DOI 10.31891/2307-5732-2024-343-6-34
УДК 004.9

ПОСТРЕЛКО ЄВГЕНІЙ
Київський державний торговельно-економічний університет
https://orcid.org/0000-0001-9730-450X
e-mail: y.postrelko@knute.edu.ua

ПРОЕКТУВАННЯ ВЕБ-СИСТЕМИ КОНТРОЛЮ РУХУ ГРОМАДСЬКОГО ТРАНСПОРТУ

У статті представлено концепцію та обґрунтовано необхідність створення веб-системи для моніторингу та управління рухом громадського транспорту в Києві. Розробка такої системи дозволить автоматизувати процеси контролю та збору даних з GPS-трекерів, моніторингу запізнень, відхилень від маршруту та зупинок поза встановленими пунктами. Запропонована веб-система орієнтована на використання транспортними департаментами для підвищення ефективності міського транспорту, забезпечення стабільного графіку руху та покращення якості послуг для пасажирів.

Ключові слова:веб-система, моніторинг громадського транспорту, дашборди, контроль руху, транспортна аналітика, GPS, бізнес-аналітика

POSTRELKO YEVHENII
State University of Trade and Economics

DESIGN OF THE PUBLIC TRANSPORT TRAFFIC CONTROL WEB SYSTEM

This article presents a comprehensive approach to the design and implementation of a web-based monitoring system specifically developed for managing and enhancing public transportation services within Kyiv. The system is designed to automate and streamline data collection from GPS-equipped public transport vehicles, allowing for continuous monitoring of key movement aspects, including route adherence, delays, unscheduled stops, and overall performance. One of the primary objectives is to support transportation departments in increasing operational efficiency by facilitating data-driven decision-making and enabling rapid responses to schedule disruptions and incidents on the road. A significant focus of the proposed solution is its intuitive user interface, which provides transport operators with access to critical data through interactive dashboards. These dashboards, powered by the Metabase business intelligence (BI) platform, allow users to dynamically visualize transport movement and track real-time data, including vehicle locations, delay patterns, and operational metrics. Metabase's flexibility enables users to customize dashboards for specific routes, stops, or time intervals, enhancing analytical insights and supporting a range of operational and strategic needs. By integrating Metabase, the web system provides structured, visually digestible information, improving transparency and enabling coordinated actions across different transport departments. The proposed web system offers substantial potential for optimizing public transportation in Kyiv. Through continuous monitoring of delays, route deviations, and schedule adherence, it can help reduce passenger wait times, increase service reliability, and improve overall service quality. Additionally, the system's data-driven approach supports long-term planning and scheduling optimizations, identifying recurring operational issues and enhancing transport authorities' ability to achieve a modernized, efficient, and user-centric public transit system that meets the demands of an evolving urban landscape.

Keywords: web system, public transport monitoring, dashboards, traffic control, transportation analytics, GPS, business intelligence

Постановка проблеми

Поточна ситуація з громадським транспортом у Києві характеризується серйозними викликами, що перешкоджають його стабільній та ефективній роботі. Нерегулярність графіків, постійні затори та технічні проблеми створюють атмосферу непередбачуваності, яка негативно впливає на пасажирів. Це підриває довіру громадян до системи громадського транспорту та ускладнює процес управління міською транспортною інфраструктурою. Без належного контролю та моніторингу стає складно приймати обґрунтовані рішення щодо оптимізації маршрутів та розкладів.

Аналіз останніх досліджень і публікацій

Дослідження у сфері проєктування веб-систем для аналітики руху громадського транспорту та контролю дотримання розкладу є численними та свідчать про зростаючий інтерес до цієї теми. Наприклад, у роботі Ваткінса та співавторів (2011) було досліджено вплив системи реального часу OneBusAway на задоволеність пасажирів та їх використання громадського транспорту [1]. Дослідження показало, що надання точної інформації в реальному часі підвищує довіру пасажирів до транспорту, зменшує відчуття часу очікування та сприяє збільшенню кількості користувачів громадського транспорту.

Інше дослідження, проведене Барчі та співавторами (2015), розглядало використання великих даних та Інтернету речей для управління системами громадського транспорту [2]. Вони розробили веб-платформу, яка дозволяє в режимі реального часу моніторити рух транспорту, аналізувати відхилення від розкладу та оптимізувати маршрути. Результати показали, що така система може значно покращити дотримання розкладу, зменшити затримки та підвищити ефективність роботи транспортної мережі.

Ще одне дослідження, проведене Ченом та співавторами (2016), зосереджувалося на використанні великих даних для аналізу поведінки пасажирів і оптимізації роботи громадського транспорту [3]. Вони розробили систему, яка збирає та обробляє дані з різних джерел, включаючи мобільні пристрої користувачів та датчики в транспортних засобах, для виявлення патернів мобільності та прогнозування попиту. Результати їхнього дослідження показали, що інтеграція великих даних у систему управління громадським транспортом дозволяє покращити планування маршрутів, підвищити точність розкладів і загальну ефективність транспортної мережі.

