3. *Юзевич В. М.* Інформаційна технологія оцінювання станів об'єктів із сталі в морському середовищі із сірководнем з використанням алгебри алгоритмів / В. М. Юзевич, О. І. Огірко // Наукові записки (Українська академія друкарства). – 2012. – № 4 (41). – С. 160–172.
4. *Сопрунюк П. М.* Оцінка поверхневої енергії сталей у сірководневих середовищах / П. Сопрунюк, В. Юзевич, О. Огірко // Фіз.-хім. механіка матеріалів. Проблеми корозії та протикорозійного захисту матеріалів. – 2000. – Т. 2, № 1. – С. 726-730.
5. . *Юзевич В. М.* Інформаційна технологія оцінювання станів об'єктів із сталі в корозійному середовищі із сірководнем на основі термодинамічної моделі / В. М. Юзевич, О. І. Огірко // Матеріали III Всеукраїнської науково-практичної конференції "Сучасні інформаційні технології в економіці, менеджменті та освіті". 21 листопада 2012 р., Львів. – С. 71-76.
6. *Юзевич В.М.,* Розрахунок впливу кількісного фактора на деформаційну характеристику поліуретанового зразка /В.М.Юзевич, Я.І Чехман // Поліграфія і видавнича справа. – Львів, 1987. – № 23. – С. 46-50.
7. *Юзевич В.М.,* Роль масштабного фактора при випробуванні поліуретанового зразка різної твердості /В.М.Юзевич, Я.І Чехман // Поліграфія і видавнича справа. – Львів, 1988. – № 24. – С. 47-49.
8. *Олійник Р.В.,* Інформаційна технологія обробки даних інформаційних систем із змінними структурою та параметрами. / Р.В.Олійник, О.І. Огірко // Комп'ютерні технології друкарства,УАД.Львів. 2016.1(35).С.87-97.

Поступила 20.04.2017р.

УДК 621

В.Р. Сподарик, Національний університет «Львівська політехніка»

ПРОЕКТУВАННЯ ГЕОІНФОРМАЦІЙНОЇ СИСТЕМИ ДЛЯ ПОШУКУ МАРШРУТІВ

Розглянуто проблему з добиранням на роботу у містах, проаналізовано варіанти вирішення цієї проблеми за допомогою пошуку попутників. Розроблено алгоритм для пошуку маршрутів за заданими координатами та проведено апробацію його роботи на прикладі веб-рішення. Такий алгоритм є універсальним і підходить для систем, де є робота з пошуком маршрутів.

Ключові слова: час пік, транспорт, пасажиропотік, гео-пошук, попутники, координування руху, оптимізація, маршрут, MongoDB.

I. Постановка проблеми

Кожного дня мільйони людей добираються на роботу громадським транспортом. Більшість з них намагаються їхати з 7 по 9 ранку. Зазвичай коли з'являється дуже багато бажаючих їхати, то громадський транспорт не може справитись і перевезти всіх. Враховуючи регулярність такої проблеми вона

отримала назву «час пік», коли всі автобуси, трамваї та тролейбуси переповнені стоять в корках. [1]

Аби виправити цю проблему було вирішено розробити систему для колективного добирання за допомогою приватного транспорту. Коли водії маючи вільне місце в авто підвозять сусідів та колег, навіть якщо в тих є власне. Також така система дозволить генерувати актуальні дані по пасажиропотоку в містах, що дозволить проводити нові дослідження в цій області.

II. Мета роботи

Реалізувати систему, яка б дозволила:

- користувачам знаходити попутників і добиратись на роботу разом збільшуючи ефективність використання транспортних засобів та доріг
- проводити аналіз пасажиропотоку для майбутніх досліджень (оптимізації маршрутів громадського транспорту та координування руху на дорозі)

Потрібно реалізувати:

- можливість зберегти маршрути користувачів (час, координати звідки і куди користувачі їдуть, наявність власного автомобіля, тощо);
- зручний пошук схожих маршрутів;
- використовувати проміжні точки при пошуку (пасажир може підсісти на середині маршруту і вийти не на кінцевій зупинці маршруту водія).

III. Опис алгоритму

Для вирішення даної задачі, було прийнято рішення використати: Django Framework як веб-сервер, не реляційну документо-орієнтовану базу даних MongoDB для збереження даних та гео-пошуку та Google Maps API для геолокації та відображення результатів на карті.

