МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Тернопільський національний економічний університет
Факультет комп’ютерних інформаційних технологій
Кафедра комп'ютерної інженерії

Баданін Роман Владленович

Програмний засіб автоматизованого тестування веб-базованої інформаційної системи / Automated Testing Software of Web-Based Information System

напрям підготовки: 123 Комп’ютерна інженерія
фахове спрямування - Комп’ютерна інженерія
Бакалаврська робота

Виконав студент групи КСМ-43/2
Баданін Роман Владленович

Науковий керівник:
Савка Н. Я.

Тернопіль - 2018

РЕЗЮМЕ

Дипломний проект містить 66 сторінок пояснюючої записки, 32 рисунок, 12 таблиць, 2 додатки та 2 аркуші формату А3.

Проаналізовано та описано етапи тестування веб-базованих інформаційних систем. Досліджено всі доступні середовища для розробки програмного модуля тестування. Обрано середовище, яке надає найбільше функціональних можливостей для розробки модулю та описано середовище для розробки програмного модуля тестування веб-додатку.

Досліджено алгоритми тестування веб-базованої інформаційної системи та розроблено алгоритм, який задовільняє всі вимоги для тестування веб-базованої інформаційної системи.

Досліджено доступні програмні модулі тестування веб-додатків, проаналізовано переваги та недоліки уже існуючих програмних рішень та на основі отриманих результатів розроблено ефективний програмний модуль тестування веб-додатку, який задовільняє поставлені відділом тестування вимоги.

Описано приклад реалізації алгоритму та досліджено ефективність модулю тестування веб-базованої інформаційної системи.

Ключові слова: АВТОМАТИЗОВАНЕ ТЕСТУВАННЯ, ІНФОРМАЦІЙНА СИСТЕМА.

RESUME

The diploma project contains 66 pages of explanatory note, 32 figures, 12 tables, 2 appendices and 2 A3 sheets.

The stages of testing web-based information systems are analyzed and described. All available environments for development of the software module of testing are investigated. The environment that provides the most functionality for the development of the module is selected and the environment for the development of the software module for testing the web application is described.

The algorithms for testing a web-based information system have been studied and an algorithm has been developed that satisfies all the requirements for testing a web-based information system.

Available software modules for testing web applications are researched, the advantages and disadvantages of existing software solutions are analyzed and based on the obtained results an effective software module for testing a web application is developed, which satisfies the requirements set by the testing department.

An example of algorithm implementation is described and the efficiency of the web-based information system testing module is investigated.

Key words: AUTOMATED TESTING, INFORMATION SYSTEM.

ЗМІСТ

ВСТУП

В сучасному світі інтернет покриває абсолютно всі галузі людської діяльності. створення сайту чи веб-додатку для свого підприємства чи компанії це вже давно не розкіш, а - необхідність. Адже на даний момент майже всі мають доступ до інтернет ресурсів, де знаходиться необхідна їм інформація про різні послуги та товари, які представляють компанії. Вся необхідна інформація в інтернет мережі представлена за допомогою веб-сторінок або веб-додатків [1].

Для того, щоб веб-додатки коректно працювали їх необхідно регулярно тестувати, для перевідки на відповідність вимогам. Саме тому тестування веб-додатків є актуальноим на даний момент, адже будь-яка компанія розробник не хоче завдавати неприємностей своїм користувачам з модулями, які не працюють або працюють не корректно, тому що виникнення помилок під час роботи з додатком призводить до втрати цінної інформації.

У компаніях, що займаються розробкою програмних продуктів, за теставання готових програмних систем відповідає відділ тестування [2]. Як свідчать проведені дослідження, велика кількість робочого часу працівників затрачається на тестування саме модуля, який відповідає за дані постачальників продукції.

Метою роботи є дослідження всіх доступних функціональних можливостей та розробити модуль тестування веб-додатку для відділу тестування. Це дозволить їм пришвидшити роботу, адже модуль покриває велику частину додатку, яка відповідає саме за роботу з постачальниками матеріалів чи послуг. Також необхідно проаналізувати роботу відділу тестування та їх завдання. Дізнатися всю необхідну інформацію про веб-додаток, який необхідно тестувати, перевірити складність тестування даного додатку. На основі отриманих даних розробити алгоритм, який дозволяє автоматизувати процес тестування модулю, який відповідає за постачальників.

1 ВЕБ-БАЗОВАНІ ІНФОРМАЦІЙНІ СИСТЕМИ ТА ЗАСОБИ ЇХ ТЕСТУВАННЯ

1.1 Інформаційні системи та їх основні характеристики

Інформаційна система забезпечує приймання інформації, її перетворення, опрацювання, збереження і передачу результатів опрацювання споживачу: людині, машині, іншій інформаційній системі. Прикладом сучасної інформаційної системи може бути редакція газети або журналу, оснащена комп'ютерною технікою.

Інформаційна система (у загальному розумінні) — це системи, яка здійснює або в якій відбуваються інформаційні процеси: пошук, збирання, зберігання, передавання й опрацювання інформації. В інформаційній системі можуть відбуватися одночасно один, два чи кілька процесів.

Опрацювання інформації залежить від змісту вхідної інформації, але під час самого опрацювання інформація не осмислюється, а лише перетворюється згідно з попередньо розробленими алгоритмами [3].

Інформаційна система (у вузькому розумінні) — це комплекс інформаційних, технічних, програмних та організаційних засобів, необхідних для автоматизованого опрацювання інформації.

В інформаційній системі відбуваються такі процеси:

- введення інформації, отриманої з джерел інформації;
- опрацювання (перетворення) інформації;
- зберігання вхідної і опрацьованої інформації;
- виведення інформації, призначеної для користувача;
- відправка / отримання інформації мережею.

Розробка інформаційної системи передбачає вирішення двох таких завдань:

- наповнення системи даними певної предметної області;

- створення (бажано графічного) інтерфейсу користувача для отримання необхідної інформації.

Дані в інформаційної системи можуть зберігати в неструктурованому або у структурованому вигляді.

Неструктуровані дані — це звичайні текстові документи (можливо, ілюстровані): статті, реферати, журнали, книги тощо. Системи, в яких зберігають неструктуровані дані, не завжди дають конкретну відповідь на запитання користувача, а можуть видати текст документа або перелік документів, у яких потрібно шукати відповідь.

Структурування даних передбачає задання правил, що визначають їхню форму, тип, розмір, значення тощо.

Бажаючи підкреслити використання електронно-обчислювальної техніки для автоматизації інформаційних процесів, сучасні інформаційні системи часто називають «автоматизованими інформаційними системами».

Як інформаційну систему можна розглянути багато об'єктів: телебачення, мережу мобільного зв'язку, цифрові фотоапарати і відеокамери, людине.

До інформаційної системи дані надходять від джерела інформації. Ці дані надсилають на зберігання в базі даних чи певного опрацювання у системі й потім передають споживачеві. Користувач системою має можливість редагувати дані, зберігати їх або використовувати з будь-якою іншою метою.

Внутрішня інформаційна основа складається з:

- засоби фіксації і збору інформації;
- засоби передачі відповідних даних та повідомлень;
- засоби збереження інформації;
- засоби аналізу, обробки і представлення інформації.

На основі описаної вище інформації я розробив алгоритм, що показує який життєвий цикл проходять данні в інформаційній системі та зобразив на рисунку 1.1.

Рисунок 1.1 - Життєвий цикл даних в інформаційній системі

Споживачем може бути людина, пристрій або інша інформаційна система. Між споживачем та власне інформаційною системою може бути встановлено зворотний зв'язок (від споживача до блоку приймання інформації) [4].

Класифікація інформаційних систем:

1) за ступенем автоматизації:

- ручні, в яких опрацювання інформації виконує людина;
- автоматизовані, в яких частину функцій (підсистем) керування або опрацювання даних здійснюють автоматично, а частину — людиною;
- автоматичні, в яких — усі функції керування й опрацювання даних здійснюють за допомого технічних засобів без участі людини;

2) за масштабом використання:

- одиночні, які реалізовано, як правило, на автономному персональному комп'ютері без обов'язкового під'єднання до комп'ютерної мережі і які містять декілька простих складових із спільним інформаційним фондом;

- групові, які орієнтовано на колективне використання інформації і найчастіше побудовано на основі локальної комп'ютерної мережі;

- корпоративні, які орієнтовано на великі компанії з підтримкою територіально віддалених комп'ютерних інформаційних вузлів і мереж. Як правило, вони мають ієрархічну клієнт-серверну структуру зі спеціалізацією серверів;

- глобальні, які охоплюють територію держави чи континенту (наприклад, Інтернет);

3) за сферою призначення (предметною галуззю, вказано лише деякі):

- економічна (функція управління на підприємстві);
- медична;
- географічна;
- адміністративна;
- виробнича;
- навчальна;
- екологічна;
- криміналістична;
- військова;

4) за місцем діяльності:

- наукові, призначені для автоматизації діяльності науковців, аналізу статистичної інформації, керування експериментом;

- автоматизованого проектування, призначені для автоматизації праці інженерів-проектувальників і розробуників нової техніки чи технологій;

- організаційного керування, призначені для автоматизації функції адміністративного (управлінського) персоналу промислових підприємств і непромислових об’єктів (банків, бірж, страхових компаній, готелів тощо) та окремими офісами (філіями);

- керування технологічними процесами, призначені для автоматизації різноманітних технологічних процесів (гнучкі виробничі процеси, металургія, енергетика тощо) [5].

Автоматизоване проектування в свою чергу допомагає здійснювати:

- розробку нових виробів і технологій їхнього виробництва;
- визначення технічних параметрів виробів;
- видаткових норм — трудових, матеріальних, фінансових;
- створення графічної документації (креслень, схем, планувань);
- моделювання проектованих об’єктів;
- створення програм для верстатів з числовим програмним керуванням.

1.2 Аналіз веб-базованих інформаційних систем

Особливості web-додатків. Всім користувачам персональних комп'ютерів відомо, що таке додаток Windows. Це одна з корисних програм, яка встановлюється на комп'ютер і працює в операційному середовищі OS Windows. Текстові та графічні редактори, медіаплейери й поштові клієнти.

Web-додаток - це комп'ютерна програма, яка може працювати тільки в браузері, як Microsoft Word працює в OS Windows. Простіше кажучи, для доступу до програми потрібно браузер та інтернет. Зберігання та обробка інформації при такій організації обчислень відбувається на віддаленому сервері, а веб-переглядач служить програмою-клієнтом і призначеним для користувача інтерфейсом. Веб-додаток отримує запит від клієнта і виконує обчислення, після цього формує веб-сторінку і відправляє її клієнту через мережу з використанням протокола HTTP. Веб-додаток может бути клієнтом інших служб, наприклад, бази даних або іншого веб-додатку, розташованого на іншому сервері.

Для web-додатку не важливо, на якій операційній системі він буде працювати. Тому, що за будь – яку роботу з веб-додатком відповідає браузер [6].

В результаті такої універсальності, постійний користувач веб-додатку може абсолютно без проблем працювати з додатком на будь-якому зі своїх девайсів. Починаючи з офісного стаціонарного комп'ютера з операційною системою Windows, закінчуючи планшетом і кишеньковим смартфоном на базі операційної системи Android чи iOS.

Всім відомі веб-сайти і соціальні мережі, по своїй суті, - справжнісінькі веб-додатки. Адже вони можуть повноцінно функціонувати тільки в ланцюжку - браузер, комп'ютер, сервер, веб-сайт [7].

Відвідувач сайта має протягом 10-15 секунд зрозуміти, як почати користуватися сайтом. Не сподівайтесь, що хтось читатиме інструкції або витратить 1-2 години на вивчення системи меню та команд.

Успіх веба багато в чому зобов'язаний тому, що велика частина теоретичних побудов, присвячених гіпертексту, була відкинута на користь простих прагматичних рішень, які і послужили основою ідеальної конструкції. RSS став, можливо, єдиним широкопоширеним веб-сервісом саме тому, що він простий. А складні корпоративні набори все ще чекають своєї години [8].

В успішних веб-проектах на сайт приходять десятки користувачів на секунду. Тому варто замислитися над оптимізацією швидкості роботи.

Через відкритий доступ до керування web-додатком, кількість хакерських атак на такі програми дуже велика. Відповідно такі системи гарно захищені.

Бази даних з мільярдами записів перестали бути екзотикою зовсім недавно, і тому ще немає інструментів для ефективного пошуку та маніпуляції такими великими кількостями інформації.

