УДК 044:338.48

Т.В. СІЧКО, Б.О. ЮРЧУК
Вінницький національний аграрний університет

РОЗРОБКА ВЕБ-ДОДАТКУ ТУРИСТИЧНОЇ ФІРМИ

Під час розробки проекту було реалізовано веб-додаток для туристичної фірми. Метою проекту є розробка веб-додатку для туристичної фірми із застосуванням сучасних методологій і технологій. Актуальність проекту полягає в його адаптації до людей з обмеженими можливостями. Розробка відбувалась повністю на стеку JavaScript технологій. За основу взято нові стандарти мов програмування та розмітки, такі як HTML5, CSS3, ES6(ECMAScript 2015). Розробка серверної частини побудована на технологіях JavaScript, зокрема використана платформа Node.js. Для роботи з великим об'ємом інформації було підключено базу даних. З огляду на стек проекту, було обрано базу типу noSQL. Проект доведено до фінальної стадії розробки та реалізовано повний купівельний цикл. Описано особливості проекту та перспективи для подальшого вдосконалення.

Ключові слова: веб-розробка, веб-додаток, html, css, javascript, node.js, express, mongo DB, noSQL, front-end, back-end.

T.V. SICHKO, B.O. YURCHUK
Vinnytsa National Agriculture University

DEVELOPMENT WEB-APPLICATION FOR TRAVEL COMPANY

A web application for a travel agency was implemented, during the development of the project. The aim of the project is to develop a web-based application for a travel company using modern methodologies and technologies. The urgency of the project is its adaptability for people with disabilities. The development took place entirely on the JavaScript technology stack. New standards for programming and mark-up languages, such as HTML5, CSS3, ES6 (ECMAScript 2015), were adopted as the basis. Development of server-side technology based on JavaScript also, the platform used Node.js. A database connected for work with a large amount of information. Given the project stack, a NoSQL database was selected (databases that implement an unconventional approach to managing the database without using a special SQL language). The project implemented by the component approach (each element of the application can function independently). In application was used SSR strategy (server-side rendering). It helps make our site "CEO-friendly" and does not load the client side of the application logic. The project brought to the final stage of development and a full purchasing cycle implemented. The features of the project and the prospects for further improvement described. The project implemented in the integrated software environment of the IDE WebStorm 2017.3.2. The project has a license "GNU GENERAL PUBLIC LICENSES". Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed. Link to the repository of an implemented web application: https://github.com/tigermax139/Go_Away-Travel-Agency.

Keywords: web development, web application, html, css, JavaScript, node.js, express, mongo DB, NoSQL, application, front-end, back-end, web-development.

Вступ. З появою веб-технологій комп'ютер починають використовувати абсолютно нові верстви населення Землі. Можна виділити дві найбільш характерні групи, що знаходяться на різних соціальних полюсах, які були стрімко залучені в нову технологію, можливо, навіть крім їх власного бажання. З одного боку, це були представники елітарних груп суспільства: керівники крупних організацій, президенти банків, топ-менеджери, впливові державні чиновники і т. ін. З іншого боку, це були представники найширших верств населення: домогосподарки, пенсіонери, діти.

Інтернет зближує, реакція на будь-яку подію поступає практично негайно, а відстані зникають. Веб-додатки істотно зменшують витрати виробника, так само як і розширюють можливості покупця – купувати будь-який товар у будь-якій країні, в будь-якому місті, у будь-який час доби. Цей момент є істотним під час переходу виробників із «звичайної» торгівлі на «електронну» [1].

Традиційний ринок Інтернет-комерції зароджувався як ринок роздрібної торгівлі. Але поступово на ринку стали з'являтися рішення, орієнтовані не на кінцевих споживачів, а на організації, так званий ринок B2B (business-to-business).

Спочатку, на ринку B2B пропонувалися продукти пов'язані власне з організацією роздрібної торгівлі (готові Інтернет-магазини, послуги з реклами, впровадження Інтернет-торгівлі в традиційні бізнес-процеси і тому подібне). Але поступово через Інтернет почали продаватися рішення безпосередньо з Інтернет не зв'язані (комерційне програмне забезпечення, послуги з автоматизації, оптова торгівля, брокерські послуги і т. ін).

