Міністерство освіти і науки України
Харківський національний університет імені В. Н. Каразіна
Факультет комп'ютерних наук
Кафедра теоретичної та прикладної системотехніки

«Затверджую»
Зав. кафедри теоретичної та
прикладної системотехніки
_____ д.т.н., проф. С. І. Шматков
«___» грудня 2022 р.

Пояснювальна записка

до кваліфікаційної роботи
магістра

на тему: «**МОДЕЛЬ СИСТЕМИ МОНІТОРИНГУ АВТОМОБІЛЬНИХ ПЕРЕВЕЗЕНЬ НА БАЗІ ТЕХНОЛОГІЇ ХМАРНИХ ОБЧИСЛЕНЬ»**

Захищено на засіданні
Атестаційної комісії № 45
протокол № __ від __.12.2022 р.
Оцінка _____ / _____
Голова Атестаційної комісії
_____**МІНУХІН С. В.**

Виконав:

студент 6 курсу, групи КУ– 61
за спеціальністю 151 – Автоматизація та комп'ютерно-інтегровані технології.
Галузь знань 15 – Автоматизація та приладобудування
Попова Анастасія Олександрівна____

Керівник:

д.т.н., с.н.с., професор кафедри теоретичної та прикладної системотехніки
Толстолузька Олена Геннадіївна_____

Рецензент:

д.т.н., доцент, завідувач кафедри безпеки інформаційних систем і технологій
Росомахін Сергій Геннадійович_____

Харків – 2022

АНОТАЦІЯ

Пояснювальна записка до магістерської атестаційної роботи складається зі вступу, трьох розділів, висновків, списку використаних джерел і двох додатків. Загальний обсяг роботи складає 74 сторінки, із яких 60 сторінок основної частини з 31 рисунками, 1 таблицею, 36 найменуваннями списку використаних джерел та двома додатками.

Метою кваліфікаційної роботи є дослідження існуючих систем відстежування, принципу їх роботи, розробка моделі системи на базі хмарних технологій, яка буде постачати дані на розроблений сайт з картою щоб відображати на ній історію місцезнаходження транспорту, базуючись на даних місцезнаходження, отриманих за допомогою розробленого пристрою для їх визначення.

Об'єкт дослідження – система відслідковування транспорту який застосовується для міських перевезень або приватних рейсів, яка допомагає диспетчеру вести спостереження за ним з ціллю моніторингу його місцезнаходжень під час виконання робочих завдань.

Предмет дослідження – системи відстежування транспорту, що використовуються на сьогоднішній день. Розглядаються способи аналізу вхідних геолокаційних даних для побудови системи їх аналізу. У роботі розкривається принцип дії системи відстежування, описуються різні засоби для визначення геолокації, розробляється прототип системи слідкування за транспортом.

Проблема, яка вирішується в кваліфікаційній роботі полягає в тому, щоб скориставшись хмарними технологіями та підходами до створення безсерверних систем мінімізувати витрати на оренду серверів, досягти розгортання всіх сервісів у хмарній системі за короткий проміжок часу, за

допомогою спрощеного мережевого протоколу, орієнтованого на обмін повідомленнями між пристроями за принципом видавець-передплатник спростити передачу даних між пристроєм відстеження та хмарною системою обробки.

Область застосування – розробка систем відстежування транспорту. Розроблений прототип системи слідкування за транспортом може використовуватися в сфері малого та середнього бізнесу де важливим є відстежування історії руху службового транспорту.

Ключові слова: AWS, GPS, IoT, MQTT, RaspberryPi, SIM7000E, AT, CloudFormation, геолокація, система відстежування, модель.

ABSTRACT

The explanatory note to the master's attestation work consists of an introduction, three sections, conclusions, a list of used sources and two appendices. The total volume of the work is 74 pages, of which 60 pages are the main part with 31 figures, 1 table, 36 names of the list of used sources and two appendices.

The purpose of the qualification work is the study of existing tracking systems, the principle of their operation, the development of a model of the system based on cloud technologies, which will supply data to the developed site with a map to display the history of the location of the transport on it, based on the location data obtained using the developed device for their definition.

The object of the research is a transport tracking system used for urban transportation or private flights, which helps the dispatcher to monitor it in order to monitor its locations during the performance of work tasks.

The subject of research is transport tracking systems used today. Methods of analyzing incoming geolocation data for building a system for their analysis are considered. The work reveals the principle of operation of the tracking system, describes various means for determining geolocation, and develops a prototype of the transport tracking system.

The problem that is solved in the qualification work is to use cloud technologies and approaches to creating serverless systems to minimize the costs of renting servers, to achieve the deployment of all services in the cloud system in a short period of time, using a simplified network protocol focused on the exchange of messages between publisher-subscriber devices to simplify data transfer between the tracking device and the cloud processing system.

The field of application is the development of transport tracking systems. The developed prototype of the transport tracking system can be used in the field of small

and medium-sized businesses where it is important to track the history of the movement of official vehicles.

Keywords: AWS, GPS, IoT, MQTT, RaspberryPi, SIM7000E, AT, CloudFormation, geolocation, tracking system, model.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ І УМОВНИХ ПОЗНАЧЕНЬ

API – Application Programming Interface;

AWS – Amazon Web Services;

CLI – Command Line Interface;

GPS – Global Positioning System;

GSM – Global System for Mobile Communications;

GNSS – Global Navigation Satellite System;

IDE – Integrated Development Environment;

IoT – Internet of Things;

MQTT – Message Queue Telemetry Transport;

NTP – Network Time Protocol;

RFB – Remote framebuffer;

SQL – Structured Query Language;

TCP – Transmission Control Protocol;

UART – Universal Asynchronous Receiver-Transmitter;

YAML – YAML Ain't Markup Language;

JSON – JavaScript Object Notation;

VNC – Virtual Network Computing.

ВСТУП

За останні роки дедалі більше стало приділятись уваги використанню відстежування пересування транспорту, приватні перевізники дедалі більше потребують контролю над пересуванням своїх підлеглих з ціллю покращення логістичних задач та надійності перевезень.

Актуальність роботи. В разі приватних автомобільних перевезень при здійсненні важливих рейсів ключову роль грає людина диспетчер. Зазвичай вона здійснює контроль водіїв та перевіряє їх слідування за маршрутом через спеціальну програму та прилад відстеження, встановлений у кожному автомобілі. Актуальність роботи полягає в тому, щоб покращити можливість відстежування місцезнаходження автомобілей, що займаються перевезеннями цінних вантажів, маючи всю історію слідування з часом перебування у кожній точці. Спростити роботу диспетчера перевезень.

Метою дослідження є проаналізувати та дослідити існуючі системи відстежування, розробити свою систему, яка зможе відображати на карті історію маршруту машин з можливістю бачити час перебування на кожній точці, базуючись на даних місцезнаходження та часу, отриманих за допомогою розробленого пристрою для їх визначення.

Об’єкт дослідження – це система відстежування транспорту, яка допомагає диспетчеру вести спостереження за ним з ціллю моніторингу його місцезнаходження.

Методи дослідження: аналіз роботи систем відстежування, розбір способів передачі даних від пристроїв, пояснення.

Предмет дослідження – системи відстежування транспорту, що використовуються на сьогоднішній день. Способи аналізу вхідних геолокаційних даних для побудови системи їх обробки. Принцип дії системи

відстежування, засоби для визначення геолокації, прототип системи слідкування за транспортом.

Завдання дослідження

1. Виконати аналіз існуючої системи відстеження місцезнаходження.
2. Виконати аналіз пристроїв що можуть ділитись своїм місцезнаходженням з іншими системами чи пристроями.
3. Розробити модель системи відстежування транспорту на базі хмарних технологій, налаштувати зв’язок між пристроєм стеження та системою для передачі даних.
4. Виконати тестування розробленої моделі системи, зробити аналіз.

РОЗДІЛ 1
АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ

Розроблено багато додатків для відстежування місцезнаходження транспорту. Системи слідкування важливі, не тільки для міського чи міжобласного видів транспорту, а й для підприємств, робота яких тісно пов’язана з перевезеннями наприклад якогось важливого вантажу чи просто передислокацією вахтового транспорту у інше місце роботи.

1.1. Огляд та постановка задачі

Основна мета будь-якої системи відстеження – оцінка локації об'єкта що відстежується з мінімальною похибкою. Диспетчери можуть отримувати інформацію про історію місцезнаходжень об’єкта та про поточне його розташування за допомогою приватного веб-сайту, доступного тільки конкретним користувачам, який буде запитувати координати з системи обробки та відображати їх на карті, малюючи маршрут. Сервіс заснований на безсерверній архітектурі з використанням хмарних технологій і може бути розташований ближче до кінцевих користувачів шляхом перенесення його у інший географічний регіон.

1.2. Існуючі рішення

Найвідомішими для відстежування транспорту є такі системи як «Easy way» (для країн СНГ), «Moovit» (працює по всьому світу) та системи слідкування за шкільними автобусами «Loqqat», «BusHive», «BussBoss».

В Україні досить популярним є додаток «Easy way». Він працює у 66 містах нашої країни та є основним способом для відстежування міського транспорту. За допомогою нього можна слідкувати не тільки за автобусами, а й за тролейбусами. Слід зазначити, що не у всіх містах даний сервіс точно

визначає місцезнаходження транспорту. У таких містах як Дніпро, Тернопіль, Полтава, Житомир, Київ, Кропивницький та Кам’янське відстежування геолокації у системі «Easy way» працює значно краще, ніж у інших тому що у цих містах використовують спосіб відстежування за допомогою отримання даних від GPS-супутників. Даний сервіс має мобільний додаток який працює під операційними системами Android та IOS.

«Moovit» є безкоштовним мобільним додатком для отримання інформації про громадський транспорт і навігацію. «Moovit» надає інформацію про транспорт в режимі реального часу, в тому числі автобуси, тролейбуси, трамваї, потяги, метрополітен і пороми. Користувачі можуть побачити довколишні зупинки і станції на карті, а також планувати поїздки на всіх видах транспорту на основі даних в реальному часі. «Moovit» доступний в більш ніж 350 містах по всьому світу.

1.3. Порівняння існуючих рішень

Розглянуті вище системи мають у собі деякі мінуси. Наприклад система «E-way» встановлюється тільки у новому міському транспорті, що поставляється з заводу уже з вбудованим GPS-трекером. Досить часто виникає збій відображення місцезнаходження транспорту, спричинений поганим зв’язком пристрою передачі даних до системи обробки або з поганою видимістю сигналів супутників. Мобільний застосунок працює не з усіма версіями операційних систем смартфонів та має досить великий об’єм займаної пам'яті.

Додаток «Moovit» є більш сучасним ніж «E-way», однак теж має у собі недоліки у вигляді не всього доступного транспорту і в більшості випадків відображається лише маршрут а не місцезнаходження транспорту тому що додаток є безкоштовним і не всі координати є легкими для отримання у тих же провайдерів. Також цей додаток є виключно мобільним.

До негативних сторін систем слідкування за шкільними автобусами можна віднести те, що всі ці засоби належать до приватних компаній та потребують щомісячної оплати послуг.

Висновки за розділом 1

Аналіз існуючих рішень з відстежування транспорту показує, що для відстеження транспорту, який рухається здебільшого у рамках міста найчастіше застосовується пристрій що може визначати своє місцезнаходження та ділитись їм за допомогою GSM технології. Принцип роботи розглянутих систем відстежування транспорту складається з двох основних етапів – визначення координат та передача їх до системи обробки.

РОЗДІЛ 2
РОЗГЛЯД СПОСОБІВ ВІДПРАВКИ ДАНИХ З ПРИСТРОЇВ

2.1 Інтернет речей

Інтернет речей описує мережу фізичних об’єктів у які вбудовано датчики, програмне забезпечення та інші технології для підключення та обміну даними з іншими пристроями та системами через Інтернет. Ці пристрої варіюються від звичайних побутових предметів до складних промислових інструментів. Типова система IoT складається з чотирьох компонентів якими є датчики і актуатори, мережевий рівень, сервісна підтримка або підтримка додатків та рівень кінцевого додатку [1,2].

Датчики є основними «речами» архітектури IoT. Вони представляють собою фізичні пристрої, які збирають інформацію з реального середовища, наприклад температуру, хімічний склад, якість повітря тощо. Іншими словами, датчики перетворюють певне фізичне явище в цифрову форму.

Актуатори – це пристрої, які спроможні приймати електричний сигнал і перетворювати його на якусь фізичну дію. Як приклад можна привести взаємодію датчиків і актуаторів таку як вимикання світла в розумному будинку, коли поруч нікого немає.

Мережевий рівень служить для комунікації пристроїв між собою і складається з бездротових протоколів, які дозволяють пристроям спілкуватися між собою, передавати дані датчиків у хмару та шифрувати всі комунікації.

Сервісна підтримка та підтримка додатків відносяться до внутрішньої інфраструктури, що керує бізнес-логікою продуктів IoT. Вони складаються з хмарних або локальних серверів і служб, збирають, зберігають і обробляють показання датчиків.

Додаток IoT – це загальний термін, що описує всі види програм, які дозволяють кінцевим користувачам спостерігати дані датчиків, взаємодіяти з підключеними пристроями та налаштовувати їх [3, 4].

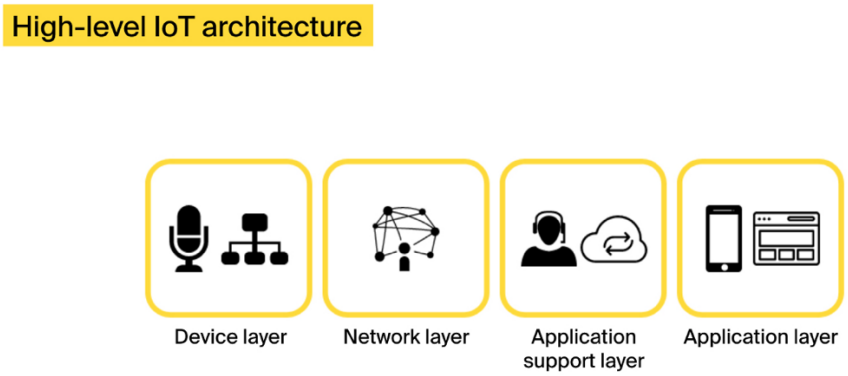


Рисунок 2.1 – Складові архітектури IoT на високому рівні

2.2 Обмін даними між пристроями

Пристрої можуть обмінюватись даними за допомогою різних способів передачі даних таких як стільниковий зв’язок, супутниковий зв’язок, Wi-Fi, Bluetooth тощо. Кожний з методів підключення відрізняється енергоспоживанням, діапазоном і пропускною здатністю. Вибір найкращого способу підключення залежить від програми IoT, але всі вони виконують те саме завдання – передають дані в хмару. Комунікаційний рівень складається з рішення фізичного зв'язку, і спеціальних протоколів, які є спеціально призначеними для використання в різних IoT середовищах (ZigBee, Thread, Z-Wave, MQTT, LwM2M). Для того щоб встановити фізичний зв’язок між пристроями зазвичай використовується спеціальна SIM-карта IoT, яка ще називається M2M SIM Card (machine-to-machine). Вона використовується в IoT пристроях для їх ідентифікації у мережі так як через них намагаються установити підключення до 2G, 3G, 4G-LTE, Cat-M, NB-IoT.

