МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ПОЛІСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет обліку та фінансів
Кафедра комп'ютерних технологій
і моделювання систем
Кваліфікаційна робота
на правах рукопису

Скоробреха Олександр Дмитрович

УДК 004:351.773

КВАЛІФІКАЦІЙНА РОБОТА

Інформаційна підсистема контролю якості продукції органічного виробництва

122 «Комп'ютерні науки»

Подається на здобуття освітнього ступеня бакалавр

кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Скоробреха О.Д.

(підпис, ініціали та прізвище здобувача вищої освіти)

Керівник роботи

Ковбасюк С.В.

д.т.н., доцент

Житомир – 2021

Висновок кафедри _____
за результатами попереднього захисту: _____

Протокол засідання кафедри _____
№ ____ від «_____» _____ 20____ р.

Завідувач кафедри _____

_____ _____ _____
(науковий ступінь, вчене звання) (підпис) (прізвище, ім'я, по батькові)
«_____» _____ 20____ р.

Результати захисту кваліфікаційної роботи

Здобувач вищої освіти _____ захистив (ла)
(прізвище ,ім'я, по батькові)
кваліфікаційну роботу з оцінкою:

сума балів за 100-бальною шкалою _____
за шкалою ECTS _____
за національною шкалою _____

Секретар ЕК

_____ _____ _____
(науковий ступінь, вчене звання) (підпис) (прізвище, ім'я, по батькові)

АНОТАЦІЯ

Скоробреха Олександр Дмитрович. Інформаційна підсистема контролю якості продукції органічного виробництва. – *Кваліфікаційна робота на правах рукопису.*

Кваліфікаційна робота на здобуття освітнього ступеня бакалавра за спеціальністю 122 «Комп'ютерні науки». – Поліський національний університет, Житомир, 2021.

В кваліфікаційній роботі викладено етапи реалізації прототипу програмного продукту для контролю якості продукції органічного виробництва.

Дослідження галузі контролю якості продукції показало, що на ринку представлено велику кількість програмних комплексів для ведення обліку основних якісних характеристик продукції та створення звітів з діаграмами та графіками. Усі програмні продукти є дорого вартісними та складними в налаштуванні під ведення конкретного типу продукції. Багато з програм вимагають використання лише однієї конкретної операційної системи, тому значно звужує потенційну аудиторію користувачів. Виходячи з такої ситуації на ринку є актуальною задачею – створення універсальної системи, яка б могла використовуватись незалежно від операційної системи чи характеристик пристрою, де вона буде використовуватись оператором.

Під час дослідження було створено прототип програмного комплексу для ведення обліку якісних характеристик продукції органічного виробництва з функціоналом автоматизованого визначення якісних показників за окремими критеріями. Прототип має панель оператора, де вносяться дані про продукцію, а також інформаційну панель, де показано графіки кількості продукції за часовою шкалою і кількість продукції в окремих категоріях. На панель виведені значення характеристики продукції за окремими критеріями оцінки якості.

Ключові слова: Інформаційна підсистема, панель оператора, органічна продукція.

ANNOTATION

Skorobrekha Alexander Dmitrovich. Information subsystem for quality control of organic products. - *Qualification work retaining on manuscript copyright.*

Qualification work for a bachelor's degree in 122 "Computer Science". – Polissia National University, Zhytomyr, 2021.

The qualification work outlines the stages of implementation of the prototype software product for quality control of organic products.

Research in the field of product quality control has shown that the market has a large number of software packages for accounting for the main quality characteristics of products and creating reports with charts and graphs. All software products are expensive and difficult to set up for a specific type of product. Many programs require the use of only one specific operating system, so it significantly narrows the potential audience of users. Based on this situation in the market is an urgent task - to create a universal system that could be used regardless of the operating system or the characteristics of the device where it will be used by the operator.

During the study, a prototype of a software package was created to keep track of the qualitative characteristics of organic products with the functionality of automated determination of quality indicators according to certain criteria. The prototype has an operator panel, where product data is entered, as well as an information panel, which shows graphs of the number of products on a time scale and the number of products in individual categories. The panel displays the values of product characteristics according to individual quality assessment criteria.

Keywords: Information subsystem, operator panel, organic products.

ЗМІСТ

Перелік умовних позначень

ППП – Прототип програмного продукту

ПО – Панель оператора

СОЯ – Система оцінки якості

СПГ – Система побудови графіків

VPS – Virtual Private Server

ВСТУП

У світі набуває популярності продукція органічної походження, зокрема такі тенденції починають прослідковуватись і в Україні. Це спричинено різко зростаючою різницею якості та безпечності продукції, які виробляються підприємствами. Органічні продукти – це не тільки овочі та фрукти, а і продукція, яка виготовляється у складі з ними. Процес визнання продукції органічною складається з аналізу дотримання стандартів виробництва та подальшого отримання сертифікату відповідності нормам.

Органічною називають продукцію, яка було виготовлена у ході сертифікованого виробництва. Сировина має бути отримана шляхом використання первинних джерел, таких як земля для вирощування культур, яка протягом більш як три роки не оброблялася з використанням елементів штучного походження. Надалі продукція, яка надходить до підприємств для переробки, має бути оброблено окремо від іншої продукції, для унеможливлення змішування. Кожен з етапів переробки має бути суворо контрольованим та мати сертифікацію відповідності нормам та стандартам.

Основними критеріями визначення якості органічної продукції є – наявність у складі продукту пестицидів, а також загальний показник чистоти. Для контролю якості вагомим значенням є показник зіпсованої продукції за різними чинниками, такими як транспортування або невідповідність умов зберігання.

Ведення статистики та обліку продукції вручну є малоефективним і тому постає необхідність використання автоматизованого програмного комплексу для вирішення питань з обліку та звітності. У випадку, коли підприємство з виробництва органічної продукції складається з багатьох філій, для ведення статистики доцільно використовувати централізовану базу, в якій будуть зберігатись усі показники за напрямами роботи підприємства та критеріями оцінки якості продукції. Використання сервера – вирішує проблему з веденням статистики та оцінки якості великих підприємств з розгалуженою структурою.

Для створення універсальності прототипу програмного продукту слід звернути увагу на те, що потенційні оператори комплексу можуть користуватися різними операційними системами, а створення версій ППП для кожної з них – довготривалий та затратний процес. Варіант використання веб – додатку для роботи з базою дає можливість використання єдиного програмного комплексу для усіх операційних систем операторів, а також значно знижує затрати на розробку та тестування такого комплексу.

Процес побудови веб-орієнтованих програмних комплексів складається з багатьох етапів, для пришвидшення їх проходження актуально використати набір інструментів та бібліотек для розробників від провідних компаній. Варіант використання систем для побудови взаємозв'язків з базою даних на основі використання уніфікованих систем, що призначені для динамічного генерування вмісту на основі запиту користувача дає змогу пропустити етапи налагодження алгоритмів запитів до полів в таблиці на вибірку та внесення даних, а також подальшого представлення їх оператору. В подальшому для процесу розвернення інтерфейсу оператора доцільно використати набір бібліотек для побудови скриптової складової ресурсу. Основними перевагами використання готових бібліотек є те, що вони уже пройшли етапи випробування на роботоздатність з усіма додатками, де вони можуть використовуватись, а також процеси оптимізації та мініфікації коду.