Сучасні додатки internet часто спілкуються між собою. Наприклад, сайт "електронний магазин" використовує сайт банку, щоб прийняти оплату за товар. Розважальні сайти показують рекламні блоки з інших сайтів і т.д.

Однією з головних характеристик сучасних інтернет-додатків є те, що вони поширюються у вигляді сервісу, а не товару. Це, у свою чергу, веде до фундаментальних змін в бізнес-моделях компаній-розробників програм.

Компанія повинна уміти управляти процесами. Мистецтво розробки додатків повинне супроводжуватись умінням організувати щоденні операції для підтримки роботи цих додатків. Розрив між софтом-артефактом і софтом-сервісом такий великий, що вже зараз не можна написати хороший продукт і забути про нього - його потрібно підтримувати щодня [9].

Не випадково інформація про системне адміністрування, обслуговування мереж, балансування навантаження і тому подібне охороняється Google, мабуть, навіть краще, ніж самі пошукові алгоритми. Google навчився автоматизувати згадані процеси, а це - ключова частина його цінової переваги перед конкурентами.

Відстежування поведінки користувачів в реальному часі дозволяє бачити, які нові властивості використовуються і як вони використовуються - і це ще одна ключова складова успіху технології. Веб-розробник одного з розкручених мережевих сервісів відзначає: "ми додаємо дві-три нових властивості в різні частини сайту щодня, і якщо користувачам вони не подобаються - ми відмовляємося від цих нововведень. Якщо подобаються - упроваджуємо на всьому сайті" [10].

Редактори Zdnet навіть дійшли висновку, що Microsoft не зможе перемогти Google: "бізнес-модель Microsoft побудована на припущенні, що користувач оновлює своє комп'ютерне оточення раз в два або три роки. Google же залежить від того, що новенького виявить користувач в своєму комп'ютерному оточенні сьогодні".

Системи, подібні вебу, RSS і AJAX, схожі тим, що особливих перешкод для їх повторного використання не існує. Велика частина корисного програмного забезпечення має відкриті тексти, а якщо і немає, то існує мало способів захистити свою інтелектуальну власність. Стандартна функція оглядача інтернет "преглянути вихідний код" дозволяє будь-якій людині скопіювати будь-яку веб-сторінку.

RSS був спроектований для того, щоб користувач міг читати контент тоді, коли це зручно йому, а не постачальникові інформації. Найуспішніші веб-

сервіси - це, як правило, такі служби, які можуть бути змінені несподіваним для їх творців чином (some rights reserved).

1.3 Аналіз алгоритмів тестування веб-базованих інформаційних систем

На даний момент з інструментів для тестування веб-додатків доступний Selenium — це об'єктно-орієнтований Java-додаток, який може аналізувати файли певної структури для того, щоб знаходити в них команди для маніпуляції браузером і команди для виконання певних дій і перевірок. Крім того, команди Selenium можна викликати з наступних мов програмування: Java, C#, Ruby, Haskell, JavaScript, Objective-C, Perl, PHP, Python, R [11]. Selenium підтримується Microsoft Internet Explorer, Google Chrome, Mozilla Suite і Mozilla Firefox для Microsoft Windows, Linux і Apple Macintosh.

В рамках проекту Selenium також випускається інструмент Selenium IDE, який являє собою версію досить популярної бібліотеки Selenium в GUI-оболонці. Реалізовано це у вигляді розширення до браузера Firefox, розміром близько 240 Кб, включаючи сам Selenium. Цей інструмент дозволяє записувати і відтворювати скрипти, що являють собою звичайні HTML-сторінки з однією таблицею, яка містить команди., але незважаючи на всі його переваги в нього є ряд недоліків:

- не підтримує Flash і Java додатки;
- не має підтримки циклів та умовних операторів;
- не має підтримки обробки помилок;
- відстутність технічної підтримки.

Модульне тестування (англ. Unit testing) — це метод тестування програмного забезпечення, який полягає в окремому тестуванні кожного модуля коду програми. Модулем називають найменшу частину програми, яка може бути протестованою. У процедурному програмуванні модулем вважають

окрему функцію або процедуру. В об'єктно-орієнтованому програмуванні — інтерфейс, клас. Модульні тести, або unit-тести, розробляються в процесі розробки програмістами та, іноді, тестувальниками білої скриньки (white-box testers).

Зазвичай unit-тести застосовують для того, щоб упевнитися, що код відповідає вимогам архітектури та має очікувану поведінку.

Метою модульного тестування є ізоляція кожної частини програми та впевненість у тому, що кожна окрема частина є коректною [12]. Модульний тест забезпечує жорсткий «контракт», за яким має працювати тестований код. Як результат, це надає деякі переваги. Модульне тестування допомагає знайти помилки раніше в циклі розробки ПЗ, що робить розробку дешевшою та швидшою.

Модульне тестування дозволяє програмісту, коли він буде змінювати код (проводити рефакторинг) бути впевненим, що модуль працює вірно (це — регресивне тестування). Оскільки модульне тестування вимагає написання тестів для всіх функцій та методів у програмі, помилки швидко локалізуються та виправляються.

Модульне тестування може бути застосоване в інтеграційному тестуванні: тестування окремих модулів та сукупності цих модулів робить інтеграційне тестування легшим. Однак модульне тестування знизу вгору не є інтеграційним тестуванням. Інтеграція з зовнішніми модулями має включатися до інтеграційних тестів, а не до модульних.

Модульні тести являють собою специфічний вид документації до системи. Розробники можуть подивитися на модульний тест, щоб дізнатися про функції, що виконує модуль, та як його застосовувати.

Unit-тест перевіряє критичні характеристики модуля. Відповідність чи невідповідність цим характеристикам демонструє коректність модуля. Модульний тест «документує» ці критичні характеристики, але не треба покладатися лише на код в документуванні ПЗ під час розробки.

Слід відзначити, що звичайна письмова документація дуже повільно реагує на зміни в коді, тоді як модульні тести завжди відображають поточний стан модуля.

1.4 Постановка задач дипломного проектування

На основі вище описаного матеріалу зрозуміло, що інформаційні системи необхідні для пошуку, збирання, зберігання, передавання й опрацювання інформації.

Веб-додаток погано піддається тестуванню, наприклад за допомогою фреймфорку Selenium це реалізувати складно, тому що він:

- не підтримує Flash і Java додатки;
- не має підтримки циклів та умовних операторів;
- не має підтримки обробки помилок.

Сценарій для тестування було реалізовано за допомогою TestCafe – це крос-платформенний фреймворк для функціонального тестування веб-додатків, який з легкістю дозволяє автоматизувати всі необхідні кроки для досягнення необхідного результату. Всі недоліки Selenium, TestCafe перетворює в свої переваги.

Для того, щоб реалізувати тестування модуля керування постачальниками в веб-додатку необхідно реалізувати наступні задачі:

- створення нового акаунту для постачальника;
- верифікація акаунту;
- перша логінізація та введення необхідних даних про користувача;
- редагування інформації;
- внесення каталогів з необхідною продукцією для клієнтів.

Зважаючи на це, в дипломній роботі необхідно виконати такі завдання:

- описати етапи тестування веб-базованих інформаційних систем;

- охарактеризувати середовище для розробки програмного модуля тестування веб-додатку;
- розробити алгоритм тестування веб-базованих інформаційних систем;
- розробити програмний модуль тестування веб-додатку;
- описати приклад реалізації розробленого алгоритму тестування;
- дослідити ефективність розробленого алгоритму;
- описати техніко-економічні показники доцільності розробки та впровадження проекту.

2 АЛГОРИТМ АВТОМАТИЗОВАНОГО ТЕСТУВАННЯ ВЕБ-БАЗОВАНОЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1 Етапи тестування веб-базованих інформаційних систем

Тестування веб-додатків — це процес технічного дослідження, призначений для виявлення інформації про якість продукту відносно контексту, в якому він має використовуватись. Техніка тестування також включає як процес пошуку помилок або інших дефектів, так і випробування програмних складових з метою оцінки. Може оцінюватись:

- відповідність вимогам, якими керувалися проектувальники та розробники;
- правильна відповідь для усіх можливих вхідних даних;
- виконання функцій за прийнятний час;
- практичність;
- сумісність з програмним забезпеченням та операційними системами;
- відповідність задачам замовника.

Оскільки число можливих тестів навіть для нескладних програмних компонент практично нескінченне, тому стратегія тестування полягає в тому, щоб провести всі можливі тести з урахуванням наявного часу та ресурсів. Як результат додаток тестується стандартним виконанням програми з метою виявлення багів (помилок або інших дефектів).

Тестування веб-додатку може надавати об'єктивну, незалежну інформацію про його якість, ризики відмови, як для користувачів так і для замовників [13].

Тестування може проводитись, як тільки створено виконуваний код (навіть частково завершено). Процес розробки зазвичай передбачає коли та як буде відбуватися тестування. Наприклад, при поетапному процесі, більшість тестів відбувається після визначення системних вимог і тоді вони реалізуються

в тестових програмах. На противагу цьому, відповідно до вимог гнучкої розробки, програмування і тестування часто відбувається одночасно.

Етапи тестування: функціональне, тестування зручності використання, тестування інтерфейсу, тестування сумісності, тестування продуктивності, тестування безпеки.

Функціональне тестування - перевіряє, чи реалізовані функціональні вимоги, тобто можливості додатку в певних умовах вирішувати завдання, потрібні користувачам. Функціональні вимоги визначають, що саме робить продукт, які завдання вирішує. Функціональні вимоги включають в себе:

- функціональна придатність;
- точність;
- можливість до взаємодії;
- відповідність стандартам та правилам;
- захищеність.

Тестування зручності використання - виконується з метою визначення зручності використання веб-додатку для його подальшого застосування. Це метод оцінки зручності продукту у використанні, заснований на залученні користувачів як тестувальників, випробувачів і підсумовуванні отриманих від них висновків [14].

Тестування інтерфейсу - це процес тестування продукту інтерфейсу користувача, для забезпечення його відповідності до даної специфікації. Зазначений етап включає в себе:

- аналіз вимог до інтерфейсу користувача;
- озробка тест-вимог і тест-планів для перевірки інтерфейсу користувача;
- виконання тестових прикладів і збір інформації про виконання тестів;
- визначення повноти покриття для користувача інтерфейсу вимогами;

- складання звітів про проблеми в разі незбігання поведінки системи та вимог або в разі відсутності вимог на окремі інтерфейсні елементи.

Тестування сумісності - Основною метою якого є перевірка коректної роботи продукту в певному середовищі. Середовище може включати в себе наступні елементи:

- апаратна платформа;
- мережеві пристрої;
- периферія (принтери, CD/DVD-приводи, веб-камери та ін.);
- операційна система (Unix, Windows, MacOS, …);
- бази даних (Oracle, MS SQL, MySQL, …);
- системне програмне забезпечення (веб-сервер, фаєрвол, антивірус і т. д.);
- браузери (Internet Explorer, Firefox, Opera, Chrome, Safari).

Тестування продуктивності - це тестування, яке проводиться з ціллю визначення, як швидко працює програма або її частина під деяким навантаженням. Тестування продуктивності намагається враховувати продуктивність на стадії включає:

- навантажувальне тестування;
- стресове тестування;
- тестування стабільності та надійності;
- об'ємне тестування.

Тестування безпеки - оцінка вразливості програмного забезпечення до різних атак. Цей етап являє собою процес, який використовується для визначення захисту даних інформаційної системи при збереженні функціональності. Шість основних аспектів безпеки, з яких складається тестування безпеки є: конфіденційність, цілісність, аутентифікація, доступність, авторизації і безвідмовность. Тестування безпеки як термін має кілька різних значень і може бути інтерпретована різними способами [15].

2.2 Методи тестування веб-додатку

З наведених вище етапів тестування алгоритм тесування зосереджений на тестуванні інтерфейсу, сумісності та функціональному тестуванні.

Функціональне тестування виконує такі задачі:

- ідентифікація множини функціональних вимог;
- ідентифікація зовнішніх функцій і побудова послідовностей функцій відповідно до їхнього використання;
- ідентифікація множини вхідних даних кожної функції і визначення областей їхньої зміни;
- побудова тестових наборів і сценаріїв тестування функцій;
- виявлення і подання усіх функціональних вимог за допомогою тестових наборів і проведення тестування помилок у програмі і при взаємодії із середовищем.