Чому обрано MongoDB — вона підтримує багато варіантів гео-пошуку «з коробки», зручна в горизонтальному та вертикальному масштабуванні, має механізми відмовостійкості.

Щоб знайти всі точки в базі даних поруч із заданою потрібно створити колекцію з даними:

{‘location’: [24.014497, 49.835352]},
{‘location’: [24.031508, 49.841948]},
{‘location’: [24.022570, 49.823601]},

Створити гео-індекс 2dsphere до цієї колекції:
db.places.ensure_index([('location', '2dsphere')])

І виконати наступний запит до бази даних, вказавши поле (location), координати (lng: 24.03, lat: 49.84) та максимальну відстань для пошуку:

```
db.places.find({
   'location': {
      '$nearSphere': {
         '$geometry': {
            'type': "Point",
            'coordinates': [ 24.03, 49.84 ]
         },
         '$maxDistance': 5000
}}})
```

В результаті отримаємо список документів, де є точка, яка поруч з шуканою. [2] [3]

Задача ускладнюється коли потрібно знайти маршрути, які складаються з багатьох точок, з врахуванням напрямку руху. Для вирішення такого завдання було розроблено наступний алгоритм.

1. Користувач вводить початкову, проміжні і кінцеву точи та час відправлення
2. Система виконує запит до Google Maps API для генерації оптимального маршруту і зберігає результат — ламану лінію маршруту в базі даних
3. Виконується пошук маршрутів за початковою точкою
4. Виконується пошук серед маршрутів за кінцевою точкою, де id-номер маршруту є в списку результатів попереднього кроку
5. Відбувається перевірка порядку входження в списку координат ламаної, аби маршрут не був у зворотню сторону
6. Перевіряється чи маршрут входить в часовий проміжок заданий користувачем
7. Результуючий список маршрутів сортується по-мінімальній відстані, яку треба пройти пішки
8. Результати виводяться користувачу

Весь процес використання системи користувачем виглядатиме як на рис.1. Користувач повинен спершу автентифікуватись за допомогою соц.мережі, створити маршрут з вказанням початкової, кінцевої точок та часу відправлення. Потім він може шукати співпадіння за цим маршрутом. А коли знайде попутника, тоді комунікувати будь-яким з доступних способів.

IV. Аналіз отриманих результатів

Тестування роботи даної системи було проведено в межах пошукового двигуна проекту «Підвезу!». Перевірено, що система коректно працювала при пошуку серед 1000 справжніх маршрутів користувачів.

На рис.2. показана форма створення маршруту користувачем. Маршрут автоматично відображається при введенні початкової і кінцевої точок. Також можна додати проміжні точки. На їх основі генерується ламана лінія, яка зберігається в базі даних.

