ISSN 2224-087X. Електроніка та інформаційні технології. 2017. Випуск 8. С. 66–73
Electronics and information technologies. 2017. Issue 8. P. 66–73

УДК 004.891

ЕКСПЕРТНА СИСТЕМА ПОШУКУ СМАРТФОНІВ, СТВОРЕНА ЯК WEB-ДОДАТОК НА ПЛАТФОРМІ NODE.JS

В. Грабовський, П.-Р. Маслій

Львівський національний університет імені Івана Франка,
вул. Ген. Тарнавського, 107, 79017 Львів, Україна
v_grabovskyi@franko.lviv.ua

Створено експертну систему продукційного типу для пошуку смартфонів з заданими користувачем параметрами у вигляді WEB-додатка. Для забезпечення використання можливостей швидкого пошуку у WEB-середовищі використано Node.js (платформу з відкритим кодом для виконання мережевих застосунків, написаних мовою JavaScript) з асинхронною моделлю запуску коду, що зумовлює велику швидкодію системи. Створена база знань та механізм пошуку дають змогу виконувати швидкий та докладний пошук потрібного смартфона з відображенням сайтів з потрібною інформацією, у тому числі сайтів інтернет магазинів, у яких його можна придбати. Передбачене редагування (зміна і поповнення) створеної бази знань.

Ключові слова: експертні системи, смартфон, Node.js, JavaScript, WEB-додаток.

Експертні системи належать до класу комп'ютерних програм, які, використовуючи закладені в них знання, можуть виконувати завдання, вирішення яких загалом притаманне людині й потребує застосування її інтелекту [1]. Основною частиною таких програм, відповідно, є база знань, до вмісту якої ставлять досить специфічні вимоги, і яку формують на основі знань фахівців-експертів у вузькій предметній галузі. Оскільки такі знання часто мають неформальний характер, можуть бути неповними й іноді навіть помилковими, то для їхнього формування, відображення і коректного використання в комп'ютерних застосунках розроблено спеціальну галузь штучного інтелекту, яка отримала назву інженерія знань [2]. Зрозуміло, що побудова таких застосунків непроста і передбачає використання досить складного програмного інструментарію – від мов програмування високого рівня до спеціально створених з цією метою середовищ [3].

Під час створення експертних систем (ЕС) залежно від задачі та моделі подання знань у багатьох випадках доцільно використовувати засоби розробки, які пропонують вирішення, що добре зарекомендували себе на практиці. До таких належать деякі спеціально створені мови програмування високого рівня – зокрема, мови декларативного програмування, які охоплюють мови функціонального (наприклад, LISP [4]) та логічного програмування (наприклад, PROLOG [5]); ці інструментальні засоби ще називають мовами штучного інтелекту. Зазвичай, такі засоби мають вбудовані механізми логічного виведення, а сама програма ЕС у них має вигляд опису особливостей предметної галузі задачі та мети, яку потрібно досягти.

Окрему нішу серед засобів створення ЕС займають так звані оболонки експертних систем [1] – середовища програмування, використання яких дає змогу створити ЕС навіть без достатніх навичок техніки програмування; їх ще називають "порожніми експертними системами". Оскільки оболонки експертних систем мають усі потрібні для успіш-

ної роботи ЕС засоби (насамперед, механізм логічного виведення та інтерфейс для спілкування з користувачами, а також усе для створення потрібної бази знань), то розробка нової системи практично зводиться до створення потрібної бази знань. До чи не найвідоміших сьогодні оболонок належать CLIPS та її різноманітні модифікації, EMYCIN, BABYLON тощо. Використання цих засобів дає змогу створити локальні ЕС, які з успіхом виконують поставлені перед ними завдання.

Однак є клас завдань, які потребують використання WEB-середовища з метою пошуку необхідної інформації для їхнього виконання. Специфіка створення таких систем потребує й використання певних інструментальних засобів. Зокрема, таке завдання можна виконати, якщо застосувати Node.js [6] – платформу з відкритим кодом для виконання мережевих додатків, написаних мовою JavaScript [7], яка доступна для застосування в будь-якій операційній системі. Для забезпечення виконання написаного коду застосовують розроблений компанією Google рушій V8. Особливістю платформи є те, що її функції не обмежені створенням серверних скриптів для WEB – платформу можна використовувати для створення як звичайних клієнтських, так і серверних мережевих програм. Важлива відмінність Node.js також та, що в ній використовують асинхронну модель запуску коду, яка ґрунтується на обробці подій у неблокувальному режимі та визначенні відповідних “зворотних викликів”, що зумовлює велику швидкодію системи.