Функціональні тести базуються на функціях і особливостях, а також взаємодії з іншими системами, і можуть бути представлені на всіх рівнях тестування.

Тести, створювані за проектною інформацією, пов'язані зі структурами даних, алгоритмами, інтерфейсами між окремими компонентами і застосовуються для тестування компонентів і їхніх інтерфейсів. Основна мета – забезпечення повноти і погодженості реалізованих функцій і інтерфейсів між ними.

В основу комбінованого методу «чорної скриньки» і «білої скриньки» покладено розбивку вхідної області функції на підобласті виявлення помилок. Підобласть містить у собі однорідні елементи, які обробляються коректно або некоректно. Для тестування підобласті застосовується виконання програми на одному з елементів цієї області [16].

Також тест виконує перевірку на сумісність додатку з різними браузерами. Тестування на сумісність є одним з декількох типів тестування

програмного забезпечення. Виконується в системі, яка побудована на основі певних критеріїв, і яка має для виконання конкретні функції у вже існуючій установці. Сумісність системи розроблюваного додатку, наприклад, іншої системи, програми, ОС, мережі, вирішують багато речей, такі як використання системи, додатку в цьому середовищі, потреби системи та додатку і т.д [17].

Зазвичай результати тестів записують в матрицю результатів тестування — це один з документів звітності тестера. Він дає можливість наочно оцінити якість ПЗ. Матриці результатів бувають різними. Основною інформацією у рядках/стовпцях матриці є номер тестового задання та або ОС, або браузер, або прізвище тестувальника (якщо їх декілька), або дата (для визначення прогресу), або інші параметри, залежно від поставленого тестерам завдання.

Після заповнення такої матриці тестер у звіті зазначає, скільки та які рядки чи стовпці складаються лише з нулів [18]. Статус завдання, для яких у стовпці містяться лише одиниці, повинен бути Tested / Closed. Кількість нулів у стовпці визначає серйозність дефекту.

Завданням тестування інтерфейсу є виявлення помилок наступного характеру:

- помилки у функціональності за допомогою інтерфейсу;
- необроблені виключення при взаємодії з інтерфейсом;
- втрата або перекручення даних, переданих через елементи інтерфейсу;
- помилки в інтерфейсі (невідповідність проектної документації, відсутність елементів інтерфейсу).

Особливості такого тестування включають:

- тест-плани для перевірки інтерфейсу користувача, як правило, являють собою сценарії, що описують дії користувача при роботі з системою;
- сценарії можуть бути записані або природною мовою, або на формальній мові якої-небудь системи автоматизації інтерфейсу;
- виконання тестів при цьому виробляється або оператором в ручному режимі, або системою, яка емулює поведінку оператора.

- при зборі інформації про виконання тестових прикладів зазвичай застосовуються технології аналізу, які виводять на екран форми та їх елементи (у разі графічного інтерфейсу) або виводиться на екран текст (у разі текстового), а не перевірка значень тих чи інших змінних, що встановлюються програмною системою;
- під повнотою покриття користувача інтерфейсу розуміється те, що в результаті виконання всіх тестових прикладів кожен інтерфейсний елемент був використаний хоча б один раз у всіх доступних режимах;
- звіти про проблеми в інтерфейсі можуть включати в себе як опис невідповідностей, вимог і реальної поведінки системи, так і опис проблем в вимогах до інтерфейсу користувача. Основне джерело проблем в таких вимогах — тести, в яких розпливчатість формулювань і неконкретність.

2.3 Алгоритм тестування веб-базованої інформаційної системи

Забезпечення якості програмного забезпечення є невід'ємною частиною життєвого циклу розробки проекту й із ускладненням тестованих програм з'явилася необхідність автоматизувати і контролювати процес тестування. Це суттєвим чином зекономить час розроки програмної ситеми та трудові ресурси, що кінцевому результаті вплине на вартість програмного забезпечення [19].

Алгоритм тестування - це послідовність точно визначених дій, що однозначно призводять до вирішення поставленого завдання шляхом перевірки кожного кроку алгоритму на виконання поставлених задач.

Даний тест написаний за допомогою TestCafe Studio дозволяє за допомогою функціонального тестування перевірити чи реалізовані функціональні вимоги, тобто можливості додатку в певних умовах вирішувати завдання, потрібні користувачам, а також протестувати інтерфейс користувача

за допомогою тестування інтерфейсу, перевірити додаток на наявність помилок, пов’язаних з відображенням елементів на сторінці у користувача. За допомогою тестування сумісності сценарій для тестування дозволяє перевірити на можливість коректної роботи в різних браузерах таких як: Internet Explorer, Firefox, Opera, Chrome, Safari.

Блок – схему алгоритму розробленого алгоритму тестування веб-базованої інформаційної системи наведено у додатку А. Алгоритм включає послідовну кількість кроків, результат завершення першого кроку слугує для початку наступного. Розглянемо детально кожен крок роботи алгоритму.

Тестування модуля керування постачальниками включає в себе такі етапи:

- створення нового акаунту для постачальника – під час реєстрації нового користувача він вводить всю необхідну інформацію про себе та компанію, яку представляє;
- верифікація акаунту – адміністратор перевіряє правильність введених даних та дає можливість новому користувачу увійти, якщо все вірно;
- перша логінізація та введення необхідних даних про користувача – після проходження верифікації користувач входить під своїм акаунтом та вводить додаткову необхідну інформацію;
- редагування інформації – після введення всіх даних можна зайти в особистий кабінет та в разі необхідності відредагувати деякі доступні дані;
- внесення каталогів з необхідною продукцією для клієнтів – після редагування акаунту користувач підгружає в систему заздалегідь готові, каталоги в яких містять всі необхідні дані про товари чи послуги які пропонує даний постачальник.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ АЛГОРИТМУ ТЕСТУВАННЯ ВЕБ-БАЗОВАНОЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1 Структура програмного модуля тестування

В даному розділі дипломної роботи описано детально структуру програмного модуля тестування. Працівники які будуть викоритовувати модуль для тестування будуть запускати його в ручну при необхідності, за допомогою командної стрічки або алгоритм буде спрацювувати автоматично кожен раз при білді додатку, для конролю працездатності.

Діаграма варіантів використання зображена на кресленні ДП.КСМ.07238/16.00.00.000 С1, на ній показано «Працівника відділу тестування», який найчастіше буде взаємодіяти з алгоритмом тестування та варіанти використання алгоритму, а саме:

- регулярний запуск тесту – регулярний періодичний запуск алгоритму тестування для моніторингу статусу додатку;
- тестування після нововведень – обов’язкове проходження тестування після додання нових можливостей в додаток, перевірка справності нових та вже існуючих функцій;
- перевірка проекту на відповідність задачам – під час тестування алгоритм перевіряє чи всі основні функції додатку виконують своє призначення;
- перевірка коректної роботи функціоналу – алгоритм тестування перевіряє коректну роботу усіх основних функцій додатку;
- запуск алгоритму на автовиконання після білду проекту;
- алгоритм після проходження виводить результати тестування;
- окрім того, що результати можна побачити на локальній машині, сповіщення про результат виконання алгоритму приходять також на пошту всім хто має відношення до проекту.

У своєму проекті я використав агрегацію для поєднання класів, адже вона включає в себе поєднання рівноправних сутностей. UML діаграма представлена на кресленні ДП.КСМ.07238/16.00.00.001 С1.

В проекті розроблено п’ять класів:

- Registration(реєстрація);
- Vendor Approve(підтвердження постачальника);
- Vendor First Login(перший логін постачальника);
- Vendor Account Information Update(оновлення інформації на акаунті);
- Vendor Upload Catalog(загрузка каталогу постачальника).

В класі Registration представлено три поля:

- vendorName – поле в яке записується назва компанії постачальника, має стрічковий тип даних;
- email – поле для унікального емейлу постачальника, на який в подальшому буде приходити уся необхідна інформація, тип даних – стрічковий;
- firstName – ім’я представника фірми постачальника, також має стрічковий тип даних.

Та сім функцій за допомогою яких модуль тестування виконує своє призначення:

- rand () - генерує випадкову стрічку з вказаних символів та вказаної довжини;
- getRandomInt (int, int) – після вказання початкового та кіцевого значення генерує випадкове ціле число;
- randNum () – генерація випадкового необходного числа;
- GetElement () – дістає ідентифікаційний номер необхідного елемента на сторінці;
- click () – приводить у дію необхідний інтерактивний елемент;
- typeText () – заповнює поле необхідною інформацією;

- setFilesToUpload () – вибирає необхідні файли і підгружає їх в додаток.

Класс Vendor Approve містить в собі два поля:

- email – емейл необхідний для того щоб попасти на акаунт апрувера;
- pass – корректний пароль до акаунту.

А також чотири функції:

- GetElement () – дістає ідентифікаційний номер необхідного елемента на сторінці;
- click () – приводить у дію необхідний інтерактивний елемент;
- typeText () – заповнює поле необхідною інформацією;
- find () – функція для пошуку необхідних елементів на сторінці.

Клас Vendor First Login має три поля:

- email – коректний емейл для того щоб потрапити на аккаунт постачальника;
- pass - вірний пароль до аккаунту;
- questions – поля для заповнення, які будуть потрібні у разі забуття паролю до аккаунту.

Функції Vendor First Login:

- GetElement () – дістає ідентифікаційний номер необхідного елемента на сторінці;
- click () – приводить у дію необхідний інтерактивний елемент;
- typeText () – заповнює поле необхідною інформацією.

В класі Vendor Upload Catalog описано два поля:

- email – коректний емейл для того щоб потрапити на аккаунт постачальника;
- pass - вірний пароль до аккаунту.

Та сім функції:

- GetElement () – дістає ідентифікаційний номер необхідного елемента на сторінці;
- click () – приводить у дію необхідний інтерактивний елемент;

- typeText () – заповнює поле необхідною інформацією;
- rand () - генерує випадкову стрічку з вказаних символів та вказаної довжини;
- wait () – для задання часу очікування модуля тестування;
- randPaths () – генерація шляхів для завантаження каталогу;
- randEndDate () – генерація випадкового часу закінчення каталогу.

В класі Vendor Account Information Update представлено два поля:

- email – коректний емейл для того щоб потрапити на аккаунт постачальника;
- pass - вірний пароль до аккаунту.

Функції класу Vendor Account Information Update:

- GetElement () – дістає ідентифікаційний номер необхідного елемента на сторінці;
- click () – приводить у дію необхідний інтерактивний елемент;
- typeText () – заповнює поле необхідною інформацією;
- rand () - генерує випадкову стрічку з вказаних символів та вказаної довжини;
- randArea () – введення випадкового місця проживання;
- pressKey () – натиснення вказаних клавіш;
- randPhNum () – введення випадкового номера телефону;
- hover () – наведення курсору на вказаний елемент додатку;
- fixture – адаптація алгоритму тестування під визначений браузер, застосовані модулі та платформи;
- await – після виконання функцій очікує на результат який повертає, якщо функція виконалася з помилкою то await повертає місце в якому сталася помилка.

3.2 Реалізація алгоритму тестування

Данний алгоритм було реалізовано в Notepad++ та відображено на рисунку 3.3. Notepad++ — текстовий редактор, призначений для програмістів і тих, кого не влаштовує функціональність програми «блокнот», що входить до складу Windows. Notepad++ базується на компоненті Scintilla (потужному компоненті для редагування), написаному на C++ з використанням тільки Windows API і STL, що забезпечує максимальну швидкість роботи при мінімальному розмірі програми. Інтерфейс Notepad++ є багатомовним (українська мова присутня). Серед особливостей програми — підсвічування синтаксису та підтримка великої кількості мов програмування (C, C++, Java, XML, HTML, PHP, JavaScript, ASCII, Visual Basic / VBScript, SQL, Ruby, CSS, Pascal, Perl і Python), багатомовна підтримка, робота з декількома документами.

Основні можливості Notepad++:

- підсвітка синтаксису тексту залежно від мови програмування, режим якої може налаштувати користувач;
- можливість згортання блоків згідно з синтаксисом мови програмування;
- WYSIWYG (друкуєш і отримуєш те, що бачиш на екрані);
- автодоповнення слова, що набирається;
- одночасна робота з великою документів;
- підтримка регулярних виразів для пошуку й заміни;
- повна підтримка перетягування фрагментів тексту;
- динамічна зміна вікон перегляду;
- автоматичне визначення стану файлу;
- підтримка великої кількості мов;
- нотатки;
- плагіни;

- запис макросу і його виконання.

