МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Житомирський державний технологічний університет

Кафедра інженерії програмного забезпечення

**ЗВІТ З ПЕРЕДДИПЛОМНОЇ ПРАКТИКИ**

Студента (ки) 2 курсу групи ПІ-47м

Галузь знань 12 «Інформаційні технології»

Спеціальність 121 «Інженерія програмного забезпечення» (спеціалізація «Інженерія програмного забезпечення»)

ступінь «**магістр»**

**(прізвище ініціали, підпис)**

Керівник: Єфремов Юрій Миколайович   
доцент кафедри ІПЗ, к.т.н.

Кількість балів: \_\_\_ Національна оцінка: \_\_\_ ECTS:\_\_\_

Члени комісії:

\_\_\_\_\_\_\_\_\_\_\_\_ \_\_С.М. Кравченко\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_ Ю.М.Єфремов \_\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_ О.В.Власенко \_\_\_\_\_

(підпис) (прізвище та ініціали)

Житомир – 2018

РЕФЕРАТ

Зм.

Арк.

№ докум.

Підпис

Дата

Арк.

2

ІПЗ.ПП.М – 121 – 18 – З

Розроб.

В.М. Данюк

Перев.

.

Ю.М. Єфремов

Н. Контр.

С.М. Кравченко

Затв.

*А.В. Панішев*

**Інтерактивний web-сервіс для пошуку домашніх тварин**

**Звіт**

Літ.

Аркушів

29

ЖДТУ, група ПІ-47м

Звіт з переддипломної практики складається з вступу, висновків, списку використаних джерел із 22 найменувань, 3 додатків, 4 таблиць та 5 ілюстрацій по тексту. Повний обсяг роботи становить 29 сторінок, з них 18 сторінок основного тексту.

Метою роботи є розробка інтерактивного web-сервісу для пошуку домашніх тварин завдяки якій користувачі зможуть розміщувати оголошення про втрачених/знайдених домашніх тварин, що містить модулі локалізації, фільтрації за основними критеріями та модерації.

В роботі визначено основні завдання на розробку системи, проаналізовано аналоги розробленої системи, обґрунтовано вибір архітектурного патерна для реалізації системи – MVVM. Наведено сценарії роботи програмного комплексу, загальна структура програмного комплексу, опис бази даних системи (Firebase Realtime NoSQL), опис алгоритмів взаємодії окремих модулів системи, опис об’єктної структури системи (діаграма компонентів), алгоритми роботи основних модулів, реалізованих з використанням мови програмування JavaScript та фреймворку Vue.

Ключові слова: БЮРО ЗНАХІДОК ДОМАШНІХ ТВАРИН, ВТРАТИВ, ЗНАЙШОВ, MVVM, FIREBASE, VUE.

ЗМІСТ

[РЕФЕРАТ 2](#_Toc530070080)

[ЗМІСТ 3](#_Toc530070081)

[ВСТУП 4](#_Toc530070082)

[ОСНОВНА ЧАСТИНА 5](#_Toc530070083)

[1. Постановка задачі 5](#_Toc530070084)

[2. Аналіз аналогів програмного продукту 7](#_Toc530070085)

[3. Вибір архітектури для інтерактивного web-сервісу по пошуку домашніх тварин 9](#_Toc530070086)

[4. Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення 11](#_Toc530070087)

[5. Визначення варіантів використання та об’єктно-орієнтованої структури системи 14](#_Toc530070088)

[6. Розробка бази даних системи 20](#_Toc530070089)

[ВИСНОВКИ 23](#_Toc530070090)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 24](#_Toc530070091)

[ДОДАТКИ 26](#_Toc530070092)

[ДОДАТОК А 27](#_Toc530070093)

[ДОДАТОК Б 28](#_Toc530070094)

[ДОДАТОК В 28](#_Toc530070095)

ВСТУП

**Актуальність теми.** Щодня десятки людей втрачають і знаходять домашніх улюбленців. Як правило, загублена тварина для її власника має набагато вищу цінність, ніж для іншої людини. Швидкий і зручний пошук та повернення знайдених/загублених домашніх тварин на сьогоднішній день дуже важливий.

**Метою даної роботи** є проектування архітектури, розробка алгоритмів роботи та реалізація ПЗ для знаходження та розміщення об’яв про знайдену/загублену домашню тварину та модулів які поліпшить їх умовам.

Для досягнення поставленої мети необхідно вирішити такі основні завдання:

* проведення системного аналізу предмету дослідження;
* визначення основних функцій та особливостей системи;
* вибір технології та засобів реалізації програмного продукту;
* проектування структурних складових та алгоритмів роботи системи.

**Об’єктом дослідження** є автоматизація управління процесами CRUD, що полегшають користувачам їх використовувати.

**Предметом дослідження** є використання веб-орієнтованих технології для автоматизації роботи інтерактивного web-сервісу для пошуку домашніх тварин.

