Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
(повна назва)

Кафедра Автоматизації проектування обчислювальної техніки
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)
(рівень вищої освіти)

Система автоматизованого формування бази проектів для верифікації.
(тема)

Виконав: студент 2 курсу, групи СКСм-19-1
Лавров А.О.
(прізвище, ініціали)
Спеціальність 123 Комп'ютерна інженерія
(код і повна назва спеціальності)
Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма Спеціалізовані комп'ютерні системи
(повна назва освітньої програми)
Керівник проф. Хаханова І.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Чумаченко С.В.
(підпис) (прізвище, ініціали)

2020 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
Кафедра Автоматизації проектування обчислювальної техніки
Рівень вищої освіти другий (магістерський)
Спеціальність 123 – Комп'ютерна інженерія
Тип програми Освітньо-професійна
Освітня програма Спеціалізовані комп'ютерні системи

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____» _____ 2020 р.

ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ (ПРОЕКТ)

Студентові Лаврову Артему Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи (проекту) Система автоматизованого формування бази проектів для верифікації

затверджена наказом по університету від "30" жовтня 2020 р. № 1489Ст
2. Термін подання студентом роботи до екзаменаційної комісії _____
3. Вхідні дані до роботи Розробити систему для автоматизації скачування Проектів з EDA Playground, написаних на мові VHDL, забезпечити доступ до файлів після скачування та вивід результатів скачування у вигляді таблиці.

4. Перелік питань, що потрібно опрацювати в роботі Огляд предметної області, аналіз ролі автоматизації в освітній сфері та його переваги, важливості верифікації моделей, аналіз існуючих програмних продуктів, аналіз та вибір засобів розробки, розробка алгоритму функціонування системи

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) 13 слайдів

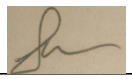
6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

7. Дата видачі завдання 01.09.2020

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Постановка завдання	01.09.2020–06.09.2020	
2	Огляд предметної області	07.09.2020–20.09.2020	
3	Аналіз та вибір засобів розробки	21.09.2020–04.10.2020	
4	Розробка структури системи	05.10.2020–18.10.2020	
5	Розробка алгоритму функціонування	19.10.2020–01.11.2020	
6	Розробка програмної частини	02.11.2020–22.11.2020	
7	Оформлення атестаційної роботи	23.11.2020–11.12.2020	
8	Передзахист	11.12.2020–15.11.2020	
9	Захист	23.12.2020–23.11.2020	

Студент  (підпис)

Керівник роботи (проекту)  (підпис)

проф. Хаханова І.В.
(посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка атестаційної роботи: 78 с., 24 рис., 7 табл., 3 дод., 16 джерел.

ФОРМУВАННЯ БАЗИ, ПРОЕКТИ, АВТОМАТИЗАЦІЯ, АВТОМАТИЧНЕ СКАЧУВАННЯ, EDA PLAYGROUND, ПАРСИНГ ВЕБ СТОРІНКИ, VHDL

Метою атестаційної роботи є розробка системи автоматизованого формування бази проектів для верифікації. У ході виконання атестаційної роботи проводився аналіз предметної області, а саме виявлення актуальності теми, огляд предмету атестаційної роботи, а саме EDA Playground, огляд існуючих програмних продуктів для автоматизованої роботи з сайтами, огляд та вибір способів взаємодії з веб сервісом, огляд та вибір способів зберігання бази проектів, розробка алгоритму роботи системи та реалізація системи.

Основним завданням є автоматизація завантаження проектів з EDA Playground за посиланнями, що передаються у програму та формування результатів завантажень у залежності від успішності завантаження проекту, що реалізується за допомогою написання програми автоматизованого формування бази проектів.

Система автоматизованого формування бази проектів для верифікації може бути застосована для прискорення та поліпшення обробки та тестування проектів, що в свою чергу може відбитися на якості та швидкості отримання результатів.

ABSTRACT

Master’s thesis: 78 pages, 24 figures, 7 tables, 3 appendices, 16 sources.

BASE FORMATION, PROJECTS, AUTOMATION, AUTOMATIC DOWNLOAD, EDA PLAYGROUND, WEBSITE PARSING, VHDL

The purpose of certification work is to develop a system of automated formation of the database of projects for verification. During the attestation work the analysis of the subject area was carried out, namely revealing the relevance of the topic, review of the subject of attestation work, namely EDA Playground, review of existing software products for automated work with sites, review and selection of ways to interact with web service, review and selection of storage methods. project bases, development of system operation algorithm and system implementation.

The main task is to automate the download of projects from EDA Playground by links sent to the program and the formation of download results depending on the success of the project download, which is implemented by writing a program of automated project database formation.

The system of automated formation of the database of projects for verification can be used to speed up and improve the processing and testing of projects, which in turn can affect the quality and speed of results.

ЗМІСТ

ДОДАТОК А Код управляючого модуля програми.....73
ДОДАТОК Б Код модуля роботи з браузером.....75
ДОДАТОК В Графічний матеріал атестаційної роботи.....78

ВСТУП

В останні кілька десятиліть одним з напрямків сучасного життя, що найбільш прогресивно розвивається є автоматизація процесів.

Автоматизація це комплекс засобів і методів для реалізації системи або систем, що дозволяють здійснювати керування процесами без втручання людини або з мінімальним його участю, наприклад для прийняття найбільш важливих і відповідальних рішень, в цих процесах.

З розвитком електронних і програмних засобів автоматизація почала розвиватися ще стрімкіше, приходячи в ті галузі, де раніше необхідна була безпосередня участь людини.

Серед усіх галузей автоматизація також може бути дуже корисна та у освітньому процесі. Це може надати такі переваги, як пришвидшення процесів, збільшення об’ємів виконаних робіт, звільнення часу, зменшення помилок у порівнянні з ручними діями, через втомленість роботи з однотипними операціями.