Можна розширити можливості Notepad++ після встановлення додаткових плагінів:

- шаблони тексту (сніпети), що вводяться за допомогою скорочень (плагін SnippetPlus);
- FTP-менеджер (плагін NppFTP);
- hex-редактор;
- автозбереження (при втраті фокусу через певний проміжок часу);
- перевірка орфографії (з використанням GNU Aspell);
- симетричне й асиметричне шифрування тексту (при встановленні плагіна NppDarkCrypt);
- підтримка Zen Coding;
- підтримка автоматизації за допомогою скриптів мовами Python, JScript, Lua та іншими;
- підтримка збереження до OneDrive і Dropbox.

Середовище реалізації зображено на рисунку 3.1.

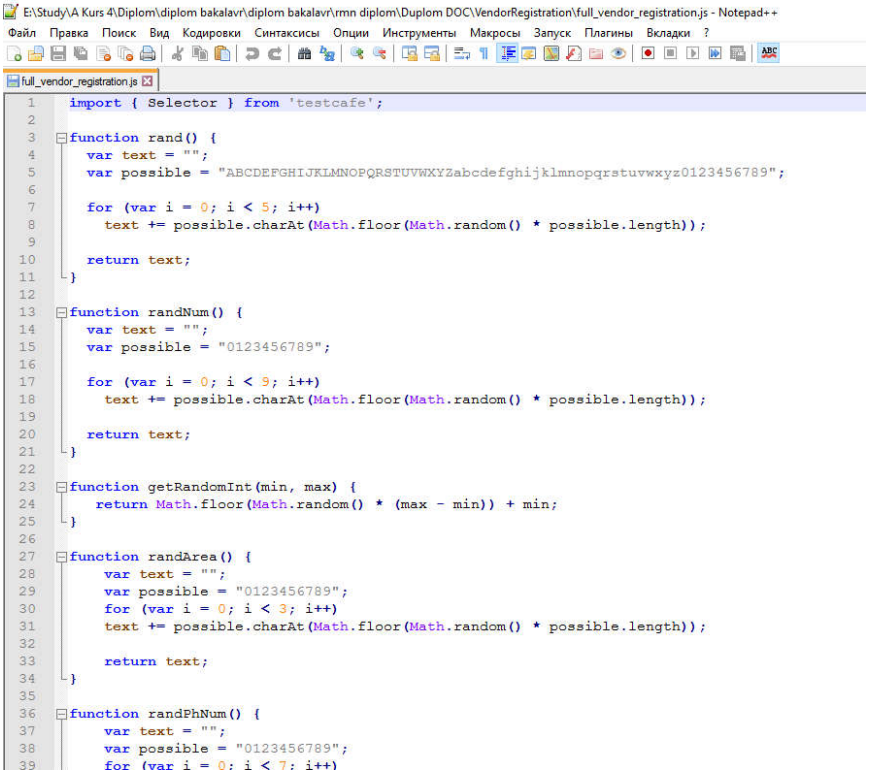


Рисунок 3.1 – Реалізація алгоритму в Notepad++

Для того, щоб написаний модуль коректно працював використано платформу Node.js версії 8.0 Node.js — платформа з відкритим кодом для виконання високопродуктивних мережевих застосунків, написаних мовою JavaScript. Засновником платформи є Раян Дал (Ryan Dahl). Платформа node.js перетворила мову JavaScript, що в основному використовувалась в браузерах, на мову загального використання з великою спільнотою розробників.

Node.js характеризується такими властивостями:

- асинхронна однопотокова модель виконання запитів;
- неблокуючий ввід/вивід;
- система модулів CommonJS;
- рушій JavaScript Google V8.

Для керування модулями використовується пакетний менеджер npm (node package manager).

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

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

За своєю суттю Node.js схожий на фреймворки Perl AnyEvent, Ruby Event Machine і Python Twisted, але цикл обробки подій (event loop) у Node.js прихований від розробника і нагадує обробку подій у веб-застосунку, що працює в браузері. При написанні програм для Node.js необхідно враховувати специфіку подієво-орієнтованого програмування.

Окрім Node.js було використано фреймворк TestCafe Studio - це крос-платформенний фреймворк для функціонального тестування веб-додатків.

Алгоритм тестування перевіряє справність усіх етапів роботи постачальника. Приклад роботи алгоритму представлено на рисунку 3.2. Алгоритм запускається за допомогою командної стрічки.

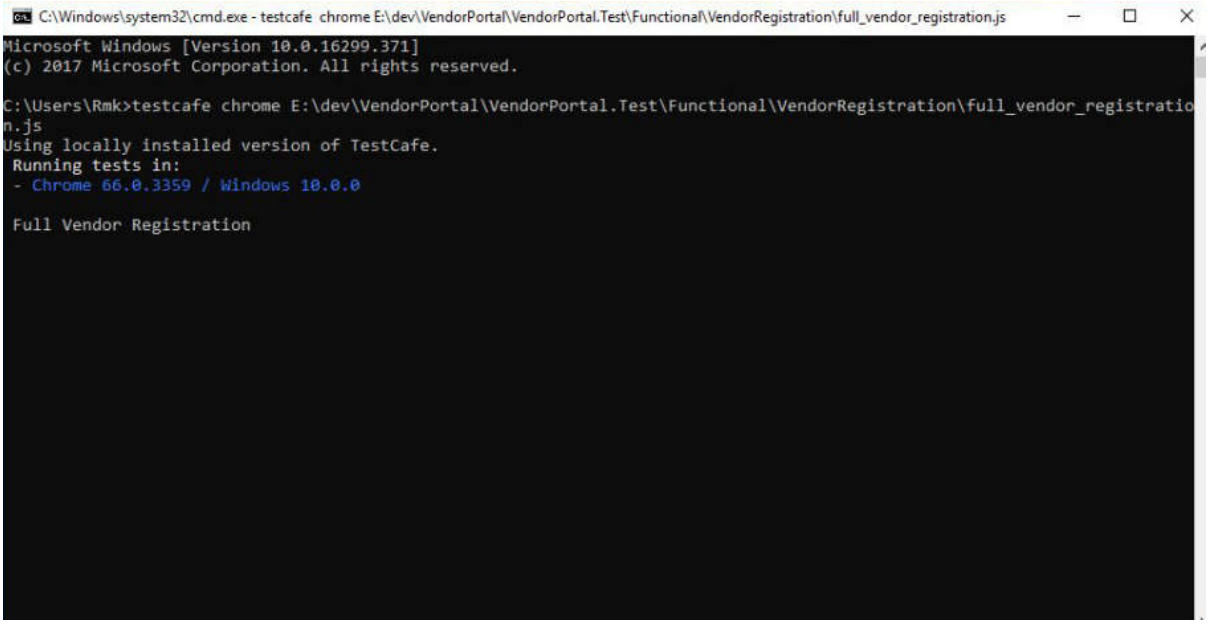


Рисунок 3.2 – Запуск алгоритму тестування

Після цього відбувається процес завантаження алгоритму, він відкривається на локальній машині та приєднується до сервера. Після цього починається процес проходження алгоритму тестування модулю постачальників, що проілюстровано на рисунку 3.3.

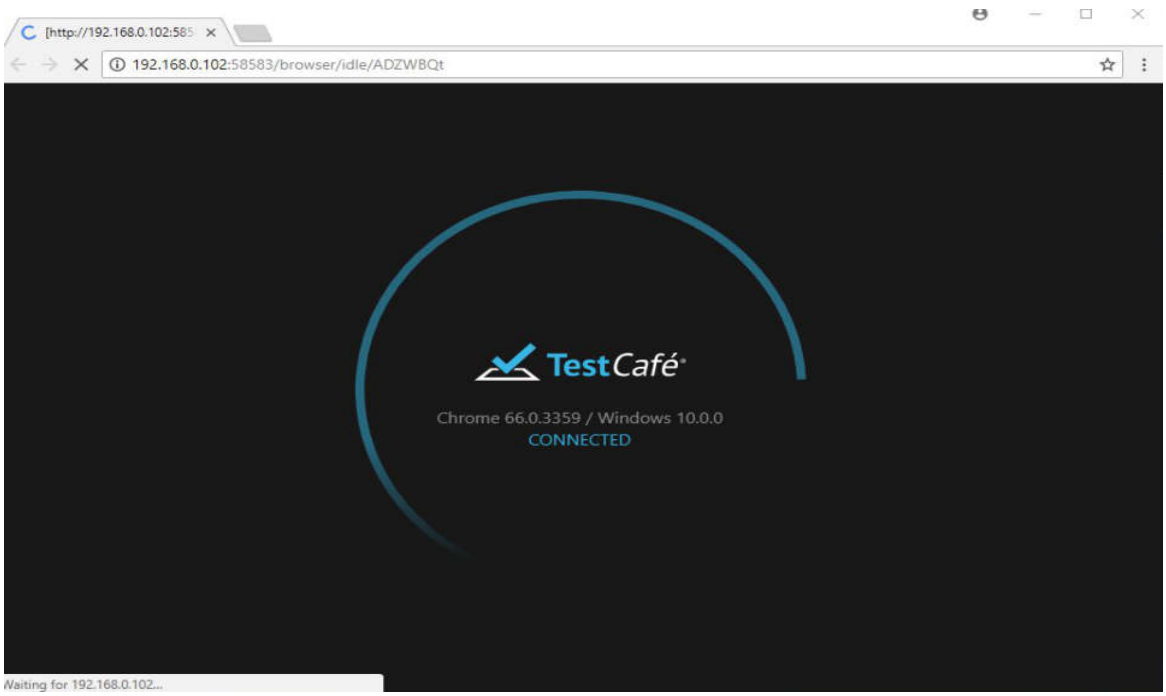


Рисунок 3.3 – Завантаження алгоритму

Процес реєстрації нового постачальника зображено на рисунку 3.4.

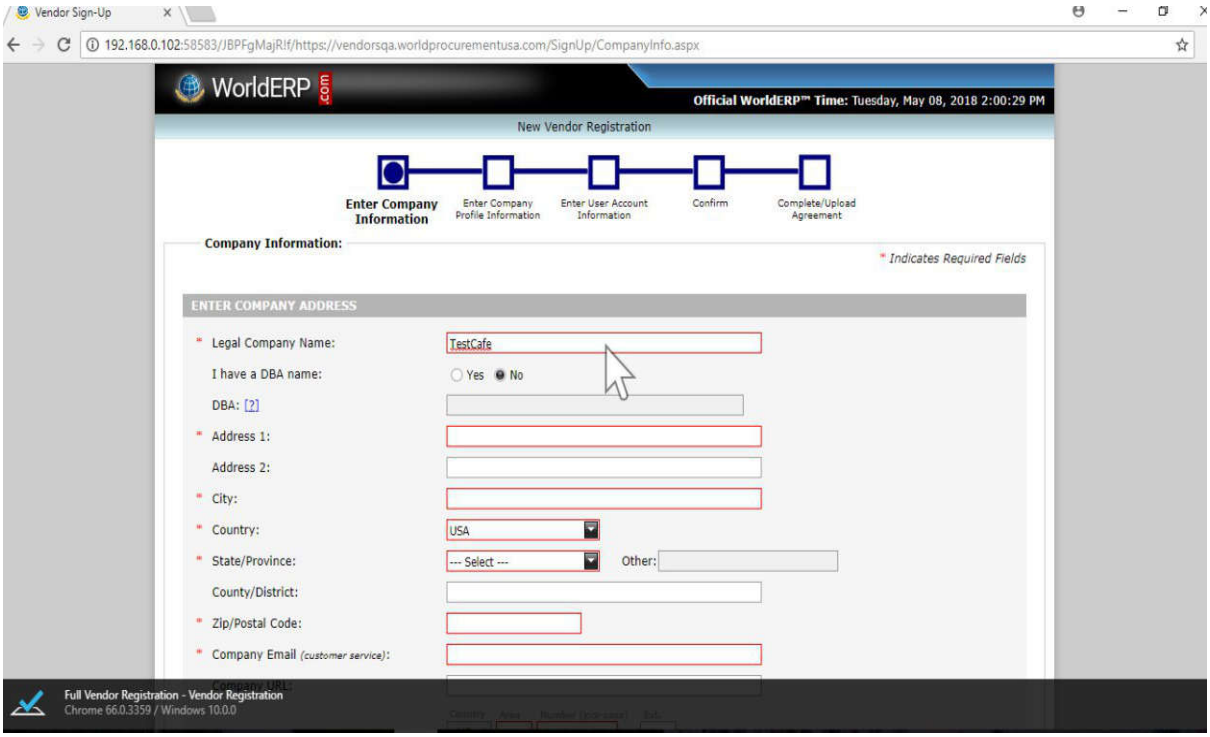


Рисунок 3.4 – Процес реєстрації постачальника

Після того як нового постачальника зареєстровано його підтверджує адміністратор. Процес підтвердження показано на рисунку 3.5.