Для того, щоб можна було швидко та зручно ділитись кодом і використовувати його багато разів, у Node.js є широка система пакетів – модулів, які розроблені в середовищі користувачів і розробників та доступні кожному через мережу. Для керування пакетами використовують пакетний менеджер NPM (Node Package Manager) [8], який дає змогу завантажувати, видаляти, запускати пакети, розроблені для роботи в середовищі NodeJS, і ділитися ними.

Платформа Node.js дуже ефективна в своєму колі завдань, стрімко розвивається і сьогодні має чи не найбільшу кількість доступних користувачам пакетів і прихильників серед усіх наявних технологій. Розробнику для модифікації чи виконання програми на локальній машині, окрім веб-переглядача, необхідно також інсталювати платформу Node.js, на якій відбувається виконання програми. Платформа доступна для будь-якої операційної системи. Завантажити її можна на офіційному сайті www.nodejs.org.

Розроблена експертна система має вигляд WEB-додатка; її структуру показано на рис. 1.

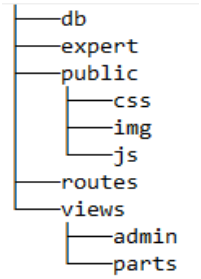

Рис. 1. Структура проекту.

Код інтерфейсу проекту міститься в директорії *views*; він розбитий на логічні та структурні частини. Наприклад, сторінки “Головна” та “Адмін” розміщені в окремих файлах. Їхня спільна частина, що підключає стилі оформлення, винесена в окремий файл; це дає змогу в одному файлі модифікувати стилі обох сторінок.

Директорія *db* містить конфігурацію підключення програми до бази даних та саму базу даних, яка є файлом, що зберігає інформацію у форматі JSON [7], тобто є набором об'єктів вигляду ключ–значення. Кожен з цих об'єктів зберігає інформацію про конкретний смартфон чи мобільний телефон. Наприклад:

{"vendor":"Apple","model":"iPhone 5,"year":"2012","price":"4999"}.

У директорії *public* розміщені стилі оформлення інтерфейсу, зображення пристроїв та код, необхідні для коректної роботи програми – зокрема, для передавання інформації про пристрій, що додається до бази даних, або інформації, за допомогою якої відбувається пошук та формується фінальний підсумок роботи ЕС. У директоріях *routes* та *expert* наявні логічний блок, що працює з даними, уведеними користувачем, та база знань, яка формує рейтинг пристроїв, відображених у базі даних, на основі наданої ним інформації. Блоки коду роботи з базою даних та пошуку вирішення розділені й містяться в окремих файлах у директоріях *public/js* та *routes*. Директорія *public/js* містить код, що виконується в клієнтській частині програми у WEB-переглядачі, а *routes* – безпосередньо у самій програмі.

Розроблена експертна система є системою продукційного типу – знання системи відображені у вигляді так званих правил-продукцій [9]. База знань є файлом з набором правил вигляду "якщо – то". Наприклад, правило, що міститься в базі знань, яке визначає як операційну систему пристрою iOS та додає до загальної оцінки пристрою 10 балів у випадку, якщо його операційна система збігається з операційною системою від компанії Apple, має такий вигляд:

```
if (wishes.apple === true) {
if (item.os === 'ios') {
item.confidence += 10
item.why.push('iOS') }}
```

Для запуску системи необхідно відкрити термінал у директорії, де зберігається проект, та ввести команду *npm run start*; уведена команда є зручним скороченням, збереженим у налаштуваннях проекту у файлі *package.json.* Там зберігаються основні відомості про проект, команди для виконання та тестування, контакти розробника тощо. Після того, як програма запущена, ми можемо відкрити головну сторінку веб-ресурсу, використовуючи локальну адресу *localhost:3000*.

Інтерфейс головної сторінки створеної експертної системи (рис. 2) містить кнопки швидкого, розширеного пошуку та адміністрування БД; за їхньою допомогою можна обрати бажаний метод взаємодії ЕС з користувачем та перейти до потрібної йому частини інтерфейсу.

