УДК 004.4
UDC 004.4

DOI: 10.33744/0365-8171-2022-112-219-226

ЗАСТОСУВАННЯ І РОЗРОБКА МОБІЛЬНИХ ДОДАТКІВ ДЛЯ ГРОМАДСЬКОГО ТРАНСПОРТУ

APPLICATION AND DEVELOPMENT OF MOBILE APPLICATIONS FOR PUBLIC TRANSPORT

Азізов Руслан Талятович, *Національний транспортний університет, аспірант кафедри інформаційних систем і технолоій, azizov1916@gmail.com*

https://orcid.org/0000-0003-2649-4822

Борецький Владислав Вікторович, *Національний транспортний університет, аспірант кафедри інформаційних систем і технолоій, vladlenius88@gmail.com*

https://orcid.org/0000-0001-5525-4604

Діхтяренко Володимир Васильович, *Національний транспортний університет, аспірант кафедри інформаційних систем і технолоій, dikhtiarenko0@gmail.com*

https://orcid.org/0000-0003-2549-4820

Куценко Олександр Іванович, *Національний транспортний університет, аспірант кафедри інформаційних систем і технолоій, alexkutsenko95@gmail.com,*

https://orcid.org/0000-0003-0047-4874

Анотація. Робота присвячена аналізу застосування та розробки мобільних додатків для громадського транспорту. Появу мобільних додатків у транспортній і логістичній індустрії можна поділити на наступні типи: оптимізація та планування маршруту, онлайн бронювання та відстеження, інформація про паркування, управління транспортом і автопарком, канал зв'язку, оптимізація розкладу транспортування/доставки. Сучасні мобільні додатки забезпечують кілька вбудованих переваг, як-от швидкість, ефективність та атмосферу без помилок. Постійне швидке оновлення розробки мобільних додатків для транспорту та логістики, ймовірно, продовжить прискорюватися на все більш насиченому ринку в наступні роки. Однак разом із такою швидкою еволюцією можливості для підвищення

прибутковості, оптимізації результатів та покращення користувацького досвіду вдосконалюються та розширюються на однаковому рівні.

Ключові слова: веб-додаток, громадський транспорт, NoSQL, архітектура клієнт-сервер.

Вступ Сьогодні за допомогою мобільних додатків можна замовити квитки на потяг або літак, переглянути карту місцевості та прокласти маршрут з точки а в точку б, переглянути карту заправок та купити бензин, замовити вантажні перевезення. За допомогою додатку компанії KLM що займається перельотами можна придбання квитків або перегляду розкладу польотів та можливих варіантів рейсу, не потрібна фізична присутність людини, отже додаток економить час та ресурси планети. Додатки для транспорту мають такі функції: продаж квитків, планування подорожей, відстеження транспортного засобу, карти, платіжні інтеграції тощо. Транспортні додатки можна поділити на 2 великі групи: додатки загального призначення для громадського транспорту та додатки транспортних постачальників. Додатки транспортних постачальників - це програми розроблені для певних транспортних агентств. На додаток до інформації про маршрут і розклад для планування поїздки, такі програми дозволяють купувати квитки онлайн (пасажир зазвичай повинен активувати віджет квитків, щоб скористатися цією функцією). інтегрований із системами бронювання квитків. Ці компанії займаються інвентаризацією місць, оскільки кожен квиток видається на конкретну поїздку. Додатки загального призначення для громадського транспорту. Ці додатки для міської навігації зазвичай підтримують десятки чи сотні міст по всьому світу та надають інформацію про варіанти поїздок на роботу різними транспортними агентствами, включаючи компанії, що пропонують велопрокат та прокат автомобілів. Moovit, Transit та Citymapper є одними з найпопулярніших додатків цього типу. Додаток Moovit – найважчий тут: він зорієнтує понад 200 мільйонів людей у понад 2500 містах у понад 80 країнах..