Незважаючи на цінні внески цих досліджень, певні проблеми залишаються невирішеними. Зокрема, попередні роботи зосереджуються на покращенні задоволеності пасажирів через надання інформації в

реальному часі та оптимізації маршрутів за допомогою великих даних та Інтернету речей. Проте вони не розглядають питання невідповідності руху транспорту до розкладу та інші проблеми, через які відслідковування транспорту на маршруті ускладнюється. Крім того, недостатньо уваги приділено розробці систем, які дозволяють відслідковувати ці аномалії та надавати зібрану інформацію у зручному для транспортних департаментів вигляді.

У своїй статті я пропоную вирішення цих проблем шляхом розробки програмного забезпечення, яке може відслідковувати відповідність часу прибуття транспорту на зупинки ,фіксувати зникнення транспорту з маршруту через втрату GPS-сигналу та інші нетипові поведінки транспорту на маршруті. Також передбачено відображення зібраної інформації у вигляді дашбордів, що дозволить оперативно аналізувати дані та приймати обґрунтовані рішення. Таким чином, моя робота заповнює прогалину в існуючих дослідженнях, пропонуючи практичне рішення для підвищення ефективності та надійності роботи громадського транспорту.

Формулювання цілей статті

Метою даної статті є представлення концепції та обґрунтування необхідності створення веб-системи для моніторингу та управління рухом громадського транспорту в Києві. Основна мета полягає у демонстрації того, як інноваційний підхід до моніторингу та управління може привести до позитивних змін у транспортній інфраструктурі міста.

Виклад основного матеріалу

Велике місто, як правило, має розвинуту систему маршрутів громадського транспорту. За даними сайту "Київпастранс" [4], станом на 30 липня 2024 року в місті Києві функціонує 92 автобусні маршрути, 22 трамвайні та 44 тролейбусні маршрути. На кожному з цих маршрутів одночасно працює декілька одиниць рухомого складу. Окрім цього, в Києві також діє значна кількість маршрутних таксі, які не є комунальним транспортом і не мають встановленого розкладу. Маршрутки становлять левову частку громадського транспорту в місті; їх рух не зупиняється навіть під час повітряної тривоги, що додає певну гнучкість для пасажирів, але також ускладнює контроль за їхньою роботою.

Хоча розклад руху громадського транспорту існує, у Києві він не завжди відповідає дійсності. Причин цього декілька: відсутність GPS-навігаторів на деяких транспортних засобах ускладнює точний контроль за їх рухом, проблема з заторами на дорогах міста постійно впливає на дотримання розкладу, а недотримання графіків водіями також додає непередбачуваності. Всі ці фактори роблять поїздки на громадському транспорті менш передбачуваними, що ускладнює планування подорожей для мешканців і гостей столиці.

Затори на дорогах не лише спричиняють затримки у русі транспорту, але й мають серйозний негативний вплив на екологію міста. Під час простою в заторах автомобілі продовжують працювати на холостому ходу, викидаючи в атмосферу значні обсяги шкідливих речовин, таких як оксиди азоту, вуглекислий газ та тверді частинки [5]. За даними Всесвітньої організації охорони здоров'я, забруднення повітря в містах є одним із головних факторів ризику для здоров'я людей [6]. Тому скорочення кількості заторів та покращення пропускної спроможності доріг може суттєво зменшити рівень забруднення повітря.

Стимулювання громадян до використання громадського транспорту замість особистих автомобілів є одним із ефективних способів досягнення цієї мети. Наявність надійного та передбачуваного громадського транспорту спонукає водіїв переосмислити свої транспортні звички. Якщо громадський транспорт буде ходити за розкладом і забезпечувати комфортні умови перевезення, більше людей будуть готові відмовитися від використання власних автомобілів. Це, у свою чергу, призведе до зменшення кількості автомобілів на дорогах, скорочення заторів та покращення екологічної ситуації в місті.

Ефективним рішенням цієї проблеми може стати розробка веб-системи для моніторингу та управління рухом громадського транспорту. Така система дозволить автоматизувати процес збору і аналізу даних, що значно зменшить потребу в людських ресурсах для моніторингу. Крім того, вона може інтегрувати дані з різних джерел, включаючи GPS-навігацію та інформацію про дорожній рух, що дозволить не лише точно відстежувати рух транспорту, але й оперативно реагувати на зміни ситуації на дорогах. Це сприятиме підвищенню ефективності роботи громадського транспорту та покращенню якості послуг для пасажирів.

Впровадження такої системи може стати ключовим кроком у вирішенні проблеми заторів та забруднення повітря в Києві. Зробивши громадський транспорт більш надійним і зручним, ми можемо спонукати більше людей користуватися ним, тим самим зменшуючи навантаження на дорожню мережу та покращуючи екологічну ситуацію. Це відповідає стратегіям сталого розвитку міста та сприяє підвищенню якості життя мешканців.