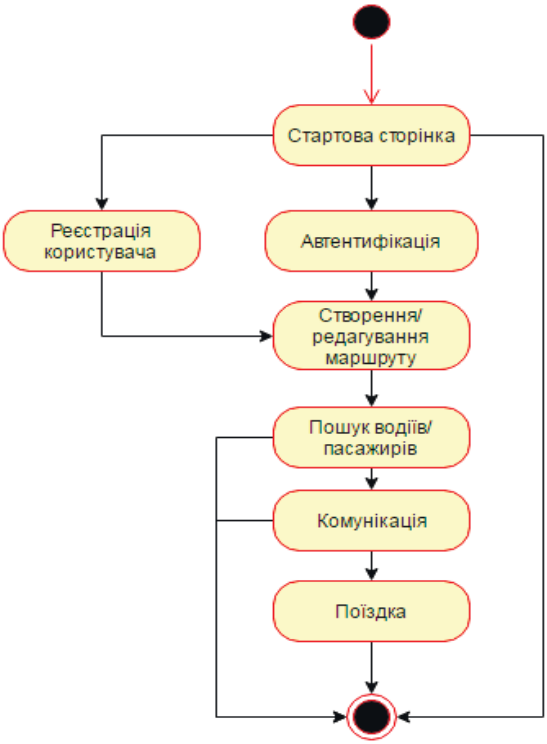

Рис.1. Діаграма активностей користувача

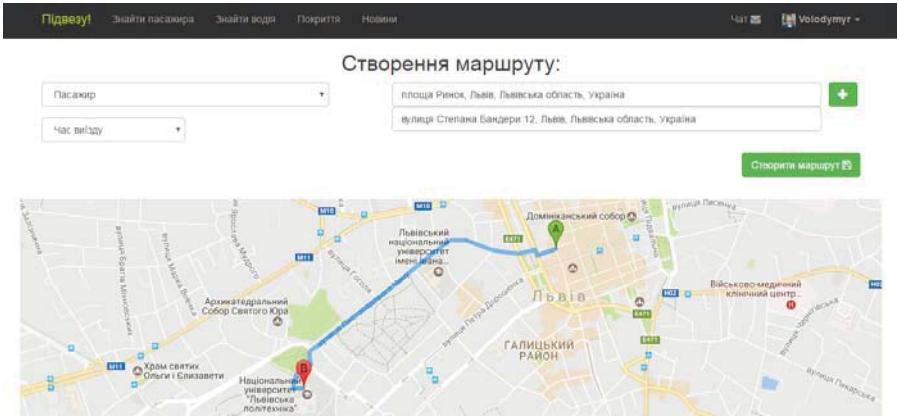

Рис.2. Збереження інформації про маршрут користувача

На рис.3. показано форму для пошуку маршрутів з координатами та діапазоном годин за якими відбувається пошук.

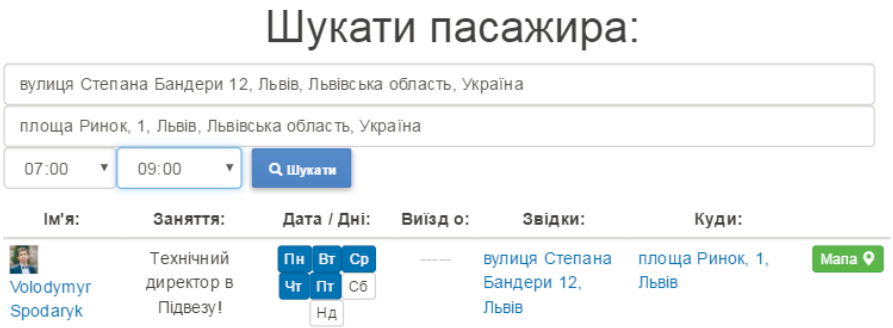

Рис.3. Форма для пошуку з результатами

Одна з проблем з якою стикнувся під час дослідження: Google Maps API та GeoJSON (який є основою гео-пошуку MongoDB) мають різний порядок для опису GPS-координат.

MongoDB – спершу longitude, потім latitude [4].

Google Maps API – спершу latitude, потім longitude [5].

V. Висновки

У роботі показано як побудувати систему для пошуку співпадінь за маршрутами користувачів. Цей принцип можна використовувати як для пошуку водіїв, які можуть підвезти так, і для пасажирів, для спільного виклику таксі, чи навіть шукати маршрути громадського чи будь-якого іншого транспорту. Алгоритм доволі універсальний і може бути перевикористаний будь-де, де потрібна робота з маршрутами. База даних MongoDB – добре працює з гео-даними «з коробки» і для пошуку потребує тільки вказання гео-індексів.

Під час тестування було виявлено, що при збільшенні кількості маршрутів понад 4000, алгоритм починає працювати занадто довго (близько 15 секунд). Планується оптимізувати його в майбутньому. Один з варіантів – розробити алгоритм з використанням іншого типу об’єктів GeoJSON, що підтримує MongoDB.

1. Е. Лобанов, в Транспортная планировка городов, М., Транспорт, 1990
2. K. Chodorow, в MongoDB: The Definitive Guide - Second edition, O’Reilly Media, Inc., 2013.
3. «$nearSphere - MongoDB Manual 3.4,» MongoDB, Inc., [3 мережі].: https://docs.mongodb.com/manual/reference/operator/query/nearSphere/.
4. «GeoJSON Objects - MongoDB Manual,» MongoDB, Inc, [3 мережі].: https://docs.mongodb.com/manual/reference/geojson/.
5. «Find or enter latitude & longitude - Computer - Google Maps Help,» [3 мережі].: https://support.google.com/maps/answer/18539.

Поступила 20.03.2017р.