В роботі було використано методи об’єктно орієнтованого проектування, теорії NoSQL баз даних, веб-програмування, об’єктно-орієнтованого програмування та дизайну інтерфейсів.

**Опис бази проходження практики**. Базою практики є ФОП Данюк Володимир Михайлович, зареєстрований за адресою м. Житомир, Льва Толстого, 15. Основний вид діяльності являється надання послуг в сфері ІТ. Під час проходження практики обіймав посаду front end developer.

ОСНОВНА ЧАСТИНА

## Постановка задачі

Основною метою створення даного веб-сервісу в тому, що кожен користувач зможе створювати або знаходити оголошення не виходячи з дому і тим самим зможе повернути знайдену/загублену домашню тварину. Одна з особливостей веб-сервісу є те, що всі оголошення позначаються на карті.

В даній роботі необхідно розробити інтерактивний web-сервіс для пошуку домашніх тварин, засоби управління інформаційним контентом та функціональні модулі, створення, редагування, видалення та пошук у відповідності із заданими даними. Для реалізації поставленого завдання, основними етапами є:

1. Розробити дизайн сайту.
   * Спроектувати загальну концепцію дизайну для головної та внутрішніх сторінок сайту: головна сторінка з картою оголошень, маркери на карті з інформацією про оголошення, спливаюче вікно с формою створення та редагування оголошень, меню; сторінка авторизації; сторінка реєстрації; сторінка профілю користувача; сторінка модерації оголошень.
   * За допомогою мови розмітки HTML 5 і таблиць стилів CSS 3 створити шаблони головної та внутрішніх сторінок.
   * Використання бібліотеки компонентів Vuetify для побудови інтерфейсів сторінки з використанням принципів Google Material Design.
2. Реалізувати основну роботу програми.

2.1. Написати програмні модулі для:

* модуль авторизації користувачів в системі. Розподіл прав доступу користувачів повинен визначатися за наступними ролями: користувач, модератор та адміністратор;
* модуль реєстрації користувачів на сайті – при заповненні обов’язкових полів та правильному введенні користувачем дані він може зареєструватися на сайті;
* модуль створення нового оголошення – після авторизації і заповнені обов’язкових полів користувач зможе створити оголошення;
* модуль фільтрації оголошень типом оголошення, типом, породою, коліром, забарвлення, віком тварини;
* модуль редагування оголошення;
* модуль модерації.

2.2. Реалізувати CRUD операції для всіх колекцій бази даних системи.

1. Реалізувати модулі виводу оголошення та статистики на сайті:
   * Статичний звіт по створених оголошеннях за місяць, кількості користувачів та інше.
   * Модуль виводу карти з оголошеннями за допомогою “Google Maps API”.
   * Контролювання нових оголошень.
2. Реалізувати структуру збереження даних наступного виду:
   * Дані про користувача (Електронна пошта, пароль, роль користувача);
   * Дані про об’яву (тип, статус, дата створення, тип тварини, порода тварини, колір тварини, забарвлення тварини, вік тварини, контактна інформація, координати місця знахідки/пропажі);
3. Створити демонстраційний та тестовий контент сайту.

Результатом виконання даних завдань ми отримаємо інтерактивний web-сервіс для пошуку домашніх тварин, який має клієнт-серверну архітектуру, безпечну базу даних, різнопланові функціональні можливості, що забезпечують коректну роботу сервера та зрозумілий клієнтський інтерфейс, а також панель модерації та адміністрації з обмеженим доступом.

## Аналіз аналогів програмного продукту

На сучасному ринку програмних засобів представлено велика кількість, але в основному вони прив’язані до одного міста. Переглянувши їх я зміг відібрати тільки 4 веб-сервіса для аналізу. Для розгляду аналогів, за критеріями популярності та позитивного рейтингу вибрані веб-сервіси наведені в табл. 2.1.

Опишемо кожну із розглянутих програмних систем.

Всеукраинского онлайн-бюро находок “Верни”. Перевагами цього сервісу є те, що він має фільтр по регіонам, для створення об’яви не потрібно реєструватись та пошук по об’явам. Недоліками сервісу є те, що він не підтримує локалізацію інших мов, в об’явах немає карти на якій відображається місце знахідки/пропажі також сервіс немає фільтра по іншим критеріям об’яви.

PETSI. Сервіс має сучасний адаптивний дизайн та пошук по об’явам. Особливість цього сервісу є те, що він має магазин та довідник порід тварин. Недоліками сервісу є те, що він не має фільтрації оголошень та локалізації для інших мов.