Однією з можливостей автоматизації є автоматизація формування бази проектів.

Такі системи можуть бути застосовані для подальшої верифікації проектів, або наприклад для створення глобальної бази проектів, яка дозволить перевіряти проекти на повторне використання. Вище описані способи застосування системи може дозволити зекономити час та сили людини, пришвидшення процесів перевірки, підвищення якості перевірок, та можливе підвищення індивідуальності проектів. Також систему можна інтегрувати в більші системи для досягнення ще більшої кількості переваг та збільшення ефективності подібних систем автоматизації [1].

Метою роботи є аналіз автоматизації в сфері освіти, аналіз та вибір засобів розробки, розробка алгоритму для системи автоматизованого формування бази проектів для верифікації та реалізація програми.

В звіті розглянуто питання розробки системи автоматизованого формування бази проектів для верифікації.

В даній атестаційній роботі пропонується автоматизувати формування бази проектів, а саме автоматизоване закачування проектів з EDA Playground, створених мовою VHDL.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Застосування системи автоматизованого формування бази проектів

В останні роки спостерігається тенденція протиріччя сучасного процесу навчання у вузі:

– постійне збільшення кількості інформації;

– скорочення аудиторного часу на процес навчання;

– необхідність дотримання єдиних вимог до рівня підготовки фахівців відповідно до освітнього стандарту при різному рівні підготовки і здібностей студентів.

Це протиріччя можна вирішити, тільки активно впроваджуючи інформаційні технології в усі види навчального процесу вузу.

Розвиток засобів обчислювальної техніки в останні роки призвів до автоматизації багатьох областей людської діяльності, в тому числі і вищої освіти. У вищих навчальних закладах можна виділити наступні приклади застосування комп'ютерних технологій та автоматизації [2, 3]:

– ведуться бази даних викладачів, студентів та інших працівників вищого навчального закладу;

– надається електронний розклад для очних і заочних форм навчання;

– створюють електронні навчальні системи: лабораторні роботи, тренажери, моделі процесів і систем;

– студенти очної, заочної та дистанційної форми навчання по-різному взаємодіють з викладачами.

Однак для всіх форм навчання деякі процеси взаємодії можна автоматизувати [4].

Для реалізації проектів автоматизації навчального процесу доцільно використовувати інтернет-технології. Це обумовлено тим, що саме інтернет-технології задовольняють вимогу загальнодоступності. Незалежно від виду

електронного пристрою, яким користуються студенти для отримання інформації, а також незалежно від операційної системи і прикладних програм на цьому пристрої, у студентів не повинно виникати проблем з отриманням або передачею інформації [5].

Використання інтернет-браузерів добре задовольняє перерахованим вимогам. Браузери забезпечують доступ з будь-якого комп'ютера, підключеного до мережі Інтернет, і надають однаковий інтерфейс доступу до даних, незалежно від програмно-апаратної платформи. Разом з тим, поширення інформації через інтернет характеризується можливістю миттєвого оновлення і необмежений за часом доступ до інформації.

Розміщення всієї необхідної інформації на сервері дозволяє здійснювати своєчасне її оновлення і перевірку викладачами. А організація постійного доступу до інформації, розміщеної на сервері, дозволяє кожному студенту або викладачеві отримувати інформацію в будь-який час дня і ночі, і що не менш важливо, з будь-якої точки планети, де є доступ до інтернет з'єднання, що особливо важливо для формування особистого навчального графіка працюючого студента або викладача [6].

Таким чином, оптимальною архітектурою автоматизованої системи взаємодії викладача та студентів слід вважати сервіс або платформу, яка базується на основі інтернет-серверу. Це можуть бути такі системи, як:

- системи для звичайного зберігання файлів;
- системи щільно інтегровані з розкладом навчального закладу;
- сторонні, більш спеціалізовані системи які використовуються для навчання.

Сторонні системи є найбільш привабливим варіантом, так як вони довели свою надійність і мають великий набір функцій. Однією з таких систем є EDA Playground. Вона дозволяє створювати і верифікувати проекти використовую веб інтерфейс.

В останні десятиліття важливою частиною розробки будь-якої системи стала верифікація. За складом проектної команди можна бачити важливість

перевірки дизайну. За статистикою типова проектна група зазвичай складається з рівної кількості інженерів-проектувальників та інженерів з верифікації. Іноді число інженерів з верифікації перевищує число інженерів-проектувальників в два рази. Це відбувається тому, що для перевірки конструкції необхідно спочатку зрозуміти специфікації, а також сам дизайн і, що більш важливо, розробити підхід до дизайну, відмінний від специфікацій. Слід підкреслити, що підхід інженера з верифікації відрізняється від підходу інженера-проектувальника. Якщо інженер з верифікації слідує тому ж стилю проектування, що і інженер-конструктор, обидва будуть робити однакові помилки, і не так багато буде перевірено. З циклу розробки проекту ми можемо зрозуміти складність перевірки дизайну.

Статистичні дані показують, що близько 70% циклу розробки проекту присвячено перевірці дизайну. Інженер-проектувальник зазвичай створює проект на основі випадків, що представляють специфікації. Проте, інженер з верифікації повинен перевіряти проект у всіх випадках, а існує нескінченна кількість випадків (принаймні, так здається). Навіть при великих витратах ресурсів на верифікацію досить складний чіп нерідко проходить через кілька операцій верифікації, перш ніж його можна буде випустити на ринок для отримання доходу. Вплив ретельної перевірки конструкції неможливо переоцінити. Несправний чіп не тільки виснажує бюджет за рахунок витрат на повторне використання, але і затримує виведення продукту на ринок, впливає на дохід, скорочує частку ринку і втягує компанію в ситуацію відставання від конкурентів. Тому до тих пір, поки люди не зможуть розробити ідеальні мікросхеми або поки виробництво мікросхем не стане недорогим і не матиме маленькі терміни виконання, верифікація залишиться однією з невід'ємних і дуже важливих частин розробки.