Метою даної роботи є аналіз застосування мобільних додатків для громадського транспорту та типові особливості їх архітектури, огляд існуючих популярних та відомих SPA додатків, технологій, що були використані при їх розробці, огляд інструментів та фреймворків, що використовуються при їх розробці.

1.Застосування мобільних додатків для громадського транспорту

Щоб підвищити ефективність роботи транспортних компаній, успішно конкурувати в сьогоднішніх умовах, варто застосовувати мобільні додатки, які допоможуть не тільки організувати контакт між усіма учасниками бізнесу, а й надати нові можливості для розширення діяльності і зниження видатків. Від зростання платформ продажу квитків та авіабілетів до мобільних кур'єрських і транспортних систем, мобільні додатки надають можливість бізнесу та споживачам у всьому секторі – так багато, тому багато підприємств у його галузях будуються на основі мобільних транспортних додатків. Для бізнесу розробка додатка для транспорту вигідна тим, що:

- автоматизуються бізнес-процеси;
- знімається навантаження з адміністраторів, знижується гостра потреба в них;
- завдання чіткі і деталізовані, так як більшість полів замовлення (за бажанням всі) – обов'язкові для заповнення;
- з'являється додатковий канал постійної реклами;
- відображаються всі пересування, зупинки, доручення виконавців;
- можна систематизувати доходи, отримувати звіти і виводити статистику, збирати відгуки і оцінки клієнтів щодо якості роботи виконавців;
- продукт забезпечує додатковий прибуток за рахунок монетизації.

Перевізники громадського транспорту мають операційні центри, які контролюють положення транспортних засобів, стан дорожнього руху та іншу важливу інформацію, необхідну для ефективного управління системою. Операційні центри відіграють важливу роль, допомагаючи підтримувати графіки та надійний рух автомобіля. При виникненні порушень співробітники операційного центру надають інструкції водіям, призначеним для мінімізації впливу. Важливо швидко усунути незначні негаразди, оскільки коли рух транзиту стає нерегулярним, транспортні засоби починають згуртовуватися, що спричиняє значні затримки та низьку якість обслуговування. Операційні центри включають:

- системи зв'язку - надавати інформацію та інструкції водіям та пасажирам;

● системи відстеження транспортних засобів - надайте інформацію про місцезнаходження всіх транспортних засобів у системі;

● плани подолання системних порушень - включає запасні транспортні засоби, що вводяться в експлуатацію за потреби.

Хоча перевізникам громадського транспорту завжди мали операційні центри, сьогодні обсяг, якість та своєчасність даних значно зросли. Це дозволяє транспортній системі швидше та ефективніше реагувати на порушення. Після того, як дані стануть доступними, ключовою вимогою є складання планів щодо усунення різних типів порушень. Центр громадського транспорту в Цюріху був піонером у розробці всебічних планів подолання проблем. Співробітники операційного центру мають готовий доступ до заздалегідь узгоджених маршрутів об'їзду для всіх автобусних і трамвайних ліній, кілька запасних транспортних засобів розташовані стратегічно в системі, які можуть вступити в експлуатацію у разі поломки. Перевізники громадського транспорту повинні надавати канали своїх даних у режимі реального часу, щоб сторонні розробники могли створювати власні програми розкладу і особливо поєднувати дані з іншими типами даних, створюючи нові інноваційні типи інформації, які могли б допомогти покращити досвід пасажирів у громадському транспорті.

Громадські транспортні агенції генерують великі обсяги даних як частину своєї щоденної діяльності.

●Системи автоматичного розташування транспортних засобів відстежують положення автобусів та поїздів та збирають постійний потік інформації.

●Системи підрахунку пасажирів, встановлені на транзитних транспортних засобах, реєструють, скільки людей сідає і виходить на кожній зупинці.

●Системи збору тарифів та смарт-картки фіксують здійснені поїздки, трансфери та схеми подорожей серед транзитних користувачів.