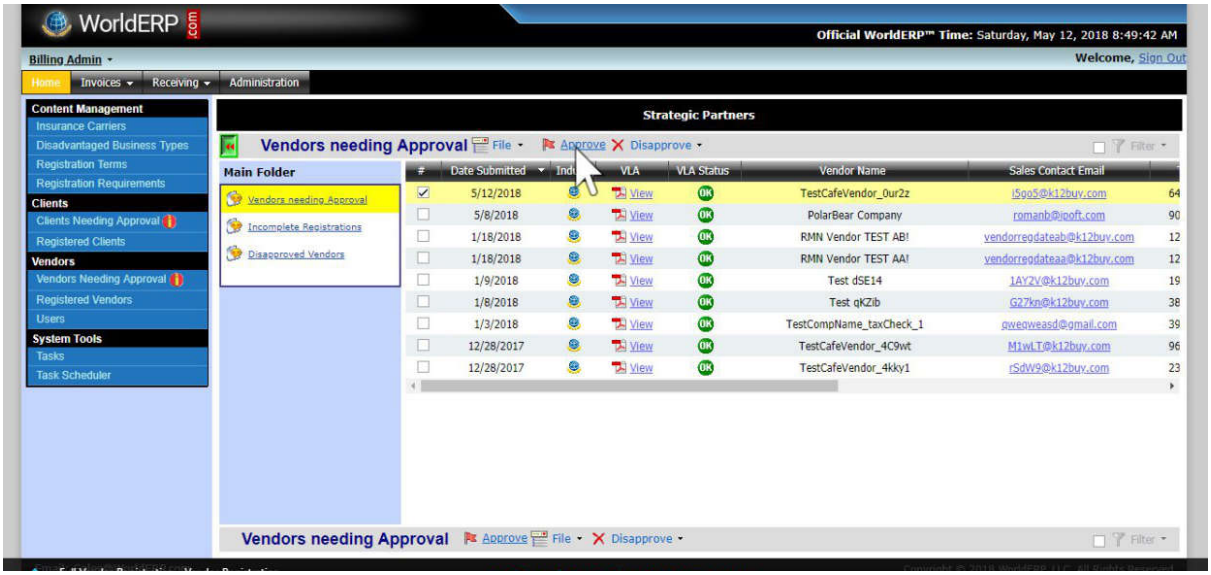


Рисунок 3.5 – Процес підтвердження постачальника

Коли акаунт постачальника підтверджено, він повинен пройти першу логінізацію. Першу логінізацію постачальника зображена на рисунку 3.6.

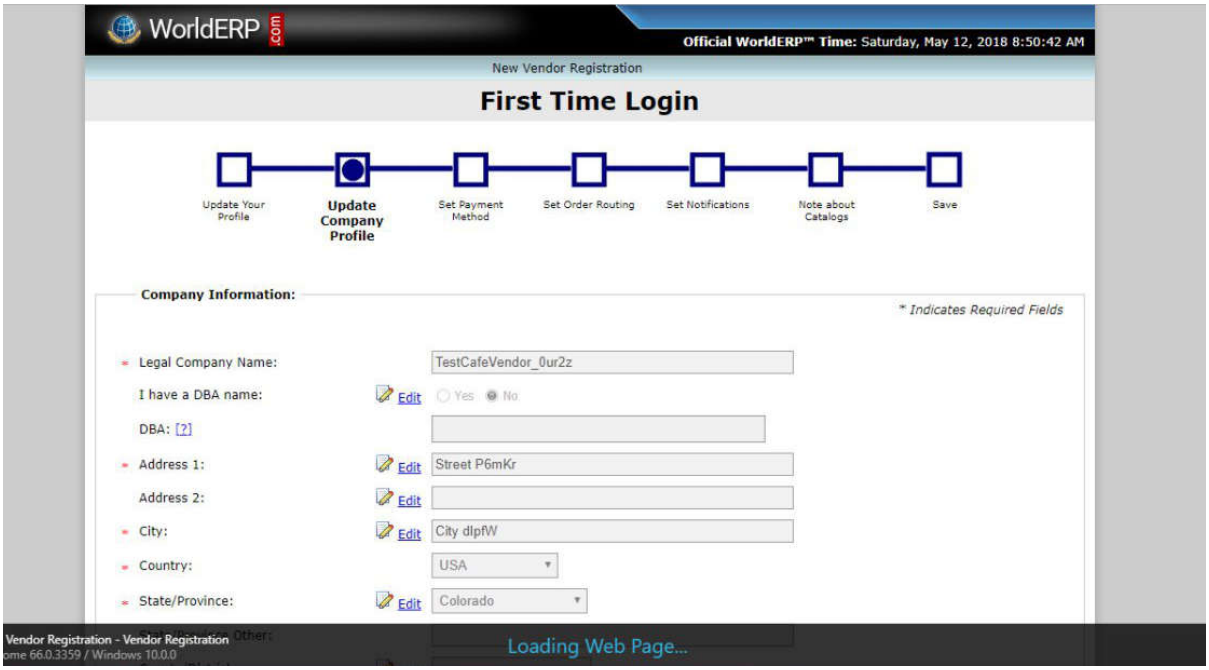


Рисунок 3.6 – Перша логінізація постачальника

Після проходження пешої логінізації постачальник має можливість підгрузити каталоги з продукцією, яку він представляє. Зображено на рисунку 3.7.

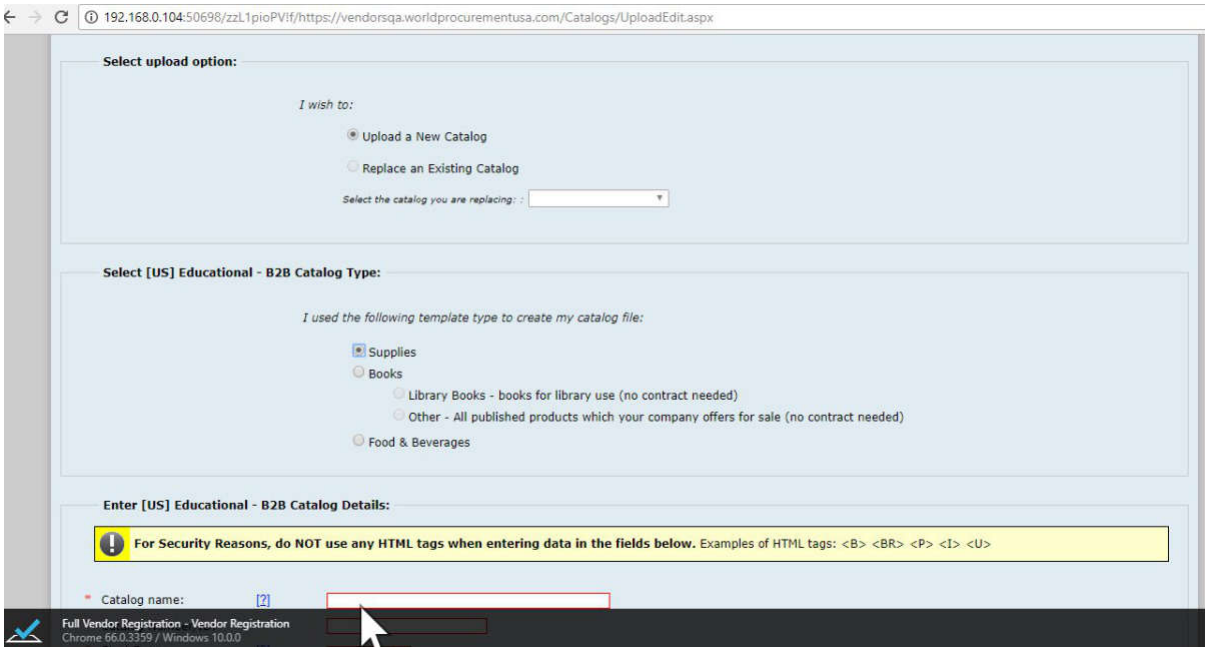


Рисунок 3.7 – Завантаження каталогів постачальника

Алгоритм тестування дуже легко та зручно реалізовувати в Notepad++, використовуючи всі доступні можливості редактора та фреймворку TestCafe.

3.3 Експериментальні дослідженя ефективності алгоритму тестування веб-базованої інформаційної системи

Розроблений алгоритм було досліджено на ефективність роботи для визначення фреймворку на якому буде реалізовано даний алгоритм.

Основні параметри фреймворків для тестування веб-додатків:

- кросбраузерна підтримка;
- технічна підтримка продукту;
- складність написання коду;

- відлагодження алгоритмів тестування та веб-додатків;
- час проходження тесту.

Результати аналізу фреймворків зображено на рисунку 3.8. На даній діаграмі зеленим прямокутником показано хороший результат, тобто фреймворк зміг виконати всі заявлені розробником задачі, жовтий – функціональні можливості виправдані частково та червоний прямокутник – фреймворк показав найгірший результат з усіх перевірених.

Рисунок 3.8 – Результат аналізу фреймворків для тестування

TestCafe показав найкращі результати під час аналізу та тестування фреймворків на відповідність вимогам, які ставить перед собою підприємство для повного та комфортного тестування веб-додатку [20].

Для проходження всіх етапів алгоритму тестування, тобто повного його завершення, необхідно в середньому вісім хвилин, результат показано на рисунку 3.9.

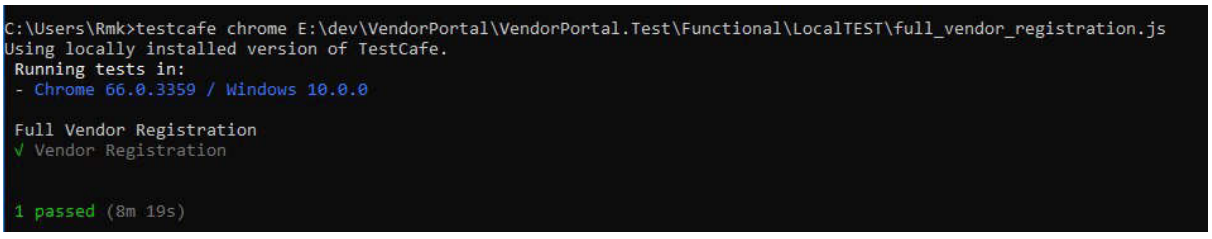


Рисунок 3.9 – Результат виконання алгоритму

Для порівняння алгоритм написаний за допомогою фреймворку Puppeteer, який був використаний для перевірки швидкості виконання та складності роботи з фреймворком виконується близько тринадцяти хвилин, результат зображено на рисунку 3.10.

Рисунок 3.10 – Результат виконання алгоритму Puppeteer

У ході аналізу фреймворків було досліджено, що працювати з кодом за допомогою фреймворку TestCafe зручніше та алгоритм виконується на п’ять хвилин швидше, результат виконання показано на рисунку 3.11.