І як було зазначено вище, на процес верифікації йде велика частина часу, яку можна скоротити шляхом автоматизації частини або всього процесу.

Для автоматизації EDA Playground, яка використовуються для навчання можна скористатись системою автоматизованого формування бази проектів для верифікації. Це дозволить скоротити час, що витрачає викладач, на перевірку робіт та дати первинні результати перевірки.

1.2 Огляд EDA playground

EDA Playground це платформа, яка надає інженерам, студентам та викладачам змогу розробляти проекти SystemVerilog, Verilog, VHDL, C++/SystemC та інші HDL. Все, що необхідно для користування платформою, це веб-браузер та підключення до мережі інтернет. EDA Playground призначена для навчання, практики, або швидкого тестування та макетування проектів. EDA Playground має велику кількість симуляторів:

- Synopsys VCS (Commercial simulator for VHDL and SystemVerilog);
- Cadence Incisive (Commercial simulator for VHDL and SystemVerilog);
- Aldec Riviera-PRO (Commercial simulator for VHDL and SystemVerilog);
- Incisive Specman Elite;
- GHDL (an open-source simulator for the VHDL language);
- Icarus Verilog;
- GPL Cver;
- VeriWell.

EDA Playground дозволяє розроблювати пристрої та писати до них тести. На рисунку 1.1 можна побачити головний інтерфейс та два вікна – праве для опису пристрою, ліве для написання тестів.

Використовуючи симулятори можна просимулювати роботу пристрою, для цього треба обрати симулятор, встановити галочку біля "Open EPWave after run", після чого запустити симуляцію, після цього відкриється вікно "EPWave" яке можна побачити на рисунку 1.2.

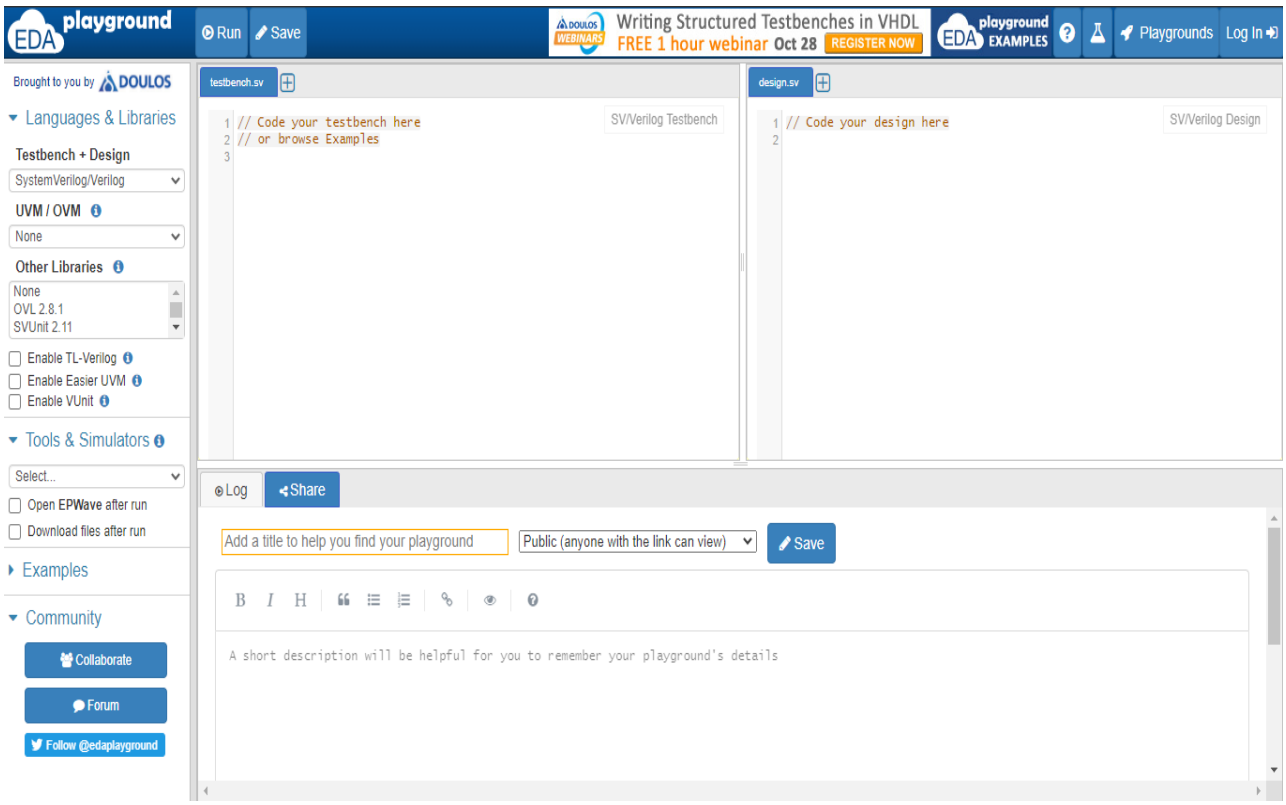


Рисунок 1.1 – Інтерфейс проекту EDA Playground

Незаперечною перевагою EDA Playground є те, що усі створені проекти можна зберігати у власному онлайн сховищі, це дозволяє вносити зміни у проект в будь-який час, або мати доступ до проекту у будь-якій точці земного шару, за умови, якщо там є інтернет з’єднання.