Одним з важливих елементів інтерфейсу оператора та системи в цілому є можливість представлення даних з таблиць у виглядів графіків та діаграм для кращого сприйняття їх людиною. Функціонал побудови таких елементів не закладений до стандартного набору веб – функцій і тому його доводиться реалізовувати за допомогою використання кількох інструментів для розробки, а саме javascript у поєднанні з можливостями css. Команда розробників з США та Європи створила набір бібліотек для пришвидшення процесу побудови та управління графіками з спільною назвою – Chartjs. Їх варіант реалізації спирається на можливості бібліотеки jQuery третьої ревізії і дає змогу динамічно взаємодіяти з графіками та варіаціями їх відображення. У зв'язку з тим, що

графіки будуються за допомогою елементу canvas, сторінки в яких залучено використання цього функціоналу будуть мати трохи більший термін рендерингу та подальшого представлення їх клієнту і тому доцільно буде створити спрощену версію представлення елементів керування системою для підвищення ефективності роботи усієї системи.

Для кінцевого поєднання інтерфейсу оператора з системами взаємодії з базою даних є необхідність використання серверної мови програмування, а саме PHP для реалізації скриптів, які будуть виконуватись на стороні сервера та обробляти події взаємодії клієнта з базою даних та формувати контент.

Мета і задачі дослідження. Збільшення ефективності роботи з оцінки якості продукції на виробництвах органіки та побудови зручного зразку програмного продукту для використання його працівниками установи.

Виконання поставлених задач має необхідність проходження етапів:

- Аналіз складових процесу оцінювання якості органічної продукції.
- Створення набору вимог до функціоналу прототипу програми.
- Створення структури взаємодії усіх систем.
- Інтеграції додаткового функціоналу для роботи системи.
- Створення системи представлення даних оператору.
- Проведення процесу публікації елементів системи на VPS.
- Випробовування системи у критичних режимах роботи під навантаження для перевірки на стабільність роботи.

Об’єкт дослідження: хід розробки системи для процесу моніторингу якості на виробництві органічної продукції.

Предмет дослідження: варіації реалізації функціоналу прототипу програмного продукту з використанням додатково – залучених бібліотек та інструментів для розробки подібних систем.

Методи дослідження. Під час виконання поставлених задач було залучено загально – наукові методи дослідження для вивчення алгоритмів та інструментів

побудови системи оцінювання та контролю якості продукції на органічному виробництві. Було проаналізовано методологію побудови та впровадження універсального ППП з базою даних та публікації його у відкритий доступ. Отримані результати досліджень показали доцільність використання інструментів для швидкої побудови веб – додатків та нестандартних елементів візуальної складової інтерфейсу оператора.

Перелік публікацій за темою роботи:

1. **Скоробреха О.Д.** «ВИКОРИСТАННЯ ФРЕЙМВОРКУ CHARTS.JS ПРИ ПОБУДОВІ ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ КОНТРОЛЮ ЯКОСТІ ПРОДУКЦІЇ». МАТЕРІАЛИ І міжнародної науково-практичної конференції ІС та КІТ – 2021«Інформаційні системи та комп'ютерно інтегровані технології: ідеї, проблеми, рішення – 2021» 1 червня 2021 року. Житомир: Поліський національний університет. С. 92-93.
2. **Скоробреха О.Д.** «ІНФОРМАЦІЙНА ПІДСИСТЕМА КОНТРОЛЮ ЯКОСТІ ПРОДУКЦІЇ ОРГАНІЧНОГО ВИРОБНИЦТВА». Збірник тез конференції «Фінансове забезпечення економіки» 1 червня 2021 року. Житомир: Поліський національний університет. С. 65 - 66.

Практичне значення одержаних результатів. Створений прототип програмного продукту може використовуватись підприємствами, які займаються виробництвом та реалізацією органічної продукції.

Структура та обсяг роботи. Кваліфікаційна робота складається зі вступу, трьох розділів та висновків, списку використаних джерел з 40 пунктів. Загальний обсяг роботи становить 40 сторінки комп'ютерного тексту, 8 рисунків.

РОЗДІЛ 1

Дослідження особливостей визначення та контролю якості органічної продукції

1.1 Визначення основних етапів та чинників при оцінці якості органічної продукції

Органічне виробництво – вид діяльності, яка пов’язана з виготовленням продукції з етапами, підготовки, обробки, змішування та пов’язані з цим процедурами, наповнення, пакування, переробки, відновлення та інші зміни стану продукції, які проводиться із дотриманням стандартів визначеними законодавством для контролю органічного виробництва та регулювання органічної продукції.

Органічним вважається продукт, який не містить ГМО, агрохімікатів, пестицидів, антибіотиків, гормональних препаратів, стимуляторів росту, штучних ароматизаторів і барвників та інших синтетичних і хімічних складових[1]. Навпаки, органічний продукт містить виключно природні речовини.

Виробництво такої продукції поєднує в собі найкращі практики з огляду на збереження довкілля, рівень біологічного різноманіття[1], відновлення родючості ґрунтів, збереження природних ресурсів, добробуту тварин, розвитку сільських територій та сприяння гармонії між людиною і природою.

Дослідження вчених свідчать, що за харчовою цінністю (у тому числі вміст вітамінів), між органічною і звичайною сільськогосподарською продукцією немає суттєвої різниці. Водночас органічні продукти у порівнянні із продуктами інтенсивного сільського господарства містять більше антиоксидантів, які сприяють профілактиці хронічних захворювань, а також мають менші концентрації важких металів та пестицидів[2].

При детальному розгляді етапів оцінки якості органічної продукції слід звернути увагу на те, що основною складовою для визначення ступеню якості

продукту є вміст пестицидів та інших агрохімікатів які можуть бути наявні в продукції, тому можна вважати це основним фактором при проведені аналізу якості таких. Ще одним чинником є чистота продукції в плані візуальних та смакових якостей, які можуть впливати на оцінку якості такого продукту при поверхневому огляді. У зв'язку з тим, що продукція такого роду не містить хімікатів та інших стимуляторів підвищення якісних характеристик слід звернути увагу на особливості зберігання та транспортування таких продуктів, адже це впливає на показник кількості зіпсованих продуктів та таких, які втратили свої якості, які впливають на їх купівлю. Тому процес контролю якості продукції органічного виробництва є невід'ємною складовою діяльності компаній, які займаються виробництвом у цій ніші.

1.2 Підбір інструментів для реалізації прототипу програмного продукту

Під час аналізу етапів оцінки якості органічної продукції постала задача реалізувати функціонал автоматизованого визначення якості таких продуктів виходячи з основних факторів, які можуть на неї впливати. Для реалізації цього необхідно використовувати колекції даних з циклічним перебором усіх параметрів з подальшим усередненням отриманих результатів. Такий функціонал присутній у всіх мовах програмування, які мають підтримку ООП. На етапі визначення інструментального функціоналу видно, що є необхідність використання єдиної бази даних в таблицях якої будуть міститися поля з усією інформацією про продукцію. Використання SQL дає можливість повноцінно розвернути ієрархію таблиць на віддаленому сервері та проводити операції по внесенню інформації, а також виконувати запити на вибірку даних з таблиць. При цьому усі запити можуть проводитись з різних місць, тому система з таким варіантом реалізації буде мати змогу використовуватись децентралізовано з будь-якого місця. Для роботи з таким сховищем підходять серверні мови програмування, а саме PHP[9]. В її основі лежить функціонал для проведення запитів до сервера та таблиць з інформацію і подальшої їх обробки, тому при роботі не буде необхідності проводити будь-які звернення до серверної частини вручну. При роботі з великими об'ємами інформації основною перевагою такої

програмування є те, що вона не навантажує ресурси клієнта, а повністю або частково залучає ресурси серверу, це дає змогу швидко виконувати запити на вибірку і циклічно обробляти та представляти інформацію за запитом. Процес написання скриптів є доволі тривалим, тому доцільно буде використати систему взаємодії для генерування динамічного вмісту користувачу. Word Press надає широкий спектр інструментів для роботи веб – додатків та створення інтерактивних проектів. Для функціонування системи такого роду необхідно залучити додаткові розширення інструментів. Набір додаткових полів дає змогу створювати динамічний вміст записів у таблицях з подальшим представленням його потенційному користувачу системи. В його основі лежить низка запитів на створення та наповнення полів додатковими значеннями, а в налаштуваннях можна вказувати умови їх приєднання до певних записів у таблицях. Також функціонал системи передбачає залучення додатку для зміни інтерфейсу створення записів для більш простої взаємодії з оператором при внесенні даних. В подальшому, коли функціонал системи буде готовий доцільно буде використовувати нестандартну систему створення візуальної складової інтерфейсу оператора, яка буде змінювати представлення даних та покращувати візуальне сприйняття інформації. Для цього доцільно буде використати - HyperText Markup Language, з його допомогою можна розділяти представлення інформації на складові і приводити її до зрозумілого для браузера вигляді. На момент розробки в доступі була остання – 5 версія мови розмітки. В цій ревізії було додано функціонал підтримки нових розділювачів (тегів).