Інші джерела даних не створюються автоматично, а вводяться працівниками агентства - такі елементи, як відстеження часу, дані про відсутність, випадки безпеки та інші джерела даних про зайнятість, зібрані на місцях. Найбільш поширені тарифи та лічильники пасажирів як найпоширеніші джерела даних. Служби дозволяють цей трансформативний аналіз даних. Кілька учасників згадували продукти вебслужб Amazon (AWS) як найважливіші у забезпеченні кількох функцій, які допомагають розбити великі дані за нижчою вартістю. Хмарні обчислення дозволяють обмінюватися даними по всій організації, забезпечуючи зручний доступ для різних підрозділів. Спеціальні програмні пакети, вбудовані в AWS, дозволяють передбачити моделювання з використанням великих наборів даних. Дані також використовуються для технічного обслуговування та експлуатації центральною системою, де використовують принципи великих даних, є удосконалення та оптимізація операцій та технічного обслуговування. Об'єднання різнорідних джерел даних в одному місці дозволило провести більш детальний аналіз обладнання та стану ремонту. Нові системи, такі як AWS, дозволяють прогнозувати моделювання з використанням інформації про стан активів та результати діяльності. Подаючи інформацію про ефективність в прогностичну модель, можна передбачити поломки шин і вивести автобуси на технічне обслуговування до того, як вони вийдуть з ладу в полі. Цей тип прогнозувальних дій може потенційно значно заощадити. використовується в інших сферах для підвищення ефективності та зменшення витрат: наприклад, для моніторингу, управління та реагування на потенційні прогули операторів. Використовуючи історичні дані про персонал, а також інформацію про погоду та інші події, прогнозна модель надає інформацію про те, де оператори можуть бути відсутніми. Це дозволяє ефективніше розміщувати операторів резервного копіювання та обмежувати перебої в роботі служби. Крім того, додаткові джерела великих даних (ефективність роботи в часі, розташування транспортного засобу) можна поєднати з графіками, щоб показати операторам, як вони працюють, порівняно з однолітками у своєму підрозділі та собі з тижня на тиждень, що призводить до загального поліпшення. застосовувати аналіз великих даних у сферах безпеки. Використання інструментів великих даних для ініціатив у галузі кібербезпеки викликає все більший інтерес у галузі громадського транспорту, оскільки загрози хакерства стають все більш досконалими. Інструменти машинного навчання аналізують різні виміри даних для нагляду за навколишнім середовищем та за користувачами мережі, щоб визначити, чи є відхилення від нормального середовища. як правило, що представляє найбільший інтерес для відділу міліції, існує можливість для інших відомств добре використовувати його.

Дані також можуть зіграти свою роль у підвищенні безпеки операцій громадського транспорту для пасажирів та людей, що знаходяться в зовнішньому середовищі. Дані збираються бортовими системами, такими як датчики запобігання зіткненням та бортові камери, які контролюють вулиці та перехрестя, що використовуються транзитними транспортними засобами. Поєднуючи ці дані з інформацією про відстеження транспортних засобів, агенції можуть визначити проблемні вулиці та перехрестя, де транзитні транспортні засоби стикаються з найбільшою кількістю конфліктів. Це також допомагає агенціям зрозуміти, які ситуації потребують додаткового навчання для операторів. Об'єднавши та очистивши широкий спектр наборів даних, можна створити модель переговорів у режимі реального часу для використання у роботі з партнерами з питань праці. Під час переговорів емулювати роботу наслідків різних пропозицій для різних категорій. Аспект моделі в реальному часі означав, що коли учасники переговорів запропонували змінити контракт, команда могла продемонструвати наслідки цього за столом переговорів, а не витрачати час в офісі за повторним запуском моделі. Дані, що використовуються для обслуговування та планування. Також варто подивитись на використання мобільних додатків для отримання даних, які допоможуть у наданні та плануванні послуг. Додаток збирає інформацію про поїздки та трансфери та навіть може відстежувати поїздки між кількома регіональними перевізниками. Це дозволяє приймати кращі рішення щодо планування послуг. Мандрівники більше не "зникають", коли вони залишають службу одного агентства для іншого. Крім того, дані стільникового телефону можуть бути використані для визначення маршрутів маршруту клієнтів громадського транспорту, щоб допомогти визначити потенційні пункти передачі та з'єднання між фіксованими маршрутами. Приватні партнери будуть важливі, оскільки вони працюють над розробкою нових додатків для збору даних про тарифи.