Які функції має виконувати така веб-система? Можна виділити наступні чотири критерії:

1. Відслідковувати зникнення та повторну появу транспорту на маршруті
2. Відслідковувати зупинки руху поза межами зупинок громадського транспорту
3. Відслідковувати зникнення транспорту з маршруту, яке може відбуватись через поломку транспорту чи GPS-трекера, втрату сигналу тощо
4. Відслідковувати відповідність графіку руху громадського транспорту та фактичну появу транспорту на зупинках

Ця веб-система призначена для використання всередині транспортних департаментів та інших відповідальних установ, що займаються моніторингом і управлінням рухом громадського транспорту. Вона

не розрахована на використання пасажирами, оскільки її основна мета – забезпечення ефективного контролю за роботою транспорту, аналіз його руху та прийняття рішень на основі зібраних даних.

Першим кроком проектування системи буде отримання API, яке надаватиме інформацію про маршрути, транспорт на них та його розміщення. Маючи таку інформацію та координати зупинок, які є на вибраному маршруті, ми можемо відслідковувати всі ці критерії, що були наведені вище. Для написання програмного коду який буде виконувати ці функції використовуватиметься фреймоворк мови програмування PHP Laravel. Для початку створимо моделі Route які будуть брати участь у цьому процесі (рис. 1).

```php
<?php

class Route extends Model
{
    protected $table = 'public_transport_routes';

    protected $fillable = [
        'number',
        'type',
        'with_tracking',
    ];

    public function stops(): BelongsToMany
    {
        return $this->belongsToMany(Stop::class);
    }

    public function delays(): HasMany{...}

    public function publicTransports(): HasMany
    {
        return $this->hasMany(PublicTransport::class, 'route_id', 'id');
    }

    public function activeDelays(): HasMany
    {
        return $this->delays()->whereNull('to');
    }

    public function hasActiveDelay(): bool
    {
        return $this->activeDelays()->count();
    }

    public static function getFirstAndLastScheduleTime(): array
    {
        return [
            'first' => Carbon::today()->startOfDay()->addHours(5)->addMinutes(0),
            'last'  => Carbon::today()->startOfDay()->addHours(22)->addMinutes(25),
        ];
    }
}
```

Рис 1. Модель Route

Модель Route зберігає інформацію про номер маршрута (number), його тип (type) та чи є функція відстежування на цьому маршруті (with tracking). Він має метод, що повертає час першої та останньої зупинки на маршруті, а також метод що визначає чи є активне запізнення на маршруті. Ця модель містить залежності із зупинками (stops), запізненнями(delays), рухомим складом(publicTransports).

Модель PublicTransport (рис. 2) зберігає інформацію про маршрут (route_id), унікальний ідентифікатор рухомого складу (trans_id), координати рухомого складу у вигляді широти та довготи (latitude та longitude відповідно). Мін містить метод що визначає чи знаходиться транспорт у русі (isMoving), чи вважається транспорт наразі нерухомим (hasActiveImmovable), чи вважається транспорт наразі зниклим (hasActiveDisappearance). Він має залежності від маршруту (route), зникнення (disappearances), нерухомого транспорту (immovables).

```php
<?php

class PublicTransport extends Model
{
    protected $table = 'public_transport';

    protected $fillable = [
        'route_id',
        'trans_id',
        'latitude',
        'longitude',
    ];

    public function route(): BelongsTo
    {
        return $this->belongsTo(Route::class, 'route_id', 'id');
    }

    public function disappearances(): HasMany
    {
        return $this->hasMany(TransportDisappearance::class, 'public_transport_id', 'id');
    }

    public function immovables(): HasMany
    {
        return $this->hasMany(TransportImmovable::class, 'public_transport_id', 'id');
    }

    public function activeDisappearances(): HasMany
    {
        return $this->disappearances()->whereNull('appeared_at');
    }

    public function hasActiveDisappearance(): bool
    {
        return $this->activeDisappearances()->count();
    }

    public function activeImmovables(): HasMany
    {
        return $this->immovables()->whereNull('to');
    }

    public function hasActiveImmovable(): bool
    {
        return $this->activeImmovables()->count();
    }

    public function isMoving(float $lat, float $lng): bool
    {
        return $this->latitude != $lat && $this->longitude != $lng;
    }
}
```

Рис 2. Модель PublicTransport

Модель TransportItemsImmovable (рис. 3) зберігає інформацію про рухомий склад який не рухається (public_transport_item_id), час зупинки (from), час відновлення руху (to), координати (lat та lng). Ця модель має залежність з моделлю рухомого складу (рис. 2) та зберігає інформацію про зупинки транспорту на маршруті.

```php
<?php

class TransportItemsImmovable extends Model
{
    protected $table = 'public_transport_items_immovable';

    protected $fillable = [
        'public_transport_item_id',
        'from',
        'to',
        'lat',
        'lng',
    ];

    public function publicTransportItem(): BelongsTo
    {
        return $this->belongsTo(PublicTransportItem::class);
    }
}
```

