УДК 004

**О. С. Дусанюк**
**В. Ю. Коцюбинський**

# ОГЛЯД ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ ТА ОСНОВНИХ АЛГОРИТМІВ СОЦІАЛЬНОЇ МЕРЕЖІ, ЩО БАЗУЄТЬСЯ НА ОСНОВІ ГЕОЛОКАЦІЇ

Вінницький національний технічний університет

**Анотація**
*У даній роботі приділено увагу технологіям та засобам для реалізації мобільної соціальної мережі, що базується на основі геолокації, а також її основним алгоритмам та функціям.*
**Ключові слова:** геолокація, соціальна мережа, геолокаційна соціальна мережа, серверна частина, Python, Flask, REST API.

**Abstract**
*This work focuses on technologies and tools for implementing a mobile social network based on geolocation, as well as its main algorithms and functions.*
**Keywords:** geolocation, social network, geolocation social network, backend, Python, Flask, REST API.

**Технології та засоби для реалізації соціальної мережі, що базується на основі геолокації**

Для розробки серверної частини будь-якого клієнт-серверного додатку необхідно визначитися із стеком (набором) технологій, який буде використовуватися для його реалізації. Для серверної частини даного додатку було обрано наступні засоби та технології: мова програмування Python, фреймворк Flask, СУБД MySQL, та Nginx разом із uWSGI в якості веб-серверу.

*Python* - це мова програмування, яка широко використовується в інтернет-додатках, розробці програмного забезпечення, науці даних і машинному навчанні (ML) [1]. Python є популярним вибором для серверної розробки з кількох причин:

1. Масштабованість: проекти, розроблені мовою Python, легко розширюються і масштабуються, що стає можливим завдяки можливості їх адаптації до високорівневої логіки.

2. Широкий спектр бібліотек і фреймворків: Python має широкий спектр бібліотек і фреймворків, таких як Django, Flask і Pyramid, які допомагають розробникам швидко й ефективно створювати веб-додатки.

3. Кросплатформеність: Python є кросплатформним, що означає, що код, написаний на одній операційній системі, можна запускати на інших операційних системах без змін.

4. Висока продуктивність: Python має чудову продуктивність і достатньо швидкий для обробки веб-запитів і обробки даних [1, 2].

*Flask* - це мікрофреймворк для Python, що дозволяє розробникам швидко й легко створювати веб-додатки та мікросервіси. Flask можна використовувати при розробці як тренувальних проектів або невеликих сайтів, яким не потрібен складний бекенд, так і API та складних проектів [3]. Деякі з ключових функцій Flask:

1. Можливість додання розробниками необхідного функціоналу до свого проекту шляхом завантаження необхідних бібліотек.

2. Підтримка RESTful архітектури та управління запитами RESTful.

3. Flask добре інтегрується з іншими популярними технологіями та фреймворками, такими як бази даних SQL, ORM тощо [3, 4].

*MySQL* - це популярна система управління реляційними базами даних із відкритим кодом, яка базується на структурованій мові запитів (SQL) [5]. Ось кілька причин, чому MySQL є хорошим вибором в якості СУБД:

1. Масштабованість: MySQL може працювати з великими базами даних і високим навантаженням трафіку, що робить його ідеальним рішенням для додатків з великими обсягами даних.

2. Гнучкість: MySQL підтримує широкий спектр типів даних, мов програмування та інструментів розробки, що робить його універсальним вибором для розробників та організацій.

3. Швидкість і продуктивність: MySQL може швидко й ефективно обробляти великі обсяги даних.

4. Безпека: MySQL має потужні вбудовані функції безпеки, включаючи підтримку шифрування, аутентифікації користувача та контролю доступу [5, 6].

*Nginx* і *uWSGI* є програмними інструментами, які часто використовуються разом для розгортання веб-програм Python.

Nginx - це високопродуктивний веб-сервер, який можна використовувати як зворотний проксі-сервер, балансувальник навантаження та HTTP-кеш. Він призначений для обробки великих обсягів трафіку і пропонує розширені функції, такі як шифрування SSL/TLS, фільтрування IP-адрес і URL-адрес, а також підтримку різноманітних веб-протоколів [7, 8].

uWSGI - це сервер веб-додатків, який може запускати програми на Python через протокол WSGI. Він призначений для одночасної обробки великої кількості запитів і забезпечує розширені функції, такі як керування процесами, балансування навантаження та кластеризація [8].

Разом Nginx і uWSGI забезпечують потужну платформу для розгортання веб-додатків Python. Типове розгортання передбачає налаштування Nginx як зворотного проксі-сервера, який пересилає запити до uWSGI, що запускає Python програму. Потім uWSGI обробляє запити та повертає відповідь Nginx, який, у свою чергу, надсилає її назад клієнту.