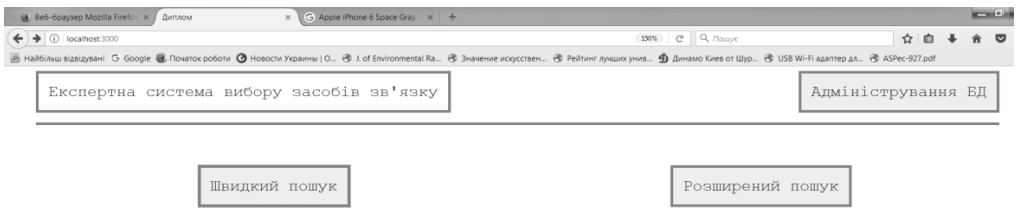

Рис. 2. Фрагмент головної сторінки розробленої ЕС.

Щоб виконати пошук рішення, користувачеві на головній сторінці програми потрібно обрати метод пошуку: “швидкий” (спрощений, за кількома наявними у користувача характеристиками пристрою) або “розширений” (детальний, за набором детальних характеристик).

Набір доступних для використання в разі “швидкого” пошуку параметрів показаний на рис. 3. У випадку, коли користувач не знає, які конкретно параметри апарату його цікавлять, він може обрати одну чи декілька з загальних характеристик і отримати вирішення, що потенційно задовольнить його потреби.

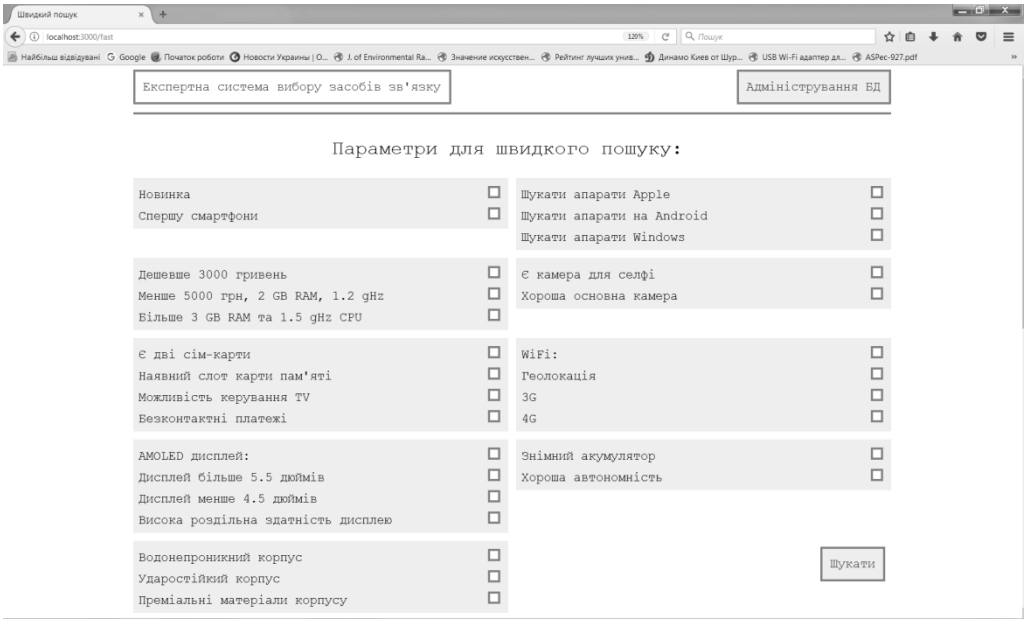

Рис. 3. Фрагмент сторінки з параметрами “швидкого” пошуку.

Після вибору бажаних параметрів і натискання на кнопку “Шукати” (див. рис. 3) код обробника події натискання спрацює і відправить інформацію в один з логічних блоків у директорії *routes* (див. рис. 1). Виконається функція *find,* яка зчитає введені користувачем дані та відправить їх у блок *answer*; цей блок, використовуючи вміст бази знань, після формування рейтингу знайдених пристроїв повертає користувачеві відшукане вирішення. Це вирішення є вибіркою з бази даних пристроїв, які найбільше відповідають заданим параметрам пошуку, відсортованою за попередньо сформованим рейтингом, яка може бути, за потреби, обмеженою (наприклад, одним, двома чи іншою кількістю вибраних екземплярів). Для кожного пристрою, знайденого ЕС унаслідок швидкого пошуку, відображено лише базову інформацію та зовнішній вигляд (фрагмент вікна з результатами пошуку для запиту “Шукати апарати Apple” показаний на рис. 4), а також прикріплено кнопку “детальніше”, призначення якої – отримання детальної інформації про конкретний пристрій через пошук у мережі інтернет; унаслідок такої деталізації у вікні виводяться сайти, на яких уся наявна в інтернет інформація про цей пристрій, у

тому числі сайти інтернет-магазинів, у яких можна придбати потрібний пристрій (фрагмент вікна з отриманою інформацією детальнішого пошуку зображено на рис. 5).