Рис. 3. Модель TransportItemsImmovable

Модель TransportItemsDisappearance (рис. 4) зберігає інформацію про рухомий склад який зник з маршруту (public_transport_item_id), координати зникнення (disappeared_lat та disappeared_lng), час зникнення (disappeared_at), координати появи після зникнення (appeared_lat та appeared_lng) та час появи (appeared_at). Дана модель має залежність з моделлю рухомого складу (рис. 2) та зберігає інформацію про зникнення транспорту з маршруту.

```php
<?php

class TransportItemsDisappearance extends Model
{
    protected $table = 'transport_items_disappearance';

    protected $fillable = [
        'public_transport_item_id',
        'disappeared_at',
        'disappeared_lat',
        'disappeared_lng',
        'appeared_at',
        'appeared_lat',
        'appeared_lng',
    ];

    public function transportItem(): BelongsTo
    {
        return $this->belongsTo(TransportItem::class, 'transport_item_id', 'id');
    }
}
```

Рис. 4. Модель TransportItemsDisappearance

Модель Stop (рис. 5) зберігає інформацію про зупинки громадського транспорту, а саме їх назву та координати. Має залежність з моделлю розкладу.

```php
<?php

class Stop extends Model
{
    protected $table = 'stops';

    protected $fillable = [
        'name',
        'latitude',
        'longitude',
    ];

    public function schedules()
    {
        return $this->hasMany(Schedule::class);
    }
}
```

Рис. 5. Модель Stop

Модель Schedule (рис. 6) зберігає інформацію про розклад прибуття громадського транспорту на зупинку та має залежність із маршрутом та зупинкою.

```php
<?php

class Schedule extends Model
{
    protected $table = 'schedules';

    protected $fillable = [
        'route_id',
        'stop_id',
        'arrival_time',
    ];

    public function route()
    {
        return $this->belongsTo(Route::class, 'route_id', 'id');
    }

    public function stop()
    {
        return $this->belongsTo(Stop::class, 'stop_id', 'id');
    }
}
```

Рис. 6. Модель Schedule

Модель RouteDelay (рис. 7) зберігає інформацію про запізнення транспорту на маршруті та має такі поля: ідентифікатор маршруту (route_id), ідентифікатор зупинки (stop_id), ідентифікатор транспорту (transport_id), час початку запізнення (delay_start), час закінчення запізнення (arrival_time). Також має залежність з моделлю маршрутів (рис. 1), моделлю транспорту (рис. 2) та моделлю зупинок (рис. 5).

Розібравшись зі моделями, які будуть брати участь у процесах, перейдемо до створення аналітичного програмного коду. Створимо клас для виконання цієї роботи.

```php
<?php

class RouteDelay extends Model
{
    protected $table = 'transport_delays';

    protected $fillable = [
        'stop_id',
        'route_id',
        'transport_id',
        'delay_start',
        'arrival_time',
    ];

    public function stop()
    {
        return $this->belongsTo(Stop::class, 'stop_id', 'id');
    }

    public function route()
    {
        return $this->belongsTo(Route::class, 'route_id', 'id');
    }

    public function transport()
    {
        return $this->belongsTo(TransportItem::class, 'transport_id', 'id');
    }
}
```

Рис. 7. Модель RouteDelay

Клас RouteDelay (рис. 8) приймає перелік діючих маршрутів з переліком транспорту на ньому та інформацією про нього. Однією з властивостей класу є перелік маршрутів які відслідковуються. Також є можливість передати час для якого збирати аналітику, за замовчуванням це час виклику класу. Головною функцією буде execute(), її будуть викликати для запуску всього процесу. У ній перевіряється чи запущена функція в час роботи роботи маршруту. Якщо так то запускається збір аналітики, якщо ні то закривається збір аналітики по всьому транспорту на маршруті.

```php
<?php

class PublicTransportMonitor
{
    public Carbon $current_time;
    public string $current_datetime;
    public array $kyiv_routes;
    public Collection $monitored_routes;

    function __construct(array $kyiv_routes, Carbon $current_time = null)
    {
        $this->current_time = $current_time ?? Carbon::now();
        $this->current_datetime = $current_time ? $current_time->format('Y-m-d H:i:s') : Carbon::now()->format('Y-m-d H:i:s');
        $this->kyiv_routes = $kyiv_routes;
        $this->monitored_routes = Route::where('with_tracking', true)
            ->with('delays', 'transportItems', 'transportItems.disappearance', 'transportItems.immovable')
            ->get();
    }

    public function execute()
    {
        DB::transaction(function () {
            if (
                $this->current_time->gt(Route::getFirstAndLastScheduleTime()['first'])
                && $this->current_time->lt(Route::getFirstAndLastScheduleTime()['last'])
            ) {
                $this->manageTransports();
            } else {
                $this->closeAllActivities();
            }
        });
    }
```