LOSTPETS. Переваги сервісу в тому, що він надає різні послуги, відображає місце знахідки/втрати на карті та має фільтрацію по оголошенням. Недоліки сервісу в тому, що він дуже повільно працює, застарілий дизайн, не коректна верстка та не має пошуку.

LUCKFIND. Переваги сервісу в тому, що він має сучасний дизайн, локалізацію трьома мовами, пошук по об’ява, відображає місце знахідки/пропажі на карті, розроблений на сучасній мові програмування AngularJS. Недоліки його в тому, що він не має фільтра, меню розміщено тільки внизу сторінки та авторизацію яку важко знайти.

Для того, щоб зробити якісь висновки потрібно порівняти їх на функціональні можливості, тому порівняння будуть наведені в табл. 2.1.

Таблиця 2.1

Функціональні можливості

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва | Всеукраинское виртуальное онлайн-бюро находок "Верни" | PETSI | LOSTPETS | Громадська організація «LUCKFIND» |
| Електронний вхід/вихід | - | + | + | + |
| Пошук | + | + | - | + |
| Фільтр | + | - | + | - |
| Служба підтримки | - | - | - | + |
| Відображення місця втрати та знахідки на карті | - | - | + | + |
| Адміністративні функції | - | - | + | + |
| Можливість залишити коментарій | + | - | + | + |
| Мобільна версія сайту | + | + | + | + |

Переглянувши список аналогів інтерактивних web-сервісів для пошуку домашніх тварин можна зробити висновки, що майже всі програми однотипні і тільки один сервіс відрізняється від інших і це “LUCKFIND”.

Таким чином, основним я зробив висновок, що мій сервіс повинен мати такі функціональні можливості:

− Фільтр за різними критеріями.

− Дані повинні оновлюватись оперативно швидко.

− Простота у використанні.

− Електронний вхід/вихід.

− Пошук.

− Локалізація.

− Відображення місця втрати та знахідки на карті.

− Модерація об’яв.

## Вибір архітектури для інтерактивного web-сервісу по пошуку домашніх тварин

Для реалізації даного проекту було використано клієнт-серверну архітектуру.



Рис. 3.1. Клієнт-серверна архітектура

Архітектура клієнт-сервер є одним із архітектурних шаблонів програмного забезпечення та є домінуючою концепцією у створенні розподілених мережевих застосувань і передбачає взаємодію та обмін даними між ними. Вона передбачає такі основні компоненти:

* набір серверів, які надають інформацію або інші послуги програмам, які звертаються до них;
* набір клієнтів, які використовують сервіси, що надаються даними серверами;
* мережа, яка забезпечує безперебійну взаємодію між клієнтами та серверами.

***Сервер* –** це набір програм, які контролюють виконання різних процесів. Відповідно, цей набір програм встановлений на будь-якому комп'ютері. Часто комп'ютер, на якому встановлено сервер, і називають сервером. Основна функція комп'ютера-сервера - по запиту клієнта запустити який-небудь певний процес і відправити клієнту результати його роботи.

***Клієнтом*** називають будь-який процес, який користується послугами сервера. Клієнтом може бути як користувач, так і програма. Основне завдання клієнта - виконання програми та здійснення зв'язку з сервером, коли цього потребує програма. Тобто клієнт повинен надавати користувачеві інтерфейс для роботи з додатком, реалізовувати логіку його роботи і при необхідності відправляти завдання серверу.

Для реалізації веб-сервісу по пошуку домашніх тварин був обраний фреймворк Vue який використовує архітектурний патерн MVVM.

В класичному варіанті, MVVM складається з трьох частин, які і дали йому назву (рис.3.2). Розглянемо їх особливості більш детально:

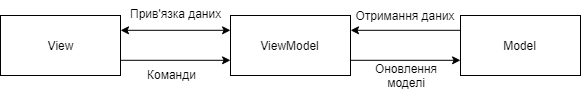


Рис.3.2. Узагальнена архітектура MVVM

MVVM полегшує відокремлення розробки графічного інтерфейсу від розробки бізнес логіки (бек-енд логіки), відомої як модель (можна також сказати, що це відокремлення представлення від моделі). Модель представлення є частиною, яка відповідає за перетворення даних для їх подальшої підтримки і використання. З цієї точки зору, модель представлення більше схожа на модель, ніж на представлення і оброблює більшість, якщо не всю, логіку відображення даних. Модель представлення може також реалізовувати патерн медіатор, організовуючи доступ до бек-енд логіки навколо множини правил використання, які підтримуються представленням.