Останні роки, мова програмування JavaScript знаходиться у стадії великих змін та інтенсивного розвитку. Спричинене це тим, що мова відноситься до слабко та динамічно типізованих, і тому «поріг входу» вважається значно нижчим ніж у конкурентів. На JavaScript раніше писали прості браузерні сценарії, а зараз на основі цієї мови існує багато Front-end фреймворків, які вже давно розроблено гігантами світу IT-індустрії: Google (Angular), Facebook (React), Yahoo (Ember) та ін. Крім того, компанією Google було створено повноцінну платформу для написання Back-end – Node.js, за допомогою якої ми можемо повністю реалізувати серверну частину сайту. Лише на сайті пакетного менеджера для Node.js - npm є більше 400 тисяч готових модулів та компонентів.

Світ мобільних пристроїв JavaScript теж не оминув. Компанія «Facebook» активно «просуває» бібліотеку React Native, яка створена для написання додатків саме під мобільні пристрої.

Сама ж «екосистема» JavaScript постійно збільшується, за словами Джеффа Етвуда (авторитетний розробник програмного забезпечення): «Будь-який додаток, який може бути написано на JavaScript, в кінцевому підсумку буде написаний на JavaScript».

Мета статті. Метою роботи є розробка веб-додатку для туристичної фірми із застосуванням сучасних методологій і технологій.

Актуальність роботи полягає в адаптації веб-додатку до людей з обмеженими можливостями. Новизною даного проекту можна вважати повну відмову від застарілих технологій типу LAMP (Linux, Apache, mySQL, PHP). Хоч мова програмування PHP, активно бореться за своє існування – попит на неї стабільно зменшується. За показниками рейтингу мов програмування TIOBE – індекс мови програмування PHP впав з 11,4% (2005 рік) до 3.42% (2018 рік), для порівняння JavaScript виріс з 1,5% (2002 рік) до майже 4% (2018 рік). На індекс у великій мірі впливає те, що за статистикою 80% сайтів написані на застарілих технологіях типу PHP, тому поки існують ці проекти – буде необхідність в їх підтримці і в мові програмування, на якій вони написані. Ми не можемо стверджувати, що вона повністю зникне протягом найближчих 5–10 років, але можемо сказати з упевненістю, що «гілку першості» вона втратила і, скоріше всього, вже не поверне. Саме тому написання проекту на лідируючих стрімких технологіях, в основі яких лежить мова програмування JavaScript буде ключовою метою даної роботи.

Виклад основного матеріалу. Платформа з відкритим кодом Node.js створена для виконання високопродуктивних мережевих додатків написаних мовою JavaScript. Засновником платформи є Раян Дал (Ryan Dahl). Платформа node.js перетворила мову JavaScript, що в основному використовувалась в браузерах на мову загального використання з великою спільнотою розробників.

Node.js характеризується такими властивостями:

- асинхронна однопотокова модель виконання запитів;
- неблокуючий ввід/вивід;
- система модулів CommonJS;
- рушій JavaScript Google V8;
- використанням пакетного менеджера npm (node package manager) для керування модулями.

Node.js призначений для відокремленого виконання високопродуктивних мережевих додатків на мові JavaScript. Функції платформи не обмежені створенням серверних скриптів для веб, платформа може використовуватися і для створення звичайних клієнтських і серверних мережевих програм. Для забезпечення виконання JavaScript-коду використовується розроблений компанією Google рушій V8.

Для забезпечення обробки великої кількості паралельних запитів у Node.js використовується асинхронна модель запуску коду, заснована на обробці подій в неблокуючому режимі та визначенні обробників зворотних викликів (callback). У якості способів мультиплексування з'єднань підтримується epoll, kqueue, /dev/poll і select. Для мультиплексування з'єднань використовується бібліотека libuv, для створення пулу потоків (thread pool) задіяна бібліотека libeio, для виконання DNS-запитів у неблокуючому режимі інтегрований c-ares. Всі системні виклики, що спричиняють блокування, виконуються всередині пула потоків і потім, як і обробники сигналів, передають результат своєї роботи назад через неіменовані канали (pipe).