Висновки по розділу 1

Дослідження критеріїв оцінки якості органічної продукція показала необхідність використання спеціальних математичних функцій для створення функціоналу визначення ключових критеріїв якості. Для роботи з інформацією доцільно використовувати базу даних у таблицях якої буде можливість зберігати ключові параметри та характеристики продукції, а для проведення запитів на вибірку та обробку інформації доцільно залучити серверну мову програмування,

яка буде виконувати автоматизовані запити на представлення контенту та внесення змін до полів бази.

РОЗДІЛ 2

Розробка структури прототипу програмного продукту

2.1 Розробка функціональної складової системи оцінювання якості

Реалізація функціоналу автоматизованої оцінки якості продукції спирається на методи мов програмування PHP та JavaScript. При встановленні основи програмного продукту на сервері та створенні бази з таблицями відбувається процес автоматизованої побудови зв'язків між полями таблиць та побудови запитів. Загальна схема представлення таблиць з ключовими полями приведена на Рис.2.1

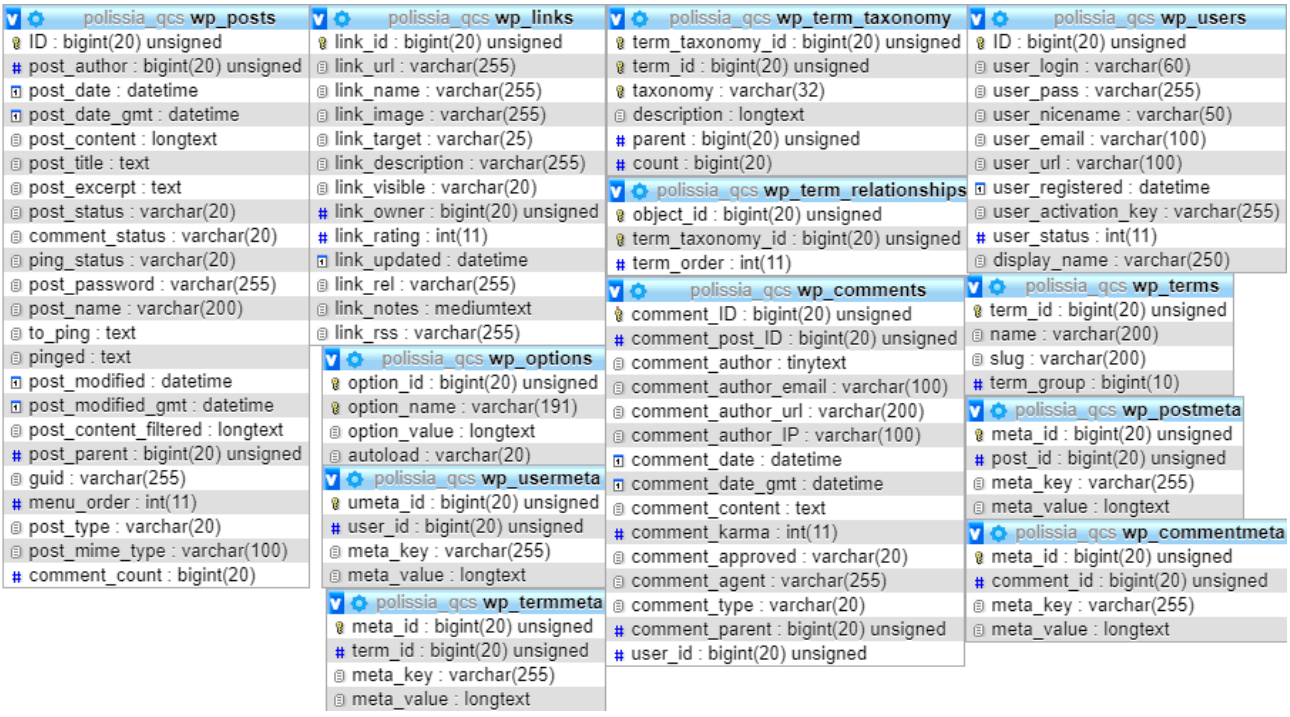

Рис.2.1 Схема таблиць бази даних

Процес отримання даних з полів відбувається автоматизовано на основі функціоналу системи керування вмістом сторінок, подальше представлення даних відбувається на основі функціоналу PHP – методів [10] для отримання та сортування даних. Для запобігання помилок та попереджень від сервера процес побудови запиту на вибірку даних починається з запиту на наявність даних по у конкретній таблиці, подальше тіло методу буде виконуватись у залежності істинності відповіді сервера про наявність даних. Подальша обробка результатів

вибірки зводиться до присвоєння унікальних ідентифікаторів тегам, в яких має зберігатися інформація. Для роботи з колекціями у Java Script існує селектор елементів за конкретним класом – getElementsByClassName. Загальне представлення рівнів доступу через окремі селекторі наведено на Рис.2.2