Проект, який зберігається онлайн, також можна завантажити на локальний комп’ютер для подальшої роботи з ним.

Для автоматизації створено вже багато програм, які вирішують різноманітні завдання. Одним з таких завдань є автоматизація створення бази проектів на основі веб сервісу EDA Playground. Таким чином доцільно буде розглянути програми для автоматизації скачування і роботи з веб інтерфейсом. Далі будуть розглянуті програми найбільш підходящі під поставлені завдання.

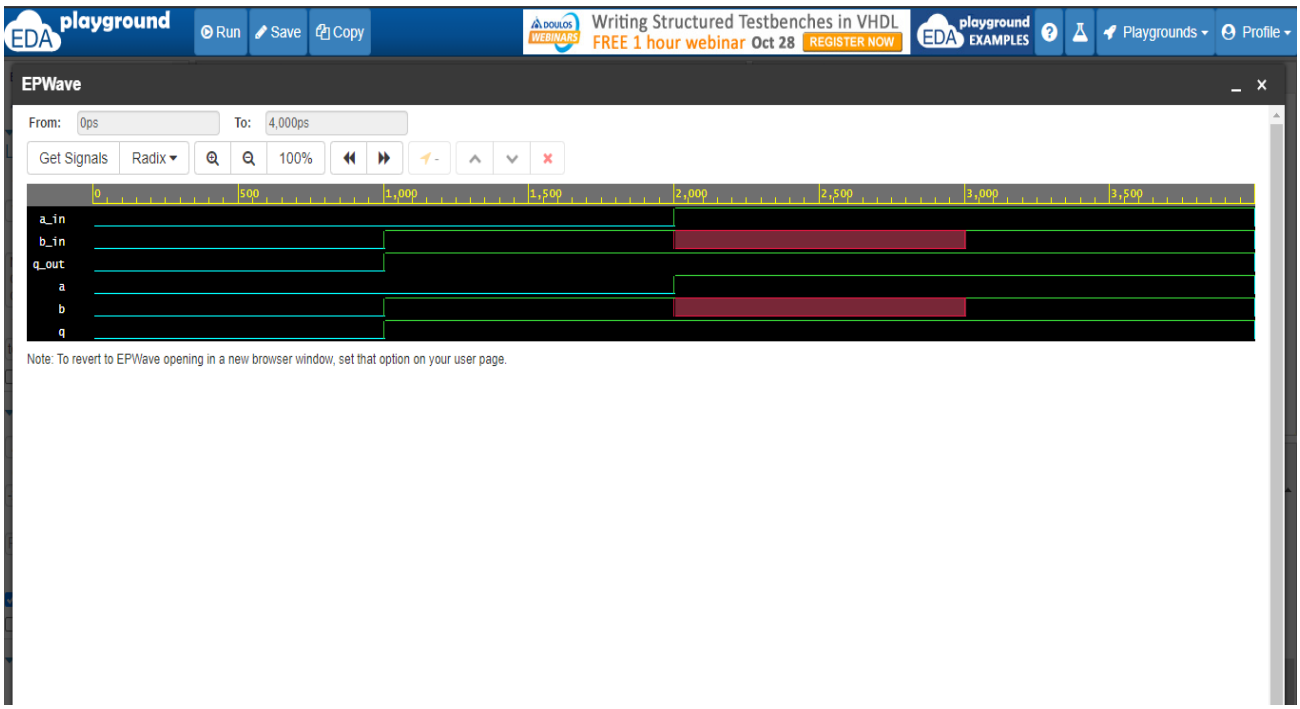


Рисунок 1.2 – Вигляд вікна EPWave

1.3 Огляд програм для автоматизованого завантаження

1.3.1 WinHTTrack

WinHTTrack – безкоштовна програма для скачування на комп'ютер цілих web-сайтів з подальшим переглядом їх в офлайн режимі. При відкритті збереженого сайту, він нічим не буде відрізнятися від онлайнової версії: повністю зберігається структура каталогів і папок, HTML, зображення та інші параметри web-сторінок.

Працювати з WinHTTrack просто, завантажити будь-якої ресурс зможе будь-який користувач. Це можливо завдяки зручному покроковому майстру, який не містить складних налаштувань і опцій. У завдання користувача входить:

- вказати URL-посилання потрібного ресурсу;
- вказати місце збереження на ПК.

Програма має наступні функції:

- режим докачування перерваних завантажень;
- режим оновлення наявних проектів;
- дозволяє копіювати вибрані сторінки сайту;
- може шукати дзеркала поточного ресурсу.

Перед початком завантаження сайту можна визначити, які типи файлів потрібно завантажувати, а кам можна пропустити, визначити назви файлів і папок, які не потрібно завантажувати. При цьому можна вказати як точні імена, так і літерні і числові поєднання, які можуть бути частиною назви. Також програма може працювати у фоновому режимі, можна встановити мінімальну швидкість завантаження. У цьому випадку вона не буде заважати виконанню інших завдань.

В налаштуваннях WinHTTrack Website Copier можна також задати обмеження на розмір файлів, причому це можна зробити окремо для HTML-файлів і файлів всіх інших типів. Крім того, можна вказати кількість одночасних з'єднань і глибину викачування, тобто скільки рівнів програма буде викачувати сторінки.

На рисунках 1.3, 1.4 та 1.5 показано інтерфейс налаштування програми.

На відміну від багатьох інших програм, де налаштування завантаження можна визначити тільки початку виконання завдання, в WinHTTrack Website Copier управляти закачуванням можна находу. Коли програма починає роботу, ви можете бачити всі посилання, які зараз закачуються, а також відсоток завантаження для кожної з них.

Рисунок 1.3 – Вікно первинного налаштування програми

Всі завантажені проекти можна переглянути за допомогою зручної команди «Перегляд» меню «Файл».