Висновки за розділом 2

Системи побудовані на базі Інтернету речей часто застосовують для того щоб фізичним об’єктам, таким як різноманітні датчики або пристрої надати змогу відправляти зібрані дані до спеціальної платформи для подальшої обробки чи обмінюватись ними між іншими такими пристроями. Платформи обробки бувають різних типів і використовуються в залежності від потреб користувача. Пристрої мають змогу відправляти та отримувати дані в основному за допомогою спеціальних протоколів передачі даних або технологій Wi-Fi, Bluetooth або GSM. В випадку з останнім використовуються SIM-карти типу IoT, котрі також мають назву M2M SIM Card (machine-to-machine). Підхід з використанням Інтернету речей для налагодження збору даних з різних пристроїв та датчиків допомагає прискорити процес обміну даними та надає змогу пристроям взаємодіяти між собою без втручання людини. Через те, що система налаштовується один раз і далі працює самостійно це може покривати великий спектр задач у різних галузях. Цей підхід потрібен для того, щоб зменшити потреби у людських ресурсах і автоматизувати взаємодію між пристроями та системою обробки даних. Така архітектура може масштабуватись із збільшенням кількості пристроїв і як правило не потребує переробки платформи обробки даних і не спричиняє великих витрат.

РОЗДІЛ 3
РОЗРОБКА МОДЕЛІ СИСТЕМИ МОНІТОРИНГУ

3.1. Вибір способу відстежування

У наш час існує як мінімум два способи відстеження розташування – станції стільникового зв'язку та супутники. Бюджетні GPS-пристрої визначають місцезнаходження лише за технологією LBS, бо власне вони не мають GPS модуля. Позиціонування за LBS зазвичай гірше за точністю, аніж GPS, тим не менш у LBS є можливість роботи у неідеальних умовах. Наприклад, його сигнали не вимагають повноцінного ясного неба і легко зловити мережі, навіть якщо пристрій знаходиться усередині будівлі або під землею [5, 6].

Для реалізації моделі системи відстежування було обрано застосовувати власний GPS-трекер. Він буде розміщуватись у транспорті та надсилати дані до системи обробки координат. Обрано GPS як спосіб визначення геолокації. Для відправки даних буде використано GSM.

3.2 Проектування трекеру

Для проектування трекеру було обрано такі елементи як Raspberry Pi 4, модуль зв'язку NB-IoT/Cat-M/EDGE/GPRS/GNSS на модулі SIM7000E (SIM7000E NB-IoT HAT).

3.2.1 Складові трекеру

Raspberry Pi 4 є одноплатним комп'ютером зручним у використанні для задач IoT, часто використовується для збору даних з датчиків.

SIM7000E NB-IoT HAT має багато комунікаційних функцій: NB-IoT, eMTC, EDGE, GPRS і GNSS. Таким чином, цей шилд добре підходить спілкування або позиціонування кількома способами.

Вигляд пристрою продемонстрований на рисунку 3.1

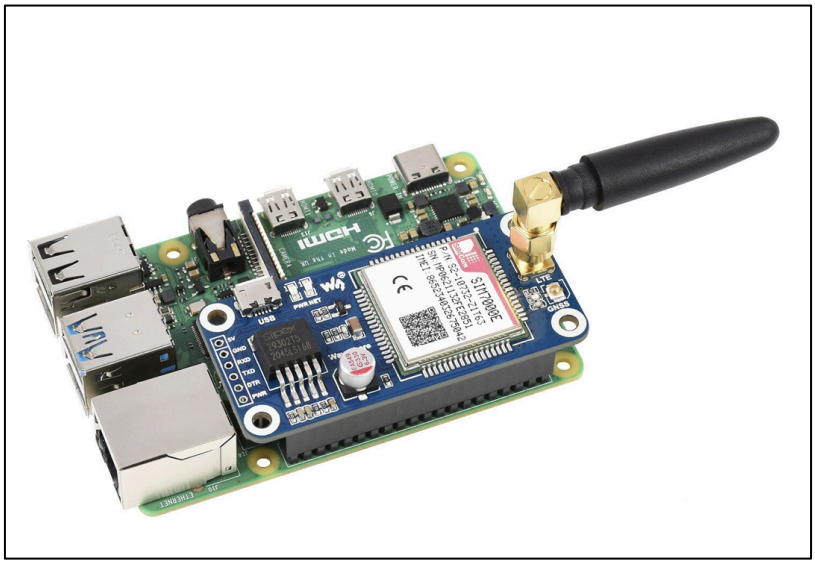


Рисунок 3.1 – Схема прототипу GPS-трекеру

3.2.2 Налаштування трекеру

Першим етапом налаштування трекеру є перевірка роботи GPS у обраному модулі зв’язку. Для того щоб перевірити здатність модулю взаємодіяти із супутниками необхідно підключити даний модуль до комп’ютера за допомогою кабелю, встановити необхідні драйвери для коректного розпізнання пристрою і ввести до модулю команди увімкнення GPS позиціонування. В даному випадку щоб вводити команди напряму до підключеного модуля зв’язку було прийняте рішення скористатись спеціальною програмою під назвою «AT Command Tester». Для підключення до AT порту в даному випадку потрібно було обрати COM5 порт комп’ютера.

Для ввімкнення геопозиціонування були введені такі команди:

- AT+CGNSPWR=1 – команда ввімкнення GNSS на модулі
- AT+CGNSTST=1 – команда для надсилання даних, отриманих від GNSS, до AT UART

Результат роботи команд показаний на рисунку 3.2.

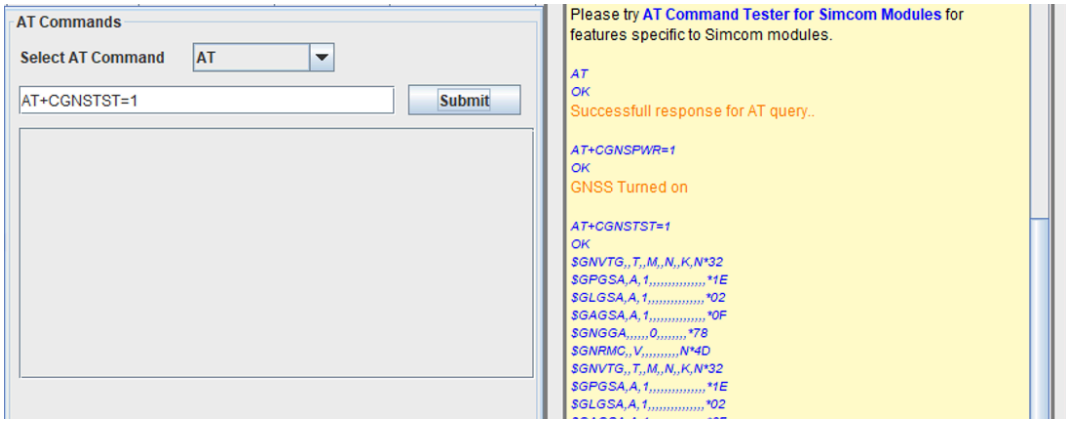


Рисунок 3.2 – Результат ввімкнення геопозиціонування

U-center – це офіційне програмне забезпечення для GNSS-приймачів від виробника U-Blox. Якщо відкрити цю програму і виставити такий самий порт як і при підключенні до модуля через «AT Command Tester» і можна побачити з якими супутниками наразі взаємодіє модуль і яку інформацію він зараз обробляє. Вигляд програми в цей момент показаний на рисунку 3.3.

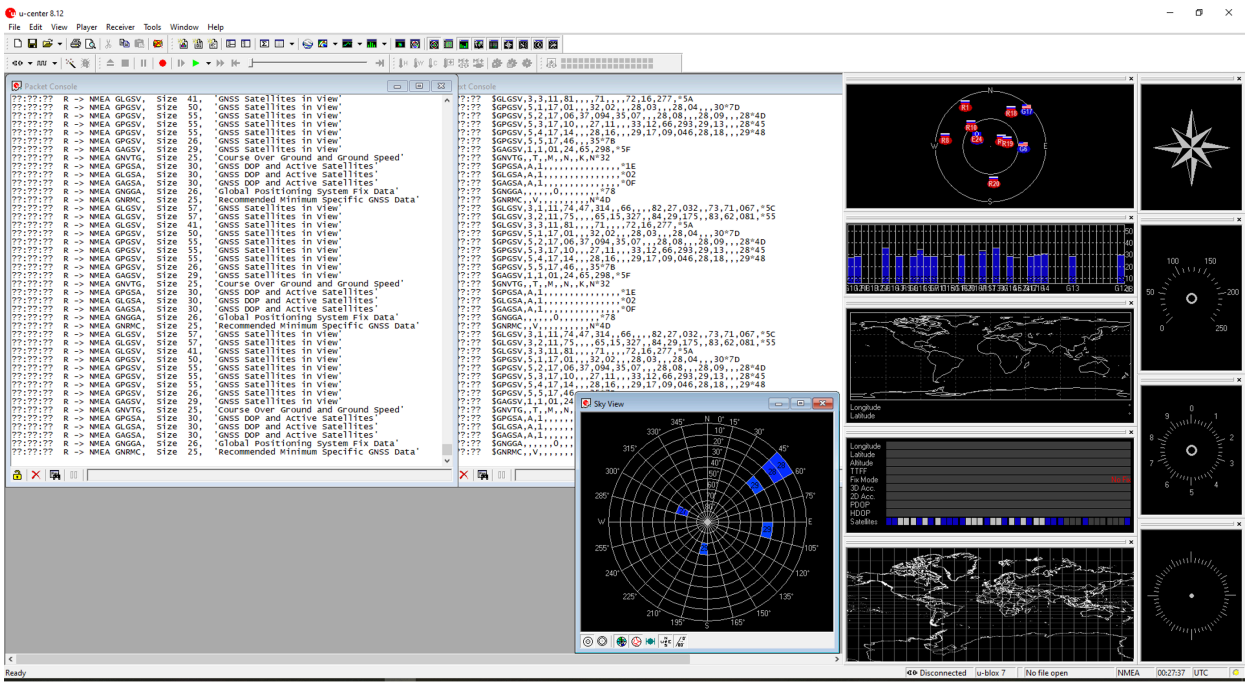


Рисунок 3.3 – Вигляд визначення модулем координат

Після двох хвилин при наявності чистого неба модуль розпізнав час, місцезнаходження та напрямок за компасом. Також у відокремленій ділянці програми можна спостерігати за тим, з якими супутниками даний модуль спілкується щоб отримати інформацію. Результат зображений на рисунку 3.4.

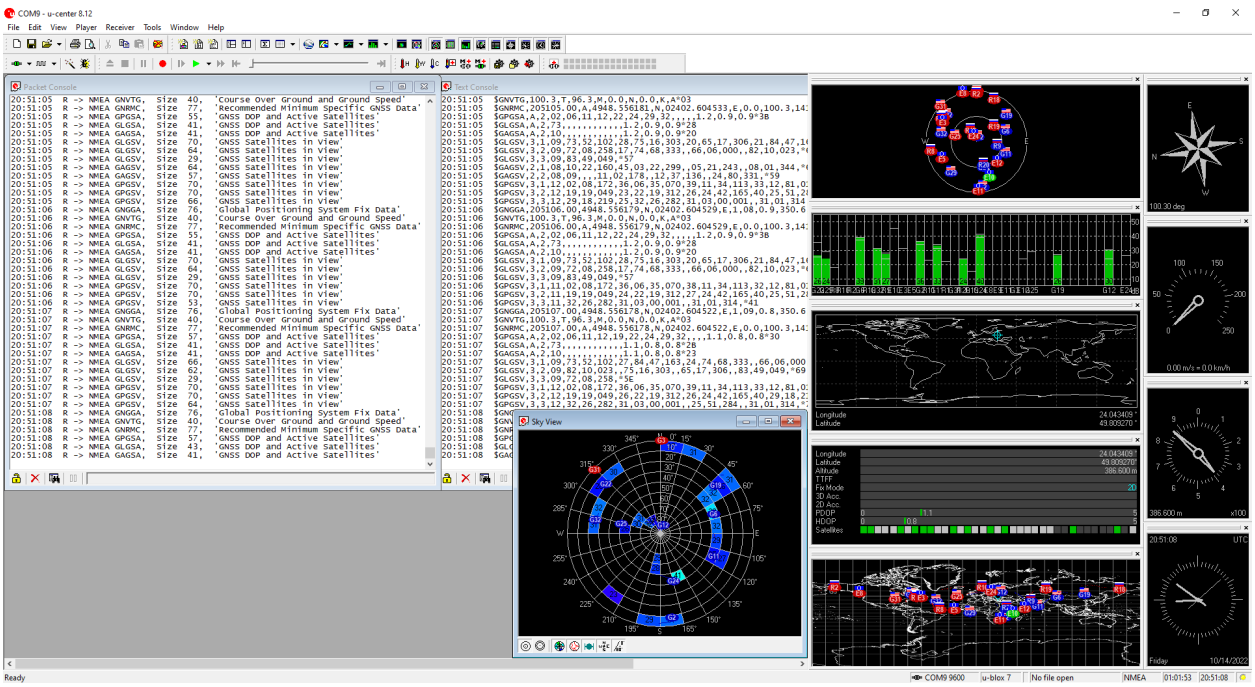


Рисунок 3.4 – Вигляд програми коли модуль отримав всі дані від супутників

Другим етапом налаштування трекеру є налаштування взаємодії з платою Raspberry Pi. На даному етапі можна одразу встановити модуль зв’язку на плату. Для налаштування взаємодії з Raspberry Pi при установці операційної системи треба задати ім’я хоста та обрати можливість підключення через SSH за допомогою логіну та паролю.

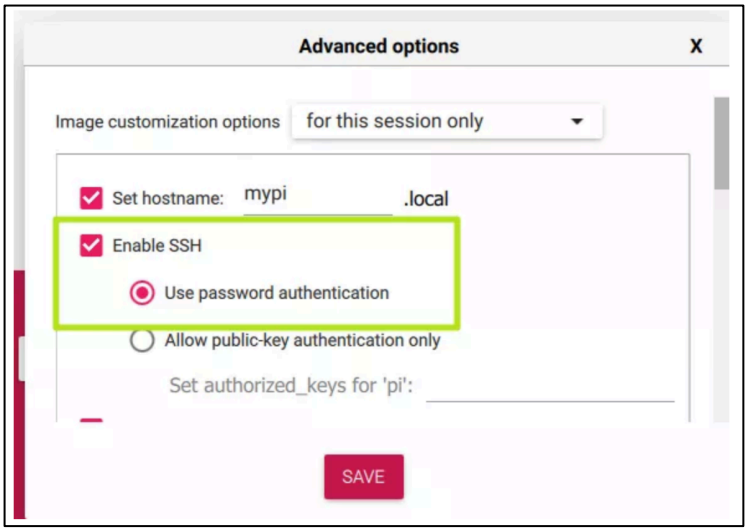


Рисунок 3.5 – Підключення hostname та SSH при налаштуванні системи

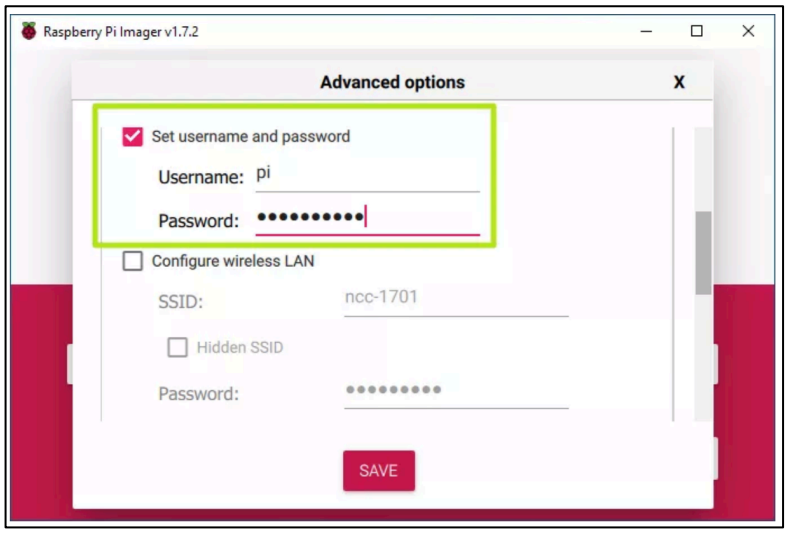


Рисунок 3.6 – Задання логіну та паролю для підключення по SSH при налаштуванні системи

Також треба встановити SSID, пароль і країну мережі Wi-Fi до якої підключений комп’ютер за допомогою котрого буде встановлене підключення до плати для її налаштування. Таким чином при старті операційної системи плата підключиться до мережі Wi-Fi. Щоб підключитись до плати треба застосувати у терміналі команди доступу за SSH.