Для взаємодії між серверною та клієнтською частинами було обрано підхід REST API. Його також називають RESTful. RESTful (Representational State Transfer) - це архітектурний стиль для створення веб-сервісів, який широко використовується для розробки масштабованих і ефективних веб-додатків. Він ґрунтується на протоколі HTTP та зосереджений на ресурсно-орієнтованій архітектурі, де дані та функції розглядаються як ресурси, до яких можна отримати доступ за допомогою стандартних методів HTTP, таких як GET, POST, PUT, DELETE тощо. Перевагами використання архітектури RESTful є:

1. Клієнт-серверна архітектура: веб-служби RESTful використовують клієнт-серверну архітектуру, яка розділяє серверну частину від клієнтської. Це полегшує підтримку та масштабування додатку.

2. Багаторівнева система: веб-сервіси RESTful розроблені як багаторівневі, що означає, що кожен компонент системи ізольований від інших компонентів. Це дозволяє спростити розробку, обслуговування та масштабованість.

3. Відсутність збереження стану: дана особливість знімає навантаження з сервера, оскільки серверу не потрібно зберігати інформацію про попередні запити клієнта.

4. Безпека: служби RESTful можуть використовувати стандартні механізми безпеки HTTP, такі як SSL/TLS, OAuth і JWT для аутентифікації та авторизації [9].

На рисунку 1 наведено Deployment діаграму.

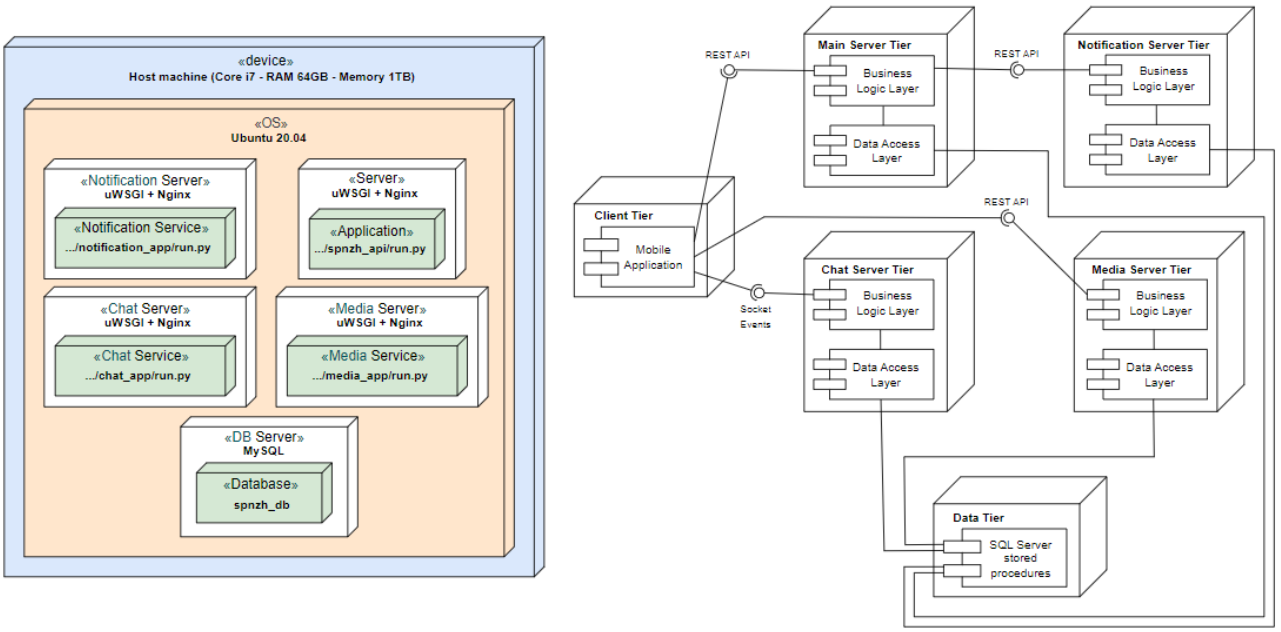


Рисунок 1 - Deployment діаграма

**Основні алгоритми та функції соціальної мережі, що базується на основі геолокації**

До основних алгоритмів та функцій соціальної мережі, що базується на основі використання геолокації, можна віднести публікацію поста, отримання списку постів, та відправку нотифікації користувачеві, якщо пост публікується в певному радіусі від нього.

Алгоритм публікації поста полягає в наступному:

1. Користувач відправляє запит на створення нової публікації на сервер. В тілі запиту надсилаються дані про пост. Серед них надсилаються поля “lat” (latitude) і “lon” (longitude), що відповідають координатам поста.