Рисунок 1.4 – Вікно налаштування відзеркалення програми

Після її вибору в браузері, який встановлений в системі за замовчуванням, відкриється згенерована програмою сторінка зі списком всіх завантажених проектів можна буде вибрати потрібний і почати перегляд сайту в режимі, відключеному від інтернету [7].

Рисунок 1.5 – Вікно додаткових налаштування програми

1.3.2 ParttGetPriceProgram

Програма із заданим інтервалом часу (або по ручній команді) викачує файли з різних джерел: листи електронної пошти, FTP або HTTP-адреси, і зберігає їх в каталог впорядкованої структури. При цьому витягує файли з архівів, може перейменовувати файли і міняти їх формат (csv, xls, txt). Можна налаштувати вивантаження оброблених файлів на сайт (через FTP-підключення). Програма буде корисна для автоматизованого скачування будь-яких файлів. Зібрані таким чином файли можна вивантажити за допомогою програми на сайт (або на будь-який FTP-сервер) або виконати інші необхідні завдання.

Програма збирає з певною періодичністю файли з різних джерел. Якщо це електронна пошта, то аналізується адреса відправника і спрацьовує налаштоване для даного відправника правило обробки. Файл з прайс-листом шукається за частковим збігом імені, в тому числі всередині архівів. Після знаходження файл перейменовується, при необхідності змінюється його формат і зберігається в каталог, перезаписуючи стару версію файлу. Для кожного файлу вказується дата останнього успішного оновлення. На рисунку 1.6 можна бачити вкладку налаштування програми. На рисунку 1.7

показаний лог роботи програми. На рисунку 1.8 показаний процес пошуку в пошті файлів для скачування.

Рисунок 1.6 – Вкладка налаштувань програми

Для FTP-серверів необхідно вказати рядок підключення відразу з ім'ям користувача і паролем. Вказати також частину імені файлу прайс-листа для його пошуку і правило, куди зберігати, яке ім'я файлу встановити і чи потрібно змінювати формат.

Рисунок 1.7 – Лог роботи програми

Аналогічно програма налаштовується для HTTP-посилань.

Рисунок 1.8 – Процес пошуку в пошті файлів для скачування

Однією з особливостей програми є те, що не з усіх сайтів виходить завантажувати файли, якщо посилання на файли доступні тільки в особистому кабінеті [8].

1.3.3 VistaTask

VistaTask являє собою ефективний і надійний інструмент для швидкої автоматизації більшості виникають при роботі в Windows завдань. Додаток відрізняється зручним, інтуїтивно зрозумілим і легко настроюється інтерфейсом, надає широкий набір підтримуваних дій і дозволяє

користувачам створювати сценарії дій без уміння програмувати – шляхом візуального вибору дій і налаштування їх параметрів. Кількість допустимих в сценаріях дій забезпечує автоматизацію як простих завдань, що виникають у домашніх користувачів, так і деяких бізнес-процесів, що актуально вже для компаній. Пакет поставляється разом з докладною документацією, доповненою серією навчальних завдань, які можна взяти за основу і відредагувати для вирішення своїх конкретних проблем. Програма порівняно нескладна в освоєнні, хоча новачкам все ж краще зупинитися на більш дешевому і простому додатку.

VistaTask підтримує дуже широке коло завдань, основними з яких можна назвати наступні:

– швидкий запуск програм, відкриття документів, виконання програм MS-DOS, запуск і зупинка сервісів, відкриття панелі управління та ін.;

– робота з вікнами: активація, відкриття на весь екран, мінімізація робочого вікна або всіх відкритих вікон, зміна розмірів робочого вікна та ін.;

– використання клавіатури, робота з мишею і використання меню, натискання певної комбінації клавіш, блокування/розблокування введення, різні варіанти переміщення і натискання миші, виділення пункту системного або призначеного для користувача меню та ін.;

– робота з файлами і папками-створення нового файлу, читання, збереження і видалення файлів, копіювання, перейменування і переміщення файлів, створення і видалення папки і т. д.;

– робота в інтернеті: завантаження інтернет-браузера, відкриття і збереження Web-сторінки, створення, відправка і видалення поштового повідомлення, скачування, видалення файлів по FTP та ін.;

– виконання системних дій-копіювання тексту в буфер обміну і вставка тексту з буфера, очищення буфера обміну, створення скріншота активного вікна, перезавантаження і виключення комп'ютера та ін.

На рисунках 1.9 та 1.10 можна бачити інтерфейс програми та сценарій та його компіляцію у виконуваний файл.

Рисунок 1.9 – Вигляд інтерфейсу програми

У списку дій можуть також бути присутніми вирази «If» і «TextLoop» – перше забезпечує виконання дій в залежності від результатів порівняння параметрів з деякими заданими значеннями, а друге дозволяє виконувати циклічно повторювані дії. Крім того, в якості дій можуть фігурувати такі операції, як перевірка існування конкретного файлу, завантаження деякої програми, відкриття певного вікна і так далі. Отриманий сценарій може бути скомпільований в exe-файл, що дозволить згодом використовувати його поза VistaTask. Для успішного налагодження складних сценаріїв в програмі передбачені можливості впровадження контрольних точок і покрокового виконання.

Рисунок 1.10 – Сценарій та його компіляція у виконуваний файл

Слід зазначити, що можливість запуску сценаріїв за розкладом в цій програмі не передбачена, так як передбачається, що сценарії запускаються безпосередньо користувачем. Це до деякої міри обмежує коло вирішуваних завдань, проте цілком можна вийти з положення і автоматизувати в VistaTask навіть ті операції, які повинні виконуватися строго за розкладом або при відсутності користувача, для чого достатньо підготувати необхідний сценарій у вигляді exe-файлу, а в планувальнику Windows встановити час його запуску [9].