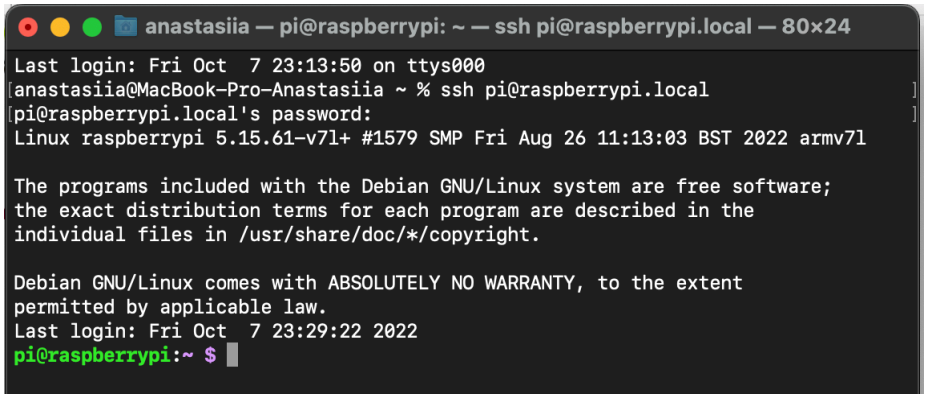


Рисунок 3.7 – Підключення по SSH до плати Raspberry Pi

Для того щоб була можливість бачити не тільки термінал а й графічний інтерфейс треба підключитись до плати за допомогою VNC. Це система віддаленого доступу до робочого столу іншого комп'ютера, що використовує протокол RFB. Для налаштування доступу за VNC треба сконфігурувати Raspberry Pi а саме за допомогою команди sudo raspi-config увійти до «Software Configuration Tool», обрати налаштування інтерфейсу (пункт під назвою Interface options) та вибрати доступ за технологією VNC.

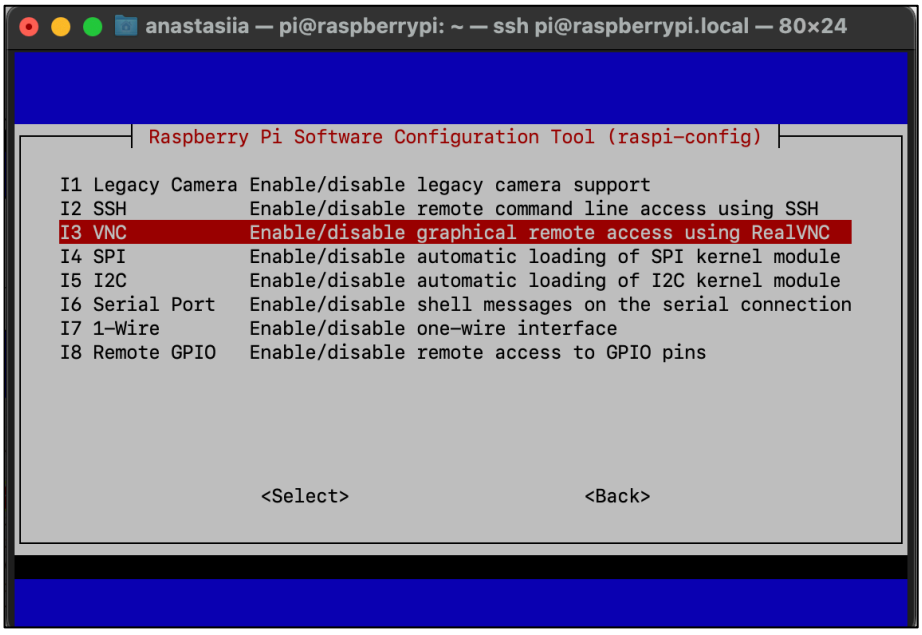


Рисунок 3.8 – Ввімкнення доступу за VNC у налаштуваннях

Щоб мати вид робочого стола Raspberry Pi через програму «VNC» треба також обрати розширення екрану у меню під назвою «Display Options». Зайти в налаштування VNC дисплею («VNC Resolution») та обрати розширення екрану розміром 1920x1080.

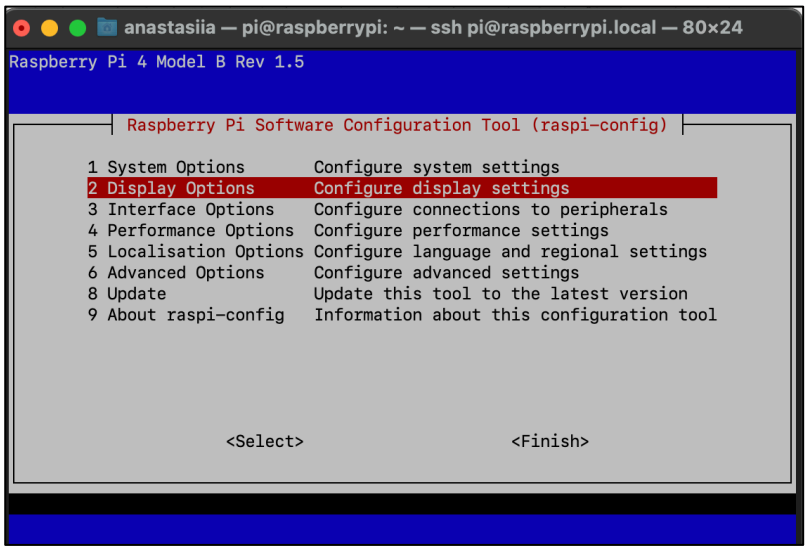


Рисунок 3.9 – Вхід до налаштування дисплею у конфігурації

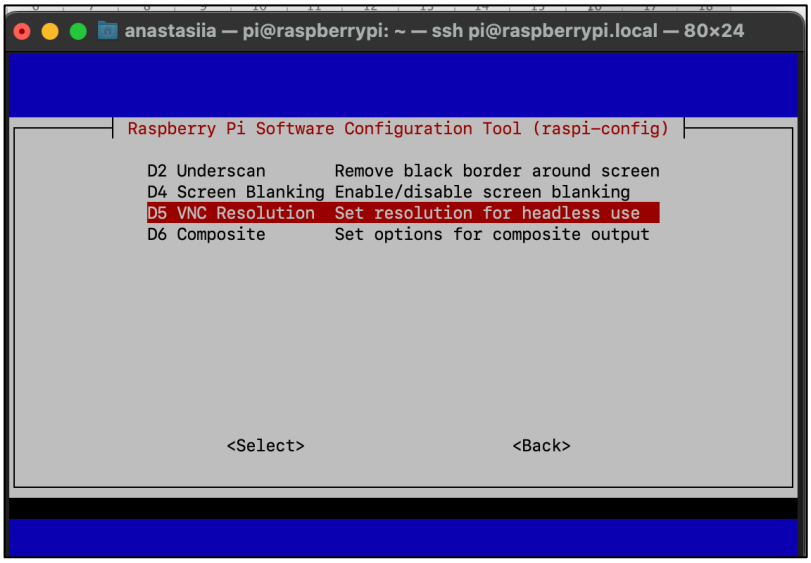


Рисунок 3.10 – Налаштування масштабу дисплею при доступі через VNC у конфігурації Raspberry Pi

Raspberry Pi містить послідовний порт UART на роз'ємі GPIO на контактах 8, TXD (GPIO 14) і 10, RXD (GPIO 15) [7].

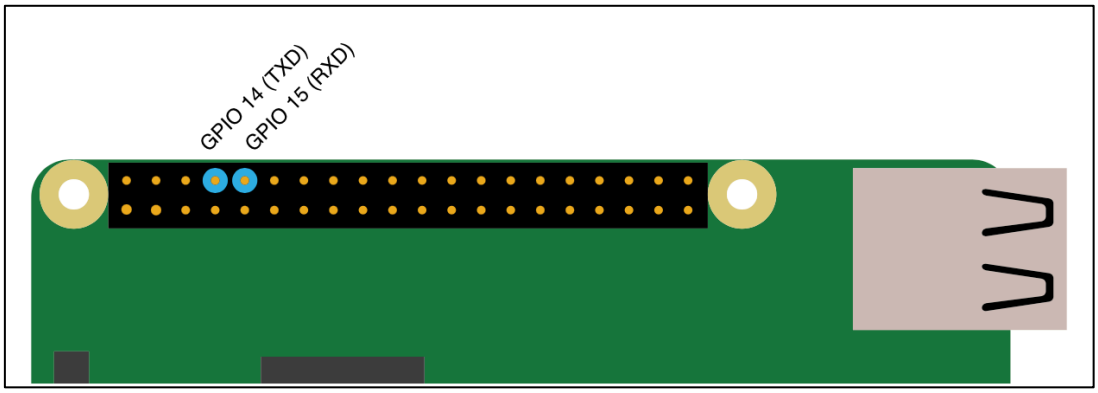


Рисунок 3.11 – Розташування TXD та RXD на роз'ємі GPIO Raspberry

Порт UART можна використовувати для прямого підключення до широкого спектру пристроїв. Оскільки він використовується для виведення консолі Linux за замовчуванням, нам потрібно змінити налаштування через те, що в нашому випадку потрібен UART для взаємодії з модулем зв'язку. Для цього треба зайти в меню «Interface Options» і далі обрати обрати «P6 Serial Port». [7]

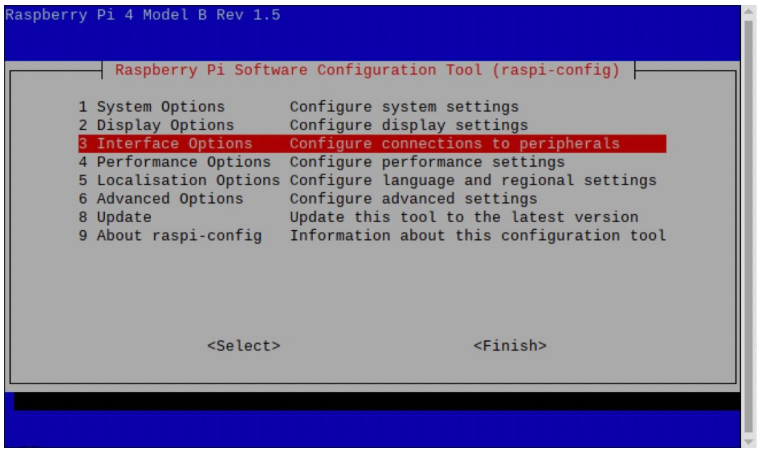


Рисунок 3.12 – Вхід до налаштування з'єднання з периферійними пристроями у конфігурації Raspberry Pi

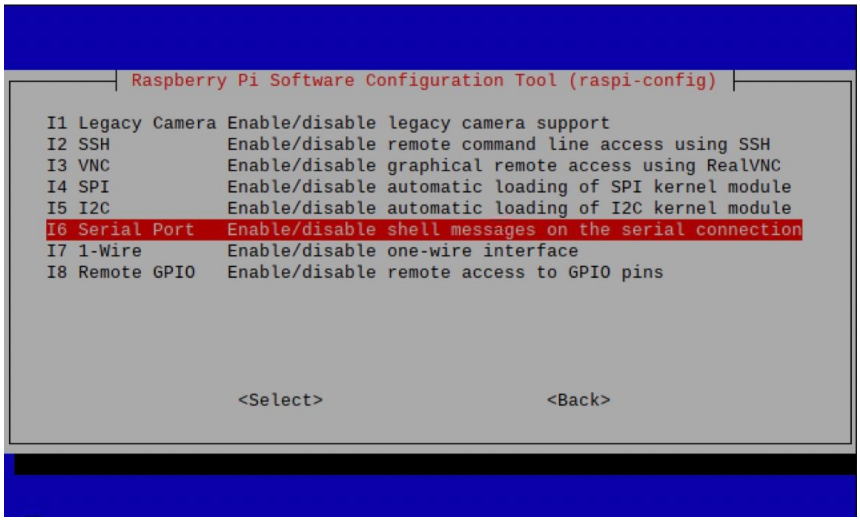


Рисунок 3.13 – Налаштування з’єднання з периферійними пристроями у конфігурації Raspberry Pi через послідовний порт

На даному етапі ми маємо налаштоване з’єднання Raspberry Pi з модулем зв’язку який буде використовуватись для відправки даних до системи обробки та саму Raspberry Pi сконфігуровану на віддалене підключення. Після зроблених налаштувань можна підключитись до пристрою за допомогою програми під назвою VNC. При першому підключенні у полі VNC Server необхідно вказати ім’я що було обране раніше для підключення по SSH на етапі налаштування операційної системи Raspberry Pi. Після цього буде запропоновано ввести логін та пароль які будуть також співпадати з такими ж параметрами для SSH з’єднання.

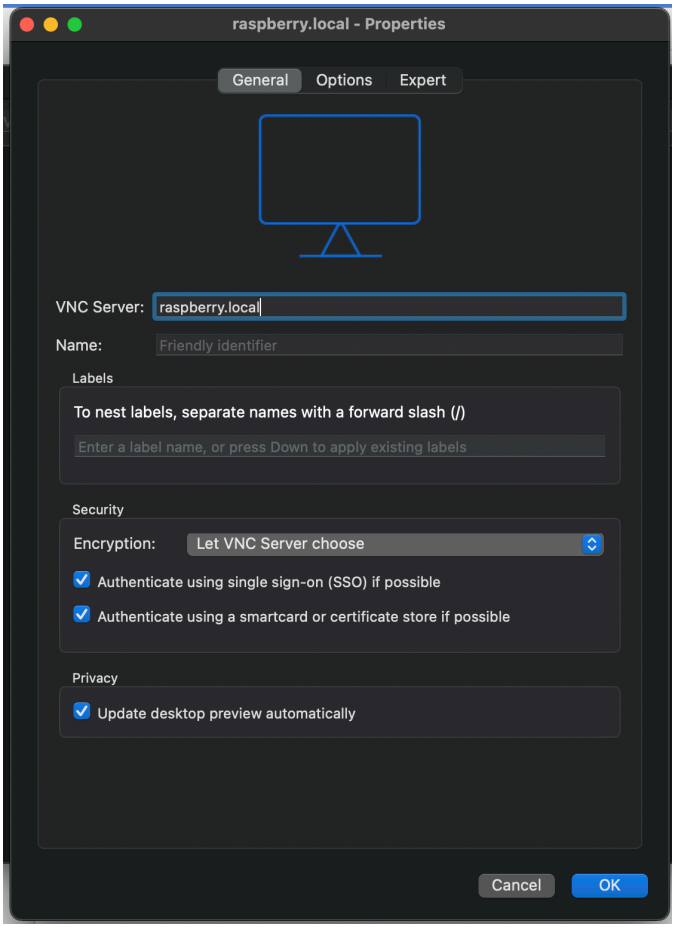


Рисунок 3.14 – Створення з’єднання через VNC до Raspberry Pi

Доступ до міні-порту UART можна отримати за допомогою /dev/ttyS0 у Linux. Треба встановити програму керування модемом і емуляції терміналу для Linux. В даному випадку на Raspberry Pi встановлюється утіліта minicom командою sudo apt-get install minicom.