2. На стороні сервера йде процес обробки даних та їх запис у базу даних.

3. Після створення поста, основний сервер повертає користувачеві дані про створений пост і надсилає запит до серверу нотифікацій на створення та відправку сповіщень. В тілі запиту надсилаються дані, необхідні для нотифікацій.

4. Сервер нотифікацій створює та відправляє сповіщення користувачам, які мають відношення до користувача який створив пост, а також тим хто знаходиться поблизу.

Data flow діаграма створення поста наведена на рисунку 2.

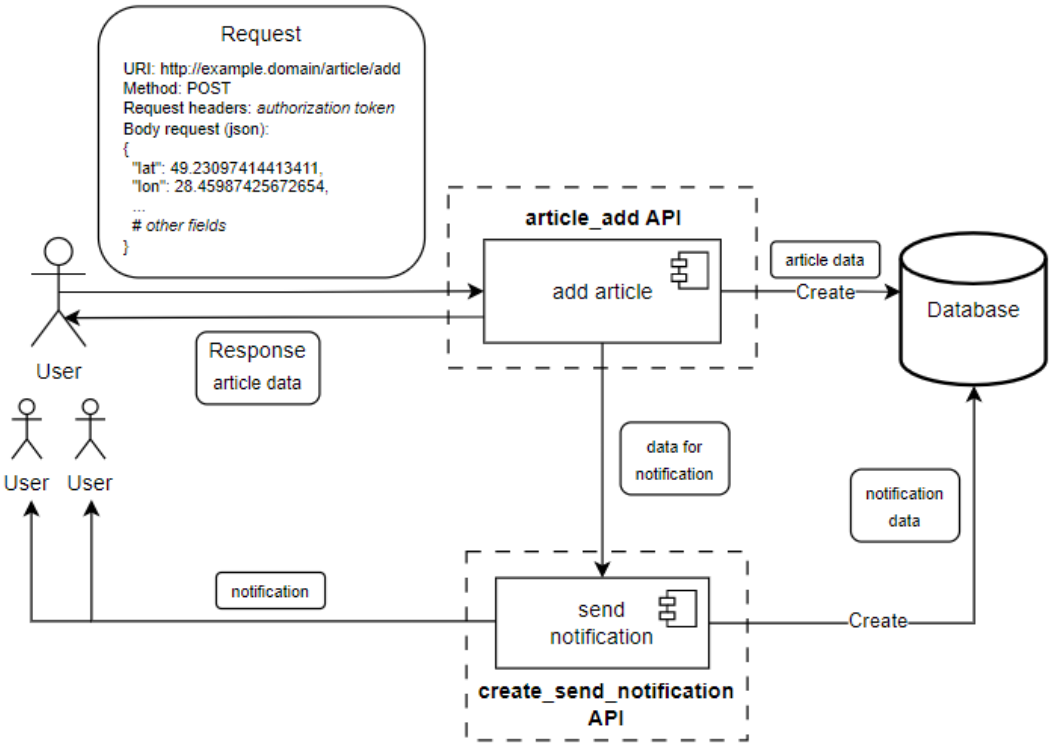


Рисунок 2 - Data flow діаграма створення поста

Алгоритм отримання списку публікацій полягає в наступному:

1. Користувач відправляє запит на отримання списку постів на сервер. За бажанням він може передати url параметри “lat” і “lon”, що відповідають його геолокації.

2. Якщо були передані параметри “lat” і “lon”, то сервер повертає користувачеві список лише тих постів, які публікувалися в певному радіусі від користувача (значення радіусу вказується в налаштуваннях користувача), в іншому ж випадку повертаються всі доступні пости.

Фрагмент sql запиту, який використовується для отримання постів, які публікувалися поблизу користувача наведено на рисунку 3.

```
...
# sql query
...
(
  6371 * acos (
    cos( radians({lat_user}) )
    * cos( radians( articles.lat ) )
    * cos( radians( articles.lon ) - radians({lon_user}) )
    + sin( radians({lat_user}) )
    * sin( radians( articles.lat ) )
  )
) AS distance
...
# sql query
...
HAVING distance < ({radius_user} / 1000)
```

Рисунок 3 - Фрагмент sql запиту

Data flow діаграма отримання списку постів наведена на рисунку 4.