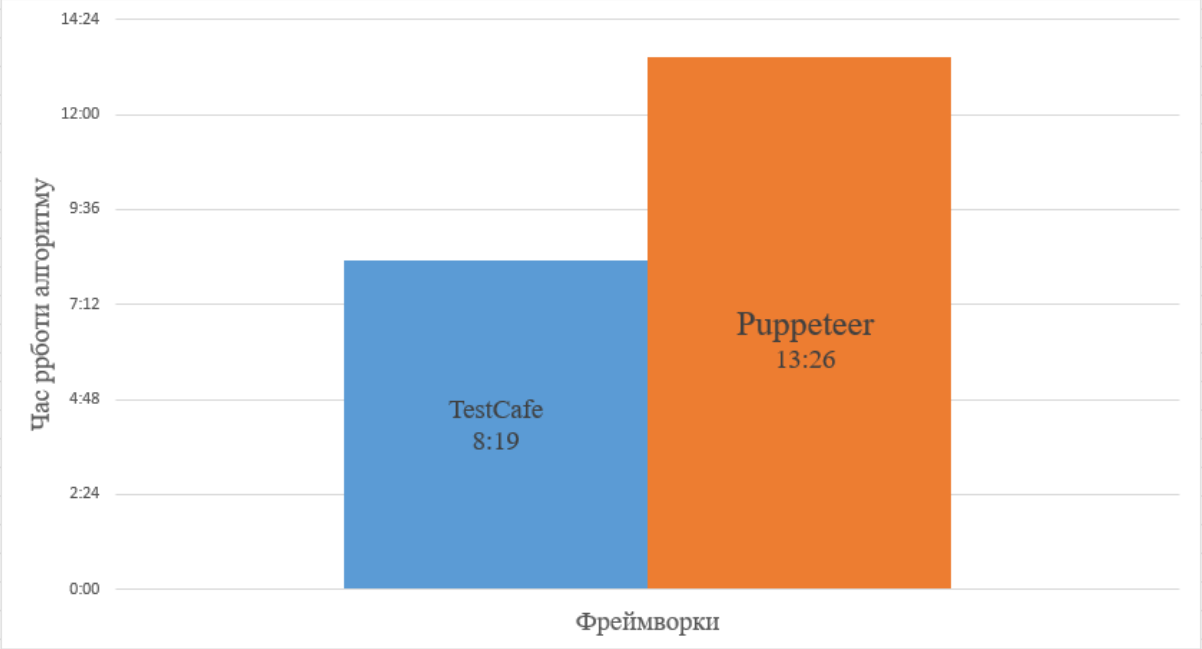


Рисунок 3.11 – Діаграма порівняння фреймворків

З даних, поданих на рисунку 3.11, можна зробити висновок, що алгоритм написаний за допомогою фреймворку TestCafe продуктивніший.

4 ТЕХНІКО-ЕКОНОМІЧНИЙ РОЗДІЛ

4.1 Розрахунок витрат на розробку програмного забезпечення

В цьому розділі дипломного проекту проводиться економічне обґрунтування доцільності розробки тестування програмного забезпечення. Зокрема, здійснюється розрахунок витрат на розробку тестування програмного забезпечення, експлуатаційних витрат, ціни споживання проектного рішення. В заключній частині визначаються показники економічної ефективності нового виду тестування програмного продукту, обґрунтовуються відповідні висновки.

Розроблене тестування програмного забезпечення призначене для пришвидшення тестування, його автоматизації та автоматизації створення логів для запису результатів і характеризується підвищеною ефективністю виконання алгоритму, що призводить до зменшення часу тестування в ручну об’єкту дослідження.

Витрати на розробку і впровадження програмних засобів ($К$) включають [21]:

$$К = К_1 + К_2$$

де $К_1$ - витрати на розробку програмних засобів, грн;

$К_2$ - витрати на відлагодження і дослідну експлуатацію програми рішення задачі на комп’ютері, грн.

Витрати на розробку програмних засобів включають:

- витрати на оплату праці розробників ($В_{ОП}$);
- витрати на відрахування у спеціальні державні фонди ($Вф$);
- витрати на покупні вироби ($Пв$);
- витрати на придбання спецобладнання для проведення експериментальних робіт ($Об$);

- накладні витрати (*Н*);
- інші витрати (*Ів*).

Витрати на оплату праці включають заробітну плату (ЗП) всіх категорій працівників, безпосередньо зайнятих на всіх етапах проектування. Розмір ЗП обчислюється на основі трудомісткості відповідних робіт у людино-днях та середньої ЗП відповідних категорій працівників.

У розробці проектного рішення задіяні наступні спеціалісти - розробники, а саме: керівник проекту; студент-дипломник; консультант техніко-економічного розділу.

Таблиця 4.1 - Вихідні дані для розрахунку витрат на оплату праці

№ п/п	Посада виконавців	Місячний оклад, грн.
1	Керівник проекту	5300
2	Консультант техніко-економічного розділу, доцент	6026
3	Студент	1400

Витрати на оплату праці розробників проекту визначаються за формулою:

$$B_{ОП} = \sum_{i=1}^{N} \sum_{j=1}^{M} n_{ij} \cdot t_{ij} \cdot C_{ij} \qquad (4.1)$$

де n_{ij} – чисельність розробників *i*-ої спеціальності *j*-го тарифного розряду, осіб;

t_{ij} – затрачений час на розробку проекту співробітником *i*-ої спеціальності *j*-го тарифного розряду, год;

C_{ij} – годинна ставка працівника *i*-ої спеціальності *j*-го тарифного розряду, грн.

Середньо годинна ставка працівника може бути розрахована за формулою:

$$C_{ij} = \frac{C_{ij}^{0}(1+h)}{РЧ_i} \quad (4.2)$$

де C_{ij} – основна місячна заробітна плата розробника *i*-ої спеціальності *j*-го тарифного розряду, грн.;

h – коефіцієнт, що визначає розмір додаткової заробітної плати (при умові наявності доплат);

$РЧ_i$ - місячний фонд робочого часу працівника *i*-ої спеціальності *j*-го тарифного розряду, год. (приймаємо 168 год.).

Таблиця 4.2 - Розрахунок витрат на оплату праці

№ п/п	Посада виконавців	Час розробки, год	Погодинна заробітна плата, грн/год.	Витрати на розробку, грн
1	Керівник проекту	16	29	580
2	Консультант техніко-економічного розділу, доцент	2	30,9	61,8
3	Студент	144	8,3	1195,2
Разом				1837

Величну відрахувань у спеціальні державні фонди визначають у відсотковому співвідношенні від суми основної та додаткової заробітних плат. Згідно діючого нормативного законодавства сума відрахувань у спеціальні державні фонди складає 20,5 % від суми заробітної плати:

$$B_{Ф} = \frac{20,5}{100} \cdot 1837 = 376,585 \text{ грн.} \quad (4.3)$$

У таблиці 4.3 наведений перелік купованих виробів і розраховані витрати на них.

Таблиця 4.3 - Розрахунок витрат на матеріали та комплектуючі

№ п/п	Найменування купованих виробів	Одиниця виміру	Ціна, грн	Кількість купованих виробів	Сума, грн	Транспортні витрати (10% від суми)	Загальна сума, грн
1	Папір (формат А4)	уп	90,0	2	120,00	8,0	128,0
2	Диски CD-R	шт	10,0	1	5,00	4,0	9,0
3	Тонер для принтера	уп	80	1	80	4,0	84,0
Разом							256

Витрати на використання комп'ютерної техніки включають витрати на амортизацію комп'ютерної техніки, витрати на користування програмним забезпеченням, витрати на електроенергію, що споживається комп'ютером. Вартість години роботи ноутбука Lenovo T-530 становить 3 грн. Середній щоденний час роботи на ноутбуці – 6 годин. Розрахунок витрат на використання комп'ютерної техніки приведений в таблиці 4.4.

Таблиця 4.4- Розрахунок витрат на використання комп'ютерної техніки

№ п/п	Назва етапів робіт, при виконанні яких використовується комп'ютер	Час використання комп'ютера, год.	Витрати на використання комп'ютера грн.
1	Проведення досліджень та оформлення їх результатів	120	360
2	Оформлення техніко-економічного розділу	8	24
4	Оформлення ДП	12	36
Разом		140	420

Накладні витрати проектних організацій включають три групи видатків: витрати на управління, загальногосподарські витрати, невиробничі витрати. Вони розраховуються за встановленими відсотками до витрат на оплату праці. Середньостатистичний відсоток накладних витрат приймемо 150% від заробітної плати:

$$H = 1{,}5 \cdot 1837 = 2755{,}5 \text{ грн.} \quad (4.4)$$

Інші витрати є витратами, які не враховані в попередніх статтях. Вони становлять 10% від заробітної плати:

$$I = 1837 \cdot 0{,}1 = 183{,}7 \text{ грн.} \quad (4.5)$$

Витрати на розробку програмного забезпечення складають:

$$K_1 = B_{ОП} + B_{Ф} + I + B_{ПВ} + H + I \quad (4.6)$$

$$K_1 = 1837 + 376{,}585 + 221 + 2755{,}5 + 183{,}7 = 5373{,}785 \text{ грн.} \quad (4.7)$$

Витрати на відлагодження і дослідну експлуатацію програмного продукту визначаємо за формулою:

$$K_2 = S_{м.г.} \cdot t_{від} \quad (4.8)$$

де $S_{м.г.}$ - вартість однієї машино-години роботи ПК, грн./год.;

$t_{від}$- комп'ютерний час, витрачений на відлагодження і дослідну експлуатацію створеного програмного продукту, год.

Загальна кількість днів роботи на комп'ютері дорівнює 20 днів. Середній щоденний час роботи на комп'ютері – 6 години. Вартість години роботи комп'ютера дорівнює 3 грн. Тому:

$$K_2 = 3*120 = 360 \quad (4.9)$$

На основі отриманих даних складаємо кошторис витрат на розробку програмного забезпечення.

Таблиця 4.5- Кошторис витрат на розробку програмного забезпечення

№ п/п	Найменування витрат	Сума витрат, грн.
1	Витрати на оплату праці	1837
2	Відрахування у спеціальні державні фонди	376,585
3	Витрати на куповані вироби	256
4	Накладні витрати	2755,5
5	Інші витрати	183,7
6	Витрати на відлагодження і дослідну експлуатацію програмного продукту	360,0
Разом		5768,785

4.2 Розрахунок ціни проекту

Для оцінки економічної ефективності розроблюваного програмного продукту слід порівняти його з аналогом, тобто існуючим програмним забезпеченням ідентичного функціонального призначення.

Експлуатаційні одноразові витрати по програмному забезпеченню і аналогу включають вартість підготовки даних і вартість роботи комп'ютера (за час дії програми):

$$E_n = E_{1n} + E_{2n} \quad (4.10)$$

де E_n - одноразові експлуатаційні витрати на ПЗ (аналог), грн.;

E_{1n} - вартість підготовки даних для експлуатації ПЗ (аналогу), грн.;

E_{2n} - вартість роботи комп'ютера для виконання проектного рішення (аналогу), грн.

Річні експлуатаційні витрати $B_{еп}$ визначаються за формулою:

$$B_{en} = E_n * N_n \quad (4.11)$$

де N_n - періодичність експлуатації ПЗ (аналогу), раз/рік.

Вартість підготовки даних для роботи на комп'ютері визначається за формулою:

$$E_{1n} = \sum_{l=1}^{n} n_i t_i c_i \quad (4.12)$$

де i - категорії працівників, які приймають участь у підготовці даних (i=1,2,...n);

n_i - кількість працівників i-ої категорії, осіб.;

t_i - трудомісткість роботи співробітників і-ої категорії по підготовці даних, год.;

c_i - середнього годинна ставка працівника i-ої категорії з врахуванням додаткової заробітної плати, що знаходиться із співвідношення:

$$c_i = \frac{c_i^0 (1+b)}{m} \quad (4.13)$$

де c_i^0 - основна місячна заробітна плата працівника i-ої категорії, грн.;

b - коефіцієнт, який враховує додаткову заробітну плату (приймемо 0,57;

m - кількість робочих годин у місяці, год.