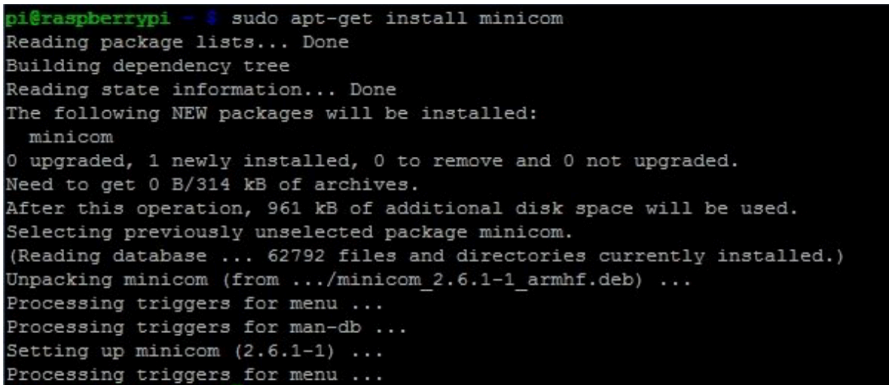


Рисунок 3.15 – Встановлення утіліти minicom

Для перевірки з’єднання з модулем зв’язку треба виконати команду minicom -D /dev/ttyS0

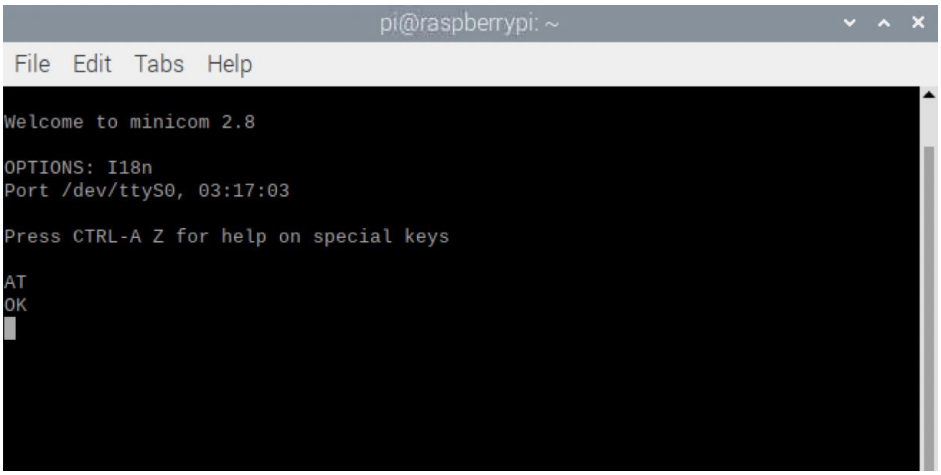


Рисунок 3.16 – Перевірка зв’язку з модулем за допомогою minicom

3.3 Проектування системи обробки

Головна частина процесу розробки – обмірковування над всією архітектурою проекту, включаючи компоненти та фізичне розташування цієї системи. Існує велика кількість способів як можна зберегти код або програму. Можна або орендувати сервер, де власне і буде розташований додаток або

скористатися хмарними рішеннями, які представляють багато готових сервісів для користування.

3.3.1 Вибір хмарної платформи

Після порівняння платформ-постачальників хмарних ресурсів для побудови нашої системи було обрано AWS (Amazon Web Services) через велику кількість сервісів не тільки для підключення пристроїв а й для подальшої їх взаємодії із побудованою на інших сервісах системою. Також ця можливість надається безкоштовно на 12 місяців користування що дуже важливо для налаштування та тестування.

Amazon залишається улюбленим постачальником хмарних технологій для багатьох компаній, тому цілком природно, що ці компанії розглядають рішення AWS IoT як логічний наступний крок у своїй цифровій трансформації. Перевагою Amazon AWS IoT є його добре структурована колекція IoT-сервісів, аналітичні служби а також продуктивність хмарної інфраструктури AWS.

Для IoT від AWS існують такі сервіси:

- AWS IoT 1-Click
- AWS IoT Analytics
- AWS IoT Button
- AWS IoT Core
- AWS IoT Device Defender
- AWS IoT Device Management
- AWS IoT Events
- AWS IoT Greengrass
- AWS IoT SiteWise
- AWS IoT Things Graph
- AWS Partner Device Catalog

- FreeRTOS

Серед найважливіших можна виділити:

- AWS IoT Core;
- AWS IoT Device Defender;
- AWS IoT Device Management;
- AWS IoT Events.
- AWS IoT Greengrass

AWS IoT Core – хмарний сервіс, що дозволяє легко і безпечно взаємодіяти з хмарними додатками та іншими пристроями. Цей сервіс може підтримувати мільярди пристроїв, а також може обробляти і маршрутизувати ці повідомлення до інших пристроїв чи ендпоінтів надійно і безпечно. AWS IoT Core полегшує використання сервісів AWS через те, що має можливість створювати спеціальні правила маршрутизації повідомлень з конкретних пристроїв або тем прямо до інших сервісів.

AWS IoT Device Defender – це сервіс, що допомагає забезпечити безпеку пристроїв IoT. Він постійно проводить аудит IoT-конфігурацій, щоб переконатися, що вони не відхиляються від найкращих практик безпеки. Сервіс надсилає попередження, якщо у налаштуваннях IoT є якісь прогалини, які можуть створити ризик безпеки, наприклад, сертифікати, що діляться на декілька пристроїв або пристрій із відкликаним сертифікатом ідентифікації, який намагається з'єднатися з AWS IoT Core. Також він дозволяє безперервно стежити за метриками безпеки пристроїв. Якщо щось не виглядає правильно, AWS IoT Device Defender надсилає попередження, щоб можна було швидко вжити заходів для усунення проблеми.

AWS IoT Device Management дозволяє легко організовувати, стежити та віддалено керувати пристроями IoT. За допомогою системи керування пристроями AWS IoT можна реєструвати під'єднані пристрої окремо або

групою, а також легко керувати дозволами, щоб пристрої залишалися захищеними. Також можна надсилати оновлення вбудованого програмного забезпечення завдяки цьому сервісу.

AWS IoT Events – це сервіс, що полегшує виявлення та реагування на події з IoT-пристроїв. Використовуючи цей сервіс можна легко виявити події через тисячі датчиків IoT, що посилають різні телеметричні дані, такі як температура з морозильної камери, вологість від дихального обладнання, а також швидкість ременя на двигуні, і сотні інших застосувань управління обладнанням. Треба обрати відповідні джерела даних, які слід задати, визначити логіку для кожної події за допомогою простих висловлювань і вибрати попередження або нетипову дію, щоб викликати цю подію. Цей сервіс автоматично запустить сповіщення та дії у відповідь на події, засновані на логіці, яку було описано. Це допомагає швидко вирішувати проблеми, зменшувати витрати на технічне обслуговування та підвищити ефективність експлуатації системи [8].

AWS IoT дозволяє вибрати найбільш відповідні та сучасні технології для вашого рішення [9]. Щоб допомогти вам керувати своїми пристроями IoT і підтримувати їх у польових умовах, AWS IoT Core підтримує такі протоколи:

- MQTT (Message Queuing and Telemetry Transport)
- MQTT over WSS (Websockets Secure)
- HTTPS (Hypertext Transfer Protocol - Secure)
- LoRaWAN (Long Range Wide Area Network)

AWS IoT Greengrass легко поширює AWS на пристрої, щоб вони могли локально працювати з даними, які вони генерують, водночас використовуючи хмару для керування, аналітики та тривалого зберігання. Завдяки AWS IoT Greengrass підключені пристрої можуть запускати функції AWS Lambda, прогнозувати на основі моделей машинного навчання, підтримувати

синхронізацію даних пристрою та безпечно спілкуватися з іншими пристроями, навіть якщо вони не підключені до Інтернету. AWS IoT Greengrass можна запрограмувати на фільтрацію даних пристрою та передачу лише необхідної інформації назад у хмару.

3.3.2 Вибір необхідних сервісів

AWS є великою платформою для розробки яка має багато готових до застосування сервісів, включаючи бази даних, віртуальні мережі, хмарні сховища, сервіси для Інтернету речей тощо. Для побудови системи обробки місцезнаходження транспорту необхідно мати одну адресу доступу до системи обробки (для веб-сайту), налаштовану переадресацію даних з трекеру на систему обробки, та саме обробники запитів, що будуть спрацьовувати на отримання даних, аналізувати отримані їх, база даних, для збереження місцезнаходження.

Для побудови системи обробки було використано такі сервіси AWS:

- API Gateway
- AWS IoT
- AWS Lambda
- S3
- DynamoDB
- CloudFormation

AWS API Gateway – це сервіс, за допомогою якого розробники налаштовують домени, створюють кінцеві точки для своїх додатків та з’єднують їх з певними ресурсами AWS для подальшої обробки, механізми авторизації, кешування та інші фічі. API Gateway – це основна частина безсерверного API, тому що відповідає за зв'язок між певним API та функцією, що обробляє запити до нього. Робота з API Gateway не потребує мінімальних платежів або стартових вкладень. Користувач сплачує за тільки отримані

визови API та переданий обсяг даних і таким чином за допомогою моделі ціноутворення API Gateway можна знизити витрати в міру масштабування використання додатка.

AWS IoT надає програмне забезпечення для пристроїв, яке може допомогти інтегрувати IoT-пристрої в рішення на основі AWS IoT. Якщо пристрої можуть підключатися до AWS IoT, цей сервіс може підключати їх до інших хмарних сервісів, які надає AWS. Брокер повідомлень AWS IoT Core підтримує пристрої та клієнти, які використовують протоколи MQTT і MQTT через WSS для публікації та підписки на повідомлення. Загалом MQTT – це протокол для передачі даних між пристроями, він вміє працювати в обох напрямах. Вміє як посилати повідомлення до хмарних сервісів, так і отримувати. Він працює за схемою підписник та той, хто публікує. Для того, щоб отримувати підписнику данні, необхідно зробити підписку на певну тему, інакше данні не можливо отримати.

Брокер AWS IoT Core також підтримує пристрої та клієнти, які використовують протокол HTTPS для публікації повідомлень. Приклад ролі AWS IoT зображений на рисунку 3.17.

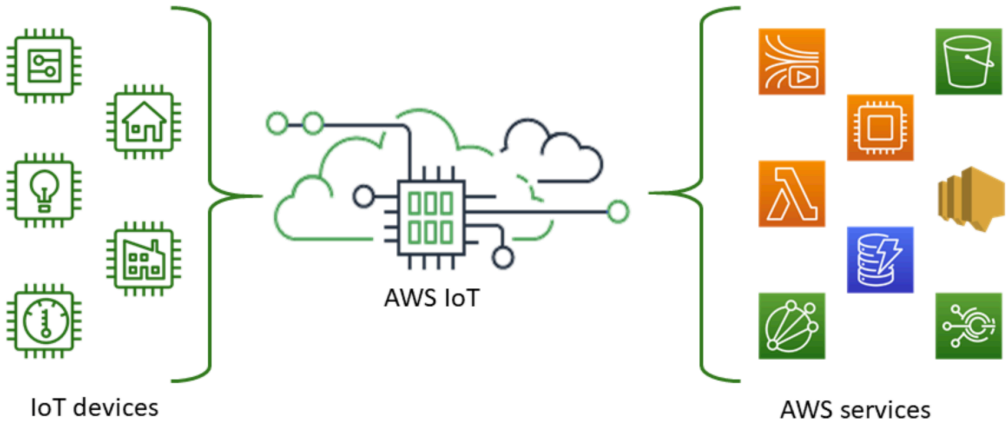


Рисунок 3.17 – Зображення головної функції сервісу AWS IoT

AWS Lambda – це сервіс безсерверних обчислень, що запускає програмний код у відповідь на певні події та відповідає за автоматичне виділення необхідних обчислювальних ресурсів для його виконання. Просто треба завантажити код у ZIP-архіві до S3 бакету, вказати сервісу шлях до нього і далі сервіс автоматично виділить обчислювальну потужність і запустить його. AWS Lambda дозволяє додавати свій код до ресурсів AWS, наприклад до бакетів сервісу Amazon S3 та таблиць Amazon DynamoDB. В результаті можна просто виконувати необхідні операції з даними на етапі їх входження або переміщення в рамках хмари.

Amazon Simple Storage Service (Amazon S3) – це масштабована, високошвидкісна веб-служба хмарного зберігання. Сервіс предоставляє можливості для онлайн-резервного копіювання та архівування даних і програм які базуються на Amazon Web Services. Amazon S3 розроблено з мінімальним набором функцій і створено, щоб спростити розробникам обчислення в масштабі Інтернету. Цей сервіс для зберігання об'єктів, що пропонує найкращі в галузі показники продуктивності, масштабованості, доступності та безпеки даних. Клієнти будь-якої величини та з будь-якої промислової галузі можуть зберігати та захищати необхідний обсяг даних. Серед клієнтів, що використовують цей сервіс є такі компанії як Ryanair, Zalando, Nasdaq та Georgia-Pacific.

Amazon S3 можна використовувати для розміщення статичного веб-сайту. На статичному веб-сайті окремі веб-сторінки містять статичний вміст. Цей сервіс має функцію розмежування доступу для того щоб зробити сайт доступним для конкретних користувачів. Для веб-сайту який розгорнутий таким чином можна задати власний домен підключаючи ще однин сервіс AWS під назвою CloudFront.

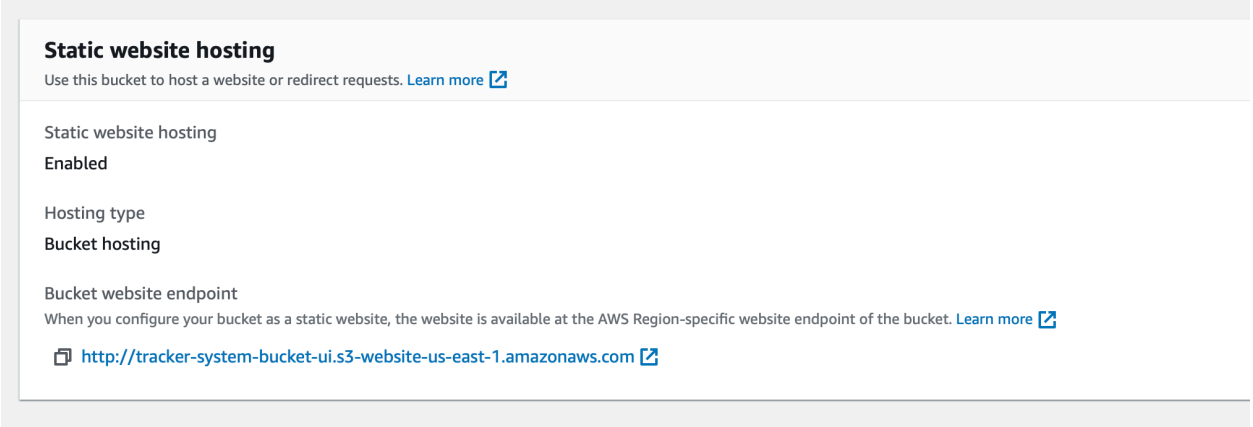


Рисунок 3.18 – Приклад включеного хостінгу на основі S3

CloudFormation – це сервіс що надає спільну мову для опису та надання всіх інфраструктурних ресурсів у хмарному середовищі AWS. Він пропонує простий спосіб моделювання необхідних ресурсів AWS, їх швидкого та одноманітного виділення, а також управління ними протягом усього життєвого циклу за рахунок роботи з інфраструктурою як із кодом. Схема принципу роботи сервісу зображена на рисунку 3.19.

Сервіс CloudFormation вводить чотири поняття:

- Шаблон (Template) – це декларативний файл коду JSON або YAML, який описує належний стан ресурсів, необхідних для розгортання програми.
- Стек (Stack) реалізує групу ресурсів, перелічених у шаблоні, та керує ним, а також дозволяє керувати станом та залежностями цих ресурсів спільно.
- Набір змін (Change Set) – це ознайомлювальна версія змін, які будуть виконані операціями зі стеком для створення, оновлення або видалення ресурсів.
- Набір стеків (Stack Set) – це група стеків із спільним керуванням, які можна реплікувати як групу.