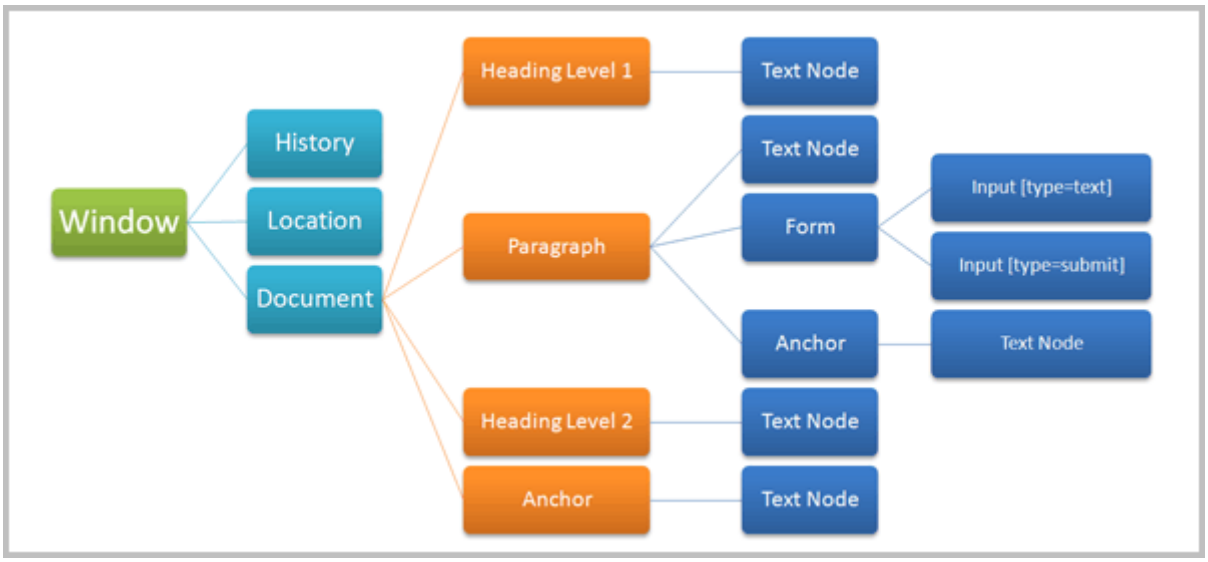

Рис. 3.2 Розподіл рівнів доступу при роботі з DOM – деревом проекту

Результатом виконання методу є масив об’єктів з усіма полями та характеристиками представлені у порядку подання їх сервером. Для створення графіків та діаграм слід створити запити на вибірку параметрів за конкретними місяцями, цей процес доцільно реалізувати за допомогою розгалужень, де кожному запису в залежності від місяця публікації буде присвоєне ім’я класу і додатковий стиль для приховування значень від відображення в інтерфейсі оператора. Робота з додатковими полями вимагає використання спеціально встановленого додатку до системи генерування вмісту, який дає змогу використовувати нестандартні запити до системи. Запити такого роду мають вигляд стандартного методу, який в якості аргументу приймає назву поля, яке необхідно витягнути з бази, а результатом повертає в місце виклику весь вміст, який зберігається в ньому. Для створення функціоналу представлення інформації за категоріями відбувається декілька запитів, які повертають колекцію записів з усіма властивостями та додатковими полями. Властивість системи керування вмістом представлення категорій з наявними в ній записами спирається на метод

- esc_html, який приймає аргумент у вигляді одного поля колекції для проведення вибірки. Надалі для формування повноцінної колекції даних для побудови графіків є необхідність отримування значення кількості записів у кожній окремій категорії. Для цього необхідно створити метод, який отримує у якості аргументу ідентифікатор категорії і в подальшому отримує доступ до колекції параметрів та полів, які в ній зберігаються. Прямого запиту на отримання кількості записів немає, тому для їх підрахунку доцільно використати цикл, який буде ітерувати записи у порядку їх подання системою керування вмістом, а по виконанні оператор – return буде повертати значення кількості елементів в поточній категорії у місце її виклику. Оформлення представлених результатів доцільно робити за допомогою рядкових тегів, які не будуть представлятися в інтерфейсу у зв'язку з накладанням властивостей відображення – none, а для виконання скриптів на отримання інформації всі значення мають унікальні ідентифікатори.

2.2 Інтеграція функціоналу API Charts.JS для побудови та відображення графіків в інтерфейсі оператора

Функціонал побудови графіків та діаграм не представлений у стандартному наборі інструментів веб – розробника, тому реалізація таких функцій лягає на створення сукупності елементів з подальшим заведенням їх до сукупного тегу – canvas, який клієнт користувача відмальовує завдяки сучасному функціоналу веб – переглядачів. Canvas API надає можливості для малювання графіки через JavaScript і HTML <canvas> – елемент. Крім усього іншого, його можна використовувати для анімації, ігрової графіки, візуалізації даних, маніпулювання зображеннями та обробки відео в режимі реального часу[3].

В основі роботи API для побудови графіків лежить методологія ООП, де процес реалізації окремих елементів системи спирається на створення окремих об'єктів з унікальними полями, які можуть приймати різні значення для кастомізації відображуваного контенту користувачу. При початку роботи з бібліотекою є необхідність задати першочергові параметри, які впливають на представлення графічного контенту оператору, до них відносяться поля:

- Назва графіку чи діаграми;
- Тип побудови графічного контенту;
- Стиль ліній;
- Розмір;
- Початок відліку;

Після задавання першочергових параметрів необхідним кроком є – створення колекції вхідних даних для подальшого представлення їх користувачу. Для цього доцільно буде використати масиви інформаційних змінних, у який зберігаються значення кількісних або якісних складових наявної продукції. Надалі процес побудови графіків зводиться до виклику стандартних методів API з передаванням їм у якості аргументів колекцій даних, які необхідно відобразити графічно. Методика отримання даних спирається на методи getElementById та getElementsByClassName при селекції окремих елементів сторінки. Суть вибірки полягає у зверненні до вмісту окремих тегів за їх унікальними ідентифікаторами або класами, це дає змогу розділяти та групувати подібну інформацію. Для реалізації адаптивності вмісту слід використовувати медіа – запити, результат виконання яких залежить від роздільної здатності клієнта оператора, тому в залежності від відповіді формується набір параметрів каскадної таблиці стилів, які визначають масштаб та дизайн відображуваних елементів користувачу. При розробці подібного функціоналу слід звернутися до документації останніх версій веб – переглядачів, в яких описано функціонал та перелік підтримуваних технологій, слід звернути увагу на те, які теги та функціональні елементи підтримує сам браузер, адже це напряму впливає на вигляд представлення окремих елементів користувачу. Для внесення розмітки на графік доцільно використовувати параметри мінімального та максимального значення відображуваної інформації, це можна реалізувати завдяки алгоритмам сортування даних, де в результаті виконання методу дані представляються у порядку зростання чи спадання, а шукані елементи знаходяться на початку та вкінці ряду даних. Після отримання значень слід знайти абсолютне значення, яке в подальшому необхідно розділити на сумарну кількість відображуваних

елементів, а після того адаптивно відобразити лінії, виходячи з результату виконання медіа – запиту. Процес налаштування та пошуку помилок при розробці інтерактивних графіків та діаграм в повній мірі спирається на роботу з консоллю розробника, адже елемент canvas не буде відображатися, якщо у процесі виконання методу конструктора цього об’єкту є хоч одна помилка або некоректно заповнене поле. При роботі з подібним функціоналом навантаження припадає на графічний процесор користувача, тому слід звернути увагу на користувачів, у який загальна потужність системи може знаходитись на низькому рівні. У ході процесу рендерингу відбувається залучення центрального та графічного процесу, основними задачами можна вважати розрахунок ключових значень та подальша побудова графіки. Цей процес досить легкий для сучасних пристроїв та займає близько 11мс. Загальну схему використання ресурсів можна розглянути на Рис. 2.3, а вихідний код у додатку А.

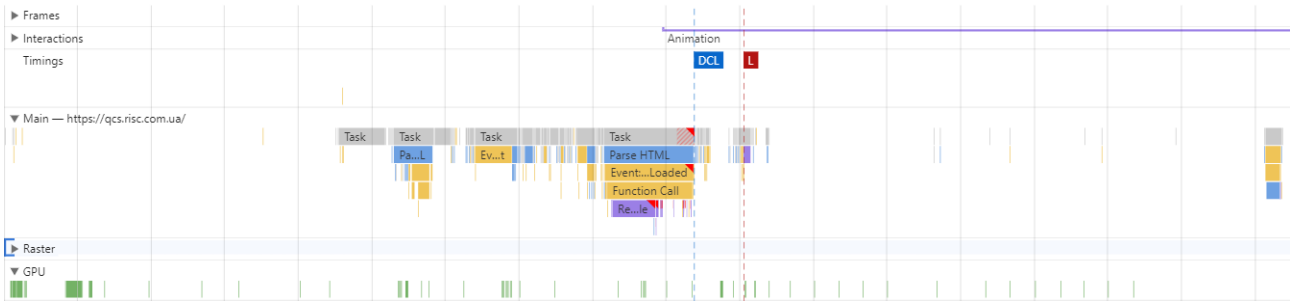

Рис.2.3 Графік використання ресурсів користувача при побудові інтерфейсу

Висновки по розділу 2

Використання інструментів та бібліотек для розширення стандартного функціоналу системи контролю та генерування вмісту дозволяє значно розширити стандартний функціонал та набір функцій. Залучення нестандартних графічних елементів, побудова яких частково або повністю спирається на використання графічного процесору, створює додаткове навантаження на ресурси користувача, але при правильному налагоджені не знижує загальну продуктивність завантаження та роботи системи.