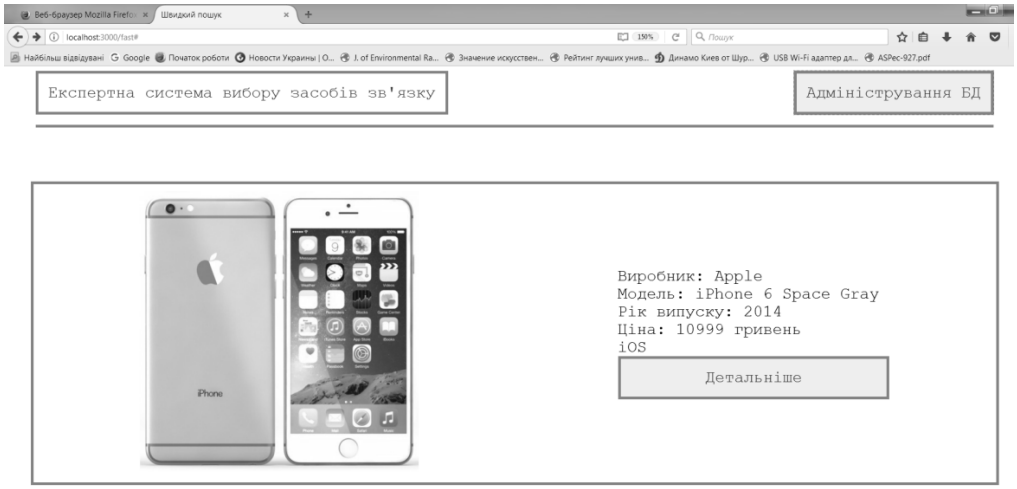

Рис. 4. Фрагмент вікна зі знайденим унаслідок “швидкого” пошуку смартфона виробництва Apple вирішенням.

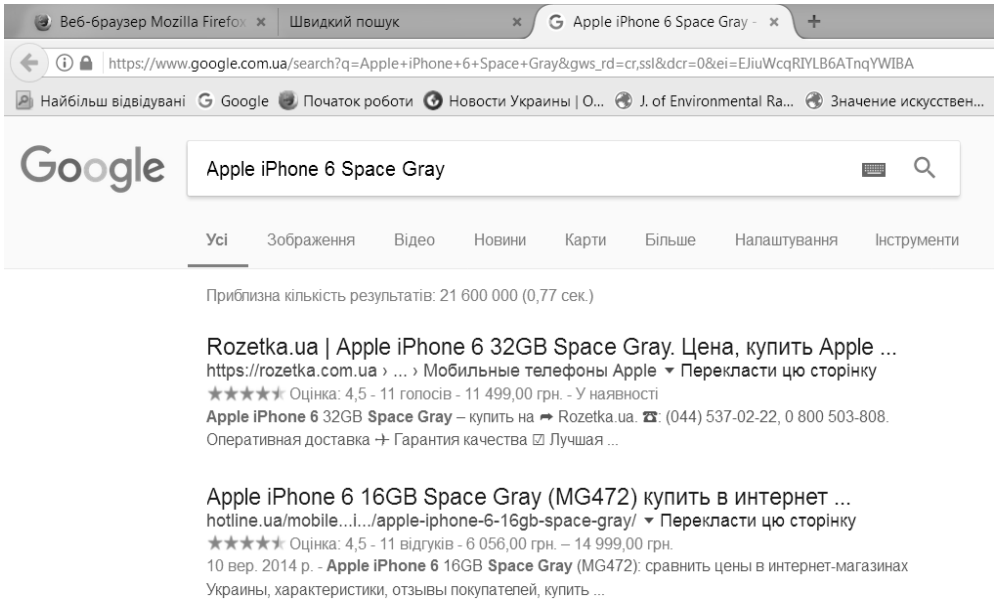

Рис. 5. Фрагмент вікна з результатами пошуку, які виводяться на екран після натискання на кнопку “Детальніше”.