Для роботи з даними як для проектного рішення так і аналогу потрібен один працівник, основна місячна заробітна плата якого складає: c^0 = 1300 грн. Тоді:

$$c_1 = \frac{1400(1+0{,}57)}{20*6} = 19{,}32 \text{ грн/год} \quad (4.14)$$

Таблиця 4.7- Розрахунок витрат на підготовку даних та реалізацію проектного рішення на комп'ютері

№	Час роботи співробітників, год.	Середньогодинна заробітна плата, грн./год.	Витрати , грн.
	Проектне рішення		
1	1	18,32	18,32
	Аналог		
1	1,5	18,32	27,48

Витрати на експлуатацію комп'ютера визначається за формулою:

$$E_{2n} = t * S_{мг} \quad (4.15)$$

де *t* - витрати машинного часу для реалізації проектного рішення (аналогу), год.;

$S_{МГ}$ - вартість однієї години роботи комп'ютера, грн./год.

$$E_{2n} = 1 * 3 = 3 \text{ грн.}; E_{2a} = 1{,}5 * 3 = 4{,}5 \text{ грн.} \quad (4.16)$$

$$E_n = 18{,}32 + 3 = 21{,}32 \text{ грн.}; E_a = 27{,}48 + 4{,}5 = 31{,}98 \text{ грн.} \quad (4.17)$$

$$B_{en} = 21{,}32 * 150 = 3198 \text{ грн.};\ B_{ea} = 31{,}98 * 150 = 4797 \text{ грн.} \quad (4.18)$$

4.3 Визначення економічної ефективності і терміну окупності капітальних вкладень

Ціна споживання - це витрати на придбання і експлуатацію проектного рішення за весь строк його служби:

$$Ц_{c(n)} = Ц_n + B_{(e)npv} \quad (4.19)$$

де $Ц_n$ - ціна придбання проектного рішення, грн.:

$$Ц_n = K(1 + \frac{П_p}{100}) + K_o + K_к \quad (4.20)$$

де K - кошторисна вартість;

$П_p$ - рентабельність;

K_o - витрати на прив'язку та освоєння проектного рішення на конкретному об'єкті, грн.;

$K_к$ - витрати на доукомплектування технічних засобів на об'єкті, грн.;

$$Ц_Д = 5733{,}785 \cdot (1 + 0{,}3) = 5905{,}80 \text{ грн.} \quad (4.21)$$

Вартість витрат на експлуатацію проектного рішення (за весь час його експлуатації), грн.:

$$B_{enpv} = \sum_{t=0}^{T} \frac{B_{en}}{(1+R)^t} \quad (4.22)$$

де B_{en} – річні експлуатаційні витрати, грн.;

T – строк служби проектного рішення, років;

R – річна ставка проценту банку.

$$B_{enpv} = \sum_{t=1}^{5} \frac{3198}{(1+0{,}02)^t} = 15676{,}47 \text{ грн.} \quad (4.23)$$

$$B_{enpv} = \sum_{t=1}^{5} \frac{4797}{(1+0{,}02)^t} = 23514{,}71 \text{ грн.} \quad (4.24)$$

Тоді ціна споживання проектного рішення дорівнюватиме:

$$Ц_{сп} = 5905{,}8 + 15676{,}47 = 21582{,}27 \text{ грн.} \quad (4.25)$$

Аналогічно визначається ціна споживання для аналогу:

$$Ц_{са} = 4500{,}0 + 23514{,}71 = 28014{,}71 \text{ грн.} \quad (4.26)$$

Економічний ефект в сфері проектування рішення:

$$E_{пр} = Ц_{п} - Ц_{а} \quad (4.27)$$

$$E_{пр} = 5905{,}8 - 4500{,}0 = 1405{,}8 \text{ грн.} \quad (4.28)$$

Річний економічний ефект в сфері експлуатації:

$$E_{кс} = B_{еа} - B_{еп} \quad (4.29)$$

$$E_{кс} = 4797 - 3198 = 1599 \text{ грн.} \quad (4.30)$$

Додатковий економічний ефект у сфері експлуатації:

$$\Delta E_{екс} = \sum_{t=1}^{T} E_{екс}(1+R)^{T-t} \quad (4.31)$$

$$\Delta E_{екс} = \sum_{t=1}^{5} 1599(1+0{,}02)^{5-1} = 8654{,}04 \text{ грн.} \quad (4.32)$$

Сумарний ефект складає:

$$E = E_{пр} + \Delta E_{екс} = 1405{,}8 + 8654{,}04 = 10059,\ 84 \text{ грн.} \quad (4.33)$$

Таблиця 4.8 - Показники економічної ефективності проектного рішення

№	Найменування	Одиниці вимірювання	Значення показників	
			Базовий варіант	Новий варіант
1	2	3	4	5
1	Капітальні вкладення	грн.	-	5768,785
2	Ціна придбання	грн.	4500,0	5905,8
3	Річні експлуатаційні витрати	грн.	23514,71	15676,47
4	Ціна споживання	грн.	28014,71	21582,27
5	Економічний ефект	грн.	-	1405,8

Продовження таблиці 4.8

1	2	3	4	5
	в сфері проектування			
6	Економічний ефект в сфері експлуатації	грн.	-	1599
7	Додатковий ефект в сфері експлуатації	грн.	-	8654,04
8	Сумарний ефект	грн.	10094, 84	

В даному розділі проведено розрахунок витрат на розробку модулю тестування веб-базованої інформаційної системи. Здійснено порівняння з існуючим аналогом, і цим показано, що даний спосіб тестування веб-базованої інформаційної системи має переваги в порівнянні з аналогами, зокрема: надійність, простота використання, гнучкість, зручність. Згідно проведеного економічного обґрунтування даний метод тестування є конкурентноздатним. Тому розробка і впровадження такого методу тестування веб-додатків є економічно доцільними.

ВИСНОВКИ

Відповідно до поставлених задач було реалізовано наступні пункти:

1. Проаналізовано та описано етапи тестування веб-базованих інформаційних систем. Досліджено всі доступні середовища для розробки програмного модуля тестування. Обрано середовище, яке надає найбільше функціональних можливостей для розробки модулю та описано середовище для розробки програмного модуля тестування веб-додатку.

2. Досліджено алгоритми тестування веб-базованої інформаційної системи та розроблено алгоритм, який задовільняє всі вимоги для тестування веб-базованої інформаційної системи.

3. Досліджено доступні програмні модулі тестування веб-додатків, проаналізовано переваги та недоліки уже існуючих програмних рішень та на основі отриманих результатів розроблено ефективний програмний модуль тестування веб-додатку, який задовільняє поставлені відділом тестування вимоги.

4. Описано приклад реалізації алгоритму та досліджено ефективність модулю тестування веб-базованої інформаційної системи.

5. Описано техніко-економічні показники доцільності розробки та впровадження проекту.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Иванов Л. В. Факторный анализ в социальных исследованиях / Л. В. Иванов, В. С. Смирнов. – М. : Наука, 1996. – 360 с.

2. Операційна система Microsoft Windows: Методичні вказівки до лабораторних робіт / Укл .: В.С. Сікора, І.В. Юрченко.- Чернівці: Рута, 2003.- 48 с.

3. Текстовий редактор Microsoft Word: Методичні вказівки до лабораторних робіт / Укл .: В.С. Сікора, І.В. Юрченко.- Чернівці: Рута, 2003.- 56 с.

4. Власов П. К. Психология менеджмента / П. К. Власов, А. В. Лепницкий, И. М. Лущихина [и др.]. – Х. :Гуманит. центр, 2007. – 320 с.

5. Харман Г. Современный факторный анализ – М. : Статистика, 1972. – 499 с.

6. Классификация и кластер / ред. Дж. Вэн Райзин. – М. : Мир, 1980. – 399 с.

7. Петров В. С. Применение методов кластерного анализа при обработке данных экспертного опроса / В. С. Петров // Изв. АН СССР. Сер.: Техн. кибернетика. – 1985. – Т. 202. - № 3. – С. 15–18.

8. Автоматизоване тестування веб-додатків ключові поняття / Інформаційний портал - Режим доступу: http://www.quality-assurance-group.com/klyuchovi-ponyattya-ta-za-yakyh-umov-dotsil/ - Дата звертання: 17 травня 2018 – Назва з екрану.

9. Тишков В. Т. Кластерный анализ в социальных исследованиях / В. Т. Тишков // Вестн. Харьк. политехн. ин та. Сер.: Техн. кибернетика и ее приложения. – 1990. – № 260. - Вып. 10. – С. 5–7.

10. Навчальний ресурс з тестування програмного забезпечення / Q & A – Режим доступу: http://qalearning.com.ua/theory/about_qa/ - Дата звертання: 20 травня 2018 – Назва з екрану.

11. Тестування веб-додатків / Писати тести - це як писати код. Тільки тести – Режим доступу: http://is.unicyb.kiev.ua/files/poliachenko_2015.pdf - Дата звертання: 22 травня 2018 – Назва з екрану.

12. Иванов Л. В. Статистический подход к обработке социоло¬гических данных / Л. В. Иванов // Труды междунар. конф. «Социология и математика». – Т. 2. – Х. : НТУ «ХПИ», 2006. – С. 5–9.

13. Гамаюн І. П. Оцінювання міри схожості між об'єктам, що характеризуються кількісними і номіналь¬ними ознаками / І. П. Гамаюн, О. М. Безменова // Інформаційні технології: наука, техніка, технологія, освіта, здоров'я. Тези доповідей XXI міжнародної науково-практичної конференції. Ч. 1 (29–31 травня 2013 р., Харків). – Х. : НТУ «ХПІ», 2013. – С. 9.

14. Національний технічний університет «Харківський політехнічний інститут». Вісник НТУ «ХПІ» / Національний технічний університет «Харківський політехнічний інститут». – НТУ «ХПІ», 2018. – Режим доступу : http://vestnik.kpi.kharkov.ua. – Дата звертання : 30 березня 2018 – Назва з екрану.

15. Азаренков В. И. Аналитическое решение уравнения тепло-проводности в задачах анализа и синтеза температурных полей радиоэлектронной аппаратуры :дис. канд. техн. наук : 01.05.02 / Азаренков – Х., 2015. – 235 с.

16. Северин В. П. Моделі та методи оптимізації показників якості систем автоматичного керування енергоблоку атомної електростанції : автореф. дис. на здобуття наук. ступеня д-ра техн. наук : спец. 05.13.07 «Автоматизація технологічних процесів» / В. П. Северин. – Х., 2007. – 65 с.

17. Джафарі Хенджані Сайед Моджтаба. Багатокритеріальний синтез інтелектуальних систем керування енергоблоків АЕС генетичними алгоритмами : автореф. дис. на здобуття наук. Ступеня канд. техн. наук : спец.

05.13.07 «Автоматизація технологічних процесів» / Джафарі Хенджані Сайд Моджтаба. – Х., 2010. – 39 с.

18. ДСТУ 3582–97. Скорочення слів в українській мові у бібліографічному описі. – К. : Держстандарт України, 1998. – 55 с.

19. ГОСТ 8.586.5–2005. Методика выполнения измерений расхода и количества жидкостей и газов с помощью сужающих устройств. – М. Стандартинформ, 2007. – 10 с.

20. Руденко В.Д. та ін. Базовий курс інформатики; за заг. ред. В.Ю.Бікова: [Навч. посіб.]. - К .: Вид. група BHV. - Кн. 1: Основи інформатики. - 2005. - 320 с.

21. Руденко В.Д. та ін. Базовий курс інформатики; за заг. ред. В.Ю.Бікова: [Навч. посіб.]. - К .: Вид. група BHV. - Кн. 2: Інформаційні технології. - 2006. - 368 с .: іл.

22. Текстовий редактор Microsoft Word: Методичні вказівки до лабораторних робіт / Укл .: В.С. Сікора, І.В. Юрченко.- Чернівці: Рута, 2003.- 56 с.

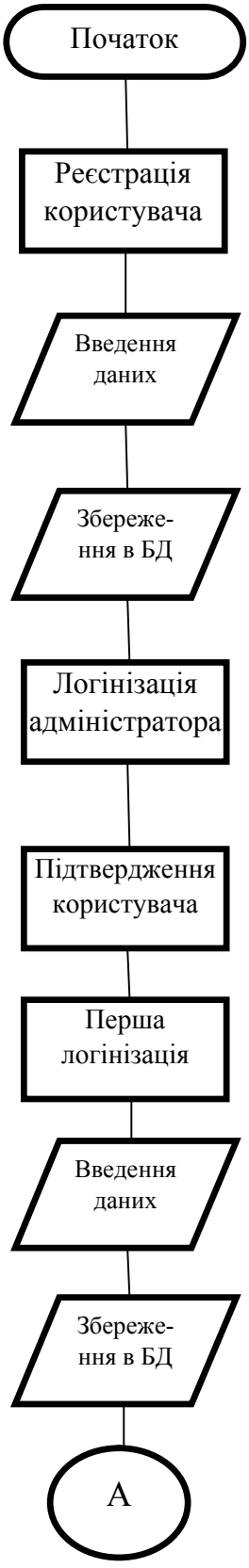
23. Семчук А.Р., Юрченко І.В. Економічна інформатика. Навчальний посібнік.- Чернівці: МВІЦ "Місто", 2008.- 426 с.