Шаблон MVVM ділиться на три частини:

1. Модель (Model), як і в класичному шаблоні MVC, Модель являє собою фундаментальні дані, що необхідні для роботи застосунку.
2. Вид/(Вигляд) (View) як і в класичному шаблоні MVC, Вигляд — це графічний інтерфейс, тобто вікно, кнопки тощо.
3. Модель вигляду (ViewModel, що означає «Model of View») з одного боку є абстракцією Вигляду, а з іншого надає обгортку даних з Моделі, які мають зв'язуватись. Тобто вона містить Модель, яка перетворена до Вигляду, а також містить у собі команди, якими може скористатися Вигляд для впливу на Модель. Фактично ViewModel призначена для того, щоб:

* Здійснювати зв'язок між моделлю та вікном.
* Відслідковувати зміни в даних, що зроблені користувачем.
* Відпрацьовувати логіку роботи View (механізм команд).

Таким чином, можна зробити висновок, що даний програмний продукт використовує архітектурні шаблони, що забезпечує технологічні можливості для побудови гнучкого додатку із дотриманням принципів ООП. Такий підхід надає можливість на успішну реалізацію проекту та подальшу його модифікацію та розвиток.

## Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення

Для створення інтерактивного web-сервісу по пошуку домашніх тварин було обрано найпоширеніші інструментальні засоби такі як: мову розмітки гіпертекстових документів HTML5, каскадні таблиці стилів CSS3, браузерну мову програмування JavaScript, прогресивний фреймворк для створення користувацьких інтерфейсів Vue, веб-сервер Apache, графічний редактор Adobe Photoshop CC 2017 та середовище розробки Visual Studio Code 1.28.2.

***HTML*** ***(HyperText Markup Language, мова розмітки гіпертексту)*** − це стандарт верстки веб-сторінок, який визначає, які елементи і як повинні розташовуватися в документі.

***CSS*** ***(Cascading Style Sheets, каскадні таблиці стилів)*** – формальна мова опису зовнішнього вигляду документа, написаного з використанням мови розмітки.

***JavaScript*** − динамічна мова програмування. Підтримує об'єктно-орієнтовані, імперативні і функціональні стилі. Є реалізацією мови ECMAScript (стандарт ECMA-262).

JavaScript зазвичай використовується як вбудована мова для програмного доступу до об'єктів додатків. Найбільш широке застосування знаходить в браузерах як мова сценаріїв для додання інтерактивності веб-сторінок.

***Vue*** - це прогресивний фреймворк для створення користувацьких інтерфейсів. На відміну від фреймворків-монолітів, Vue створений придатним для поступового впровадження. Його ядро в першу чергу вирішує завдання рівня уявлення (view), що спрощує інтеграцію з іншими бібліотеками та існуючими проектами. З іншого боку, Vue повністю підходить і для створення складних односторінкових додатків (SPA, Single-Page Applications), якщо використовувати його спільно з сучасними інструментами та додатковими бібліотеками.

Існує багато прогресивних фреймворків які підходять для реалізації такого веб-сервісу, серед них найпопулярніші це Angular 2, React та Vue. Щоб визначитись який з них краще, потрібно ї порівняти.

Таблиця 4.1

Порівняння прогресивних фреймворків

|  |  |  |  |
| --- | --- | --- | --- |
| Пункти вирішення | Angular 2 | React | Vue |
| Стабільність | Так | Так | Так |
| Підтримка сильною спільнотою | Так, за ним стоїть величезна спільнота та компанія Google. | Так, величезна спільнота та Facebook підтримує його. | Не такака величезна, але досить велика, і його підтримують Laravle та Alibaba. |
| Хороша документація | Так | Так | Так |
| Легко вивчити | Ні, потрібно знати Typescript | Частково | Так |
| Інтеграція з Bootstrap | Так | Так | Так |
| Розмір | 566К | 139К | 58.8К |
| Можливість повторно використовувати код | Так | Ні, тільки CSS | Так, HTML та CSS |
| Швидкість програмування | Повільно | Нормально | Швидко |
| Реактивність | Частково | Так | Так |
| Компонента основа | Так | Так | Так |
| CLI | Так | Так | Так |

Проаналізувавши табл. 4.1., можна зробити висновок, що всі фреймворки хороші але один з них являється фаворитом і це Vue. Він швидкий у вивчені, на ньому дуже швидко програмувати, він займає дуже мало розміру, що дуже важливо на даний момент, так як більшість користувачів інтернету використовують свій смартфон для доступу до інтернету. В наш час не дуже швидкий швидкий мобільний інтернет, тому мала розмірність є дуже важливою перевагою.

***Vuex*** - патерн управління станом + бібліотека для додатків на Vue.js. Він служить централізованим сховищем даних для всіх компонентів програми з правилами, що гарантують, що стан може бути змінено тільки передбачуваним чином. Vuex інтегрується з офіційним розширенням vue-devtools, надаючи «з коробки» такі передові можливості, як «машину часу» для налагодження і експорт/імпорт зліпків стану даних.