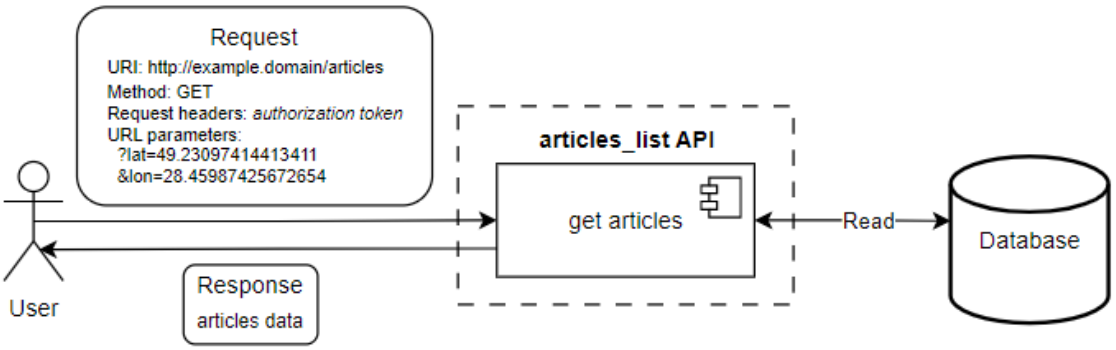


Рисунок 4 - Data flow діаграма отримання списку постів

Для відправки нотифікацій користувачам, при публікації поста поруч, необхідно отримати безпосередньо самих користувачів. Для цього треба обчислити відстань між користувачем та місцем публікації поста і дізнатися чи є дана відстань меншою за радіус користувача. Sql запит для даних розрахунків наведено на рисунку 5.

```
SELECT us.user_id,
       us.notif_around_me,
       us.around_radius,
       up.last_lat, up.last_lon,
       (
         6371 * acos (
           cos( radians(up.last_lat) )
           * cos( radians( {post_lat} ) )
           * cos( radians( {post_lon} ) - radians(up.last_lon) )
           + sin( radians(up.last_lat) )
           * sin( radians( {post_lat} ) )
         )
       ) AS distance
FROM user_settings AS us
LEFT JOIN fts_user_profile AS up ON us.user_id = up.user_id
WHERE us.notif_around_me = true
HAVING distance < (us.around_radius / 1000)
```

Рисунок 5 - Sql запит для отримання користувачів поблизу

**Висновки**

Отже, огляд технологій та основних алгоритмів для реалізації мобільної соціальної мережі, що базується на основі геолокації, дає цінну інформацію для майбутніх досліджень. Це відкриває можливості для подальшого дослідження в області аналізу соціальних мереж, дозволяючи краще зрозуміти соціальні взаємодії та поширення інформації.

Крім того, результати дослідження можуть сприяти розробці послуг, заснованих на геолокації, міського планування, транспортування тощо. Використовуючи алгоритми на основі геолокації, дослідники можуть оптимізовувати системи, створювати персоналізовані послуги та вдосконалювати механізми реагування на надзвичайні ситуації для різних спільнот.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. What is Python? [Електронний ресурс] – Режим доступу: https://aws.amazon.com/what-is/python/, вільний.

2. Особливості та переваги Python [Електронний ресурс] – Режим доступу: https://spacelab.ua/articles/osobennosti-i-preimushestva-python/, вільний.

3. FLASK PYTHON FRAMEWORK [Електронний ресурс] – Режим доступу: https://quintagroup.com/cms/python/flask, вільний.

4. Flask [Електронний ресурс] – Режим доступу: https://devopedia.org/flask, вільний.

5. What is MySQL? Everything You Need to Know [Електронний ресурс] – Режим доступу: https://ua.talend.com/resources/what-is-mysql/, вільний.

6. MySQL Features [Електронний ресурс] – Режим доступу: https://www.javatpoint.com/mysql-features, вільний.

7. What Is Nginx? A Basic Look at What It Is and How It Works [Електронний ресурс] – Режим доступу: https://kinsta.com/knowledgebase/what-is-nginx/, вільний.

8. The Layered World Of Web Development: Why I Need NGINX And UWSGI To Run A Python App? [Електронний ресурс] – Режим доступу: http://www.ines-panker.com/2020/02/16/nginx-uwsqi.html, вільний.

9. What is RESTful API [Електронний ресурс] – Режим доступу: https://aws.amazon.com/what-is/restful-api/, вільний.

***Дусанюк Олександр Сергійович*** – студент групи 1АКІТ-22м, факультету інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, м. Вінниця, aleksdusanuyk45@gmail.com

***Коцюбинський Володимир Юрійович*** – к.т.н., доцент, Вінницький національний технічний університет, м. Вінниця, vkotsyubinsky@gmail.com

***Dusaniuk Oleksandr S.*** – student of 1ACIT-22m, Faculty of Intelligent Information Technologies and Automation, Vinnytsia National Technical University, Vinnytsia, aleksdusanuyk45@gmail.com

***Kotsiubynskyi Volodymyr Y.*** – PhD, Associate Professor, Vinnytsia National Technical University, Vinnytsia, vkotsyubinsky@gmail.com