Рис. 8. Початок класу PublicTransportMonitor

У функції manageTransports() (рис. 9) відбувається порівняння вже існуючих у базі даних записах про транспорт з тими, що надійшли по API. Ті рухомі склади що є у переліку з API і яких немає в базі даних будуть додані в базу даних. Для них будуть записані актуальні координати, ідентифікатор маршруту та ідентифікатор транспорту (рис. 8).

```php
protected function manageTransports()
{
    foreach ($this->monitored_routes as $route) {
        if (in_array($route->id, array_keys($this->kyiv_routes))) {
            $route_transport_ids = $route->transportItems->pluck('kyiv_id')->toArray();

            $new_transport_ids = array_diff(array_keys($this->kyiv_routes[$route->id]), $route_transport_ids);
            $disappeared_transport_ids = array_diff($route_transport_ids, array_keys($this->kyiv_routes[$route->id]));
            $existing_transports_ids = array_intersect($route_transport_ids, array_keys($this->kyiv_routes[$route->id]));

            $this->addNewTransport($route->id, $new_transport_ids);
            $this->logDisappearedTransport($disappeared_transport_ids, $route);
            $this->updateExistingTransport($existing_transports_ids, $route);
            $this->logImmovableTransport($existing_transports_ids, $route);
        }
    }
}

protected function addNewTransport(int $route_id, array $new_transport_ids)
{
    foreach ($new_transport_ids as $kyiv_id) {
        TransportItem::create([
            'route_id' => $route_id,
            'kyiv_id'  => $kyiv_id,
            'lat'      => $this->kyiv_routes[$route_id][$kyiv_id]['lat'],
            'lng'      => $this->kyiv_routes[$route_id][$kyiv_id]['lng'],
        ]);
    }
}
```

Рис. 9. Функції запуску аналітики та збереження нового транспорту

Транспорт, який вже збережений у базі даних та який не надійшов по API буде вважатись таким що зник та буде записаний у базу даних як зниклий (рис. 10). В базу даних буде записана інформація про місце та час зникнення транспорту. Інформація про транспорт що є і в базі даних і який надійшов у переліку з API буде оновлена (рис. 9). Якщо транспорт був зниклим то буде записана інформація про координати його появи та час його появи.

```php
protected function logDisappearedTransport(array $disappeared_transport_ids, Route $route)
{
    $transports = $route->transportItems->whereIn('kyiv_id', $disappeared_transport_ids);

    foreach ($transports as $transport) {
        if (!$transport->hasOpenDisappearance()) {
            TransportItemsDisappearance::create([
                'transport_item_id' => $transport->id,
                'disappeared_at' => $this->current_datetime,
                'disappeared_lat' => $transport->lat,
                'disappeared_lng' => $transport->lng,
            ]);
        }
    }
}

protected function updateExistingTransport(array $existing_transports_ids, Route $route)
{
    $transports = $route->transportItems->whereIn('kyiv_id', $existing_transports_ids);

    foreach ($transports as $transport) {
        $transport_info = $this->kyiv_routes[$route->id][$transport->kyiv_id];

        if ($transport->hasOpenDisappearance()) {
            foreach ($transport->openDisappearance() as $disappearance) {
                $disappearance->appeared_at = $this->current_datetime;
                $disappearance->appeared_lat = $transport_info['lat'];
                $disappearance->appeared_lng = $transport_info['lng'];
                $disappearance->save();
            }
        }

        if ($transport->isMoving($transport_info['lat'], $transport_info['lng'])) {
            $transport->lat = $transport_info['lat'];
            $transport->lng = $transport_info['lng'];
            $transport->save();
        }
    }
}
```

Рис.10. Функції збереження зниклого та оновлення існуючого транспорту

Функція logImmovableTransport (рис. 11) отримує перелік існуючого транспорту в базі даних та API методі. Далі перевіряється чи однакові координати були у транспорта до цієї перевірки і зараз та чи не вважається цей транспорт нерухомим наразі. І якщо ці два критерії негативні то створюється запис про нерухомий стан транспорту, в якому зберігається інформація про час та місце зупинки транспорту. Якщо ж транспорт рухається то відбувається оновлення інформації про його нерухомий стан, а саме додається інформація про час початку руху.

```php
protected function logImmovableTransport(array $existing_transports_ids, Route $route)
{
    $transports = $route->transportItems->whereIn('kyiv_id', $existing_transports_ids);

    foreach ($transports as $transport) {
        $transport_info = $this->kyiv_routes[$route->id][$transport->kyiv_id];

        if (!$transport->isMoving($transport_info['lat'], $transport_info['lng']) && !$transport->hasOpenImmovable()) {
            TransportItemsImmovable::create([
                'transport_item_id' => $transport->id,
                'from' => $this->current_datetime,
                'lat' => $transport_info['lat'],
                'lng' => $transport_info['lng'],
            ]);
        } elseif ($transport->isMoving($transport_info['lat'], $transport_info['lng'])) {
            if ($transport->hasOpenImmovable()) {
                foreach ($transport->openImmovable() as $immovable) {
                    $immovable->to = $this->current_datetime;
                    $immovable->save();
                }
            }
        }
    }
}
```