***Apache*** – це найбільш розповсюджений веб-сервер, який являється безкоштовним та призначений для використання під управлінням більшості відомих операційних систем. Apahce у якості протоколу передачі даних використовує протокол ***HTTP***.

Порівняльний аналіз СКБД дозволяє раціонально вибрати систему керування базами даних для проекту. У якості альтернатив будуть розглянуті наступні СКБД: Firebase Realtime Database, MongoDB, MySQL. Всі бази безкоштовні, Firebase Realtime Database (частково).

Усі обрані системи керування базами даних підходять для проведення аналізу й порівняння тому що їх можна використовувати с фреймворком Vue.

Таблиця 4.2

Аналіз загальних показників СКБД

|  |  |  |  |
| --- | --- | --- | --- |
| Показники | Firebase Realtime Database | MongoDB | MySQL |
| Модель первинної бази даних | Сховище документів | Сховище документів | Реляційна |
| Вторинна модель баз даних | - | Сховище ключів | Сховище документів та ключів |
| Доступна тільки як хмарна служба | Так | Ні | Ні |
| Серверні операційні системи | Хостинг | Linux, OS X, Solaris, Windows | FreeBSD, Linux, OS X, Solaris, Windows |
| Схема данных | Без схеми | Без схеми | Так |
| Зумовлені типи даних | Так | Так | Так |

Продовження таблиці 4.2

|  |  |  |  |
| --- | --- | --- | --- |
| Вторинні індекси | Так | Так | Так |
| Підтримка SQL | Ні | Запити тільки для читання SQL-запитів через коннектор MongoDB для BI | Так |
| API і інші методи доступу | Android, IOS, JavaScript API, RESTful HTTP API | Пропріетарний протокол з використанням JSON | Власний API, ADO.NET, JDBC, ODBC |
| Мови програмування | Java, JavaScript, Objective-C | C, C#, C++, Java, JavaScript, PHP, Lua, Python та ін. | C, C#, C++, Java, JavaScript, PHP, Perl, Python та ін. |
| Збережені процедури | обмежена функціональність з використанням «правил» | JavaScript | Так |
| Тригери | Зворотні виклики запускаються при зміні даних | Ні | Так |
| Транзакції | Так | Багатодомені транзакції ACID з ізоляцією моментальних знімків | ACID |

Проаналізувавши табл. 4.2 можна зробити висновок, що для даного веб-сервісу підходить Firebase Realtime Database.

***База даних Firebase Realtime*** є хмарною базою даних. Дані зберігаються як JSON і синхронізуються в реальному часі з кожним підключеним користувачем. Коли створюється крос-платформені додатки з iOS, Android і JavaScript SDK, всі користувачі діляться одним екземпляром бази даних Realtime і автоматично отримують оновлення з новітніми даними.

## [Визначення](file:///C:\Users\Anna\AppData\Roaming\Skype\бакалаврат\fragment_bakalavr.docx#_Toc212894719) варіантів використання та об’єктно-орієнтованої структури системи

Автоматизований веб-сервіс для пошуку домашніх тварин має основні цілі: проект створюється з метою пошуку втрачених домашніх тварин або знаходженні господаря тварини.

**Вимоги користувачів**

Зовнішні користувачі мають доступ до наступних функцій:

1. Перегляд основної інформації про сайт, його основні сторінки.

2. Реєстрація на сайті.

3. Аутентифікація на сайті.

4. Створення, редагування та видалення об’яв.

5. Фільтрація об’яв за різними критеріями.

6. Перегляд огошень на карті.

7. Локалізація трьома мовами: англійська, російська та українська.

Внутрішні користувачі – адміністратори сервісу мають доступ до додаткових функцій, таких як:

1. Редагування БД.

2. Редагування дизайна сервісу.

3. Модерація оголошень.

4. Перегляд статистики сервісу.

5. Монетизація сервісу.

З діаграми прецендентів яка зображена на рис. А.1 ми бачимо, що для звичайного користувача даний програмний продукт буде доступний не повністю. Йому буде доступно лише перегляд оголошень та фільтрація. Для того щоб він був доступний повністю користувачу потрібно зареєструватись.

Після реєстрація він стає зареєстрованим користувачем і вже зможе створювати, редагувати та видаляти свої оголошення про знайдену/загублену домашню тварину. Також якщо у зареєстрованого користувача щось не виходить він може звернутися за допомогою до модератора, який в свою чергу відповість йому на поставлені запитання.

В даному програмному продукті є модератор який перевіряє добавлені оголошення користувачів. Він може прийняти або відхилити об’яви користувачів які не є коректні.

Також є адміністратор який слідкує щоб програмний продукт працював швидко та стабільно. Він планує стратегію по просуванню веб-сервісу в мережі інтернет. Аналізує відвідуваність, залучення відвідуваності, конверсію. Розробляє, підтримує концепцію веб-сервісу, покращує його роботу. Удосконалює та виправляє поточні структури веб-сервісу. При появі помилок він швидко виправляв їх, щоб вони не впливали на процес роботи користувачів з програмним продуктом.