РОЗДІЛ 3

Інтеграція прототипу програмного продукту на зовнішньому сервері

3.1 Зміна параметрів функціонування системи для роботи на віртуальному сервері

Реалізація та випробування проекту у повній мірі проходить на локальному сервері, який може бути фізичним або віртуальним. При встановленні системи керування контентом процес створення таблиць з полями та зв’язків між ними проходить автоматично, спираючись на стандартні серверні запити. В базі даних формуються стартові посилання на корінь проекту, а також створюється наповнення файлу, в якому містяться усі функціональні особливості роботи системи. В першу чергу необхідно зробити резервну копію усіх таблиць бази даних у форматі sql з кодуванням UTF-8, це дає змогу проводити маніпуляцію над вмістом у звичних текстових редакторах. При зміні адреси кореня проекту необхідно виходити з налаштувань сервера, а саме провести всі налаштування систем безпеки та інтеграція сертифікатів, а потім змінити домашню адресу проекту у файлі налаштувань на шлях до кореневого каталогу сервера, де буде розміщуватись проект. Правильність зміни налаштувань можна перевірити тільки після вивантаження усіх файлів проекту в каталог серверу. Процес вивантаження проходить при використанні стандартних протоколів передачі файлів, однак доцільно увімкнути функцію підтвердження передачі файлу аби запобігти втрати інформації. Після чого можна в пошуковому рядку веб – переглядача ввести адресу каталогу сервера з проектом, а як результат отримати попередження про неможливість отримання доступу до бази. Саму базу можна завантажити через стандартний набір інструментів СУБД – PHPMyAdmin. Особливу увагу слід приділити налаштуванням, які визначають квоту розміру передаваних файлів, адже у разі коли хоч одна з таблиць буде завантажена не у повній мірі, уся база втратить працездатність і система перестане функціонувати. У разі використання файлів контенту, які мають в своїй назві символи кирилиці, в налаштуваннях сервера потрібно увімкнути автоматичне конвертування

рядкових змінних у юнікод для запобігання конфлікту символів і втрати доступу до файлів проекту. Загальні налаштування PHP наведено на Рис.3.1

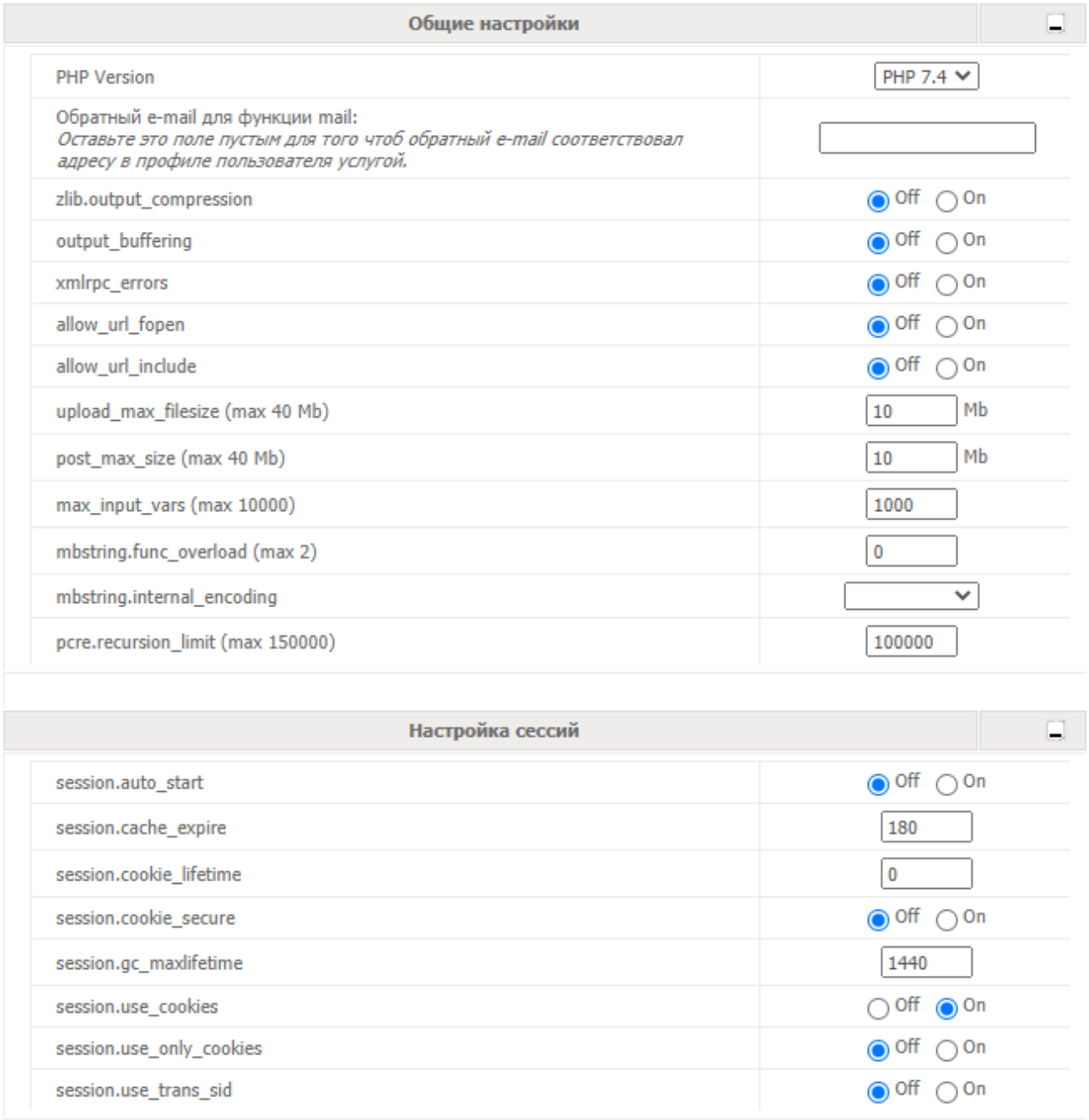

Рис.3.1 Налаштування PHP для роботи системи керування вмістом

Важливою складовою роботи системи є правильний вибір ревізії PHP, адже систему керування вмістом має функціонал автоматичного оновлення, і розробники завжди спираються на нові функції при побудові інструментів та методів роботи. На даний момент остання стабільна версія – 7.4.

3.2 Тестування роботи підсистеми контролю якості продукції органічного виробництва у критичних режимах роботи для виявлення вразливих місць

Перевірка проекту на стабільність роботи – один з ключових етапів інтеграції сервісу для подальшої його роботи. В першу чергу доцільно буде виконати перевірку швидкості представлення інтерфейсу користувача, адже це визначальний параметр при взаємодії з оператором. Виконання перевірки можна зробити при залученні сторонніх ресурсів для оцінки швидкодії веб – додатків, один з найбільш розширених звітів надає сервіс BE-ONE. Звіт швидкодії представлення інтерфейсу представлено на Рис.3.2.