Рисунок 3.19 – Схема принципу роботи сервісу CloudFormation

DynamoDB – це розміщена в хмарі база даних NoSQL, яку пропонує AWS. Це повністю керована безсерверна база даних на основі пар «ключ-значення», створена для запуску високопродуктивних програм у будь-якому масштабі. Сервіс пропонує вбудований захист, можливість безперервного резервного копіювання, автоматичної реплікації в кількох географічних регіонах та інструменти імпорту та експорту даних. Корпорації, такі як Zoom, Snap, Dropbox і Disney, використовують сервіс DynamoDB для виконання своїх робочих навантажень.

У нашому випадку буде використано одну таблицю «coordinates-table» яка є сховищем для отриманих координат транспорту.

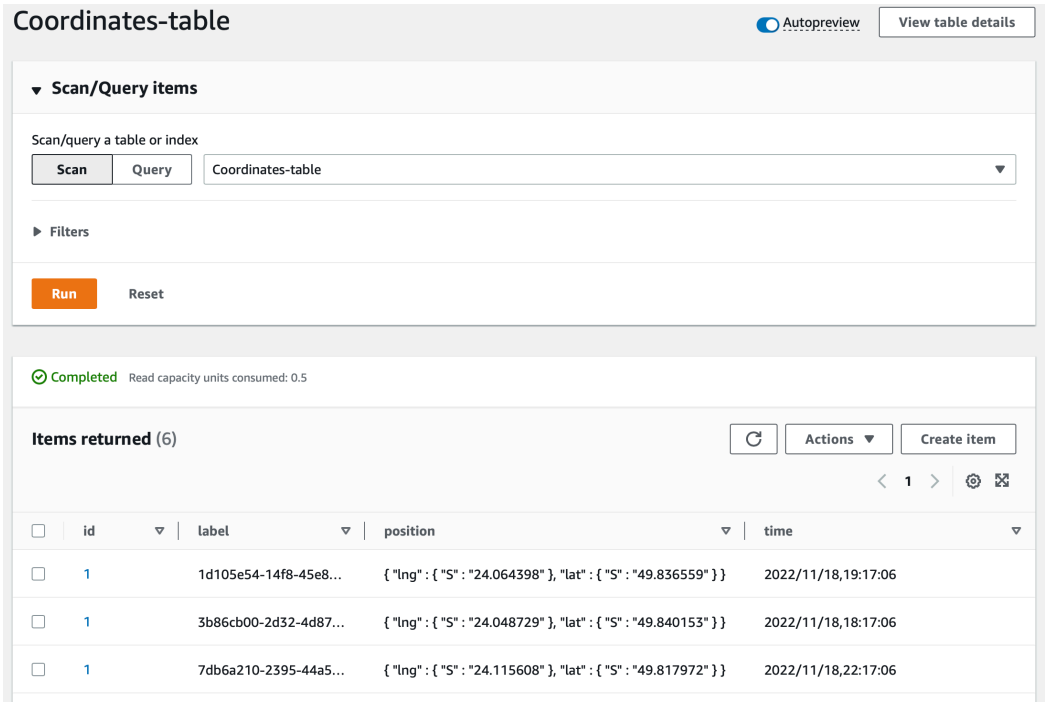


Рисунок 3.20 – Вигляд бази даних

Для основних дій пов’язаних з системою обробки було побудовано дві лямбди на мові програмування Python. Перша лямбда служить для обробки даних, надісланих до AWS IoT, друга зчитує дані з бази даних та передає до веб-сайту.

Лямбда, що є обробником для даних, отриманих з трекеру розбирає отримані параметри, що складаються з координат місцезнаходження транспорту (широти та довжини), номеру машини та часу (рисунок 3.21) та зберігає їх у таблицю «coordinates-table».

```
{
   "id":"1",
   "position":{
      "lat":"49.817972",
      "lng":"24.115608"
   },
   "time":"2022/11/18,19:17:06"
}
```


Рисунок 3.21 – Вигляд отриманих даних від трекеру

Друга лямбда виступає коннектором для веб-сайту та системи обробки, вона закріплена за API Gateway «TrackingSystemApi» та приймає запити від веб-сайту і виходячи з них дістає з таблиці бази даних історію руху транспорту. Завдяки цій лямбді на сайті відображається маршрут і поточне положення.

Загальна архітектура прототипу системи відстежування зображена на рисунку 3.22s.

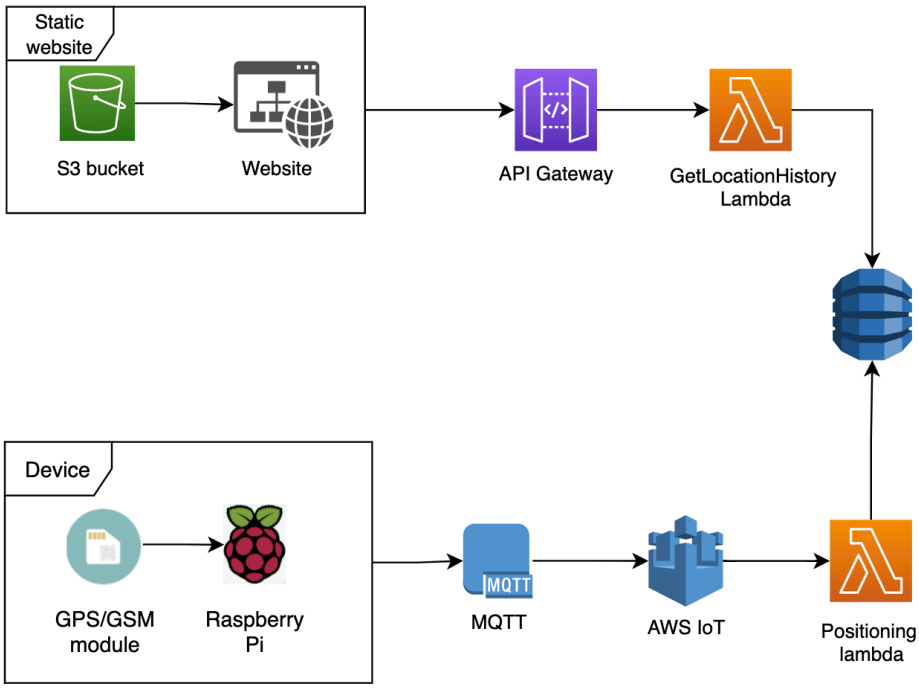


Рисунок 3.22 – Архітектура системи відстежування

3.3.3 Розгортання системи

Сервіси, обрані для побудови архітектури системи обробки можна як створювати вручну в консолі AWS так і розгорнути їх за допомогою такого сервіса AWS як CloudFormation. Для цього вся архітектура повинна бути написана в одному файлі формату YAML або JSON. Цей сервіс дозволяє розгорнути всю написану там інфраструктуру в Amazon за пару хвилин. Він створює такий об’єкт як стек до якого будуть прив’язані всі сервіси, це поліпшує їх пошук серед багатьох різних сервісів та спрощує подальшу взаємодію з ними таку як оновлення, додавання інших сервісів у додаток чи їх видалення.

Для того щоб використовувати CloudFormation потрібно встановити на комп’ютер утіліту AWS CLI яка є уніфікованим інструментом для керування

сервісами AWS. Далі треба створити користувача у консолі AWS в аккаунті. Даний користувач буде напряму брати участь у розгортанні системи обробки, тому можна дати йому ім’я адміністратора.

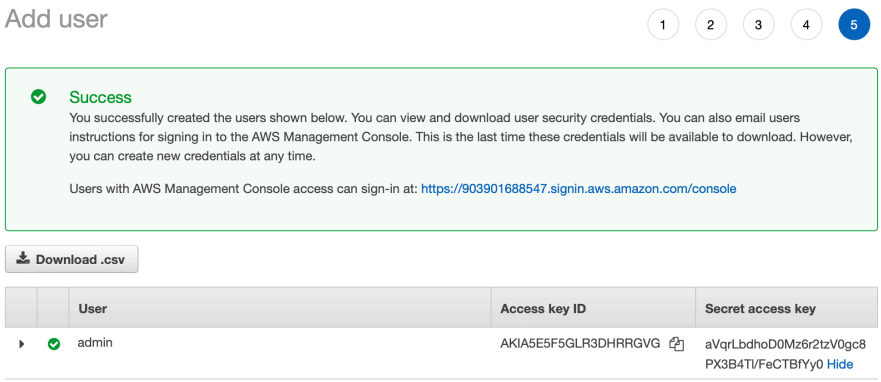


Рисунок 3.23 – Створення користувача admin для проведення розгортання інфраструктури через AWS CLI

Після отримання облікових даних для користувача, необхідно сконфігурувати AWS CLI на використання цього профілю.

```
anastasiia@MacBook-Pro-Anastasiia tracker-system % aws configure --profile admin
AWS Access Key ID [None]: AKIA5E5F5GLR3DHRRGVG
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-1
Default output format [None]: json
```

Рисунок 3.24 – Створення профілю користувача admin у AWS CLI

Маючи налаштований профіль для взаємодії з сервісами можна створювати інфраструктуру системи.

Лямбди, що використовуються в поточній системі написані з використанням мови програмування Python.

При створенні архітектури сервісів за допомогою CloudFormation, лямбди зазвичай беруть код з вже існуючого сховища в Amazon. В якості такого сховища в даному випадку буде виступати сервіс S3 в якому будуть зберігатись архіви з кодом. Назва архівів буде співпадати з назвою лямбд.

Створимо бакет під назвою tracker-system-bucket-code в регіоні us-east-1 та завантажимо туди два архіви з кодом командами що показані на рисунку 3.25.

```
anastasiia@MacBook-Pro-Anastasiia cloudformation % aws --profile admin s3api create-bucket --bucket tracker-system-bucket-code --region us-east-1 --acl private
{
    "Location": "/tracker-system-bucket-code"
}
anastasiia@MacBook-Pro-Anastasiia cloudformation % cd ../lambdas/PositioningLambda
anastasiia@MacBook-Pro-Anastasiia PositioningLambda % aws --profile admin s3api put-object --bucket tracker-system-bucket-code --key PositioningLambda.zip --region us-east-1 --body ./PositioningLambda.zip
{
    "ETag": "\"308694383fc8ba9c953698e032f0f5a0\""
}
(venv) anastasiia@MacBook-Pro-Anastasiia GetLocationHistoryLambda % aws --profile admin s3api put-object --bucket tracker-system-bucket-code --key GetLocationHistoryLambda.zip --region us-east-1 --body ./GetLocationHistoryLambda.zip
{
    "ETag": "\"308694383fc8ba9c953698e032f0f5a0\""
}
```

Рисунок 3.25 – Створення S3 бакету та завантаження туди двох архівів з кодом лямбд

Далі необхідно створити стек додатку, тобто описати всі сервіси що будуть використовуватись у одному файлі. CloudFormation як було позначено раніше підтримує файли у форматах YAML або JSON. Останній є більш приємним для сприйняття людиною, ніж JSON, тому саме такий формат часто використовують для опису сценаріїв програмного забезпечення.

Для взаємодії з кінцевим користувачем було вирішено побудувати веб-сайт у якому буде зображений маршрут транспорту. Для цього буде використаний сервіс Google Maps. Для його використання необхідно створити проект у сервісі Google Cloud та створити API ключ який відноситься до нього.

Для використання ключа тільки у додатках пов’язаних із JavaScript до нього було додано обмеження під назвою Maps JavaScript API.

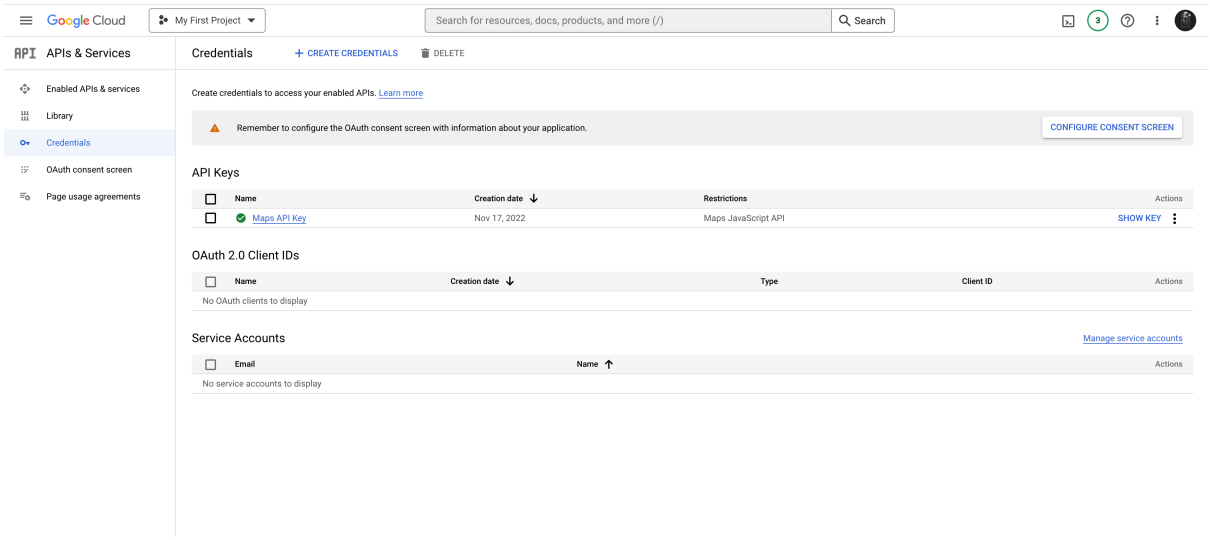


Рисунок 3.26 – Створення API ключа для використання Google Maps

Для того щоб у користувача був доступ до розробленого веб-сайту було прийняте рішення застосувати функцію S3 бакету під назвою static website hosting, налаштувати права доступу.

Рисунок 3.27 – Налаштування веб-сайту у файлі з описом інфраструктури

Створений файл інфраструктури містить в собі опис розгортання усіх потрібних сервісів, налаштування прав доступу між ними та створення бази даних.

При розгортанні стеку необхідно вказати опцію --capabilities CAPABILITY_IAM через те що в файлі інфраструктури описано створення ролей та полісі і тому треба вказати сервісу CloudFormation що так і потрібно. Розгортання системи обробки зображено на рисунку

```
anastasiia@MacBook-Pro-Anastasiia cloudformation % aws --profile admin cloudformation deploy --template-file tracker-syste
m-template.yaml --stack-name tracker-stack --region us-east-1 --capabilities CAPABILITY_IAM

Waiting for changeset to be created..
Waiting for stack create/update to complete
Successfully created/updated stack - tracker-stack
```

Рисунок 3.28 – Розгортання інфраструктури системи обробки в AWS за допомогою AWS CLI та CloudFormation

Як результат розгортання ми можемо зайти в консоль AWS, перейти до сервісу CloudFormation та побачити створений стек з усіма ресурсами.