У багатьох людей згадка про "базу даних" відразу викликає асоціації з MySQL, таблицями та SQL-запитами. MySQL, як і Oracle чи Postgresql є реляційною СКБД (системою керування базами даних). Інформація в них зберігається у вигляді зв’язаних таблиць. Кожна таблиця має власну схему (структуру), яка включає назву таблиці та набір полів (атрибутів, колонок). Сукупність усіх таблиць становить схему реляційної бази даних.

Чим складніша схема бази, тим складнішими стають запити на отримання інформації. Виконання складних запитів збільшує навантаження на сервер бази даних. Наразі набирають розповсюдження бази даних, які не мають схеми NoSQL БД. Збереження даних у таких базах є значно менш затратним. NoSQL розшифровується як Not Only SQL.

MongoDB документоорієнтована система управління базами даних (СУБД) з відкритим вихідним кодом, не потребує описання схеми таблиць. Класифікується як NoSQL, використовує JSON-подібні документи і схему бази даних. Написана мовою C++. Розробляється з жовтня 2007 року компанією 10gen. Зберігає всі дані у форматі бінарного JSON (BSON). Вже має широке застосування у реальних проектах.

Для веб-сайтів органів державної влади, органів місцевого самоврядування та територіальних громад, а особливо для веб-сайтів через які можна замовити отримання публічних та адміністративних послуг важливим є питання доступності для людей з вадами зору.

Нажаль питання стандартів або методичних рекомендацій щодо розробки адаптованих веб-сайтів в Україні залишається не вирішеним. Єдиним варіантом на сьогодні залишається використання нормативів і рекомендацій інших країн та міжнародних методичних рекомендацій.

На сьогодні можна виділити два основних документи, визнаних на міжнародному рівні, що описують стандарти та рекомендації доступності веб-сайтів для людей з обмеженими можливостями, це:

1. The Web Content Accessibility Guidelines 2.0 (WCAG 2.0) – остання версія методичних рекомендацій розроблена міжнародним консорціумом World Wide Web Consortium (W3C). W3C – це міжнародна організація, що розробляє відкриті стандарти для мережі Інтернет, створює специфікації для HTML, CSS, тощо. Консорціум W3C створив окрему ініціативу щодо доступності веб-ресурсів «Web

Accessibility Initiative» (WAI), метою якої є саме розробка стандартів доступності для веб-браузерів, засобів розробки веб-контенту для людей з обмеженими можливостями. Результатом роботи робочої групи з веб-контенту ініціативи WAI стала розробка зазначених методичних рекомендацій щодо доступності веб-сайтів – WCAG.

2. Розділ 508 Закону про Реабілітацію США. Зазначений закон з'явився у 1973 році та був першою законодавчою ініціативою створення рівних умов для людей з обмеженими можливостями. У 1998 році у розділ 508 зазначеного закону були внесені зміни, що заборонили федеральному уряду закуповувати товари та послуги, які не є повністю доступні для людей з обмеженими можливостями, включаючи послуги веб-дизайну. На базі цього розділу законодавчого акту були створені відповідні стандарти, опубліковані 21 грудня 2000 року.

Основний принцип закладений у зазначених вище методичних рекомендаціях та стандартах – це максимальна адаптація веб-сайтів до інструментів доступності, таких як текстове озвучування або використання без прив'язки до кольору.

Це включає в себе наступне:

• Заповнення атрибутів альтернативного тексту (alt) для всіх графічних елементів, що мають недекоративну функцію, з чітким описом зображення та його функції. На рис. 1 наведено приклад заповнення атрибутів альтернативного тексту карток турів одразу при генерації з бази даних.

```
<div class="tours__items">
    <% tours.forEach((tour, index) => { %>
    <div class="tour tour-features " data-features="<%= tour.tourFeatures %>" data-country="<%= tour.tour(
        data-price="<%= tour.tourPrice %>">
        <div class="tour__visible">
            <div class="tour__img-container">
                <img src="../img/country/<%= tour.tourImg %>" alt="<%= tour.tourCountry %>" width="600"
                    height="200" class="tour__img">
```

Рис. 1. Заповнення атрибутів альтернативного тексту карток турів одразу при генерації з бази даних

• Додання окремих файлів субтитрів для всіх аудіо та відео-матеріалів.