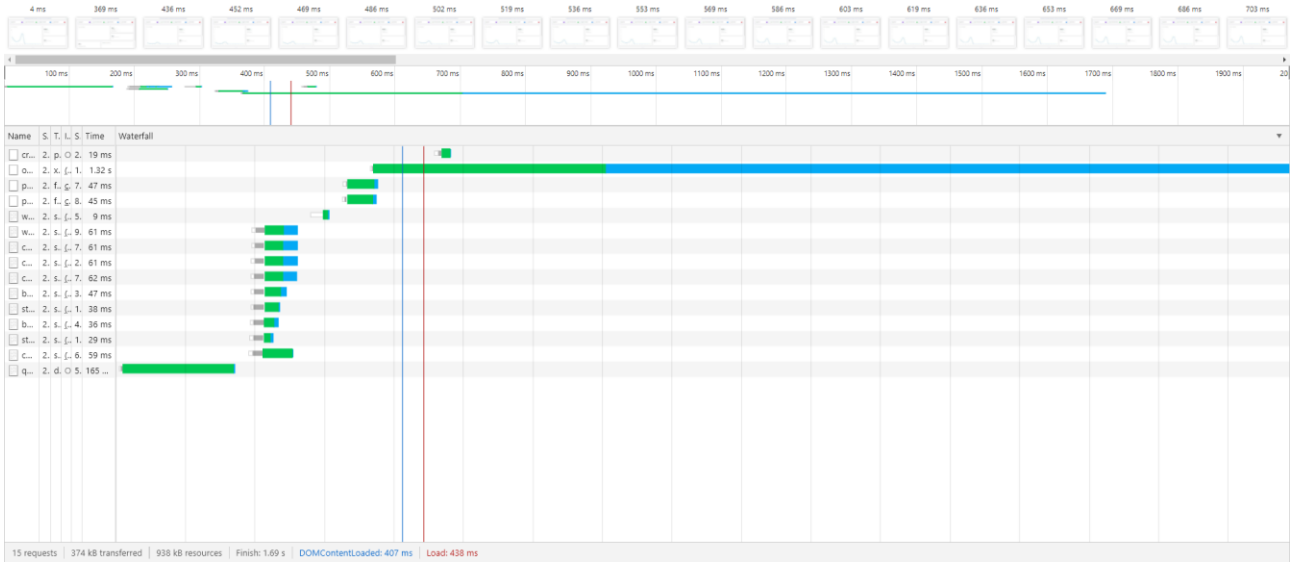

Рис.3.2 Звіт про швидкість завантаження проекту користувачем

Оцінка використання ресурсів користувача дає представлення про ступінь навантаження, яке створює сервіс на сторону клієнта і дає змогу оцінити мінімальні системні вимоги для коректної роботи прототипу програмного продукту. Повний аналіз включає в себе етапи перевірки навантаження на мережу, постійну та оперативну пам’ять та процесор користувача. Звіт випробувань представлено на Рис.3.3.

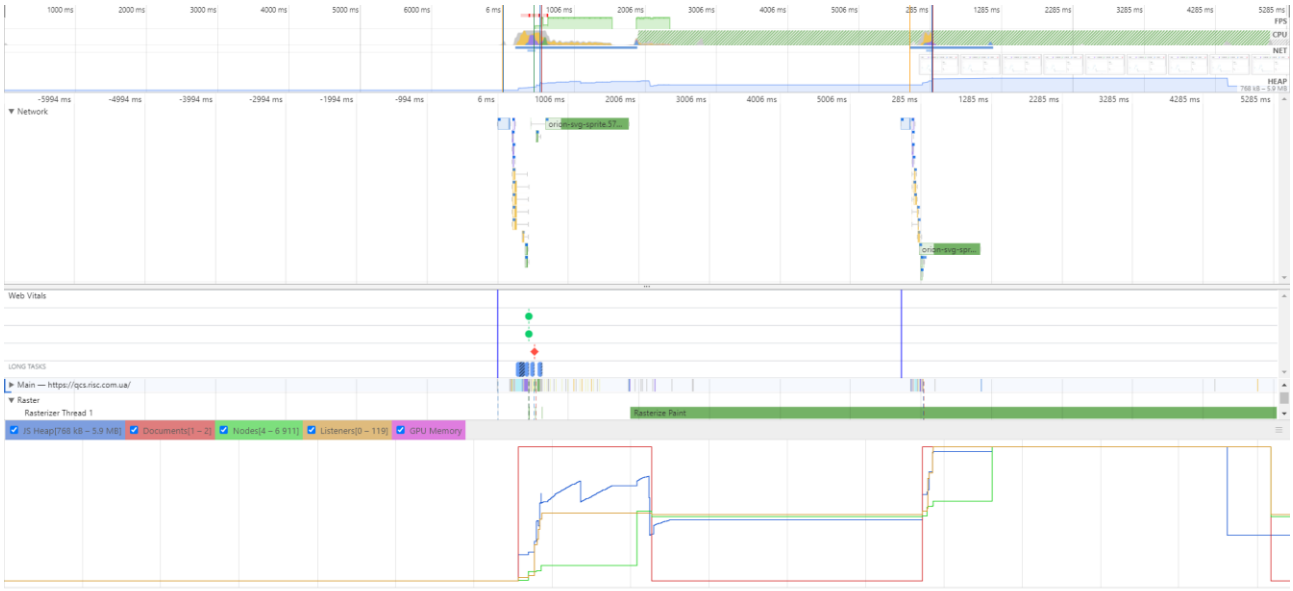

Рис.3.3 Звіт про навантаження системи користувача прототипу програмного продукту

Оцінюючи результати представленого звіту можна зробити висновки, що при роботі сервісу ресурси користувача навантажується досить слабо і найбільше навантаження відбувається на дисковий простір клієнта на протязі 24.74мс, що не є причиною для створення рекомендованих параметрів до ресурсів клієнта і свідчить про високий рівень оптимізації сервісу в цілому.

Висновки по розділу 3

Оцінка ефективності роботи системи показала пряму залежність між кількістю залучених бібліотек та навантаженням на ресурси користувача. Процес оптимізації – невід'ємна частина усіх етапів розробки та інтеграції системи. Особливої уваги потребують налаштування сервера для коректної роботи усіх інструментів та систем, а також запобігає виникненню помилок та конфліктів версій використовуваних ресурсів.

ВИСНОВОК

У ході виконання роботи було опрацьовано методологію побудови інформаційної підсистеми контролю якості продукції органічного виробництва та її інтеграції для подальшого використання користувачами. Було опрацьовано законодавчі підстави для розробки критеріїв оцінки органіки та побудовано алгоритми їх підрахунку. Реалізовано прототип програмного комплексу оператора для ведення статистики та представлення звітності про діяльність компанії, що веде діяльність у галузі виробництва органіки. У ході роботи було виконано такі етапи:

1. Проаналізовано алгоритми та критерії оцінки якості органічної продукції та реалізовано систему автоматизованого збору інформації для подальшої її представлення у вигляді вагових коефіцієнтів ключових критеріїв оцінки якості.
2. Інтегровано систему керування вмістом та підключено функціонал додаткових плагінів та бібліотек для розширення набору інструментів розробника та адміністратора ресурсу.
3. Розроблено інтерфейс оператора та підключено функціонал системи побудови графіків та діаграм на основі сторонніх бібліотек. Налагоджено автоматизовану оцінку ключових параметрів для створення градацій графіків представлення ключової інформації.
4. Проведено випробування системи на стійкість шляхом тестування в критичних режимах роботи та оцінено стабільність роботи та швидкодії. Проведено аудит системи та ресурсів користувача при їх взаємодії та представлено загальні звіти про стабільність роботи систем.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Вимоги до виробництва органічної продукції в Україні. URL: https://vetif.gov.ua/451.html. Електронний ресурс.
2. Як зміняться правила роботи для виробників та ринку органічних продуктів в Україні. URL: http://nvppoint.com/uk/yak-zminyatsya-pravila-roboti-dlya-virobnikiv-ta-rinku-organichnih-produktiv-v-ukra%D1%97ni-2/ Електронний ресурс.
3. Canvas API URL: https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API?retiredLocale=uk Електронний ресурс.
4. Руководство по PHP. Справочник языка. URL: https://www.php.net/ Електронний ресурс.
5. Chart.js URL: https://www.chartjs.org/docs/latest/ Електронний ресурс.
6. JS ES6 для начинающих. URL: https://monsterlessons.com/project/series/es6-dlya-nachinayushih . Електронний ресурс.
7. Be1.ru - проверка сайта URL: https://be1.ru/ Електронний ресурс.
8. Application,Types,Example,Advantages. URL: https://www.guru99.com/what-is-dbms.html. Електронний ресурс.
9. PHP Functions URL: https://www.w3schools.com/php/php_functions.asp. Електронний ресурс.
10. How to Define and Call a Function in PHP - Tutorial Republic URL: https://www.tutorialrepublic.com/php-tutorial/php-functions.php Електронний ресурс.
11. WebSocket - Современный учебник JavaScript. URL: https://learn.javascript.ru/websockets Електронний ресурс.
12. HTML Canvas Tutorial. URL: https://www.w3schools.com/graphics/canvas_intro.asp Електронний ресурс.
13. CSS transitions and hover animations, an interactive guide URL: https://www.joshwcomeau.com/animation/css-transitions/ Електронний ресурс.