Розширений пошук дає змогу знайти смартфон за конкретними характеристиками; список характеристик з’являється у вікні після натискання на кнопку “Розширений по-

шук”. На відміну від швидкого пошуку, у цьому режимі формується точніший рейтинг пристроїв, оскільки в процесі пошуку вирішення може бути задіяно багато характеристик смартфону.

Для отримання вирішення в разі використання “розширеного” пошуку необхідно в сторінці “Розширеного пошуку” задати значення кожного з бажаних параметрів шуканого пристрою. У підсумку пошуку у вікні буде виведено результат, який містить розміщені за рейтингом (задоволенням вимог користувача) пристрої – їхній вигляд та параметри; першим у вікні буде показаний смартфон, параметри якого найповніше задовольняють вимоги користувача (вигляд частини вікна з результатами розширеного пошуку показаний на рис. 6).

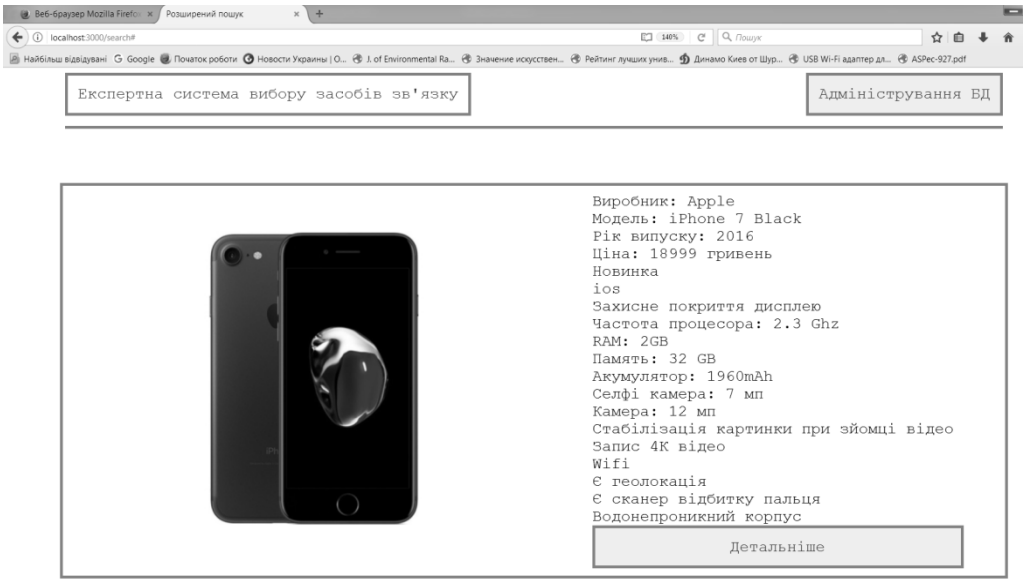

Рис. 6. Фрагмент вікна зі знайденим унаслідок “розширеного” пошуку вирішенням.

Передбачена можливість адміністрування бази даних (БД), тобто як додавати нові пристрої в базу, так і видаляти з неї наявні.

Щоб видалити потрібний елемент БД, необхідно натиснути на кнопку “Адміністрування БД” (див. рис. 2–6), увійти на сторінку “Адміністрування БД”, знайти на ній потрібний пристрій і натиснути на прикріплену до нього кнопку “Видалити”. Після натискання блок *delete*, використовуючи унікальний ідентифікатор, знайде відповідний елемент у базі даних та видалить його разом з файлом закріпленого за ним зображення; ім’я цього файлу міститься в записі бази даних про конкретний елемент. У підсумку інтерфейс сторінки буде перемальовано з урахуванням оновленої бази даних.

Процедура додавання в базу даних дещо подібна до процедури пошуку. Для цього потрібно зайти на сторінку “Адміністрування БД” і натиснути на розміщену тут кнопку “Додати”. У полях вікна, яке з’являється, увести відповідні дані, що відповідають пристрою, а також створити файл з його зображенням, вказати шлях до цього файлу й знову натиснути на кнопку “Додати”. Функція, подібна до функції пошуку рішення, зчитує

поля введених даних та відправляє їх у блок *add,* де отримані дані належно інтерпретуються, додаються у базу даних; надалі введений пристрій відображатиметься у вікні пошуку.

Отже, використання платформи Node.js дає змогу створити експертну систему, яка виконує пошук потрібного користувачам пристрою та інформації про нього з застосуванням не тільки створеної бази знань, а й ресурсів WEB-середовища. Створений застосунок допомагає користувачу відшукувати потрібний засіб зв'язку у швидкому та розширеному режимі залежно від повноти інформації, а також отримувати оперативну інформацію про наявність потрібного пристрою на ринку продаж та можливість його придбання через інтернет-магазини. Також реалізовано можливість редагування створеної бази даних користувачем.