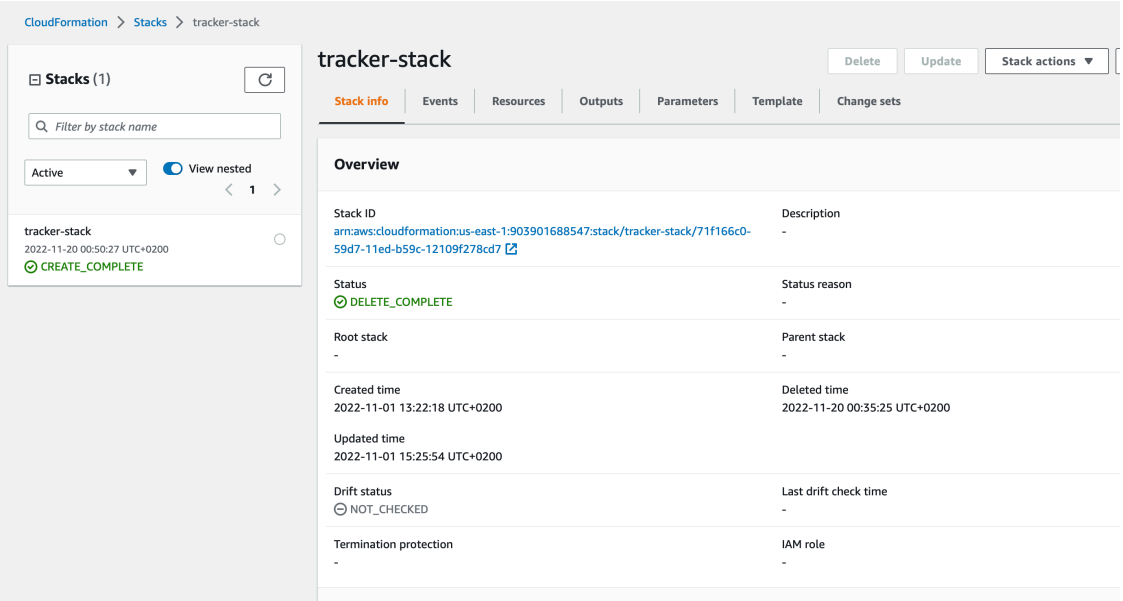


Рисунок 3.29 – Результат розгортання стеку

Після створення стеку системи обробки ми можемо побачити два створених бакети у сервісі S3 як зображено на рисунку 3.21. Один бакет був створений раніше для зберігання коду лямбд, інший створився під час розгортання стеку, для того щоб в нього завантажити веб-сайт для взаємодії користувача з системою.

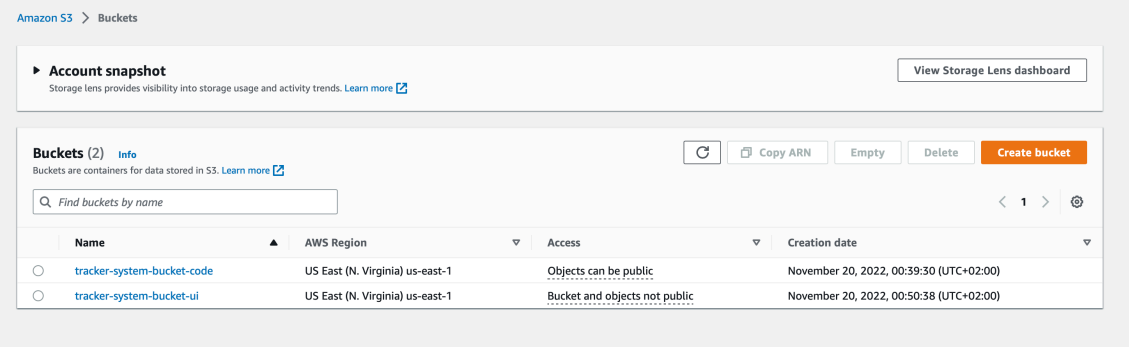


Рисунок 3.30 – Два бакети у сервісі S3, необхідні для роботи системи

Для того щоб розмістити розроблений веб-сайт у бакеті необхідно скопіювати файли веб-сайту до нього командою AWS CLI що показана на рисунку 3.31.

```
(venv) anastasiia@MacBook-Pro-Anastasiia tracker-system % aws --profile admin s3 cp dist/tracker-system s3://tracker-system-bucket-ui/ --recursive
upload: dist/tracker-system/styles.ef46db3751d8e999.css to s3://tracker-system-bucket-ui/styles.ef46db3751d8e999.css
upload: dist/tracker-system/index.html to s3://tracker-system-bucket-ui/index.html
upload: dist/tracker-system/favicon.ico to s3://tracker-system-bucket-ui/favicon.ico
upload: dist/tracker-system/runtime.2518ed19d3452aab.js to s3://tracker-system-bucket-ui/runtime.2518ed19d3452aab.js
upload: dist/tracker-system/3rdpartylicenses.txt to s3://tracker-system-bucket-ui/3rdpartylicenses.txt
upload: dist/tracker-system/polyfills.d97102cdd7f12880.js to s3://tracker-system-bucket-ui/polyfills.d97102cdd7f12880.js
upload: dist/tracker-system/main.960a9e97c3ede9c5.js to s3://tracker-system-bucket-ui/main.960a9e97c3ede9c5.js
```

Рисунок 3.31 – Перенесення веб-сайту у бакет

Скопійовані файли у бакеті після виконання цієї команди будуть виглядати як показано на рисунку 3.32.

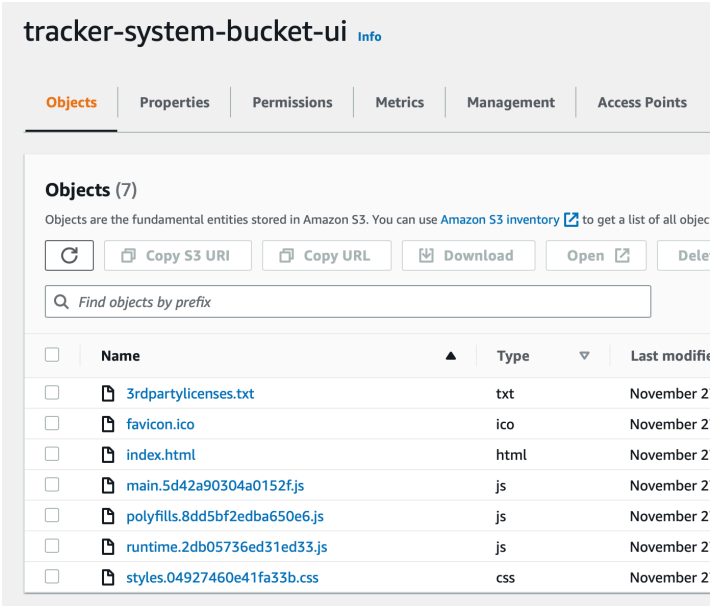


Рисунок 3.32 – Вигляд сайту розташованого в S3

Таким чином за посиланням, що вказане у S3, користувач, якому був виданий доступ до веб-сайту, зможе його побачити.

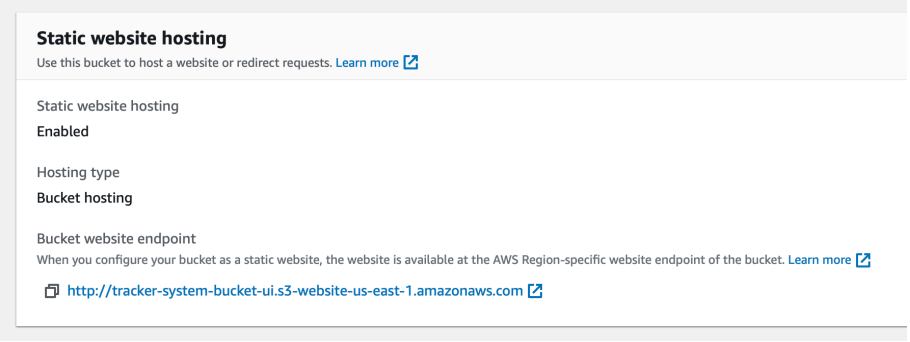


Рисунок 3.33 – Посилання на веб-сайт, що знаходиться у S3 бакеті

3.4 З’єднання системи обробки з трекером

Для того щоб налаштувати відправку даних з трекеру до системи обробки буде використовуватись сервіс AWS IoT та протокол передачі даних MQTT. Для цього створимо пристрій у AWS IoT та завантажимо сертифікат пристрою, приватний ключ та кореневий сертифікат.

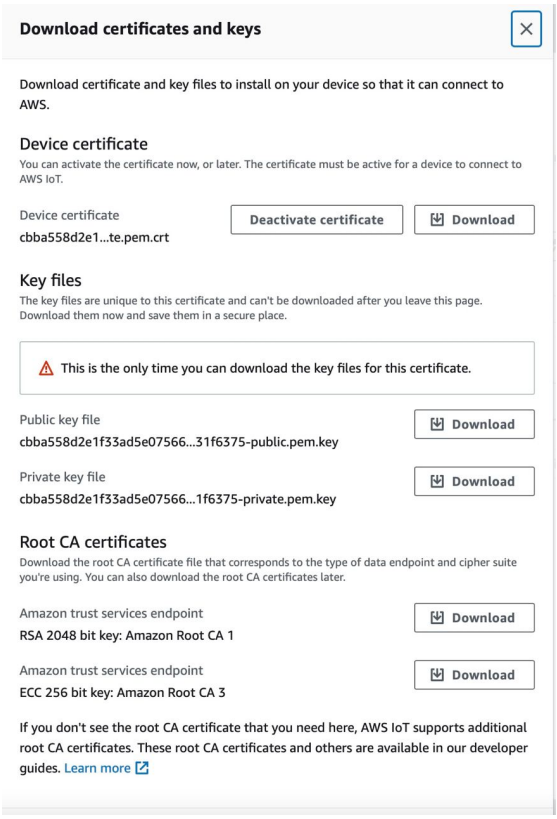


Рисунок 3.34 – Згенеровані сертифікати для доступа пристрою в AWS

Для того щоб перенести запропоновані сертифікати до модулю зв’язку можна скористатися двома способами.

У першому способі можна завантажити сертифікати через створення файлів сертифікатів у модулі за допомогою AT команд [10]. Необхідно ввести наступні команди:

1. AT+CFSINIT
2. AT+CFSWFILE
3. AT+CFSTERM

Перша команда проведе ініціалізацію файлової системи модуля. Далі необхідно завантажити необхідні сертифікати за допомогою другої команди. Повний її синтаксис виглядає наступним чином: AT+CFSWFILE=<index>,<filename>,<mode>,<filesize>,<input time>

Index – у цьому параметрі необхідно буде вказати директорію з можливими значеннями 0, 1, 2, 3. Відповідно із переліченими номерами індекса позначаються наступні директорії custapp, fota, datatx, customer. Для нашого завдання необхдно саме 3 індекс – customer.

Filename – у цьому параметрі задається ім’я файлу, але є валідація за його довжиною, не більше ніш 50 символів.

Mode – у цьому параметрі встановлюється властивість запису файлу або на початок файлу, або у кінець файлу. Тому є можливість встановити параметри 0, 1. Якщо встановлюється параметр 0, то при існуванні файлу нові дані будуть записані на початок файлу. При встановленні параметру 1, за умови існування файлу, дані будуть записані у кінець файлу.

Filesize – цей параметр визначає кількість байтів у файлу. Це значення не може бути більшим, аніж 10240 байтів.

Input time – параметр визначається у мілі секундах, значення не повинно перевищувати 10000 мс, у випадку, якщо встановлене значення вичерпується завантаження файла у файлову систему переривається з таймаут помилкою.

У випадку завантаження файлу-сертифікату на наш модуль це буде виглядати наступним чином:

```
AT+CFSINIT
OK
AT+CFSWFILE=3,"isrgrootx2.crt",0,543,10000
DOWNLOAD

OK
AT+CFSRFILE=3,"isrgrootx2.crt",0,2000,0
+CFSRFILE: 543
ItgKbppbd9/w+kHsOdx1ymgHDB/qo0IwQDAOBgNVHQ8BAf8EBAMCAQYwDwYDAlF9

OK
AT+CFSTERM
OK
```

Рисунок 3.35 – Перенесення сертифікату у модуль за допомогою АТ-команд

Другим способом перенесення сертифікатів у модуль зв'язку є застосування програми EFS Explorer. Ця програма є файловим менеджером файлової системи телефонів або модулей зв'язку. Для того, щоб побачити, що міститься у файловій системі нашого модуля необхідно при запуску застосунку обрати необхідний модуль, а далі обрати позначку із зеленим циліндром вказану на рисунку 3.25 що означає переключення на інший вид:

Рисунок 3.36 – Значок переключення файлової системи модулю

Після виконаних дій ми побачимо зміст файлової системи модуля. Для того, щоб побачити розташування сертифікатів у файловій системі модуля, необхідно перейти до директорії «customer»:

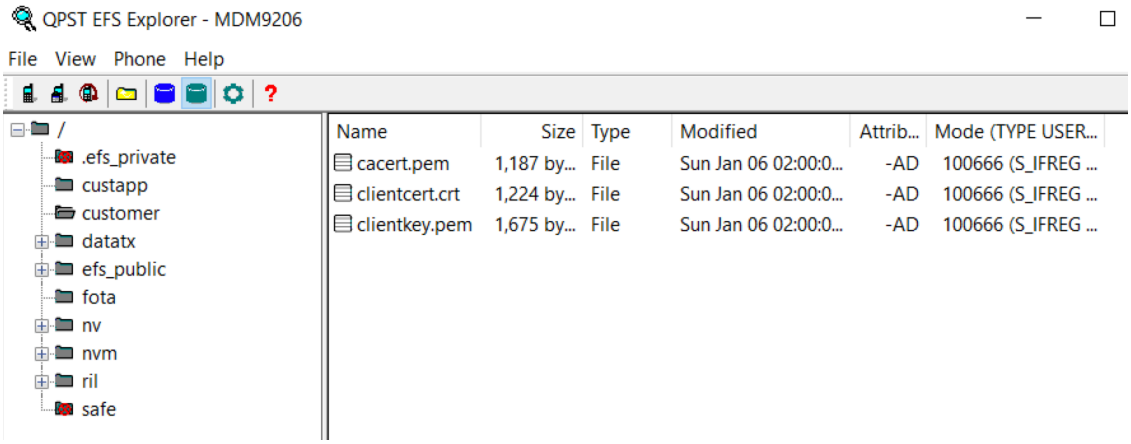


Рисунок 3.37 – Розташування сертифікатів у файловій системі модуля

Для того, щоб відправляти дані з трекеру до хмарної платформи необхідно за допомогою мови програмування Python вводити в модуль зв'язку перелік АТ-команд, зображених на рисунку 3.38:

1. AT+CREG = 1

   За допомогою цієї команди дозволяється модулю під’єднатися до мережі.

2. AT+COPS=4,2,”25503”

   За допомоги цієї команди виконується реєстрація модуля у мережу GSM/UMTS/EPS оператора Київстар. Для того, щоб дізнатися код оператора для рєєстрації, необхідно виконати команду “AT+COPS=?”. Вона поверне список із доступними у вашій локації операторами.

3. AT+CMNB=3

   Встановлення вибору зв’язку між CAT-M і NB-IOT. Крім цього є можливість вибору окремо або CAT-M, індекс 1, або NB-IOT, індекс 2.

4. AT+CNACT=1,”cmnet”

   Відкриваємо безпровідне з’єднання вказуючи це першим параметром, а другим необхідно вказати APN. Другий параметр відповідає за вибір GGSN або сторонніх пакетів данних мережі.

5. AT+SAPBR=3,1,”Contype”,”GPRS”

   Виконується конфігурування bearer профілю, для запита на NTP сервер.

6. AT+SAPBR=3,1,”APN”,”CMNET”

   Вказується APN для bearer профілю.

7. AT+SAPBR=1,1

   Після конфігурування необхідно відкрити профіль для подальших запитів у GPRS мережі.

8. AT+CNTPCID=1

   Встановлюємо для NTP bearer профіль 1

9. AT+CNTP=”pool.ntp.org”,8

   Подальші команди будуть конфігурувати дані для синхронізації часу модуля у мережі. Тобто, встановлення актуального часу. Це необхідно,

бо стандартно модуль стартує із некоректною датою та часом. У цій команді встановлюється NTP сервер та часовий пояс, який вираховується за формулою 4*X– де X поточна часова зона.

10. AT+CNTP

    Синхронізація часу модуля з часом мережі.

11. AT+CSSLCFG=”sslversion”,0,3

    Встановлюємо версію для SSH з’єднання з AWS MQTT сервером.

12. AT+SMSSL=1,"clientcert.crt", "certificate.crt"

    Встановлення необхідних сертифікатів для проходження аутентифікації в AWS.

13. AT+SMCONF-"URL", "a2mblhli26wqer-ats.iot.us-east-1.amazonaws.com", "9883" – встановлення MQTT посилання та встановлення порту, на який буде орієнтуватися модуль при з’єднанні.