Щоб полегшити обмін даними та доступ до інформації для користувачів. В рамках співпраці між Google та Портлендом, штат Орегон, агентством громадського транспорту (TriMet) була представлена GTFS (General Transit Feed Specification - це специфікація для надання інформації про розклад громадського транспорту і супутніх геоданих). Щоб не плутати цю специфікацію з розширенням GTFS Realtime, її також називають GTFS Static і Static Transit. За допомогою GTFS транспортні агентства можуть зробити дані доступними для пасажирів і розробників додатків. Завдяки своїй простоті, як невеликі транзитні агенції, так і великі, можуть публікувати свої дані за низькою вартістю. Ця специфікація визначає шість обов'язкових файлів, відокремлених комами (CSV), та сім необов'язкових, загалом 13 у повному наборі даних. Разом вони описують зупинки, маршрути та розклад руху цілої транзитної системи. Ці файли надаються в основному для розробників і можуть розглядатися як таблиці реляційної бази даних. Діаграма, показана на рисунку 1.1, ілюструє різні файли та спосіб їх зв'язку.

2. Розробка мобільних додатків для громадського транспорту

Розробка додатків для мобільних пристроїв - це процес, при якому додатки розробляються для невеликих портативних пристроїв, таких як КПК, смартфони, стільникові телефони. Ці додатки можуть бути встановлені на пристрій в процесі виробництва, завантажені користувачем за допомогою різних платформ для поширення ПО або бути веб-додатками, які обробляються на стороні клієнта (JavaScript) або сервера.

Коли запит клієнта здійснюється через RESTful API (REST — це набір архітектурних обмежень, розробники API можуть реалізувати REST різними способами), він передає представлення про стан ресурсу запитувачу або кінцевій точці. Ця інформація, або представлення, надається в одному з кількох форматів через HTTP: JSON (нотація об'єктів Javascript), HTML, XLT, Python, PHP або звичайним текст. JSON є найбільш популярним форматом файлів, оскільки, незважаючи на свою назву, він не залежить від мови, а також його можна читати як людям, так і машинами.

Слід також мати на увазі, що заголовки та параметри також важливі в методах HTTP-запитів, оскільки вони містять важливу інформацію як метадані запиту, його авторизація, єдиний ідентифікатор ресурсу (URI), кешування, файли cookie та більше. Існують заголовки запиту та відповіді, кожен із власною інформацією з'єднання HTTP та кодами стану. Для того, щоб API вважався RESTful, він повинен відповідати наступним критеріям;

- архітектура клієнт-сервер, що складається з клієнтів, серверів і ресурсів, із запитами, які керуються через HTTP;

- зв'язок клієнт-сервер без стану, що означає, що інформація про клієнта не зберігається між запитами на отримання, і кожен запит є окремим і не підключеним;
- дані з кешуванням, які спрощують взаємодію клієнту-серверу;
- єдиний інтерфейс між компонентами, щоб інформація передавалася у стандартній формі. Для цього необхідно аби:
  - ресурси, що запитуються, були ідентифікованими та відокремленими від представлень, надісланих клієнту;
  - клієнт міг маніпулювати ресурсами за допомогою представлення, яке він отримує, оскільки представлення містить достатньо інформаці, щоб зробити це;
  - самоописні повідомлення, що повертаються клієнту, мали достатньо інформації, щоб описати, що клієнт може з цим робити;
  - гіпертекст/гіпермедіа був доступний, що означає, що після доступу до ресурсу клієнт повинен мати можливість використовувати гіперпосилання, щоб знайти всі інші доступні дії, які можна виконати;
- багатошарова система, яка організує кожен тип серверів (відповідальних за безпеку, балансування навантаження тощо), що мають відношення до отримання інформації по запиту в ієрархії, невидимій для клієнта;
- code-on-demand (опціонально): можливість надсилати код, що можна виконати, із сервера клієнту за запитом, що розширює функціональні можливості клієнта.