• Спрощення та коректне створення таблиць з чітким зазначенням рядків і стовпчиків заголовків та відповідності комірок таблиці заголовкам.

• Адаптація для перегляду сторінки без прив'язки до кольорів.

• Створення форм внесення інформації з послідовними логічними переходами між полями, наявністю всіх підписів та підказок щодо заповнення (рис. 2).

```
<form class="cost__form" action="#" method="post">
    <label for="dateIn" class="cost__label">
    Check-out
        <input type="text" class="cost__input" id="dateIn" name="dateIn"
               title="Введите дату заезда" min="10" max="10" required>
        <span></span>
    </label>
    <label for="dateOut" class="cost__label">
       Check-in
        <input type="text" class="cost__input" id="dateOut" name="dateOut"
               title="Введите дату возвращения" min="10" max="10" required>
        <span></span>
    </label>
```

Рис. 2. Додавання підписів до полів вводу через тег label

В сучасній веб-розробці є велика кількість готових фреймворків, які вже мають комплексні і готові рішення для більшості поставлених задач. Але фронтова частина проекту була реалізована лише за допомогою HTML та CSS. Для прискорення роботи використовувались препроцесори Pug та Sass, а також таск-менеджер Gulp 4 версії.

Перед початком налаштування проекту потрібно зробити два головні кроки:

• здійснити ініціалізацію та налаштування package.json (файл з повними відомостями про проект, автора, підключені модулі та їх залежності) для коректної роботи нашого проекту та можливості повторного використання і редагування.

• здійснити налаштування репозиторію контролю версій (ми будемо використовувати систему Git за допомогою сервісу GitHub).

Всі пакети (додаткові модулі) для нашої програми ми будемо встановлювати за допомогою пакетного менеджера npm. Установка пакетів відбувається з командної стрічки CLI з поміткою "--save", для збереження в package.json.

Налаштування середовища розробки є важливою частиною та фундаментом будь-якого проекту, правильні налаштування допоможуть зекономити багато часу при розробці проекту. Наприклад, наш таск-менеджер сам компілює файли стилів, розставляє вендорні префікси, та сам оновлює сторінку браузера при зміні в структурі.

Верстка проекту виконується за стандартами HTML5, CSS3 та ES6. Це надає кращі можливості для подальшої підтримки та розробки.

Також за методологію розробки було взято BEM (Block Element Modifier – одна із фундаментальних методологій, яка описує норми написання ефективного коду, який можна перевикористовувати).

На рис. 3 представлено зразок використання методології BEM у проекті на прикладі найменування класів.

Під час стилізації сторінки використовувались такі особливості CSS 3, як плавні переходи (transition) та анімації (@keyfames).

```
<ul class="nav">
    <li><a href="/index">Главная</a></li>
    <li class="sub-menu">
        <button class="sub-menu__btn" onclick="dropDown('#dropdown-1')" tabindex="0">
        <ul class="sub-menu__container sub-menu-js" id="dropdown-1">
            <li class="sub-menu__item"><a href="/tours/hot">Hot tours</a></li>
            <li class="sub-menu__item"><a href="/tours/top">Top tours</a></li>
            <li class="sub-menu__item"><a href="/tours">All tours</a></li>
```

Рис. 3. Зразок використання методології BEM у проекті на прикладі найменування класів

На рис. 4 наведено приклад анімації «підстрибування» кнопки, яка в подальшому буде активована при наведенні курсора миші. Анімація реалізована за допомогою властивостей атрибута transform: translateY (зміщення по вертикалі) scale (пропорційна зміна розмірів).

```
@keyframes bonce {
  from{transform: translateY(0) scale(1);}
  40% {transform: translateY(15%);}
  50% {transform: translateY(-10%) scale(1.2);}
  70% {transform: translateY(10%);}
  80% {transform: translateY(-5%);}
  to {transform: translateY(0) scale(1);}
```