***Функціональні вимоги:***

1. Авторизація користувачів в системі: У веб-сервісі для пошуку домашніх тварин має бути можливість реєстрації та авторизації користувача, при повторному відвідуванню сайту та присвоєння йому відповідної ролі (зареєстрований користувач, модератор, адміністратор).

2. Збереження нових даних введених користувачами: Система повинна зберігати нові оголошення при створені, редагуванні та при редагуванні інформації про користувача. Також надавати можливість користувачу шукати та переглядати об’яви.

3. Модерація: Модератор має можливість переглянути, схвалити або відхилити оголошення.

4. Реалізація фільтрації за різними критеріями оголошень на сайті.

5. Валідація введених даних.

***Нефункціональні вимоги:***

1. Навчання роботи з системою:

* час, потрібний для навчання з інструментами роботи з інформаційною системою для звичайних користувачів – 10-20 хвилин, а для досвідчених – 10 хвилин;
* час відповіді системи для звичайних запитів не повен перевищувати 2 секунди, а для більш складних запитів — 3 секунд;
* інтерфейс представлення програми повинен бути інтуїтивно зручним для користувача;
* надійність;
* доступність – час, потрібний для обслуговування системи не повинен перевищувати 2% від загального часу роботи;
* середній час безперервної роботи – 365 робочих днів;
* максимальна норма помилок та дефектів в роботі системи – 1 помилка на 1000 запитів користувача.

1. Продуктивність:

* швидке отримання результатів при здійсненні функцій програми;
* cистема повинна підтримувати мінімум 100 одночасно працюючих користувачів, пов’язаних з спільною базою даних.

1. Можливість експлуатації:

* даний програмний продукт повинен збільшувати споживання ресурсів системи при збільшенні навантаження на нього.
* оновлення версій компонентів програми повинно автоматично встановлюватися на клієнтську частину даного програмного продукту.

Для реалізації проекту була вибрана технологія MVVM, та браузерна мова програмування JavaScript з використанням фраємворку Vue.

На рис. Б.1 наведено діаграму компонентів та класів. Для даного програмного продукту було реалізовано 13 компонентів та 3 класи.

Компонент Navigation – відноситься до навігації по сайту, він має наступні методи:

* onLogout(), викликається коли користувач припиняє сеанс роботи в якості зареєстрованого користувача;
* addModerationTab(navigationMenu), викликається щоб добавити новий пункт меню “Модерація” для модератора.

Компонент Signup – призначений для реєстріції користувачів, він має наступні методи:

* onSignup(), викликається для створення нового користувача;
* onDismissed(), викликається щоб закрити помилку, яка з’явилась при реєстрації користувача.

Компонент Signin – призначений для авторизації користувач, він має наступні методи:

* onSignin(), викликається для авторизації користувача;
* onDismissed(), викликається щоб закрити помилку, яка з’явилась при авторизації користувача.

Компонент Home – призначений для відображення головної сторінки, він має наступні методи:

* setLatLng(event), потрібен для відображення спливаючого вікна для створення оголошення;
* setNewCenter(), потрібен для задання нових координат для центру карти;
* getDistanceBetween(markerPosition), потрібен для розрахунку дистанції між маркером на карті і центром карти;
* getCurrentLocation(), потрібен для визначення поточного місця геолокації;
* setNewCenterPosition(position), потрібен для задання початкового центру на карті;
* getAdverts(), потрібен для отримання всіх оголошень, які підійшли під фільтр;
* filterRadius(adverts), фільтрує оголошення за введеним радіусом.

Класс Store – призначений для зберігання стану програми, також він служить центральним сховищем даних для всі компонентів програми з правилами, що гарантують, що стан може бути змінено тільки передбачуваним чином, він має наступні методи:

* loadPetTypes(), потрібен для завантаження видів тварин з БД;
* loadPetBreeds(), потрібен для завантаження порід тварин з БД;
* loadPetColors(), потрібен для завантаження кольорів тварин з БД;
* loadPetColorings(), потрібен для завантаження забарвлень тварин з БД;
* loadAdverts(), призначений для завантаження всіх оголошень з БД;
* createAdvert(advertData), викликається для створення оголошення;
* updateAdvert(payload), викликається для оновлення даних про оголошення;
* deleteAdvert(id), викликається для видалення оголошення з відповідним id;
* updateInfoWindow(payload), викликається для оновлення даних в спливаючому вікні з інформацією про оголошення;
* editAdvert(payload), викликається для зміни даних в спливаючому вікні при оновлені даних про оголошення;
* updateAdvertFilter(payload), викликається для оновлення даних фільтра;
* autoSign(payload), викликається для автоматичної авторизації користувача;
* signUserUp(payload), викликається для реєстрації користувача;
* signUserIn(payload), викликається для авторизації користувача;
* loadUserData(userId), потрібен для завантаження інформації про користувача з БД;
* updateLocale(payload), викликається для оновлення локалізації в програмі;
* logout(), викликається для припинення сеансу роботи в якості зареєстрованого користувача.