14. A set sequence of CSS rules URL: https://marksheet.io/css-animations.html Електронний ресурс.
15. Chart.js URL: https://www.npmjs.com/package/websocket Електронний ресурс.
16. Node Packaged Masterfully URL: https://monsterlessons.com/project/series/es6-dlya-nachinayushih Електронний ресурс.
17. Web Server - Tutorialspoint URL: https://www.tutorialspoint.com/internet_technologies/web_servers.htm Електронний ресурс.
18. Install and Configure Apache. URL: https://ubuntu.com/tutorials/install-and-configure-apache#1-overview. Електронний ресурс.
19. Server: Web vs. Application. URL: https://www.javatpoint.com/server-web-vs-application. Електронний ресурс.

ДОДАТКИ

Додаток А – Вихідний код системи побудови лінійних графіків

```
"use strict";
var January = document.getElementsByClassName('January');
var JanuaryCount = 0;
for (var i = 0; i < January.length; i++) {
    JanuaryCount += Number.parseInt(January[i].innerHTML);
    console.log(January[i].innerHTML);
}
var February = document.getElementsByClassName('February');
var FebruaryCount = 0;
for (var i = 0; i < February.length; i++) {
    console.log(i);
    FebruaryCount += Number.parseInt(February[i].innerHTML);
    console.log(FebruaryCount);
}
var March = document.getElementsByClassName('March');
var MarchCount = 0;
for (var i = 0; i < March.length; i++) {
    MarchCount += Number.parseInt(March[i].innerHTML);
    console.log(MarchCount);
}
var April = document.getElementsByClassName('April');
var AprilCount = 0;
for (var i = 0; i < April.length; i++) {
```

```
    AprilCount += Number.parseInt(April[i].innerHTML);
    console.log(AprilCount);
}
var May = document.getElementsByClassName('May');
var MayCount = 0;
for (var i = 0; i < May.length; i++) {
    MayCount += Number.parseInt(May[i].innerHTML);
    console.log(MayCount);
}
var June = document.getElementsByClassName('June');
var JuneCount = 0;
for (var i = 0; i < June.length; i++) {
    JuneCount += Number.parseInt(June[i].innerHTML);
    console.log(JuneCount);
}
var July = document.getElementsByClassName('July');
var JulyCount = 0;
for (var i = 0; i < July.length; i++) {
    JulyCount += Number.parseInt(July[i].innerHTML);
    console.log(JulyCount);
}
var August = document.getElementsByClassName('August');
var AugustCount = 0;
for (var i = 0; i < August.length; i++) {
    AugustCount += Number.parseInt(August[i].innerHTML);
    console.log(AugustCount);
```

```
}
var September = document.getElementsByClassName('September');
var SeptemberCount = 0;
for (var i = 0; i < September.length; i++) {
    SeptemberCount += Number.parseInt(September[i].innerHTML);
    console.log(SeptemberCount);
}
var October = document.getElementsByClassName('October');
var OctoberCount = 0;
for (var i = 0; i < October.length; i++) {
    OctoberCount += Number.parseInt(October[i].innerHTML);
    console.log(OctoberCount);
}
var November = document.getElementsByClassName('November');
var NovemberCount = 0;
for (var i = 0; i < November.length; i++) {
    NovemberCount += Number.parseInt(November[i].innerHTML);
    console.log(NovemberCount);
}
var December = document.getElementsByClassName('December');
var DecemberCount = 0;
for (var i = 0; i < December.length; i++) {
    DecemberCount += Number.parseInt(December[i].innerHTML);
    console.log(DecemberCount);
}
var min = 10000000000000;
```

```
var max = 0;

var total = 0;

if (JanuaryCount > max) max = JanuaryCount;

if (JanuaryCount < min) min = JanuaryCount;

if (FebruaryCount > max) max = FebruaryCount;

if (FebruaryCount < min) min = FebruaryCount;

if (MarchCount > max) max = MarchCount;

if (MarchCount < min) min = MarchCount;

if (AprilCount > max) max = AprilCount;

if (AprilCount < min) min = AprilCount;

if (MayCount > max) max = MayCount;

if (MayCount < min) min = MayCount;

if (JuneCount > max) max = JuneCount;

if (JuneCount < min) min = JuneCount;

if (JulyCount > max) max = JulyCount;

if (JulyCount < min) min = JulyCount;

if (AugustCount > max) max = AugustCount;

if (AugustCount < min) min = AugustCount;

if (SeptemberCount > max) max = SeptemberCount;

if (SeptemberCount < min) min = SeptemberCount;

if (OctoberCount > max) max = OctoberCount;

if (OctoberCount < min) min = OctoberCount;

if (NovemberCount > max) max = NovemberCount;

if (NovemberCount < min) min = NovemberCount;

if (DecemberCount > max) max = DecemberCount;

if (DecemberCount < min) min = DecemberCount;
```

```
var total = JanuaryCount + FebruaryCount + MarchCount + AprilCount + MayCount + JuneCount
+ JulyCount + AugustCount + SeptemberCount + OctoberCount + NovemberCount +
DecemberCount;

document.getElementById('all').innerHTML = total + ' Тон';

var wheat = document.getElementsByClassName('Пшениця');

var wheatCount = 0;

for (var i = 0; i < wheat.length; i++) {

    wheatCount += Number.parseInt(wheat[i].innerHTML);

}

var corn = document.getElementsByClassName('Кукурудза');

var cornCount = 0;

for (var i = 0; i < corn.length; i++) {

    cornCount += Number.parseInt(corn[i].innerHTML);

}

var pomace = document.getElementsByClassName('Макуха');

var pomaceCount = 0;

for (var i = 0; i < pomace.length; i++) {

    pomaceCount += Number.parseInt(pomace[i].innerHTML);

}

var rapeseed = document.getElementsByClassName('Ріпак');

var rapeseedCount = 0;

for (var i = 0; i < rapeseed.length; i++) {

    rapeseedCount += Number.parseInt(rapeseed[i].innerHTML);

}

var sunflower = document.getElementsByClassName('Соняшник');

var sunflowerCount = 0;
```

```
for (var i = 0; i < sunflower.length; i++) {
    sunflowerCount += Number.parseInt(sunflower[i].innerHTML);
}
var soy = document.getElementsByClassName('Соя');
var soyCount = 0;
for (var i = 0; i < soy.length; i++) {
    soyCount += Number.parseInt(soy[i].innerHTML);
}
var min1 = 10000000000000;
var max1 = 0;
if (wheatCount > max1) max1 = wheatCount;
if (wheatCount < min1) min1 = wheatCount;

if (cornCount > max1) max1 = cornCount;
if (cornCount < min1) min1 = cornCount;

if (pomaceCount > max1) max1 = pomaceCount;
if (pomaceCount < min1) min1 = pomaceCount;

if (rapeseedCount > max1) max1 = rapeseedCount;
if (rapeseedCount < min1) min1 = rapeseedCount;

if (sunflowerCount > max1) max1 = sunflowerCount;
if (sunflowerCount < min1) min1 = sunflowerCount;

if (soyCount > max1) max1 = soyCount;
```

```
if (soyCount < min1) min1 = soyCount;

var total = JanuaryCount + FebruaryCount + MarchCount + AprilCount + MayCount + JuneCount
+ JulyCount + AugustCount + SeptemberCount + OctoberCount + NovemberCount +
DecemberCount;

var elementPesticide = document.getElementsByClassName('pesticide');

var counter = 0;

var pesticides = 0;

for (counter = 0; counter < elementPesticide.length; counter++) {

   pesticides += Number.parseInt(elementPesticide[counter].innerHTML);

   console.log(Number.parseInt(elementPesticide[counter].innerHTML));

}

document.getElementById('pestic').innerHTML = (pesticides / counter).toFixed(2) + '%';

var elementPolution = document.getElementsByClassName('polution');