Рис. 4. Реалізація анімації «підстрибування» на CSS3

Далі розглянемо динамічні компоненти, реалізовані за допомогою JavaScript. Перший компонент – це модальне вікно. Йому потрібно реалізувати два стани: видимий (відкрите) та прихований (вікно закрите). Знайдемо власне вікно та кнопку відкриття за допомогою методу querySelector (шукає перший збіг в DOM-дереві). Далі присвоїмо кнопці відкриття «слухача» та обробника події кліку. А в обробнику вже будемо додавати нашому вікну клас, який зробить його видимим (рис. 5).

```
const open = document.querySelector(buttonId);
const modal = document.querySelector(modalId);
open.addEventListener('click', function () {
       modal.classList.add('modal-content-show');
       overlay.classList.add('modal-content-show');
});
```

Рис. 5. Реалізація методу відкриття модального вікна

Для зручності, метод закриття модального вікна було реалізовано трьома різними методами:

- клік по кнопці закриття вікна в правому верхньому кутку;
- клік по блоку overlay, що знаходиться на фоні форми пошуку;
- натиснувши кнопку Esc на клавіатурі;

Всі методи реалізовані описаним вище способом за допомогою «слухача» та додавання/видалення класів, які забезпечують видимість (рис. 6).

```
const overlay = modal.querySelector('.overlay');
const close = modal.querySelector('.modal-close');
close.addEventListener('click', function () {
    modal.classList.remove('modal-content-show');
    overlay.classList.remove('modal-content-show');
});
overlay.addEventListener('click', function () {
    modal.classList.remove('modal-content-show');
    overlay.classList.remove('modal-content-show');
});
```

Рис. 6. Реалізація методу закриття модального вікна

Наступним реалізованим компонентом буде перемикач зовнішнього вигляду карток турів (рис. 7, 9). Важливу роль в перебудові сітки карток відіграє більше CSS, ніж JavaScript. Завдяки гнучкій розкладці flex-box ми можемо будувати та перебудовувати сітку сайту згідно з тим, як нам потрібно.

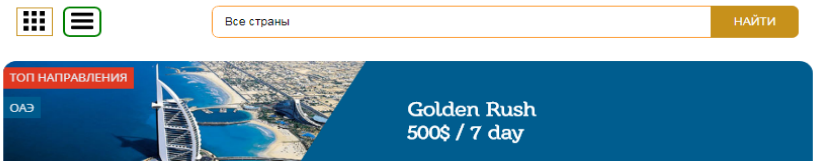

Рис. 7. Вигляд карток у режимі «повного перегляду»

В режимі «повного перегляду» кожен наступний тур розташовується один під одним. Для розкладки встановлено напрямок головної осі зверху-вниз (рис. 8).

```
.tours__container {
        display: flex;
        flex-direction: column;
        justify-content: center;
        width: 100%;
        padding: 10px;}
```

Рис. 8. Розкладка карток в режимі «повного перегляду» за допомогою flex-box

В режимі «швидкого перегляду» картки мають втричі менші розміри і розташовуються одна коло одної, також змінено зовнішній вигляд картки (рис. 9).

Рис. 9. Вигляд карток у режимі «Швидкий перегляд»

Для отримання такого результату нам необхідно: зменшити ширину картки, а також напрямок головної осі. Але крім цього треба дозволити переніс карток, яким не вистачає місця по ширині, на новий ряд розкладки (рис. 10).

```
   .tour-short {
display: flex;
flex-direction: row;
flex-wrap: wrap;
align-content: center;}
   .tour-short .tour{
flex-basis: 32.5%;
margin-right: 0.5%;}
```

Рис. 10. Розкладка карток в режимі «швидкого перегляду» за допомогою flex-box

Далі ми розглянемо побудову серверної частини сайту. За каркас нашої back-end частини взято мінімалістичний і гнучкий веб-фреймворк для додатків Node.js - Express, що надає великий набір функцій для мобільних і веб-додатків. Маршрутизація визначає, як додаток відповідає на клієнтський запит до конкретної адреси (кінцевої точки), яким є URL (або шлях) і метод запиту HTTP (GET, POST і т.ін.).