Рис. 11. Функція запису інформації про нерухомий стан транспорту

Функція checkTransportDelays (рис. 12) перевіряє відповідність між графіком прибуття транспорту на певну зупинку та фактичним місцем розташування транспорту. Спочатку функція отримує всі заплановані прибуття на зупинки для заданого маршруту на поточний час. Для кожної зупинки перевіряється, чи знаходиться якийсь із транспортних засобів на координатах цієї зупинки. Якщо транспортний засіб присутній на координатах зупинки, оновлюється запис про затримку, додаючи час фактичного прибуття. Якщо жоден транспортний засіб не знаходиться на координатах зупинки, створюється новий запис про затримку транспорту, фіксуючи початок затримки.

```php
protected function checkTransportDelays(Route $route)
{
    $schedules = Schedule::where('route_id', $route->id)
        ->whereTime('arrival_time', $this->current_time->format('H:i'))
        ->get();

    foreach ($schedules as $schedule) {
        $stop = $schedule->stop;
        $stop_coords = ['lat' => $stop->latitude, 'lng' => $stop->longitude];
        $matched = false;

        foreach ($this->kyiv_routes[$route->id] as $transport_id => $coords) {
            if ($coords['lat'] == $stop_coords['lat'] && $coords['lng'] == $stop_coords['lng']) {
                $matched = true;

                $delay = RouteDelay::where('stop_id', $stop->id)
                    ->where('route_id', $route->id)
                    ->where('transport_id', $transport_id)
                    ->whereNull('arrival_time')
                    ->first();

                if ($delay) {
                    $delay->arrival_time = $this->current_datetime;
                    $delay->save();
                }
            }
        }

        if (!$matched) {
            RouteDelay::create([
                'stop_id' => $stop->id,
                'route_id' => $route->id,
                'transport_id' => $transport_id,
                'delay_start' => $this->current_datetime,
            ]);
        }
    }
}
```

Рис. 12. Функція запису інформації про запізнення транспорту

Розглянувши те яким чином дані отримуються та записуються в базу даних, перейдемо того як ми можемо обробляти ці дані та отримувати певне уявлення про стан роботи громадського транспорту. Одним із інструментів аналізу та візуалізації даних є системи бізнес-аналітики. Системи бізнес-аналітики (Business Intelligence, BI) є важливими інструментами для прийняття рішень на основі даних [7]. На ринку систем бізнес-аналітики існують багато готових рішень які мають свої переваги та недоліки. Та в даній статі буде розглянуто застосування системи Metabase.

Metabase має відкритий вихідний код, що робить її доступною та гнучкою для налаштувань під специфічні потреби [8]. Вона підтримує широкий спектр баз даних, включаючи MySQL, PostgreSQL, та MongoDB, що дозволяє легко інтегрувати її з нашою базою даних. Metabase також має простий у використанні інтерфейс і потужні інструменти для створення дашбордів та аналітичних звітів. Metabase можна підключити до існуючої бази даних через нативний коннектор. Файл конфігурації визначає підключення до бази даних, яке використовується для отримання даних і побудови звітів. Після налаштування Metabase зчитує структуру бази даних і дозволяє створювати запити та дашборди через веб-інтерфейс.

Перейдемо до прикладів запитів, які допоможуть Metabase отримати інформацію з нашої бази даних. На рисунках 13 та 14 наведені приклади запитів які система бізнес-аналітики може відправляти в базу даних веб-системи для створення звітів та дашбордів. Перший запит повертає інформацію про активний транспорт на маршруті, другий визначає зупинки з найбільшою кількістю затримок прибуття транспорту.

```
SELECT r.number AS route_number, COUNT(pt.id) AS transport_count
FROM public_transport_routes r
JOIN public_transport pt ON r.id = pt.route_id
GROUP BY r.number;
```

Рисунок 13. Запит інформації про активний рухомий склад на маршруті

```
SELECT s.name AS stop_name, COUNT(rd.id) AS delay_count
FROM stops s
JOIN transport_delays rd ON s.id = rd.stop_id
GROUP BY s.name
ORDER BY delay_count DESC;
```

Рис. 14. Запит інформації про зупинки з найбільшою кількістю затримок

Відображення інформації на дашборді надає значні переваги для користувачів, оскільки дозволяє отримувати візуально зрозумілі дані в реальному часі, що сприяє швидшому прийняттю рішень. Дашборди забезпечують інтерактивність, дозволяючи користувачам досліджувати дані глибше та фокусуватися на ключових показниках ефективності. Metabase надає можливість інтегрувати дашборд у вашу веб-систему [8], що дозволяє відображати аналітику безпосередньо у внутрішніх або зовнішніх додатках, створюючи єдину інформаційну платформу.