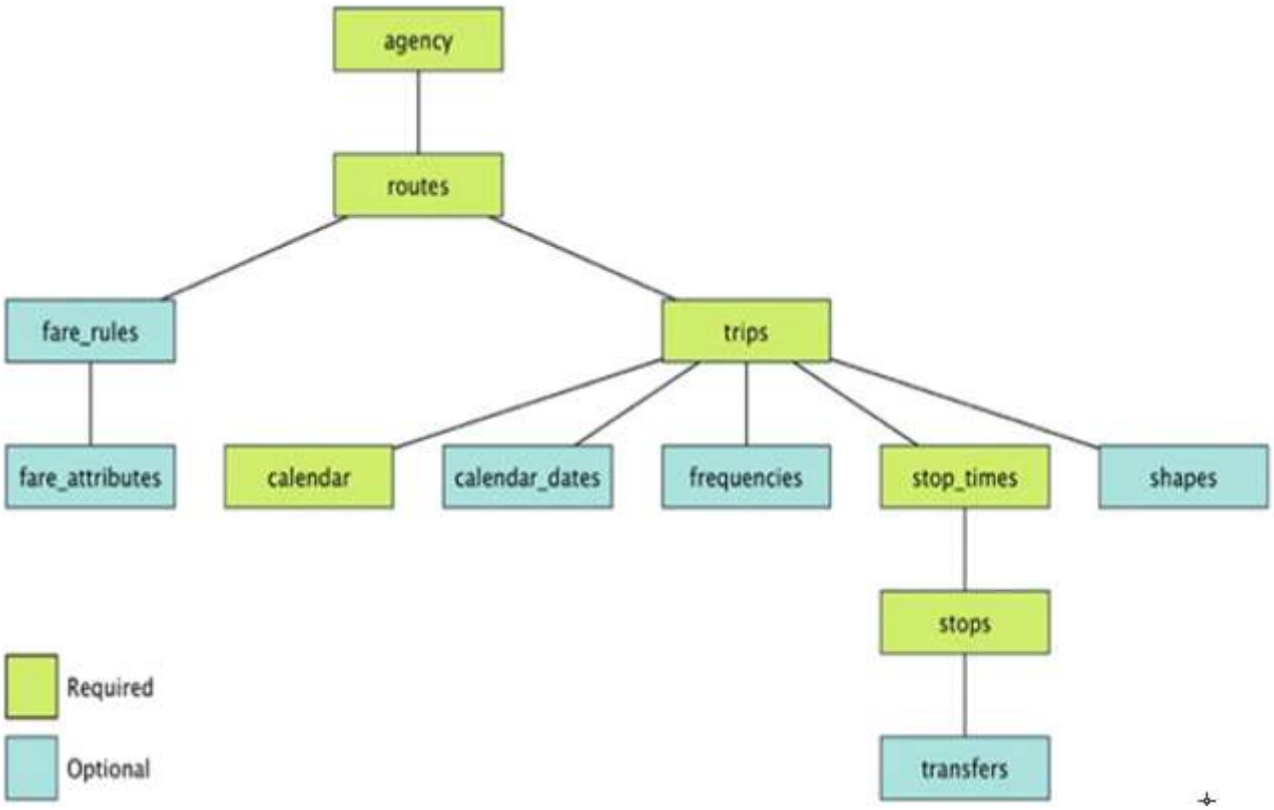

Рисунок 1 – GTFS структурний набір файлів
Figure 1 – GTFS structured file set

Хоча REST API має відповідати усім цим критеріям, він все ще вважається легшим у використанні, ніж, наприклад, протокол SOAP (Протокол доступу до простих об'єктів), який має конкретні вимоги, як-от обмін повідомленнями XML, а також вбудовану безпеку та відповідність транзакціям, які роблять його повільніше і важче.