14. AT+SMCONF="CLIENTID" "raspberry"

    Встановлення MQTT параметр client id для ідентифікації модуля у хмарних сервісах.

15. AT+SMCONF="KEEPTIME, 60

    Встановлення MQTT параметр на очікування 60 секунд.

16. AT+SMCONN

    Встановлення з’єднання з сервером

17. AT+SMPUB-"raspberry", 96, 1, 0

    Опублікувати повідомлення до теми raspberry. Також вказується довжина повідомлення другим параметром.

18. AT+SMDISC

    Після надсилання необхідних пакетів до певної теми закриваємо з’єднання MQTT.

19.AT+CNACT=0

Закриваємо безпровідне з’єднання у мережі на модулі [10].

```
AT+CREG=1
AT+COPS=4,2,"25503"
AT+CMNB=3
AT+CNACT=1,"cmnet"
AT+SAPBR=3,1, "Contype","GPRS"
AT+SAPBR=3,1, "APN","CMNET"
AT+SAPBR=1,1
AT+CNTPCID=1
AT+CNTP="pool.ntp.org",8
AT+CNTP
AT+CSSLCFG="sslversion",0,3
AT+SMSSL=1,"clientcert.crt","certificate.crt"
AT+SMCONF="URL","a2mblhli26wqer-ats.iot.us-east-1.amazonaws.com","8883"
AT+SMCONF="CLIENTID","raspberry"
AT+SMCONF="KEEPTIME",60
AT+SMCONN
AT+SMPUB="raspberry",96,1,0
AT+SMDISC
AT+CNACT=0
```

Рисунок 3.38 – AT-команди для відсилання координат до системи обробки

Для того щоб всі дані, які приходять з пристрою перенаправлялись на лямбду, яка відповідає за збереження координат до бази даних, створимо правило, що буде їх переправляти.

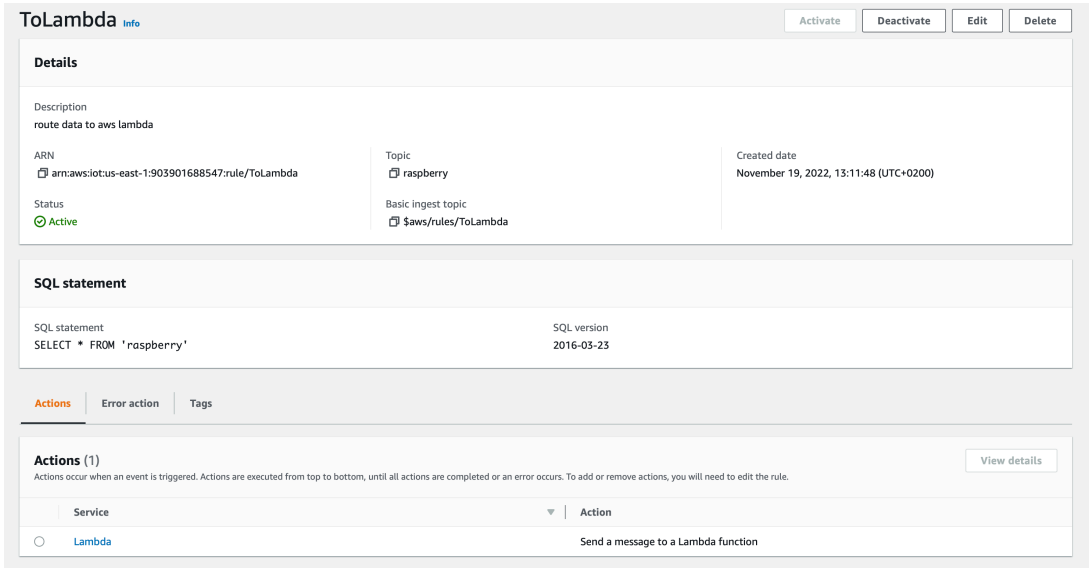


Рисунок 3.39 – Створення правила що буде перенаправляти дані з MQTT теми до лямбди

Після того як створене правило, до лямбди, що була завантажена раніше, та вказана в правилі як точка доставки повідомлень з теми, буде прикріплений тригер. Це означає, що до неї будуть надходити усі події, зв’язані з цією темою, тобто події, що були створені трекером.

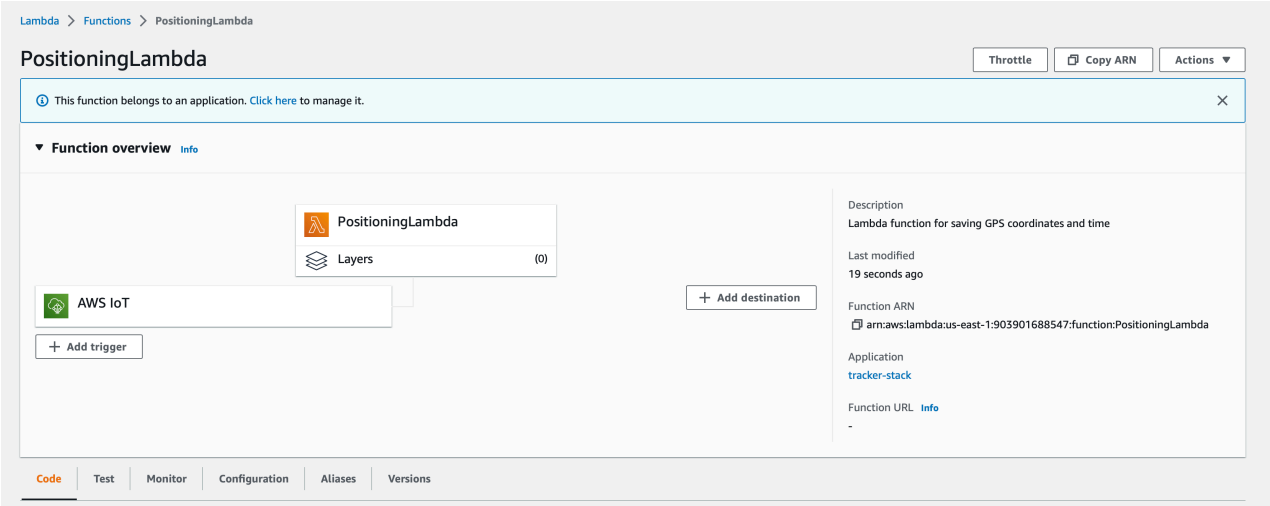


Рисунок 3.40 – Вигляд лямбди після успішного створення правила

Таким чином всі повідомлення, що будуть надходити в тему з назвою «raspberry» будуть автоматично викликати лямбду, що буде валідувати та зберігати отримані дані до бази даних.

Висновки за розділом 3

У Розділі 3 виконано аналіз існуючих сервісів IoT, відібрані необхідні для побудови системи обробки координат. Побудована архітектура системи, обрані компоненти для побудови трекеру. Піднята хмарна інфраструктура та налаштоване з’єднання трекеру із системою.

РОЗДІЛ 4
РЕЗУЛЬТАТИ РОБОТИ СИСТЕМИ

4.1 Досягнення мети роботи

У роботі був проведений аналіз систем відстежування транспорту, досліджені підходи до розробки таких систем та розроблений прототип такої системи з обробкою координат, що розташована повністю у хмарній платформі і завдяки цьому може надавати координати до багатьох користувальницьких інтерфейсів, взаємодіяти із багатьма пристроями відстеження, витримує досить велику кількість навантажень, виділяє обчислювальні потужності в залежності від потреб та має дуже зручний сервіс моніторингу здоров’я застосованих хмарних ресурсів та багато інших вбудованих властивостей.

4.2 Тестування роботи системи

Зібраний прототип GPS-трекеру зображений на рисунку 4.1.

Рисунок 4.1 – Вигляд пристрою

Після написання коду для роботи трекеру, створення системи обробки за допомогою хмарної платформи, створення користувацького інтерфейсу, було вирішено провести тестування.

Після вмикання трекеру на налаштування потрібно приблизно 2 хвилини. Модуль зв’язку налаштовується досить швидко, практично одразу після вмикання. При пошуку стільникової мережі модуль блимає раз в секунду, що означає тривалість пошуку. При реєстрації в стільниковій мережі даний модуль блимає раз у три секунди. На налаштування GPS після його ввімкнення може піти до 3 хвилин, в залежності від погодних умов. Найкращі результати модуль видає при ясному небі тому що так він швидше встановлює зв’язок із супутниками.

На рисунку 4.2 зображені логи трекеру одразу після його вмикання. Після визначення координат за допомогою GPS відбувається відправка даних до системи обробки.

Рисунок 4.2 – Приклад визначення координат пристроєм та їх відправки

Інтерфейс сервісу AWS IoT має можливість підписуватись на теми і таким чином прослуховувати повідомлення в них. Під час підписання на тему під назвою «raspberry» можна побачити дані, які в поточний момент приходять від трекеру. На рисунку 4.3 зображена підписка на цю тему.

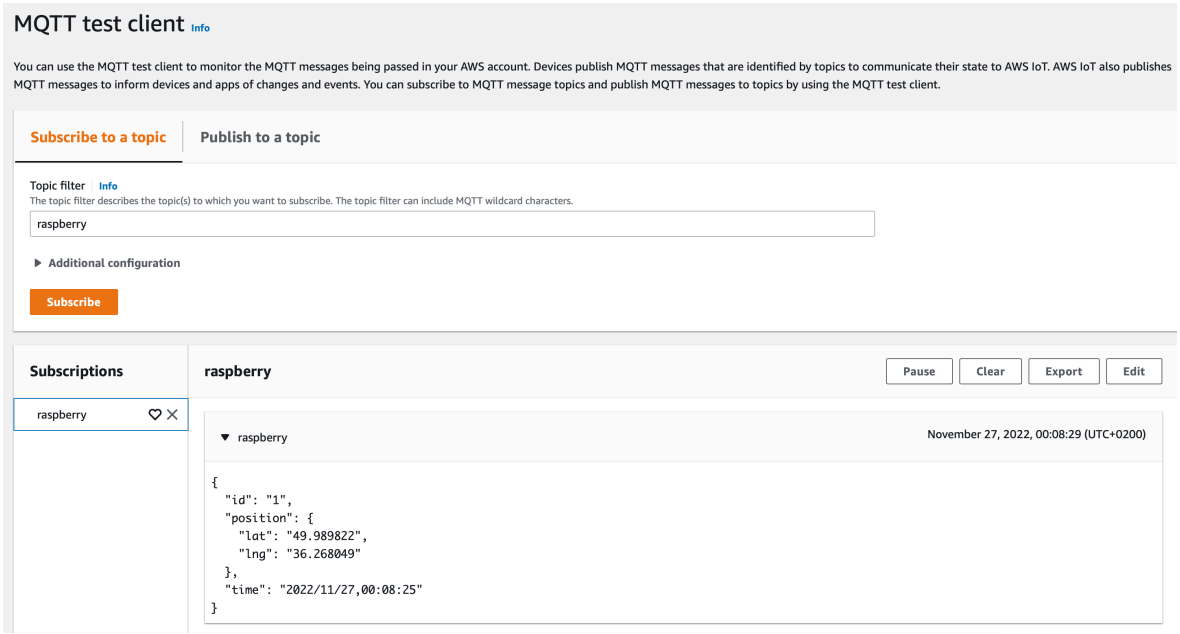


Рисунок 4.3 – Приклад даних отриманих від пристрою

Після того, як в тему «raspberry» надходять повідомлення, спрацьовує правило яке надсилає повідомлення до лямбди.

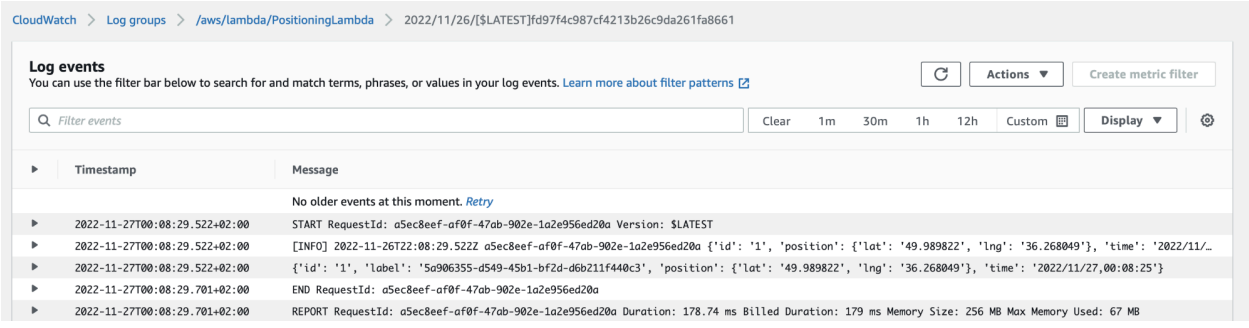


Рисунок 4.4 – Приклад логів лямбди яка спрацювала на подію надходження даних з пристрою

Лямбда записує координати до бази даних і після цього нова точка зображується на сайті. Щоб попасти на сайт, треба ввести посилання з S3 бакету. На рисунку 4.5 продемонстрований вигляд сайту. Тут можна побачити чисту карту міста без маркерів.

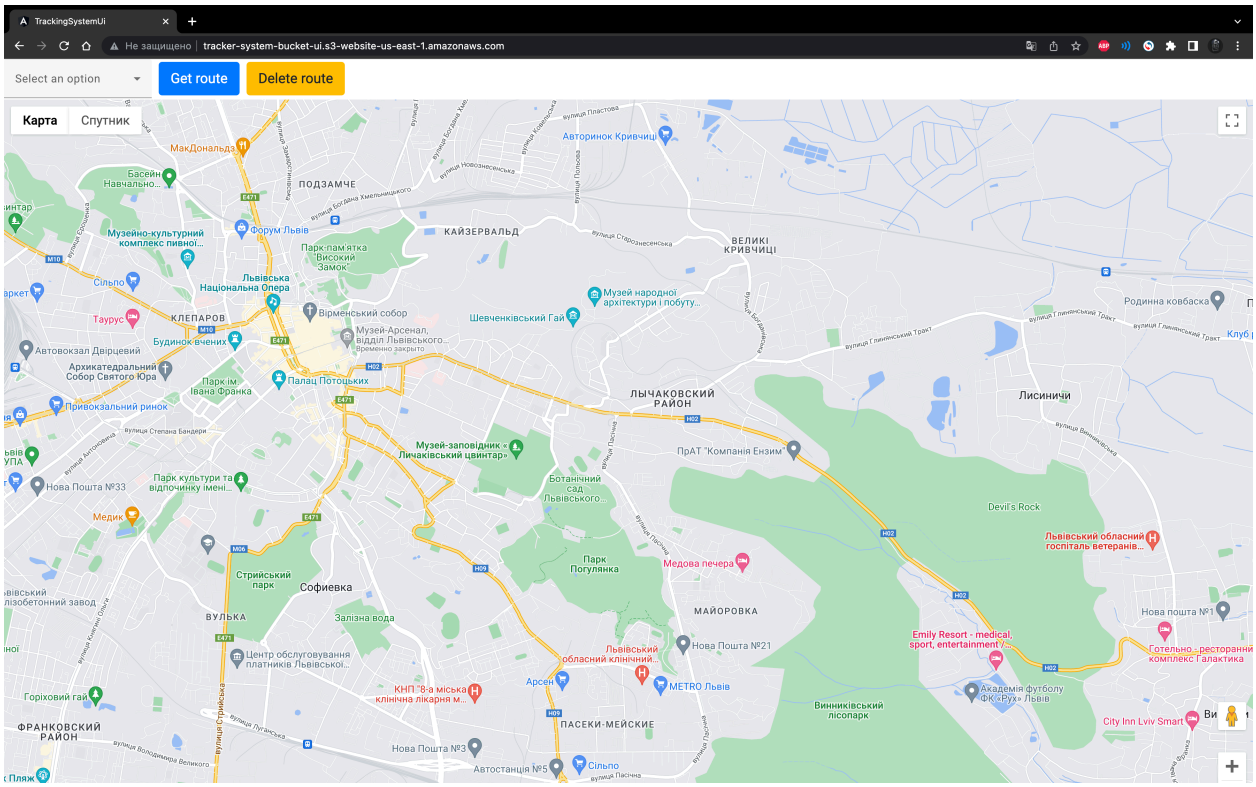


Рисунок 4.5 – Вигляд веб-сайту при першому заході на нього

На рисунку 4.6 зображено розташування в лівому верхньому куті поля вибору машини та кнопки для зображення її маршруту та прибирання.