Підключити Metamask можна за допомогою конфігурації у стилі YAML, як наведено у прикладі (рис. 15). Після підключення Metabase до бази даних та створення дашборду ми можемо використовувати створений нами дашборд у веб-системі.

```
- name: MyDatabase
  engine: mysql
  host: localhost
  port: 3306
  database: my_database
  username: my_user
  password: my_password
```

Рис.15. Приклад підключення Metabase до веб-системи

Одним з найпростіших способів додавання дашборду є використання iframe: *<iframe src="https://your-metabase-url/public/dashboard/your-dashboard-id" frameborder="0" width="800" height="600" allowtransparency></iframe>*

Підключивши Metabase до бази даних, створивши дашборди та підключивши їх до веб-системи ми можемо отримати візуалізацію зібраної раніше статистики прямо у веб-системі, не переходячи у стороні сервіси. дашборд “Кількість транспорту на маршруті” (рис. 16) є візуалізацією запиту з рис. 12, а дашборд “Кількість затримок на зупинках” (рис. 17) є візуалізацією результатів запиту з рис. 13.

Рисунок 16. Приклад дашборду "Кількість транспорту на маршруті"

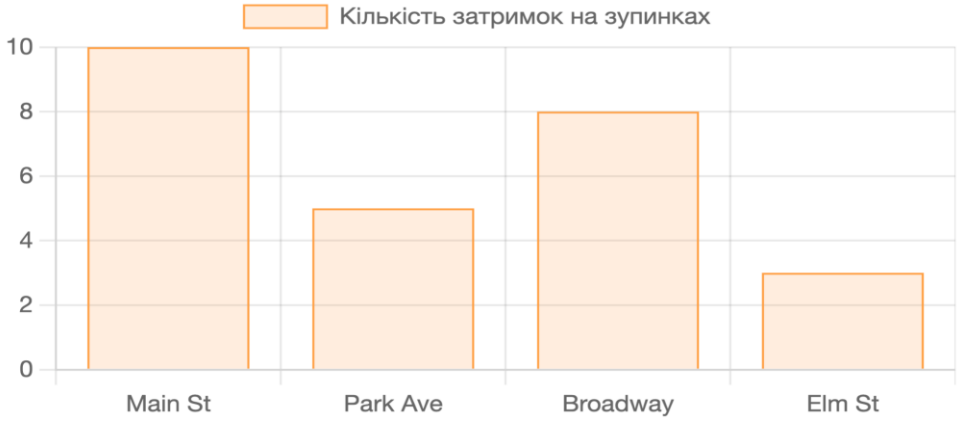

Рис. 17. Приклад дашборду "Кількість затримок на зупинках"

В процесі проектування веб-системи контролю руху громадського транспорту в Києві було закладено основи для створення потужного інструменту, який дозволить ефективно моніторити і управляти транспортними потоками. На початковому етапі було розроблено концепцію та визначено ключові моделі, що відповідатимуть за зберігання та обробку інформації про маршрути, транспортні засоби, їх зупинки, запізнення та інші важливі параметри. Ці моделі слугуватимуть базисом для подальшого розвитку системи, забезпечуючи автоматизацію збору та аналізу даних, що є важливим кроком для підвищення точності і своєчасності надання інформації користувачам.

Таким чином, запроектована система стане важливим інструментом для покращення роботи громадського транспорту, забезпечуючи більш точний контроль, своєчасний аналіз та оперативне реагування на зміни в ситуації на дорогах. Це сприятиме підвищенню якості послуг для пасажирів і покращенню загальної транспортної інфраструктури міста.

Висновки

Проведене дослідження з проєктування веб-системи контролю за рухом громадського транспорту та дотриманням ним розкладу показало значний потенціал для покращення транспортної інфраструктури Києва. Інтеграція з API забезпечує оперативне надходження актуальних даних про транспортні засоби та їх розташування в реальному часі. Це дозволяє системі ефективно моніторити рух транспорту, оперативно виявляти відхилення від графіка та реагувати на них, що є критичним для підвищення якості послуг громадського транспорту.

Використання системи бізнес-аналітики Metabase для аналізу та візуалізації зібраних даних значно спрощує процес обробки інформації. Наочні дашборди, які відображають ключові показники ефективності роботи транспорту в реальному часі, сприяють прийняттю більш обґрунтованих управлінських рішень. Це підвищує загальну ефективність управління транспортом та дозволяє швидко адаптуватися до змінних умов на дорогах.