1.3.4 Висновки огляду програм автоматизації

При огляді програм для автоматизації було розглянуто три програми, а саме:

- WinHTTrack;
- ParttGetPriceProgram;
- VistaTask.

WinHTTrack – це програма для скачування веб сторінок, сайтів і файлів з FTP серверів. Ця програма не підходить, так як вона не зможе завантажити необхідні проекти з сайту EDA Playground, оскільки для скачування проекту після переходу на сторінку проекту необхідно провести деякі маніпуляції, щоб почалося скачування.

ParttGetPriceProgram – програма для скачування файлів з сайтів. Ця програма може логінитися на сайті, але так як HTML сторінка проекту є динамічно змінною, а скачування проекту в EDA Playground починається тільки після натискання кнопок на сторінці, Ця програма також не підходить для виконання завдання, так як не може виробляти дії на сторінці.

VistaTask – це програма призначена для автоматизації широкого спектра функцій, з усіх програм ця підходить найбільше, але у неї є один недолік, а саме оскільки ця програма управляє елементами комп'ютера, тим самим не дозволяючи працювати за цим комп'ютером під час виконання сценарію, що може зайняти тривалий час, що є неприпустимим, тому ця програма також не підходить для виконання завдання.

Оскільки жодна з розглянутих програм не задовольнила всіх вимог, необхідно створити програму призначену для вирішення поставленого завдання.

1.4 Постановка завдання

Мета атестаційної роботи – вибір методів та засобів розробки, розробка алгоритму роботи та реалізація системи автоматизованого формування бази проектів для верифікації.

Основні завдання проектування наступні:

– аналіз відомих способів автоматизованої взаємодії з веб інтерфейсом;

– аналіз способів зберігання бази проектів;

– аналіз мов програмування, які найбільш підходять для реалізації поставленої задачі;

– розробка алгоритму роботи системи;

– розробка програмного засобу.

2 ОГЛЯД ТА ВИБІР МЕТОДІВ, ЗАСОБІВ ДЛЯ РЕАЛІЗАЦІЇ СИСТЕМИ

2.1 Способи взаємодії з веб сервісом

При автоматизованій роботі з веб-сервісом є декілька способів взаємодії з віддаленим сервером:

- за допомогою HTTP запитів;
- API;
- за допомогою автоматизованої навігації веб сторінкою.

2.1.1 HTTP запити

Перший метод достатньо простий, підходить для парсинга сайтів.

Парсинг – це синтаксичний аналіз сайтів, який проводить спеціальна програма або скрипт. Зібрана інформація може надаватися в певному виді, за певними правилами, алгоритмами і проводиться на одній з мов програмування.

Об'єктом може бути довідник, інтернет-магазин, форум, блог і абсолютно будь-який інтернет-ресурс.

Парсинг сайтів – це найкращий спосіб автоматизувати процес збору та збереження інформації. Завдяки парсеру можна створювати і оновлювати сайти, схожі з оформлення, змісту та структури.

Іншими словами, за допомогою HTTP запиту парсер отримує вихідний код сторінки, програма проходить по ньому, як по звичайному тексту, і знаходить деякі відповідності, які записані в програмний код. Він опрацьовує таким чином всю сторінку та зберігає те, що закладено у її підпрограмах. Останній крок – збереження в зручному для вас форматі даних. Тобто програми або скрипти будуть зберігати інформацію у тому вигляді, у якому її запрограмували, для подальшої обробки цієї інформації, або виводити її у

тому чи іншому вигляді. Для реалізації цього способу необхідне використання бібліотек.

На мові Python є три бібліотеки для реалізації подібного функціоналу. Перша це pandas, друга Requests і третя urllib/urllib2.

Бібліотека pandas це високорівнева Python бібліотека для аналізу даних. Вона є високорівневою, тому що побудована вона поверх більш низькорівневої бібліотеки NumPy (написана на сі), що є великим плюсом в продуктивності. В екосистемі Python, pandas є найбільш просунутою і швидко розвивається бібліотекою для обробки та аналізу даних. Ця бібліотека дозволяє брати дані з різноманітних джерел, один з яких це HTTP запити, і парсить вміст сторінки.

Requests – це модуль Python, який ви можете використовувати для надсилання всіх видів HTTP-запитів. Це проста у використанні бібліотека з безліччю функцій, починаючи від передачі параметрів в URL-адресах до відправки користувальницьких заголовків і перевірки SSL. У бібліотеці requests всі заголовки і параметри додаються дуже просто, також і обробка відповідей сервера. Бібліотека requests працює на базі urllib2, але в requests реалізована вся складна логіка.

urllib/urllib2 – це модуль Python, який створений, щоб допомогти працювати з URL-адресами. Вони визначає функції та класи для обробки URL-адрес. Модулі мають свої функції і класи, які допомагають в роботі з URL-basic і digest аутентифікації, перенаправлення, cookie і багато іншого.

Незважаючи на схожі назви ці модулі незначно відрізняються. urllib2 в якості аргументу може приймати Request object, щоб додавати заголовки до запиту і інше, в той час як urllib може приймати тільки стрінговий URL. Бібліотека urllib має метод urlencode, який використовується для кодування рядка у вигляд, який відповідає правилам даних у запитах, а urllib2 не має такої функції. Через це urllib і urllib2 часто використовуються разом.

На мові Perl є дві бібліотеки для реалізації подібного функціоналу. Перша це LWP (скорочення від "Library for WWW in Perl"), а друга AnyEvent::HTTP.