counter = 0;

var polution = 0;

for (counter = 0; counter < elementPolution.length; counter++) {

   polution += Number.parseInt(elementPolution[counter].innerHTML);

   console.log(Number.parseInt(elementPolution[counter].innerHTML));

}

document.getElementById('polut').innerHTML = 100 - (polution / counter).toFixed(2) + '%';

var elementBad = document.getElementsByClassName('bad');

counter = 0;

var bad = 0;

for (counter = 0; counter < elementBad.length; counter++) {

   bad += Number.parseInt(elementBad[counter].innerHTML);

   console.log(Number.parseInt(elementBad[counter].innerHTML));
```

```
}
document.getElementById('bads').innerHTML = (bad / counter).toFixed(2) + '%';


var currentMonth = document.getElementsByClassName('currentMonth');
var currentMonthCount = 0;
for (var i = 0; i < currentMonth.length; i++) {
    currentMonthCount += Number.parseInt(currentMonth[i].innerHTML);
}
document.getElementById('curProd').innerHTML = currentMonthCount + 'т'
console.log('Current month count: ' + currentMonthCount);
document.getElementById('finQuality').innerHTML = (100 -
((Number.parseInt(document.getElementById('bads').innerHTML.toString().replace('%', ' ')) +
Number.parseInt(document.getElementById('pestic').innerHTML.toString().replace('%', ' ')) + (100
- Number.parseInt(document.getElementById('polut').innerHTML.toString().replace('%', ' '))) /
3)).toFixed(2) + '%';
var totalMonth = 0;
var todat = document.getElementById('curMonth').innerHTML;
console.log(todat);
if (todat == '01') {
    totalMonth = document.getElementsByClassName('January').length;
    console.log(totalMonth);
}
if (todat == '02') {
    totalMonth = document.getElementsByClassName('February').length;
    console.log(totalMonth);
}
if (todat == '03') {
```

```
    totalMonth = document.getElementsByClassName('March').length;
    console.log(totalMonth);
}
if (todat == '04') {
    totalMonth = document.getElementsByClassName('April').length;
    console.log(totalMonth);
}
if (todat == '05') {
    totalMonth = document.getElementsByClassName('May').length;
    console.log(totalMonth);
}
if (todat == '06') {
    totalMonth = document.getElementsByClassName('June').length;
    console.log(totalMonth);
}
if (todat == '07') {
    totalMonth = document.getElementsByClassName('July').length;
    console.log(totalMonth);
}
if (todat == '08') {
    totalMonth = document.getElementsByClassName('August').length;
    console.log(totalMonth);
}
if (todat == '09') {
    totalMonth = document.getElementsByClassName('September').length;
    console.log(totalMonth);
```

```
}
if (todat == '10') {
    totalMonth = document.getElementsByClassName('October').length;
    console.log(totalMonth);
}
if (todat == '11') {
    totalMonth = document.getElementsByClassName('November').length;
    console.log(totalMonth);
}
if (todat == '12') {
    totalMonth = document.getElementsByClassName('December').length;
    console.log(totalMonth);
}
console.log(document.getElementById('permonth').innerHTML);
document.getElementById('permonth').innerHTML = Number.parseInt(totalMonth);

document.getElementById('ColCat').innerHTML =
document.getElementById('colcat').innerHTML;
document.addEventListener("DOMContentLoaded", function() {

    var lineChart1 = new Chart(document.getElementById("lineChart1"), {
        type: "line",
        options: {
            tooltips: {
                mode: "index",
                intersect: false,
```

```
    callbacks: {
        label: function(tooltipItems, data) {
            return tooltipItems.yLabel.toString() + "Тон";
        },
    },
},
hover: {
    mode: "nearest",
    intersect: true,
},
scales: {
    xAxes: [{
        gridLines: {
            display: false,
            drawBorder: false,
        },
    }, ],
    yAxes: [{
        ticks: {
            max: (Number.parseInt(Number.parseInt(max) / 100) + 1) * 100,
            min: 0,
        },
        gridLines: {
            display: false,
            drawBorder: false,
        },
```

```
            }, ],
        },

        legend: {
            display: false,
        },
    },
    data: {
        labels: ["Січень", "Лютий", "Березень", "Квітень", "Травень", "Червень", "Липень",
"Серпень", "Вересень", "Жовтень", "Листопад", "Грудень"],
        datasets: [{
            label: "Кількість продукції",
            fill: true,
            lineTension: 0.4,
            backgroundColor: "transparent",
            borderColor: window.colors.blue,
            pointBorderColor: window.colors.blue,
            pointHoverBackgroundColor: window.colors.blue,
            borderCapStyle: "butt",
            borderDash: [],
            borderDashOffset: 0.0,
            borderJoinStyle: "miter",
            borderWidth: 3,
            pointBackgroundColor: "blue",
            pointBorderWidth: 5,
            pointHoverRadius: 5,
```

```
            pointHoverBorderColor: "#fff",
            pointHoverBorderWidth: 1,
            pointRadius: 0,
            pointHitRadius: 1,
            data: [JanuaryCount, FebruaryCount, MarchCount, AprilCount, MayCount, JuneCount, JulyCount, AugustCount, SeptemberCount, OctoberCount, NovemberCount, DecemberCount],
            spanGaps: false,
        }, ],
    },
});


var lineChart2 = new Chart(document.getElementById("lineChart2"), {
    type: "line",
    options: {
        tooltips: {
            mode: "index",
            intersect: false,
        },
        scales: {
            xAxes: [{
                gridLines: {
                    display: false,
                    drawBorder: false,
                },
                ticks: {
```

```
            fontColor: "#adb5bd",
        },
    }, ],
    yAxes: [{
        ticks: {
            max: Number.parseInt(Number.parseInt(max1 / 100) + 1) * 100,
            min: 0,
        },
        gridLines: {
            display: false,
            drawBorder: false,
        },
    }, ],
},
legend: {
    display: false,
},
},
data: {
    labels: ["Кукурудза", "Макуха", "Пшениця", "Ріпак", "Соняшник", "Соя"],
    datasets: [{
        label: "Кількість продукції",
        fill: true,
        lineTension: 0.4,
        backgroundColor: "transparent",
        borderColor: window.colors.primary,
```

```
            pointBorderColor: window.colors.primary,
            pointHoverBackgroundColor: window.colors.primary,
            borderCapStyle: "butt",
            borderDash: [],
            borderDashOffset: 0.0,
            borderJoinStyle: "miter",
            borderWidth: 3,
            pointBackgroundColor: window.colors.primary,
            pointBorderWidth: 5,
            pointHoverRadius: 5,
            pointHoverBorderColor: window.colors.primary,
            pointHoverBorderWidth: 1,
            pointRadius: 0,
            pointHitRadius: 1,
            data: [cornCount, pomaceCount, wheatCount, rapeseedCount, sunflowerCount,
soyCount],
            spanGaps: false,
          }, ],
      },
    });



});
```