Запропонована система не лише покращує контроль і моніторинг, але й створює умови для стимулювання мешканців користуватися громадським транспортом замість особистих автомобілів. Підвищення надійності та передбачуваності громадського транспорту може зменшити кількість заторів, покращити екологічну ситуацію та підвищити загальну мобільність населення.

Перспективи подальших досліджень

Подальший розвиток проєкту може включати розширення функціоналу системи, впровадження прогнозних моделей для оптимізації маршрутів та інтеграцію з іншими міськими інформаційними системами.

Це сприятиме створенню комплексної платформи для управління міською транспортною інфраструктурою, яка відповідатиме сучасним вимогам до розумних міст.

Література

1. Watkins, K. E., Ferris, B., Borning, A., Rutherford, G. S., & Layton, D. (2011). Where is my bus? Impact of mobile real-time information on the perceived and actual wait time of transit riders. *Transportation Research Part A: Policy and Practice*, 45(8), 839–848. https://doi.org/10.1016/j.tra.2011.06.010
2. Barchi, G., De Martinis, V., & Ricci, S. (2015). Big Data and Internet of Things for the management of public transportation systems. *Procedia Computer Science*, 52, 17–24. https://doi.org/10.1016/j.procs.2015.05.003
3. Chen, C., Ma, J., Susilo, Y., Liu, Y., & Wang, M. (2016). The promises of big data and small data for travel behavior (aka human mobility) analysis. *Transportation Research Part C: Emerging Technologies*, 68, 285–299. https://doi.org/10.1016/j.trc.2016.04.005
4. Київпастранс. (n.d.). *Розклад руху громадського транспорту*. Отримано з https://kpt.kyiv.ua/schedule
5. Європейське агентство з охорони навколишнього середовища. (2016). *Емісії атмосферних забруднювачів від транспорту*. Отримано з https://www.eea.europa.eu/data-and-maps/indicators/transport-emissions-of-air-pollutants-8
6. Всесвітня організація охорони здоров'я. (2018). *9 з 10 людей у світі дихають забрудненим повітрям, але все більше країн вживають заходів*. Отримано з https://www.who.int/ukraine/news/detail/02-05-2018-9-out-of-10-people-worldwide-breathe-polluted-air-but-more-countries-are-taking-action
7. Dedić, N., & Stanier, C. (2016). Measuring the success of changes to existing business intelligence solutions to improve business intelligence reporting. У P. Ceravolo, J. Sillitti, M. Morisio, & M. García Rodríguez (Ред.), *Lecture Notes in Business Information Processing* (т. 268, с. 225–236). Springer International Publishing. https://doi.org/10.1007/978-3-319-49944-4_17
8. Metabase. *Metabase Documentation*. Отримано з https://www.metabase.com/docs/latest/

References

1. Watkins, K. E., Ferris, B., Borning, A., Rutherford, G. S., & Layton, D. (2011). Where is my bus? Impact of mobile real-time information on the perceived and actual wait time of transit riders. Transportation Research Part A: Policy and Practice, 45(8), 839–848. https://doi.org/10.1016/j.tra.2011.06.010
2. Barchi, G., De Martinis, V., & Ricci, S. (2015). Big Data and Internet of Things for the management of public transportation systems. Procedia Computer Science, 52, 17–24. https://doi.org/10.1016/j.procs.2015.05.003
3. Chen, C., Ma, J., Susilo, Y., Liu, Y., & Wang, M. (2016). The promises of big data and small data for travel behavior (aka human mobility) analysis. Transportation Research Part C: Emerging Technologies, 68, 285–299. https://doi.org/10.1016/j.trc.2016.04.005
4. Kyivpastrans. (n.d.). Rozklad rukhu hromadskoho transportu. Otrymano z https://kpt.kyiv.ua/schedule
5. Ievropeiske ahentstvo z okhorony navkolyshnoho seredovyshcha. (2016). Emisii atmosfernykh zabrudniuvachiv vid transportu. Otrymano z https://www.eea.europa.eu/data-and-maps/indicators/transport-emissions-of-air-pollutants-8
6. Vsesvitnia orhanizatsiia okhorony zdorovia. (2018). 9 z 10 liudei u sviti dykhaiut zabrudnenym povitriam, ale vse bilshe krain vzhyvaiut zakhodiv. Otrymano z https://www.who.int/ukraine/news/detail/02-05-2018-9-out-of-10-people-worldwide-breathe-polluted-air-but-more-countries-are-taking-action
7. Dedić, N., & Stanier, C. (2016). Measuring the success of changes to existing business intelligence solutions to improve business intelligence reporting. U P. Ceravolo, J. Sillitti, M. Morisio, & M. García Rodríguez (Red.), Lecture Notes in Business Information Processing (t. 268, s. 225–236). Springer International Publishing. https://doi.org/10.1007/978-3-319-49944-4_17
8. Metabase. Metabase Documentation. Otrymano z https://www.metabase.com/docs/latest/