LWP (скорочення від "Library for WWW in Perl") це дуже популярна група модулів мови Perl для доступу до даних в мережі Internet. Як і більшість модулів мови Perl, на кожен модуль зі складу LWP присутня документація з повним описом його інтерфейсу. Однак, в LWP є безліч модулів, для яких досить складно знайти документацію по, здавалося б, найпростішим речам. Один з модулів дозволяє просто отримати документ, що знаходиться за певною адресою (URL), найпростіший спосіб зробити це – використовувати функції модуля LWP:: Simple. Функції LWP:: Simple зручні тільки для простих випадків, але ці функції не підтримують тіньових посилок(далі cookies) і перевірки автентичності(далі authorization); вони також не дозволяють встановлювати будь-які параметри HTTP запиту; і головне, вони не дозволяють зчитувати рядки заголовка в HTTP відповіді (особливо повний текст повідомлення в разі HTTP помилки (HTTP error message)). Для доступу до всіх цих можливостей, ви повинні використовувати весь набір класів LWP [10].

LWP містить безліч класів, але головні два, які ви повинні розуміти – це LWP::UserAgent і HTTP::Response. LWP:: UserAgent це клас для "віртуальних браузерів, які ви будете користуватися для виконання запитів. HTTP::Response це клас для відповідей (або повідомлень про помилку), які ви отримуєте назад, після запиту.

AnyEvent::HTTP являє собою не блокуючий HTTP/HTTPS клієнт. Є підтримка проксі, keep-alive з'єднань і сесій, всіх HTTP методів і обробки cookie і багато іншого, і все це на низькому рівні. Він може відстежувати перенаправлення, підтримує проксі-сервери і автоматично обмежує кількість підключень значеннями, зазначеними в RFC. Не реалізована підтримка HTTP-авторизації [11].

2.1.2 API

API (Application Programming Interface) являє собою сукупність різних інструментів, функцій, реалізованих у вигляді інтерфейсу для створення нових додатків, завдяки якому одна програма буде взаємодіяти з іншою.

Основним завданням створення API була дати можливість програмістам істотно полегшити задачу при розробці різних додатків за рахунок використання вже готового коду (будь-якої стандартної функції, процедури, структури або постійного значення, які будуть в подальшому виконуватися в кінцевому продукті).

API визначає можливу функціональність, яку певна програма в формі бібліотеки або модуля зможе виконувати, при цьому API дозволяє абстрагуватися від способу реалізації функціоналу.

Різні компоненти програми завдяки API отримують можливість взаємодії один з одним, або з віддаленими ресурсами. Так, наприклад компанія з надання будь-яких послуг може створити API для найпростішої інтеграції їх сервісів в додаток клієнта. Це здешевлює розробку програми за рахунок того, що клієнтам не потрібно буде витрачати час на розробку своїх API, до того ж це дозволить більш ретельно відтестувати та усунути всі недоліки в наданих API, які розробляє постачальник.

Веб сервіс EDA Playground немає офіційних API для розробки сторонніх додатків для взаємодії з EDA Playground, а жодні сторонні API не підходять для взаємодії з EDA Playground.

2.1.3 Автоматизована навігація веб сторінкою

Автоматизована навігація веб сторінкою іноді дозволяє зробити те, що неможливо отримати за допомогою парсинга та API. У випадку, якщо немає API або за допомогою запитів або парсинга не можливо отримати потрібний результат, можна використовувати автоматизовану навігацію веб сторінкою. Це дозволить імітувати людські дії для отримання бажаного результату, при цьому без втручання самої людини. Цей спосіб вимагає наявності необхідної

для навігації бібліотеки, а також знань HTML для складання програми навігації сайтом. Такий вид взаємодії дозволяє отримати данні недоступні через API або через запроси та парсинг сторінок сайту. Для реалізації автоматизованої навігації веб сторінкою необхідне використання бібліотек.

На мові Python є одна бібліотека для реалізації автоматизованої навігація по сторінці, яка називається Selenium.

Selenium WebDriver – це програмна бібліотека для управління браузерами за допомогою WebDriver. WebDriver являє собою драйвери для різних браузерів і клієнтські бібліотеки на різних мовах програмування, призначені для управління цими драйверами. Бібліотеки WebDriver доступні на мовах Java, .Net(C#), Python, Ruby, JavaScript, драйвери реалізовані для браузерів Firefox, InternetExplorer, Safari, Andriod, iOS (а також Chrome і Opera). Selenium дозволяє отримувати результати виконання HTTP запиту і проводити різноманітні маніпуляції з вмістом, наприклад натискати на елементи веб сторінки, вводити текст в поля і переміщатися по сторінці і посиланнях в цілому [12, 13].

На мові Perl є дві бібліотеки для реалізації автоматизованої навігація по сторінці. Перша – це WWW::Mechanize::Chrome, а друга Win32::GuiTest.

WWW::Mechanize::Chrome (WMC) використовує інструменти розробника, вбудовані в Chrome і браузери, подібні Chrome, для програмного управління екземпляром браузера. WMC можете також використовуватися для автоматизації рутинних завдань, тестування веб-додатків і збору інформації з веб сторінок. Зазвичай WMC використовується як для запуску хост-екземпляра браузера, так і для надання клієнтського екземпляра браузера. Хост-екземпляр браузера видно на робочому столі (якщо браузер не працює в режимі «headless», і в цьому випадку він не буде відкриватися у вікні). Екземпляр клієнта – це програма Perl, яка використовує модуль WMC для виконання команд для управління екземпляром браузера. Ця бібліотека дозволяє отримувати результати виконання HTTP запиту і проводити різноманітні маніпуляції з вмістом, наприклад натискати на елементи веб

сторінки, вводити текст в поля і переміщатися по сторінці і посиланнях в цілому. Все це відбувається в результаті команд, які відправляються вашим клієнтом хосту з використанням протоколу Chrome DevTools, який реалізує протокол http для відправки структур даних JSON. Хост також відповідає клієнту за допомогою JSON, щоб описати завантажені веб-сторінки. WMC зручно приховує складність зв'язку нижнього рівня між клієнтським і хост-браузерами і обертає їх в об'єкт Perl, щоб надати прості у використанні методи [14].

