], threat=data['threat'],
checked=data['checked'], user=request.user)
        if res:
           return HttpResponse(200)
        response = JsonResponse({'error': 'Something wrong on
server side, oooops...'})
        response.status_code = 500
        return response
    else:
        response = JsonResponse({'error': 'Condition check not
satisfied, permission error'})
        response.status_code = 403
        return response


@csrf_exempt
def institutePoint(request, key):
    print(key)
    if request.method == 'POST' and check_key(key):
```

```
        data = json.loads(request.body)
        if save_point(lat=data['latitude'],
lng=data['longtitude'], desc=data['description'],
adrs=data['address'], threat=data['threat'], bkey=key):
           return HttpResponse(200)
        response = JsonResponse({'error': 'Something wrong on
server side, oooops...'})
        response.status_code = 500
        return response
    else:
        response = JsonResponse({'error': 'Not authorized
request...'})
        response.status_code = 403
        return response


def updatePoint(request):
    if request.method == 'POST' and (request.user.is_superuser
or request.user.is_staff):
        data = json.loads(request.body)
        res = update_point(data['old_latitude'],
data['old_longtitude'], data['latitude'], data['longtitude'],
data['description'], data['address'], data['threat'],
data['checked'])
        if res:
           return HttpResponse(200)
        response = JsonResponse({'error': 'Something wrong on
server side, oooops...'})
        response.status_code = 500
        return response
    else:
        response = JsonResponse({'error': 'Condition check not
satisfied, permission error'})
        response.status_code = 403
        return response


def deletePoint(request):
    if request.method == 'DELETE' and (request.user.is_superuser
or request.user.is_staff):
        data = json.loads(request.body)
        if drop_point(data['latitude'], data['longtitude']):
           return HttpResponse(200)
        response = JsonResponse({'error': 'Something wrong on
server side, oooops...'})
        response.status_code = 500
```

```
        return response
    else:
        response = JsonResponse({'error': 'Condition check not
satisfied, permission error'})
        response.status_code = 403
        return response


def banUser(request):
    if request.method == 'DELETE' and (request.user.is_superuser
or request.user.is_staff):
        data = json.loads(request.body)
        res = ban_user(data['latitude'], data['longtitude'],
data['reason'])
        if res == True:
           return HttpResponse(200)
        else:
            response = JsonResponse({'error': res})
            response.status_code = 500
            return response
    else:
        response = JsonResponse({'error': 'Condition check not
satisfied, permission error'})
        response.status_code = 403
        return response
```