Компонент AdvertsList – призначений для відображення списку оголошень, він має наступні методи:

* onClosePopup(), викликається для закриття спливаючого вікна редагування оголошення;
* onOpenEditPopup(advert), викликається для відкриття спливаючого вікна редагування оголошення;
* onDeleteAdvert(id), викликається для видалення оголошення з відповідним id;
* onApproveAdvert(id), викликається для схвалення оголошення модератором;
* onRejectAdvert(id), викликається для відхилення оголошення модератором.

Компонент AdvertPopup – призначений для відображення спливаючого вікна створення/редагування оголошення, він має наступні методи:

* createMarker(), викликається для створення оголошення;
* onUpdateAdvertInfo(), викликається для оновлення інформації про оголошення;
* hidePopup(), викликається для закриття спливаючого вікна;
* loadedPetBreeds(petType), викликається для завантаження інформації порід тварини певного типу;
* onChangePetType(value), викликається при змінні типу тварину;
* onCancelEdit(), викликається для відхилення змін в оголошенні.

Таким чином, дана система реалізує функціонал, що передбачає реєстрацію та авторизацію користувачів, створення оголошень. Також містить методи редагування та видалення цих даних та модерування оголошень модератором. Даний функціонал передбачає фільтрацію оголошень за різними критеріями.

## [Розробка бази](file:///C:\Users\Anna\AppData\Roaming\Skype\бакалаврат\fragment_bakalavr.docx#_Toc212894723) даних системи

Реалізована база даних Firebase Realtime складається з 6 документів, які містять у собі всі дані для роботи програми.

Структура бази даних наведена на рис. В.1.

Документ «users» призначений для зберігання інформації про користувачів, він має наступні поля:

1. id – унікальний ідентифікаційний номер користувача;
2. email – електрона пошта;
3. password – пароль;
4. role – роль користувача.

Документ «adverts» призначений для зберігання інформації про оголошення, він має наступні поля:

1. id – унікальний ідентифікаційний номер оголошення;
2. id\_pet\_type – унікальний ідентифікаційний номер виду тварини;
3. id\_pet\_breed – унікальний ідентифікаційний номер породи тварини;
4. id\_pet\_color – унікальний ідентифікаційний номер кольору тварини;
5. id\_pet\_coloring - унікальний ідентифікаційний номер забарвлення тварини;
6. id\_user - унікальний ідентифікаційний номер користувача;
7. typeMarker – тип оголошення;
8. petAge – вік тварини;
9. status – статус оголошення;
10. position – позиція оголошення на карті;
11. contactInfo – контактна інформація;
12. dateCreate – дата створення оголошення.

Документ «pet\_breeds» призначений для зберігання інформації про породи тварини, він має наступні поля:

1. id – унікальний ідентифікаційний номер породи тварини;
2. id\_pet\_type - унікальний ідентифікаційний номер виду тварини;
3. en – назва на англійській мові;
4. ru – назва на російській мові;
5. ua – назва на українській мові.

Документ «pet\_colorings» призначений для зберігання інформації про забарвлення тварини, він має наступні поля:

1. id – унікальний ідентифікаційний номер забарвлення тварини;
2. en – назва на англійській мові;
3. ru – назва на російській мові;
4. ua – назва на українській мові.

Документ «pet\_colors» призначений для зберігання інформації про кольори тварин, він має наступні поля:

1. id – унікальний ідентифікаційний номер кольору тварини;
2. en – назва на англійській мові;
3. ru – назва на російській мові;
4. ua – назва на українській мові.

Документ «pet\_types» призначений для зберігання інформації про види тварин, він має наступні поля:

1. id – унікальний ідентифікаційний номер виду тварини;
2. en – назва на англійській мові;
3. ru – назва на російській мові;
4. ua – назва на українській мові.