Дотримання цієї архітектури також спрощується використанням унітарної мови для обох сторін.

Бази даних NoSQL набагато краще підходить для зберігання таких даних, як вміст статті, публікації в соціальних мережах, дані датчиків та інші типи неструктурованих даних, які не впишуться в таблицю. Бази даних NoSQL були побудовані з урахуванням гнучкості та масштабованості, яка відповідає моделі BASE-узгодженості (Basically Available, Soft state, Eventual consistency), що означає:

- Базова доступність – база даних може не отримати запитувані дані, або дані можуть перебувати в стані, що змінюється, або непослідовно.
- М'який стан – стан бази даних може змінюватися з часом.
- Побічна послідовність – база даних з часом стане послідовною, і дані будуть розповсюджуватися скрізь у якийсь момент майбутнього.

Здатність баз даних NoSQL горизонтально масштабувати пов'язана з відсутністю структури даних. Оскільки NoSQL вимагає значно меншої структури, ніж SQL, кожен збережений об'єкт є майже самостійним та незалежним. Таким чином, об'єкти можуть бути легко збережені на декількох серверах без необхідності з'єднання. Посилаючись на статтю ресурсу з програмування (https://www.freecodecamp.org/news/nosql-databases-5f6639ed9574/) визначають 4 типи баз даних NoSQL:

- сховище ключових значень (база даних «ключ-значення») (Key Value) – Riak, Voldemort, Redis;
- графові бази даних – Neo4J, HyperGraphDB;
- сховище з широким стовпчиком – Cassandra, HBase;
- бази даних документів – MongoDB;

MongoDB – кросплатформна документо-орієнтована система керування базами даних. Являється NoSQL базою даних, тобто має механізм зберігання та маніпулювання даними, який відрізняється від функціоналу релятивних баз даних.

Зображення карт в Google Maps та API JavaScript Maps розбиваються на "плитки" та "рівні масштабування". При низькому рівні масштабування невеликий набір плиток карти охоплює широку область; при вищому рівні масштабування плитки мають вищу роздільну здатність і покривають меншу площу. У наведеному нижче списку показано приблизний рівень деталізації, який можливо побачити на кожному рівні масштабування:

1: Світ
5: Материк / континент
10: Місто
15: Вулиці
20: Будинки

Класом JavaScript, що представляє карту, є клас Map. Об'єкти цього класу визначають єдину карту на сторінці. (Можна створити більше одного екземпляра цього класу - кожен об'єкт буде визначати окрему карту на сторінці.)

Також застосовуються служби **Directions**, що дозволяє отримати доступ до маршрутизації автомобіля, їзди на велосипеда, пішки та громадського транспорту за допомогою API маршрутів за допомогою HTTP-запиту. Точки маршруту надають можливість змінити шлях через певне місце. Відправлення, пункти призначення та маршрутні точки вказуються у вигляді текстових рядків (наприклад, "Chicago, IL" or "Darwin, NT, Australia") або як координати широти/довготи.

Та **Geolocation**, що допомогаю знайти місцезнаходження та радіус точності на основі інформації з таких речей, як вежі стільникового зв’язку та точки доступу Wi-Fi. Це в першу чергу використовується там, де GPS неможливий або не підходить.

Node, Express та MongoDB доволі потужний технічний стек, який дозволяє побудувати додаток з урахуванням понять REST. Завдяки ним у розробника є все необхідне для розробки. Node.js виступає як база серверної частини, через його оболонку код нативного JavaScript виконується поза браузером, що власне і дозволяє використовувати його для backend. Але чистий JavaScrip, особливо для серверної сторони, місцями може бути доволі незручний, наприклад, для маршрутування користувача по сайту потрібно прийняти URL-шлях, який опрацьовується методом GET, але за час користування переходів по сторінках може бути багато і писати для кожного такого переходу новий метод не найкраща практика. Для вирішення такої проблеми є фреймворк Express, який не лише оптимізує маршрутизацію (об’єкт Route), а й узагалі поліпшує створення серверу та впровадження бізнес-логіки.