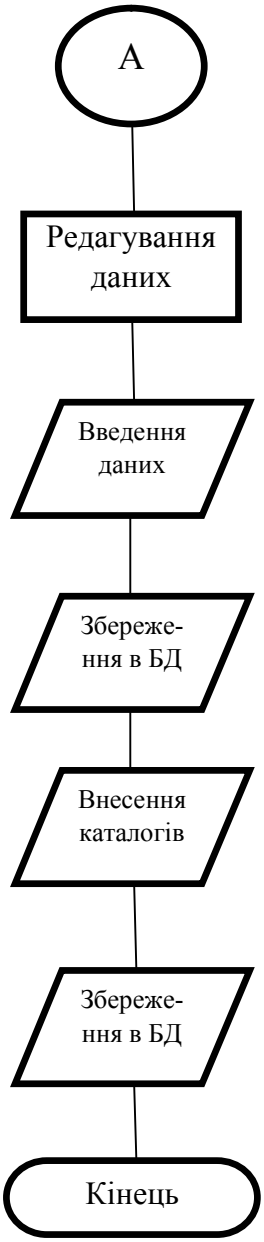
24. Методичні рекомендації до виконання дипломного проекту з освітньо-кваліфікаційного рівня «Бакалавр» напряму підготовки 6.050102 «Комп’ютерна інженерія» фахового спрямування «Комп’ютерні системи та мережі» / О.М. Березький, Л.О. Дубчак, Р.Р. Трембач, Г.М. Мельник, Ю. М. Батько, С.В. Івасьєв / Під ред. О.М. Березького. – Тернопіль: ТНЕУ, 2016. – 60 с.

25. Методичні вказівки до написання техніко-економічного розділу для дипломних проектів на здобуття освітньо-кваліфікаційного рівня «Бакалавр» напряму підготовки 6.050102 «Комп’ютерна інженерія» / І.Р. Паздрій. – Тернопіль: ТНЕУ, 2015. – 36 с.

ДОДАТОК А

Алгоритм роботи модуля тестування




A
Редагування даних
Введення даних
Збереже-ння в БД
Внесення каталогів
Збереже-ння в БД
Кінець

ДОДАТОК Б

Лістинг коду модуля тестування

```
import { Selector } from 'testcafe';
function rand() {
  var text = "";
  var possible =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxy
z0123456789";
  for (var i = 0; i < 5; i++)
    text += possible.charAt(Math.floor(Math.random() * possible.length));
  return text;
}
function randNum() {
  var text = "";
  var possible = "0123456789";
  for (var i = 0; i < 9; i++)
    text += possible.charAt(Math.floor(Math.random() * possible.length));
  return text;
}
function getRandomInt(min, max) {
   return Math.floor(Math.random() * (max - min)) + min;
}
function randArea() {
        var text = "";
        var possible = "0123456789";
        for (var i = 0; i < 3; i++)
        text += possible.charAt(Math.floor(Math.random() *
possible.length));
        return text;
```

```
}
function randPhNum() {
        var text = "";
        var possible = "0123456789";
        for (var i = 0; i < 7; i++)
        text += possible.charAt(Math.floor(Math.random() *
possible.length));
        return text;
}
// ///
function randEndDate() {
   var date = new Date();
   date.setDate(date.getDate() + 30).toLocaleString();
   date = date.toLocaleDateString('en-US');
   return date;
}
function randPaths() {
   var paths = [
      "E:/TestCafe_catalogs/catalog_0.xlsx",
      "E:/TestCafe_catalogs/catalog_1.xlsx",
      "E:/TestCafe_catalogs/catalog_2.xlsx",
      "E:/TestCafe_catalogs/catalog_3.xlsx",
      "E:/TestCafe_catalogs/catalog_4.xlsx",
      "E:/TestCafe_catalogs/catalog_5.xlsx"
   ];
   var res = "";
   var length = paths.length;
      var rnd = Math.floor(Math.random() * length);
      res = paths[rnd];
```

```
    return res;
}
// ///
var email = rand() + '@k12buy.com';
var pass = 'path';
var questions = 'Test';
var vendorName = 'TestCafeVendor_' + rand();
var firstName = 'TestCafeVendor_' + rand();
const GetElement = Selector(value => {
    return document.getElementById(value) ||
document.getElementsByClassName(value);
});
fixture('Full Vendor Registration')
.page
('https://vendorsqa.worldprocurementusa.com/Login.aspx?ReturnUrl=%2
fHomeNew.aspx');
        test('Vendor Registration', async t =>{
                await t
                        .setTestSpeed(0.5)
                        .maximizeWindow()
                        .click(Selector('a').withAttribute('href', '/SignUp/'))
        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnNext_CD'))
        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_ucCompan
yInfo_tbxCoName_I'), vendorName)
        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_ucCompan
yInfo_usAddressEdit_tbxAddress1_I'), 'Street ' + rand())
        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_ucCompan
yInfo_usAddressEdit_tbxCity_I'), 'City ' + rand())
        .click(GetElement('ctl00_ctl00_cphMain_cphMain_ucCompanyInf
o_usAddressEdit_ddlStates_I'))
```

```
        .click(GetElement('ctl00_ctl00_cphMain_cphMain_ucCompanyInfo_usAddressEdit_ddlStates_DDD_L_LBI'+getRandomInt(1, 51)+'T0'))

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_ucCompanyInfo_usAddressEdit_tbxZip_I'), randNum())

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_ucCompanyInfo_tbxCompanyEmail_I'), email)

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_ucCompanyInfo_pfnPhone_tbxArea_I'), randNum())

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_ucCompanyInfo_pfnPhone_tbxNumber_I'), randNum())

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_ucCompanyInfo_ddlStatesInc_I'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_ucCompanyInfo_ddlStatesInc_DDD_L_LBI'+getRandomInt(1, 51)+'T0'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_ucCompanyInfo_ddlIncType_I'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_ucCompanyInfo_ddlIncType_DDD_L_LBI'+getRandomInt(1, 3)+'T0'))

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_ucCompanyInfo_tbxTaxID_I'), randNum())

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_ucCompanyInfo_cbxBillToSame_S_D'))

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_ucCompanyInfo_tbxBillToATTN_I'), randNum())

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnNext_CD'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_BusinessTypeEdit_cbxRetailer_S_D'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_lbtnAdd'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_k12CommoditySearch_dvxTreeList_R-10000000_D'))
        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnContinue_CD'))

        .setFilesToUpload(GetElement('ctl00_ctl00_cphMain_cphMain_ucAttachment_dvxUploadControl_TextBox0_Input'), './w9test.txt')
```

```
        .click(GetElement('ctl00_ctl00_cphMain_cphMain_ucAttachment_
dvxUploadButton_CD'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnNext_CD'))

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxFirstNa
me_I'), firstName)

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxLastNa
me_I'), rand())

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxJobTitle
_I'), 'Chef')

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxEmail_I
'), email)

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxEmailC
onfirm_I'), email)
        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_pfnPhone1
_tbxArea_I'), randNum())

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_pfnPhone1
_tbxNumber_I'), randNum())

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_cbCarriers_I'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_cbCarriers_DD
D_L_LBI1T0'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnNext_CD'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnConfirm'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_cbAcceptVLA
_S_D'))

        .click(GetElement('signature'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnSign'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnDone'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnDone_CD')
)

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnContinue'))

        //VENDOR APPROVE!

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxEmail_I
'), 'billing@k12buy.com')
```

```
        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxPass_I')
, 'path')
        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnSignIn_CD'
))
.click(Selector('a').withAttribute('href','/Billing/Vendors/VendorSignUp.a
spx'))
        .click(Selector('#ctl00_ctl00_ctl00_cphMain_cphMain_cphMain_x
gvGrid_DXMainTable').find('td').withText(vendorName))
        .click(GetElement('ctl00_ctl00_ctl00_cphMain_cphMain_cphMain
_k12GridToolbarTop_lbApproveTop'))
        .click(GetElement('ctl00_ctl00_ctl00_cphSignIn_lbtnLogOut'))
        //VENDOR FIRST LOGIN!
        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxEmail_I
'), email)
        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxPass_I')
, pass)
        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnSignIn_CD'
))
        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxPasswo
rd'), pass)
        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxPasswo
rdConfirm'), pass)
        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxQuestio
n1'), questions)
        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxQuestio
n2'), questions)
        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxQuestio
n3'), questions)
        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnNext_CD'))
        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnNext_CD'))
        .click(GetElement('ctl00_ctl00_cphMain_cphMain_cbxPaymentP
O'))
        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnNext_CD'))
```

```
        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxInitials'
), 'TST')

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnNext_CD'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnNext_CD'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnNext_CD'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnSave_CD'))

if (await
GetElement('ctl00_ctl00_k12PopupDevEx_dvxPopup_dvxBtnContinue').
exists && await
GetElement('ctl00_ctl00_k12PopupDevEx_dvxPopup_dvxBtnContinue').
visible)

await
t.click(GetElement('ctl00_ctl00_k12PopupDevEx_dvxPopup_dvxBtnCon
tinue'));

        await t

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnLoginNow_
CD'))

        // VENDOR UPLOAD CATALOG

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxEmail_I
'), email)

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxPass_I')
, pass)

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnSignIn'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_pnlK12Buy_w
gIndustryK12Buy_lnkB2BCatalogs'))//Click view/upload

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_lnkUploadCata
log'))// press upload catalog

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_rbnSupplies'))/
/ check radio button supplies

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbCatalog
Name'), rand())// enter catalog name

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbCatalog
Number'), rand())// enter catalog number
```

```
        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_catalogStar
tDate_I'), randEndDate())// enter catalog start date
        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_catalogEnd
Date_I'), randEndDate())// enter catalog end date

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbDescripti
on'), rand())// enter catalog description

        .setFilesToUpload(GetElement('ctl00_ctl00_cphMain_cphMain_dv
xUpload_TextBox0_Input'), randPaths())// insert path to file for
download

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnUpload',
{timeout:20000}))// click upload button

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnValidate',
{timeout:20000}))// click validate button

        .wait(2000)

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnPreviewCat
alog', {timeout:20000}))// click preview button

.click(Selector('img').withAttribute('alt', 'close'))// close preview window

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnSubmitCata
log', {timeout:20000}))// click submit button

        .click(GetElement('ctl00_ctl00_cphSignIn_lbtnLogOut',
{timeout:20000}))// click LogOut button

        //VENDOR ACCOUNT INFORMATION UPDATE!

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxEmail_I
'), email)

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxPass_I')
, pass)

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnSignIn_CD'
))

        .hover(Selector('#ctl00_ctl00_cphSignIn_dvxOptions_DXI0_T').fi
nd('span').withText('Personal Options'))

        .click(Selector('a').withAttribute('href',
'Account/UserProfile.aspx'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_LinkButton8'))
```

```
        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_txtOldPass
word'), pass)

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_txtPasswor
d'), pass)

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_txtConfirm
Password'), pass)

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnSave_CD'))


        .click(GetElement('ctl00_ctl00_cphMain_cphMain_LinkButton12')
)

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_tbxFirstName')
)

        .pressKey('ctrl+a delete')
        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxFirstNa
me'), 'TestCafeVendor_' + rand())

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_LinkButton11')
)

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_tbxLastName')
)

        .pressKey('ctrl+a delete')

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxLastNa
me'), 'Test ' + rand())

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_LinkButton1'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_tbxJobTitle'))

        .pressKey('ctrl+a delete')

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxJobTitle
'), 'Test ' + rand())

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_LinkButton2'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_tbxPhone1Are
a'))

        .pressKey('ctrl+a delete')

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxPhone1
Area'), randArea())
```

```
        .click(GetElement('ctl00_ctl00_cphMain_cphMain_tbxPhone1Nu
mber'))

        .pressKey('ctrl+a delete')

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxPhone1
Number'), randPhNum())

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_LinkButton4'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_tbxMobile1Are
a'))

        .pressKey('ctrl+a delete')

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxMobile
1Area'), randArea())

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_tbxMobile1Nu
mber'))

        .pressKey('ctrl+a delete')

        .typeText(GetElement('ctl00_ctl00_cphMain_cphMain_tbxMobile
1Number'), randPhNum())

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_hgTerritories_c
tl03_Icon'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_hgTerritories_c
tl03_ChildTemplate_country2state_dgStates_ctl73_cbxSelected'))

        .click(GetElement('ctl00_ctl00_cphMain_cphMain_btnSave_CD');

        });
```

ДОДАТОК В

Довідка про використання

Завідувачу кафедри
комп’ютерної інженерії
д.т.н., проф. О.М. Березькому

ДОВІДКА ПРО ВИКОРИСТАННЯ

Виконана студентом групи КСМ-43/2 факультету комп’ютерних інформаційних технологій Тернопільського національного економічного університету Баданіним Р. В. бакалаврська робота на тему «Програмний засіб автоматизованого тестування веб-базованої інформаційної системи» має практичне значення і планується до використання.

Провідний програміст WorldERP _____ Степаненко А. В.

підпис