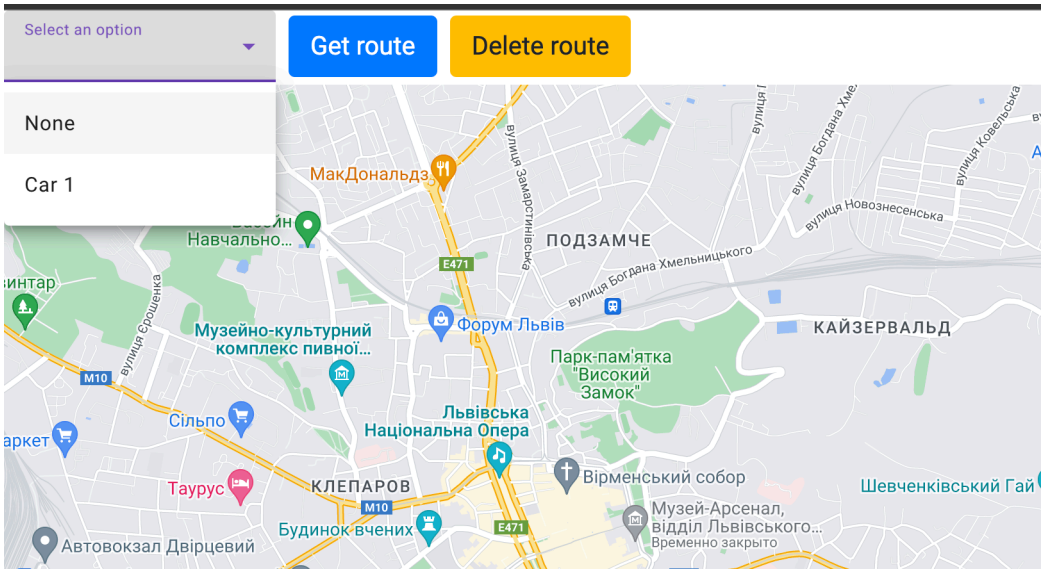


Рисунок 4.6 – Опція вибору машини

Після вибору машини та натисканні на натисканні на кнопку для зображення її маршруту на сайті з’явиться її маршрут. Маркери представляють собою координати передані трекером до системи. Кожен маркер пронумерований в часовому порядку і з’єднаний з попереднім та наступним маркерами як зображено на рисунку 4.7.

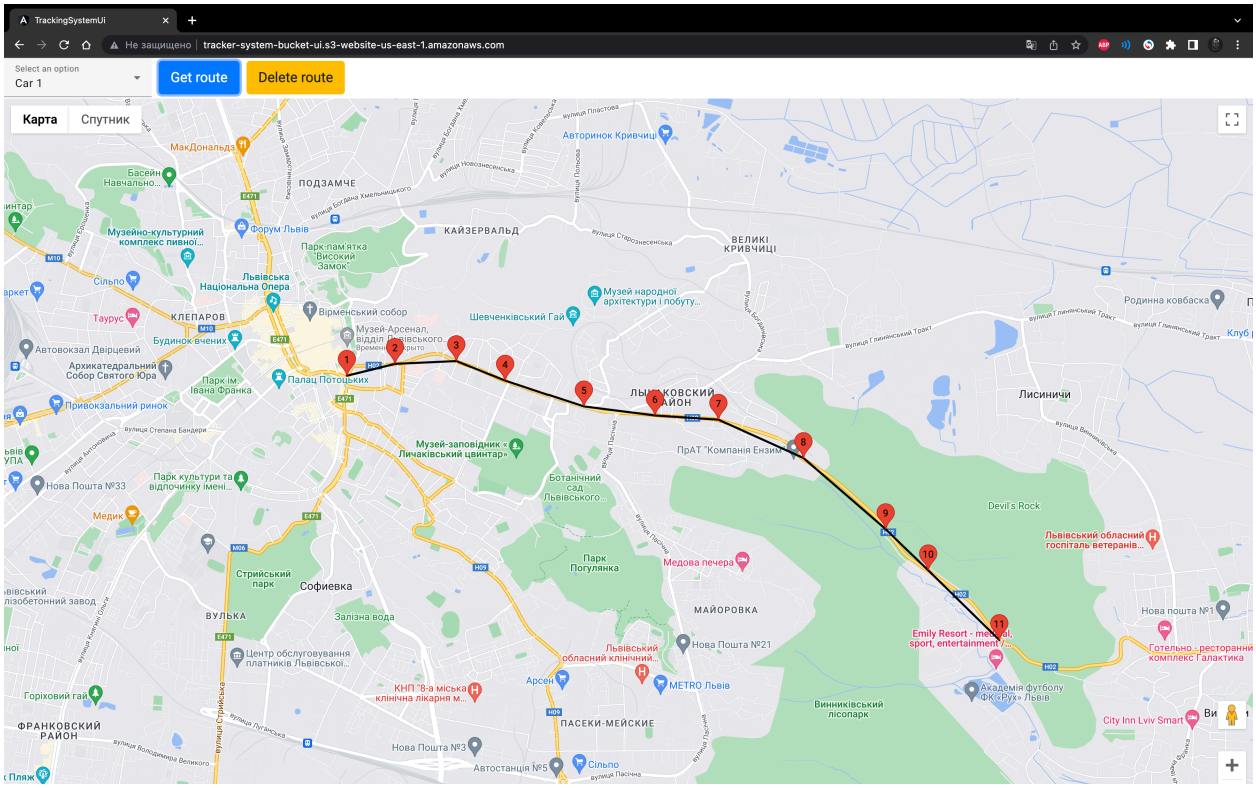


Рисунок 4.7 – Вигляд отриманого маршруту

При натисканні на кожен маркер на мапі з’явиться детальна інформація про цю точку. В даному випадку тестування такою інформацією виступає час.

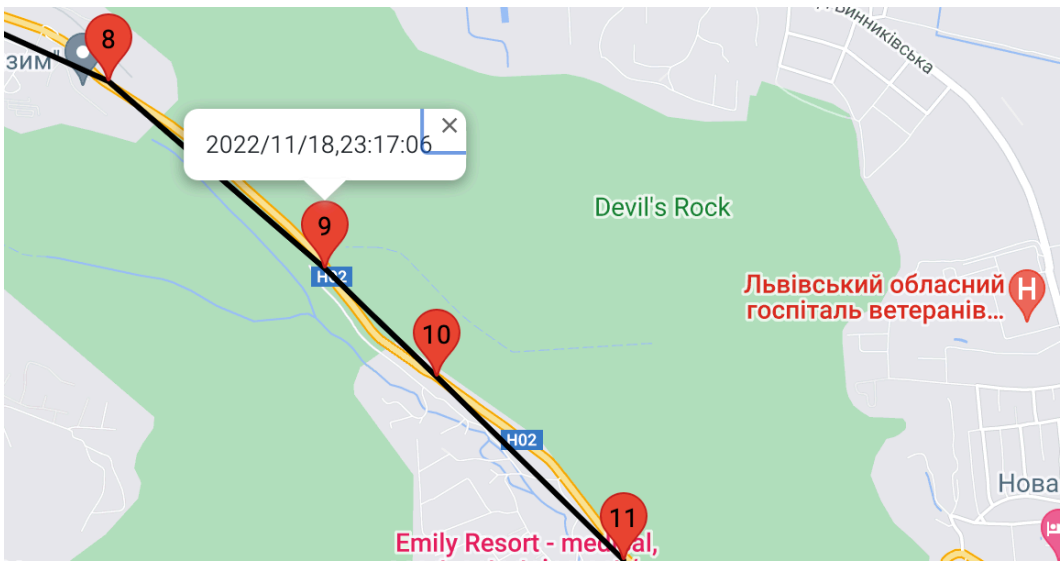


Рисунок 4.8 – Вигляд деталей обраної точки маршруту

Висновки за розділом 4

У розділі 4 була протестована робота розробленої моделі системи відстежування а саме трекеру, системи обробки координат та веб-сайту. Був виконаний вхід на сайт та протестований його інтерфейс, переглянутий маршрут першого транспорту, перевірена нумерація слідування маркерів за маршрутом. Протестована швидкість роботи системи обробки, її реагування на з’явлення нових координат від пристрою.

ВИСНОВКИ

Під час виконання кваліфікаційної роботи було розглянуто підхід до розробки моделі системи відстежування на базі хмарних технологій. Розроблений відстежуючий трекер що складається з двох частин – модуля зв’язку з GPS та управляючої плати. Розроблена модель системи відстежування користуючись Internet of Things для спілкування відстежуючого пристрою з системою обробки координат, що в свою чергу розташована повністю у хмарній платформі. Також розроблений веб-сайт на якому є можливість обрати конкретний транспорт та продивитись його маршрут по точкам, кожна з котрих є маркером на карті і при натисканні надає інформацію часу коли тут знаходився транспорт.

Системи відстежування транспорту знаходяться на важливому місці серед застосунків для підприємств які займаються перевезенням вантажу на далекі дистанції чи часто передислоковують свій транспорт у інше місце.

Розроблений прототип трекеру не залежить від типу транспорту, що відстежується, тобто від буде з’єднуватись із системою відстежування без додаткових налаштувань.

Система обробки координат розташована у хмарі, де оплата за сервіс надходить не за час його використання а за кожен запит до нього, таким чином вона теж є досить гнучкою та може бути масштабована на використання більшої кількості трекерів не збільшуючи вартість обслуговування системи. Надсилання координат є налаштоване тільки на розроблений веб-сайт.

Розроблений веб-сайт який теж розташований в хмарі доступний тільки для конкретного користувача підвищує приватність використання через те, що доступний не всім.

Розроблена модель системи обробки координат може бути легко перенесена у інший географічний регіон задля пришвидшення доставки контенту кінцевим користувачам. Використаний підхід до її розробки є досить гнучким через те що описана інфраструктура всіх сервісів знаходиться лише в одному файлі і розгортається за лічені секунди. Також в даному підході можливе використання версіонування інфраструктури, що дозволяє мати можливість одночасно розгорнутих двох версій системи для тестування.

Всі сервіси, що використані для побудови інфраструктури системи обробки мають логування у спеціальні лог-группи, що поліпшує відстежування стану системи.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. What is IoT: [Електронний ресурс],URL: https://www.oracle.com/internet-of-things/what-is-iot/ (Дата звернення: 15.09.2022).
2. How much does IoT cost: [Електронний ресурс],URL: https://becominghuman.ai/how-much-does-iot-cost-3255fe9225a5 (Дата звернення: 15.09.2022).
3. What is IoT (Internet of Things): [Електронний ресурс],URL: https://aws.amazon.com/what-is/iot/?nc1=h_ls (Дата звернення: 20.09.2022).
4. IoT Architecture – Detailed Explanation): [Електронний ресурс],URL: https://www.interviewbit.com/blog/iot-architecture/ (Дата звернення: 27.09.2022).
5. LBS: [Електронний ресурс],URL: http://www.smartphone.ua/w_location-based-service.html. (Дата звернення: 01.10.2022).
6. 3 способа определения местоположения: [Електронний ресурс] URL: https://watchgps.com.ua/2016/09/09/kak-opredelyaetsa-mestopolojenie/. (Дата звернення: 01.10.2022).
7. Setup Serial in Raspberry : [Електронний ресурс],URL: https://www.abelectronics.co.uk/kb/article/1035/serial-port-setup-in-raspberry-pi-os (Дата звернення: 15.10.2022).
8. Internet of Things (IoT): [Електронний ресурс],URL: https://docs.aws.amazon.com/whitepapers/latest/aws-overview/internet-of-things-services.html (Дата звернення: 20.10.2022)
9. What is AWS IoT: [Електронний ресурс],URL: https://docs.aws.amazon.com/iot/latest/developerguide/what-is-aws-iot.html (Дата звернення: 20.10.2022)

10. SIM7000 Series_AT Command Manual: [Електронний ресурс],URL: https://cdn.geekfactory.mx/sim7000g/SIM7000%20Series_AT%20Command%20Manual_V1.06.pdf (Дата звернення: 02.11.2022)

ДОДАТКИ

Додаток А

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Харківський національний університет імені В. Н. Каразіна

Факультет комп'ютерних наук
Кафедра теоретичної та прикладної системотехніки
Рівень вищої освіти (освітньо-кваліфікаційний рівень) Магістр
Галузь знань: 15 – Автоматизація та приладобудування
Спеціальність: 151 – «Автоматизація та комп'ютерно-інтегровані технології»

ЗАТВЕРДЖУЮ
Завідувач кафедри теоретичної
та прикладної системотехніки
_____д.т.н., проф. Шматков С. І.

«10» 12 2021року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Попової Анастасії Олександрівни
(прізвище, ім'я, по батькові студента)

1. Тема роботи Модель системи моніторингу автомобільних перевезень на базі технології хмарних обчислень

керівник роботи Толстолузька Олена Генадіївна, д.т.н., с.н.с., професор кафедри теоретичної та прикладної системотехніки
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від "___"_____20__року №_____

2. Строк подання студентом роботи 30 листопада 2022

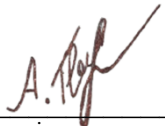
3. Перелік питань, які потрібно розробити.
1. Проведення аналізу проблем, які виникають в галузі відстежування транспортних перевезень;
2. Вивчення існуючих програмних реалізацій з відстежування транспорту;
3. Провести аналіз вимог до програмного рішення;
4. Проектування комп'ютерної моделі;
5. Розробка моделі системи моніторингу на основі комп'ютерної моделі;
6. Тестування розробленої моделі.

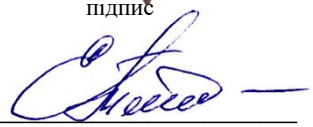
4. План роботи

№ з/п	Назви етапів роботи	Термін виконання етапів роботи
1.	Аналіз сучасного стану систем моніторингу транспортних перевезень	11.10.2021 - 09.01.2022
2.	Визначення вимог щодо систем, які розробляються	10.01.2022 - 01.02.2022
3.	Проектування архітектури моделі системи моніторингу	02.02.2022 - 30.03.2022
4.	Розробка моделі системи	01.04.2022 - 01.05.2022
5.	Розробка технічного завдання	01.05.2022 - 25.05.2022
6.	Розробка програми і методики тестування розробленої моделі	26.05.2022 - 20.06.2022
7.	Розробка рекомендацій щодо застосування розробленої моделі	21.06.2022 - 15.07.2022
8.	Написання пояснювальної записки	16.07.2022 - 01.09.2022
9.	Оформлення звіту за результатами науково-дослідницької практики	11.10.2022 - 13.10.2022
10.	Оформлення звіту за результатами переддипломної практики	13.10.2022- 23.11.2022
11.	Підготовка доповіді на тему дипломної роботи на науково-технічну конференцію	16.10.2022 - 01.11.2022

12.	Підготовка до попереднього захисту роботи	02.11.2022 - 14.11.2022
13.	Представлення дипломної роботи науковому керівнику	15.11.2022 - 23.11.2022
14.	Представлення дипломної роботи науковому рецензенту	24.11.2022 - 30.11.2022

5. Дата видачі завдання 10.12.2021

Студент А.О.Попова

ініціали, прізвище підпис

Керівник роботи О.Г.Толстолузька

ініціали, прізвище підпис