Список використаної літератури

1. *Джарратано Дж.* Экспертные системы. Принципы разработки и программирование / Джозеф Джарратано, Гари Райли ; 4-е изд. ; [пер. с англ.]. – М. : Изд. дом "Вильямс", 2007. – 1152 с.
2. *Частиков А. П.* Разработка экспертных систем. Среда CLIPS / А. П. Частиков, Т. А. Гаврилова, Д. Л. Белов. – СПб. : БХВ-Петербург, 2003. – 608 с.
3. *Кургаєв О. П.* Методи та системи штучного інтелекту : конспект лекцій для студентів напряму підготовки 6.050101 "Комп'ютерні науки" денної та заочної форм навчання / О. П. Кургаєв. – К. : НУХТ, 2014. – 279 с.
4. *Хювёнен Э.* Мир Лиспа. Т. 1. Введение в язык Лисп и функциональное программирование / Э. Хювёнен, И. Сеппянен. – М. : Мир, 1990. – 332 с.
5. *Стобо Д. Ж.* Язык программирования Пролог / Д. Ж. Стобо ; [пер. с англ.] – М. : Радио и связь, 1993. – 368 с.
6. Node.js в действии / М. Кантелон, М. Хартер, Т. Головайчук, Н. Райлих. – СПб. : Питер, 2014. – 548 с.
7. Современный учебник JavaScript [Электронный ресурс]. – Режим доступа : https://learn.javascript.ru/
8. What is npm? [Electronic resource]. – Mode of access : https://docs.npmjs.com/getting-started/what-is-npm/
9. *Grosan C.* Intelligent Systems. A Modern Approach / Crina Grosan, Ajith Abraham // Intelligent Systems Reference Library. – Springer, 2011. – Vol. 17. – 450 p.

Стаття: надійшла до редакції 29.09.2017,
доопрацьована 09.10.2017,
прийнята до друку 10.10.2017.

EXPERT SYSTEM OF SMARTPHONES SEARCH, CREATED AS A WEB APPLICATION ON THE BASIS OF THE NODE.JS PLATFORM

V. Grabovskyi, P.-R. Masliy

Ivan Franko National University of Lviv,
107 Tarnavsky St., UA–79017 Lviv, Ukraine
v_grabovskyi@franko.lviv.ua

We present an expert system that searches for smartphones with user-specified parameters. Our system is created in the form of a web application. To ensure a possibility of fast searching in web environment, a Node.js platform is used, which represents an open-source cross-platform system for executing different network applications. Moreover, it is based on an asynchronous code-launch model, which enables high query-processing performance. Notice that Node.js has been written in 2009 as a special server-side solution for JavaScript, revealing high efficiency in the field of web-searching. In particular, it has been aimed at receiving and responding to various HTTP requests. The platform is now rapidly developing. Among currently existing tools of this type, it has the largest number of custom packages available, as well as human supporters that employ it.

Our application is a rules-based expert system, also known as a production system, of which knowledge base contains domain knowledge coded in the form of rules. The project-interface code is located in a '*views*' directory. It is divided into logical and structural units. In particular, pages "Home" and "Admin" are located in separate files. A common part of the code that connects the styles of their designs is also organized as a separate file, which allows modifying the styles of both pages in a single file. The styles of the interface design, the device images and the code, which are necessary for correct operation of the program, are located in a '*public*' directory. The database-connection code and the database code, which in itself represents a file that stores information in the JSON format, are located in a directory '*db*'. '*routes*' and '*expert*' directories contain a logical block that works with the data entered by a user, as well as a block that generates ranking of the devices presented in the database. The codes of the database and the search-engine blocks are separated, being located in '*public/js*' and '*routes*' directories, respectively. Finally, a '*public/js*' directory includes a code executed on a client side of the program in a web browser, whereas the code executed directly in the program is located in a '*routes*' directory.

The knowledge base and the search system created by us enable one to perform quick and expanded searching for smartphones with desired characteristics. Moreover, the software finds the websites with all the necessary information, including the websites of online stores where the smartphones can be purchased. Abilities for editing (modifying and adding) the knowledge base by users are also provided.

Key words: expert systems, smartphones, Node.js, JavaScript, WEB application.