ВИСНОВКИ

Під час виконання даної роботи було зроблено такі ключові висновки:

1. Здійснено аналіз напрямків використання інформаційних технологій для автоматизації роботи програмного продукту. Проведений аналіз предметної області дозволив правильно поставити основну задачу веб-сервісу для пошуку домашніх тварин та необхідність реалізації засобів управління контентом сайту, написання функціональних модулів, створення, редагування, видалення, фільтрація оголошень.
2. Проаналізовано аналоги програмного продукту, такі як: «Всеукраинское виртуальное онлайн-бюро находок «Верни»», PETSI, LOSTPETS, громадська організація «LUCKFIND» та визначено, що основним рисами нової системи мають бути наявний веб-інтерфейс, модуль створення, редагування та видалення оголошень, модуль модерації, модуль локалізації, модуль фільтрації та реєстрації з наступною авторизацією.
3. Обрано архітектурні шаблони – клієнт-серверний та паттерн MVVM. Встановлено, що даний програмний продукт використовує архітектурні шаблони, які забезпечують технологічні можливості для побудови гнучкого додатку із дотриманням принципів ООП.
4. Для реалізації було вибрано наступні засоби: мову розмітки гіпертекстових документів HTML5; каскадні таблиці стилів CSS3; браузерну мову програмування JavaScript; фреймворк Vue; середовище керування базами даних Firebase Realtime; графічний редактор Adobe Photoshop CC 2017; середа розробки Visual Studio Code 1.28.2.
5. Визначено функціонал, який передбачає реєстрацію та авторизацію користувачів, створення, редагування та видалення оголошень. Веб-сервіс містить методи модерації та фільтрації оголошень.
6. Створено базу даних у відповідності до вимог реляційної моделі та встановили зв'язок за ідентифікаційними полями. База даних складається з 6 документів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

**Книги:**

1. Петюшкин А. В. «HTML. Экспресс-курс», – СПб.: БХВ – Петербург, 2003,– 256 с.
2. Шеллі Пауерс, Вивчаючи Javascript, М.: МетаТехнология, 2006, 240 с.
3. С. Стефанов, Шаблони Javascript. СПб.: БХВ, Петербург, 2013, 124 с.
4. Д. Герман, Ефективний Javascript. М.: БИНОМ, 2016, 236 с.
5. М. Фаулер, К. Скотт. UML в кратком изложении. Применение стандартного языка объектного моделирования. М: «Мир», 1999, 191 с.
6. М. Гленфорд, Т. Баджетт, С. Кори, Искусство тестирования программ, 3-е изд. Москва: Дилектика, 2012, 345 с.
7. Мартин Фаулер. Шаблоны корпоративных приложений (Signature Series) = Patterns of Enterprise Application Architecture (Addison-Wesley Signature Series). — М.: «Вильямс», 2012. — 544 с.
8. Э. Гамма, Р. Хелм, Р. Джонсон, Дж. Влиссидес. Приемы объектно-ориентированного проектирования. Паттерны проектирования = Design Patterns: Elements of Reusable Object-Oriented Software. — СПб: «Питер», 2007. — С. 366.
9. Стив Макконнелл. Совершенный код = Code complete. — СПб.: Питер, 2005. — С. 896.
10. Джозеф Шмуллер. Освой самостоятельно UML 2 за 24 часа. Практическое руководство = Sams Teach Yourself UML in 24 Hours, Complete Starter Kit. — М.: Вильямс, 2005. — 416 с.

**Електронні ресурси:**

1. Довідниково-інформаційний портал htmlbook [Електронний ресурс]. Режим доступу до ресурсу: http://htmlbook.ru/.
2. Современный учебник Javascript [Електронний ресурс]. Режим доступу до ресурсу: http://learn.javascript.ru.
3. Vue прогресивний JavaScript-фреймворк [Електронний ресурс] – Режим доступу до ресурсу: https://vuejs.org.
4. Що таке Vuex? [Електронний ресурс] – Режим доступу до ресурсу: https://vuex.vuejs.org.
5. Material Design Component Framework [Електронний ресурс] – Режим доступу до ресурсу: https://vuetifyjs.com.
6. Vue Router: Введення [Електронний ресурс] – Режим доступу до ресурсу: https://router.vuejs.org.
7. Vue I18n [Електронний ресурс] – Режим доступу до ресурсу: https://kazupon.github.io/vue-i18n.
8. Документація webpack [Електронний ресурс] – Режим доступу до ресурсу: https://webpack.js.org.
9. Babel - The compiler for next generation JavaScript [Електронний ресурс] – Режим доступу до ресурсу: https://babeljs.io.
10. Документація firebase [Електронний ресурс] – Режим доступу до ресурсу: https://firebase.google.com/docs.
11. Google Maps API - Google Developers [Електронний ресурс] – Режим доступу до ресурсу: https://developers.google.com/maps/documentation.
12. DB-Engines - Knowledge Base of Relational and NoSQL Database Management Systems [Електронний ресурс] – Режим доступу до ресурсу: https://db-engines.com.

# ДОДАТКИ

# ДОДАТОК А

Варіанти використання веб-сервісу для пошуку домашніх тварин



# ДОДАТОК Б

Діаграма компонентів та класів

# ДОДАТОК В

Структурна схема БД