Тобто, Back-end частина буде відповідати за обробку запитів до сервера та подальшу роботу з ними (обробка та відправлення відповіді). Під відповіддю сервера мається на увазі одна або декілька функцій. В більшості випадків ми будемо використовуємо функцію res.render (рис. 11), оскільки контент всіх сторінок нашого сайту буде генеруватись безпосередньо сервером. Форматування шаблонів було виконане за допомогою шаблонізатора ejs.

```
app.get('/index', (req, res) => {
    res.render('index');
});
```

Рис. 11. Відправка шаблонів сторінок за допомогою функції res.render

Однак є випадки коли потрібно застосувати інші методи відправки з сервера. Наприклад, middleware, який виводить всі тури по конкретній країні (рис. 12) в разі, якщо країну вказано некоректно або користувач випадково надіслав некоректний запит пошуку, сервер просто виконає res.redirect (автоматичний перехід) на сторінку з усіма турами.

```
app.get('/tours/country', (req, res) => {
    let country;
    if(req.query.c){
        country = req.query.c;
    }else {
        res.redirect('/tours');
    }
    tourView(req, res,{tourCountry: country},'tours_bd');
});
```

Рис. 12. Перенаправлення роута за допомогою функції res.redirect

Також сервер може відправити користувачу дані без перезавантаження сторінки (рис. 13). Для цього було використано технологію AJAX, яка працює через звичайний http-request. AJAX надсилає дані на сервер, а сервер, в свою чергу, намагається їх опрацювати та повернути власне результат своєї роботи (успіх або помилку). Відбувається це шляхом надсиланням змінних StatusCode. Якщо StatusCode буде рівний 200, то запит вважається успішним.

```
app.post('/callback', (req, res) => {
    const {customerName, customerNumber, customerMessage, status} = req.body;
        MongoClient.connect(urlDB, (err, db) => {
        db.collection("callback").insertOne(customer, (err, result) => {
            if(err) {
                return res.status(400).send();
            }
            db.close();
            res.status(200).send();});
    });
});
```

Рис. 13. Відправка сервером StatusCode після обробки запиту

Коли проект стає високонавантаженим або працює з великою кількістю інформації, виникає необхідність підключення бази даних. Опишемо роботу з СУБД mongo DB. База даних створена на основі сервісу mLab, що дозволяє працювати з нею незалежно від розташування сервера з нашим додатком.

Розглянемо роботу з даною СУБД на прикладі функції tourView (функція, яка виводить відфільтровані тури на сторінку). Розглянемо її роботу детальніше (рис. 14).

```
function tourView(req, res, sample = {}, render) {
    MongoClient.connect(urlDB, (err, db) => {
        db.collection('tours').find(sample).toArray((err, tours) => {
            db.close();
            const size = tours.length;
            const limit = 10;
            const maxPage = Math.ceil(size/limit);
            const skip = (page - 1) * limit;
    MongoClient.connect(urlDB, (err, db) => {
            db.collection('tours').find(sample).skip(skip).limit(limit).toArray(
               (err, tours) => {
                   db.close();
                   if(tours.length === 0) {
                         res.redirect('/tours');
                    }else {
                        res.render(render, {
                        tours,
                        page: page,
                        maxPage: maxPage
                            })
                    });
```

Рис. 14. Код функції tourView

На вході функція отримує параметри:

- req / request – дані, які надіслав користувач;
- res / response – дані, які має повернути сервер;
- sample – запит на вибірку з бази даних, за замовчуванням він порожній, що означає вибірку всіх елементів колекції. (Примітка: використання у функціях змінних з дефолтними значеннями стало доступно лише у новому стандарті мови JavaScript – ES6. Якщо ви використовуєте ES5 або нижчі версії, то код у вас не буде підтримуватись).

render – назва сторінки, яку серверу необхідно надіслати у відповідь. Функція tourView, крім вибірки за турами, генерує клієнтську пагінацію, що зроблено для кращої доступності нашого додатку та зручності при роботі з ним. Під час запиту до колекції визначається кількість елементів у ній (метод length). Далі за допомогою вбудованих в наш СУБД клієнт методів: skip (скільки записів пропустити), limit (максимальна кількість записів на 1 сторінці), ми здійснюємо новий запит до бази даних.

Висновки. Розроблений для проекту додаток задовольняє всі обов'язкові вимоги технічного завдання. Front-end частина реалізована нативним кодом без використання фреймворків та бібліотек готових рішень. Back-end частина реалізована за допомогою платформи Node.js, а для спрощення маршрутизації та відправки даних використано фреймворк Express. Також підключено базу даних типу noSQl – mongo DB. Підключення та робота з базою відбувалась через стандартний mongoClient.