Порівнявши вище види баз даних можна побачити переваги СУБД на основі NoSQL – відсутність дещо комплексних зв'язків, схем, простіша структура зберігання даних, можливість виконати запит однією строчкою, коли у SQL БД для нього знадобилось б 20. Серед представників таких БД було розглянуто MongoDB через її тип – документо орієнтований. Через специфіку завдання проекту він найбільш відповідний – інформацію про геолокацію, маршрути та користувачів зручніше зберігати як документ з полями.

Взаємодіяти з базою даних через сервер клієнт зможе завдяки JavaScript, що використовується для наповнення веб-сторінки «життям». Завдяки скриптам, написаним на цій мові, події на сторінці можуть бути опрацьовані та мати явний для користувача результат, дані можуть бути передані до серверу, який в свою чергу, провівши розрахунки, вибере результат з бази даних та поверне стороні клієнта, аби JavaScript через розмітку сторінки вивів користувачу бажане.

Для того, щоб користувач залишився задоволений не лише функціональним результатом, а й узагалі процесом роботи з ресурсом, рекомендовано впроваджувати зрозумілий, доступний, адаптивний інтерфейс за допомогою семантичного використання елементів HTML та креативного стилізування CSS.

Висновки

1. В роботі на етапі аналізу предметної області було досліджено предметну область, виявлені напрямки, методи та вплив на розвиток транспортних систем.
2. На етапі проектування програмної системи вивчено принципи роботи та архітектуру додатку. Вивчено інтерфейс серверної частини та способів інтеграції.
3. Проведено детальний аналіз існуючих ресурсів-аналогів. На основі проведеного аналізу визначено переваги та обґрунтовано використання вибраних засобів.
4. Було розглянуто вимоги до сучасного веб-ресурсу та допоміжні засоби розробки. У процесі розробки використовувалися кілька пакетів та бібліотек для середовища Node.js, фреймворк express, хмарний сервіс для бази даних MongoDB, служби Google Maps.

Визначено функціональні можливості розроблюваного додатку. При створенні системи враховано функціональні вимоги.

Перелік посилань

1. Google Maps Platform [Електронний ресурс] : The Directions API quickstart. – Режим доступу: https://developers.google.com/maps/documentation/directions/quickstart?hl=en_US
2. Netguru.com [Електронний ресурс] : Use Node.js backend. – Режим доступу: https://www.netguru.com/blog/use-node-js-backend
3. Webplatform.githab.io [Електронний ресурс] : The purpose of JavaScript. – Режим доступу: https://webplatform.github.io/docs/concepts/programming/the_purpose_of_javasc ript/
4. MDN Web Docs [Електронний ресурс] : Web technology for developers. – Режим доступу: https://developer.mozilla.org/en-US/docs/Web
5. w3school.com [Електронний ресурс] : JavaScript Tutorial. – Режим доступу: https://www.w3schools.com/js/default.asp
6. npmjs.com [Електронний ресурс] : CLI commands npm-init. – Режим доступу: https://docs.npmjs.com/cli/init.html
7. Express [Електронний ресурс] : 5.x API. – Режим доступу: http://expressjs.com/en/5x/api.html
8. CSS-TRICKS [Електронний ресурс] : A complete Guide to Flexbox. – Режим доступу: https://css-tricks.com/snippets/css/a-guide-to-flexbox/
9. METANIT.com [Електронний ресурс] : JSON и AJAX. – Режим доступу: https://metanit.com/web/nodejs/4.6.php
10. bem.info [Електронний ресурс] : Методология. Документация. – Режим доступу: https://ru.bem.info/methodology/key-concepts/
11. w3c [Електронний ресурс] : HTML 5.3 – Режим доступу: https://www.w3.org/TR/html53/
12. w3c [Електронний ресурс] : Node.js MongoDB. – Режим доступу: https://www.w3schools.com/nodejs/nodejs_mongodb.asp
13. Google Maps Platform [Електронний ресурс] : Maps JavaScript API. Overwiev – Режим доступу: https://developers.google.com/maps/documentation/javascript/overview
14. MongoDB [Електронний ресурс] : Docs Home. Develop Applications. MongoDB