Розроблений додаток має ряд особливостей, зокрема:

- Використано стек технологій пов'язаних з JavaScript.
- Проект реалізовано компонентно, що дозволяє перевикористовувати код та в подальшому успішно підтримувати проект.
- Проект має завершений «ланцюжок» від потенційного покупця до купленого квитка.

Однак, будь-який продукт потребує доопрацювань і наш додаток – не виключення. Оскільки, він був написаний в наукових цілях, потрібно розробити ряд компонентів необхідних для повної реалізації його на ринку, а саме:

- Використання Mongoose для роботи з БД в цілях збільшення захищеності та типізованості вхідних даних.
- Створення можливості реєстрації на сайті.
- Доопрацювання поштових модулів для налаштування підписки на новини та отримання на пошту готового квитка.

Також, дотримуючись тенденцій сучасних веб-додатків, такі дії як: перехід по сторінках сайту, запити на сервер, відправка форм повинні відбуватися без перезавантаження сторінки браузера. Тому подальша реалізація додатку можлива лише у вигляді SPA (Single Page Application – додаток на одну сторінку). А також підтримки технологій Shadow DOM або Virtual DOM (віртуальне завантаження компонентів сайту), такі інструменти нам надають сучасні фреймворки React та Vue 2.0 відповідно.

Крім цього, об'єктивним на нашу думку буде реалізація API нашого додатку за допомогу технології REST запитів, яка зараз набирає популярність – GraphQL. Ну і звісно, обов'язково, потрібно провести юніт-тестування кожного компоненту, тут знову ж таки на допомогу приходять нові технології – бібліотека для тестування Jest або Mocha.

Література

1. Офіційна документація по роботі з фреймворком Express [Електронний ресурс]. – Режим доступу : http://expressjs.com/

2. Офіційна документація по роботі з платформою Node.js [Електронний ресурс]. – Режим доступу: https://nodejs.org/uk/

3. Офіційний сайт пакетного менеджера npm [Електронний ресурс]. – Режим доступу : https://www.npmjs.com/

4. Офіційний сайт сервісу GitHub [Електронний ресурс]. – Режим доступу : https://github.com/

5. Офіційний сайт бази даних mongo DB [Електронний ресурс]. – Режим доступу : https://www.mongodb.com/

6. Переклад книги Node Hero від RisingStack [Електронний ресурс]. – Режим доступу : https://medium.com/devschacht/node-hero-6a07ef8d822d\

7. Онлайн-підручник вивчення мови JavaScript [Електронний ресурс]. – Режим доступу : https://learn.javascript.ru/

8. Офіційний сайт бібліотеки jQuery UI [Електронний ресурс]. – Режим доступу : https://jqueryui.com/

9. Офіційний сайт компанії EGAP [Електронний ресурс]. – Режим доступу : http://egap.in.ua/

10. Офіційний сайт організації W3C [Електронний ресурс]. – Режим доступу : https://www.w3.org/.

References

1. Official documentation for working with the Express [Electronic resource] – Access mode : http://expressjs.com/

2. Official documentation for working with the Node.js platform [Electronic resource] – Access mode : https://nodejs.org/uk/

3. The official site of the package manager npm [Electronic resource] – Access mode : https://www.npmjs.com/

4. Official site GitHub services [Electronic resource] – Access mode : https://github.com/tigermax139

5. The official database of mongo DB [Electronic resource] – Access mode : https://www.mongodb.com/

6. Translation of Node Hero book from RisingStack [Electronic resource] – Access mode : https://medium.com/devschacht/node-hero-6a07ef8d822d\

7. Online tutorial for JavaScript learning[Electronic resource] – Access mode : https://learn.javascript.ru/

8. Official site jQuery UI library of user interfaces [Electronic resource] – Access mode : https://jqueryui.com/

9. Official site EGAP company [Electronic resource] – Access mode : http://egap.in.ua/

10. The World Wide Web Consortium (W3C) [Electronic resource] – Access mode : https://www.w3.org/

Рецензія/Peer review : 25.04.2018 р. Надрукована/Printed :08.07.2018 р.
Рецензент: д.т.н., проф. Матвійчук В.А.