Drivers. Node.js. Retrieve Data. – Режим доступу: https://docs.mongodb.com/drivers/node/current/fundamentals/crud/read-operations/retrieve/

15. Google Maps Platform [Електронний ресурс] : Distance Matrix API. Overwiev – Режим доступу: https://developers.google.com/maps/documentation/distance-matrix/overview

16. StackOverflow [Електронний ресурс] : Public. Questions. Calculate distance between two points in google maps V3 – Режим доступу: https://stackoverflow.com/questions/1502590/calculate-distance-between-two-points-in-google-maps-v3

17. Subbu Allamaraju RESTful Web Services Cookbook / Subbu Allamaraju// O'Reilly Media / Yahoo Press. – 2010. – 316.

18. https://developers.google.com/transit/gtfs?hl=ru

19. C.J. DateAn Introduction to Database / C.J. DateAn // – Pearson – 2003.

APPLICATION AND DEVELOPMENT OF MOBILE APPLICATIONS FOR PUBLIC TRANSPORT

Azizov Ruslan T., National Transport University, graduate student of the Department of Information Systems and Technologies, azizov1916@gmail.com
Dikhtyarenko Volodymyr V., National Transport University, graduate student of the Department of Information Systems and Technologies, dikhtiarenko0@gmail.com
Boretskyi Vladyslav V., National Transport University, graduate student of the Department of Information Systems and Technologies, vladlenius88@gmail.com, https://orcid.org/0000-0001-5525-4604
Kutsenko Alexander I., National Transport University, graduate student of the Department of Information Systems and Technologies, alexkutsenko95@gmail.com , https://orcid.org/0000-0003-0047-4874

Abstract. The work is devoted to the analysis of the use and development of mobile applications for public transport. The emergence of mobile applications in the transport and logistics industry can be divided into the following types: route optimization and planning, online booking and tracking, parking information, transport and fleet management, communication channel, transportation/delivery schedule optimization. Modern mobile apps provide several built-in benefits such as speed, efficiency and a bug-free environment. The constant rapid development of mobile app development for transportation and logistics is likely to continue to accelerate in an increasingly saturated market in the coming years.

Key words: web application, public transport, NoSQL, client-server architecture.

.

Referens

1. https://developers.google.com/maps/documentation/directions/quickstart?hl=en_US
2. https://www.netguru.com/blog/use-node-js-backend
3. https://webplatform.github.io/docs/concepts/programming/the_purpose_of_javasc ript/
4. https://developer.mozilla.org/en-US/docs/Web
5. https://www.w3schools.com/js/default.asp
6. https://docs.npmjs.com/cli/init.html
7. http://expressjs.com/en/5x/api.html
8. https://css-tricks.com/snippets/css/a-guide-to-flexbox/
9. https://metanit.com/web/nodejs/4.6.php
10. https://ru.bem.info/methodology/key-concepts/
11. https://www.w3.org/TR/html53/
12. https://www.w3schools.com/nodejs/nodejs_mongodb.asp
13. https://developers.google.com/maps/documentation/javascript/overview
14. https://docs.mongodb.com/drivers/node/current/fundamentals/crud/read-operations/retrieve/
15. https://developers.google.com/maps/documentation/distance-matrix/overview
16. https://stackoverflow.com/questions/1502590/calculate-distance-between-two-points-in-google-maps-v3
17. Subbu Allamaraju RESTful Web Services Cookbook / Subbu Allamaraju// O'Reilly Media / Yahoo Press. – 2010. – 316.
18. https://developers.google.com/transit/gtfs?hl=ru

C.J. DateAn Introduction to Database / C.J. DateAn // – Pearson – 2003