Win32::GuiTest – Perl-модуль для автоматизації операцій з графічним інтерфейсом Win32. Він надає набір методів для управління віконними дескрипторами, елементами управління і імітації призначеного для користувача введення. Цей модуль дозволяє взаємодіяти з віконним інтерфейсом Windows, або управляти інтерфейсом за допомогою імітації введення з клавіатури [15].

У таблиці 2.1 можна побачити зведення за всіма способами і бібліотекам.

Таблиця 2.1

	Python	Perl
HTTP запити	pandas, Requests, urllib / urllib2	LWP, AnyEvent::HTTP
API	–	–
Автоматизована навігація веб сторінкою	Selenium	WWW::Mechanize::Chrome, Win32::GuiTest

2.2 Огляд способів зберігання бази проектів

Так як при автоматизації будь-якого процесу необхідні вхідні дані і вихідні дані, то ці дані потрібно будь-яким чином задавати і зберігати. У випадку з автоматизованою системою створення бази проектів є три способи зберігання інформації:

- зберігання в базі даних;
- зберігання в хмарному сервісі;
- зберігання на жорсткому диску.

2.2.1 Бази даних

База даних – це впорядкований набір структурованої інформації або даних, які зазвичай зберігаються в електронному вигляді в комп'ютерній системі. База даних зазвичай керується системою управління базами даних (СКБД). Дані разом з СКБД, а також додатки, які з ними пов'язані, називаються системою баз даних, або, для стислості, просто базою даних.

Дані в найбільш поширених типах сучасних баз даних зазвичай формуються у вигляді рядків і стовпців в ряді таблиць, щоб забезпечити ефективність обробки і запитів даних. Потім можна легко отримувати доступ до даних, керувати ними, змінювати, оновлювати, контролювати і впорядковувати. У більшості баз даних для запису і запитів даних використовується мова структурованих запитів (SQL).

Далі варто розглянути переваги та недоліки використання СКБД, тобто СУБД, призначених для широкого використання в структурах інформаційних систем.

Переваги СКБД.

Більшість сучасних СУБД розрахована на роботу в багатозадачному і багатокористувацькому режимі. Перше означає, що СУБД може запускати одночасно кілька завдань (потоків) обробки даних. Наприклад, запускати запити від користувача і, одночасно, за розкладом запускати будь-які завдання управління даними, скажімо, резервне копіювання, обмін даними з іншими СКБД (реплікації) і т.п. у багатопроцесорних системах Сучасні СКБД можуть економно розподіляти процесорне навантаження між усіма виконуваними потоками (розподіляти потоки між усіма процесорами).

З іншого боку, сучасні СУБД дозволяють одночасно реєструватися безлічі користувачів. При цьому, використовуючи багатозадачність, СУБД може одночасно обробляти запити від безлічі користувачів або джерел. Всі сучасні СУБД підтримують транзакційні механізми і блокування для безпечної роботи одночасно декількох користувачів.

Система безпеки. Всі СКБД, особливо промислового типу (сервери баз даних), підтримують систему обмеження доступу для різних користувачів і груп користувачів. Використовуючи цю можливість, можна побудувати ефективну систему захисту конфіденційності даних. При розробці інформаційної системи на основі СУБД можна просто скористатися вже готовою системою безпеки або вдосконалити її, інтегрувавши систему безпеки ІС в систему безпеки СКБД. СКБД надає різні механізми обмеження доступу до даних: за допомогою уявлень, збережених процедур і функцій, через безпосереднє обмеження доступу до таблиць (обмеження операцій, які може виконувати користувач в таблиці). Крім цього, СКБД дозволяють зберігати дані в зашифрованому вигляді.

Резервне копіювання. Резервне копіювання – це простий і разом з тим найбільш ефективний спосіб захисту даних від пошкодження. Сучасні СУБД мають досить потужною системою резервного копіювання. Ця система дозволяє робити резервне копіювання без участі користувача за розкладом або настанню будь-якої події. Крім цього, сучасні СУБД дозволяють копіювати не всю базу даних, а тільки ті зміни, які були в ній зроблені з часу останнього копіювання (диференціальне копіювання). Також СУБД підтримують і дзеркальні бази даних, які гарантовано містять копію тієї чи іншої бази даних.

Підтримка транзакційних механізмів. Всі сучасні СКБД підтримують транзакційні механізми. Ці механізми є одним з важливих засобів захисту даних від збоїв. Транзакції гарантують, що навіть після збою дані будуть перебувати в несуперечливому стані. Використання транзакцій в багатокористувацької роботі призводить до взаємодії транзакцій один з одним, що може привести до отримання неадекватної інформації від інформаційної системи. Для запобігання такій ситуації в СКБД передбачені досить складні механізми блокувань, що дозволяють усунути небезпечну взаємодію.

Контроль цілісності даних. Всі СКБД підтримують в тому чи іншому вигляді контроль цілісності баз даних. Цей контроль дозволяє гарантувати унікальність ключів, несуперечливість даних, що зберігаються в стовпцях таблиць, відповідність зовнішніх і первинних ключів. Крім цього, СКБД надають найбільш потужний механізм підтримки корпоративної цілісності – тригери.

Підтримка стандартів. Різні СКБД в тій чи іншій мірі підтримують існуючі стандарти, зокрема стандарти SQL. Наявність тієї чи іншої підтримки стандарту дасть певну гарантію (хоча і не повну) використовувати наступні версії даної СКБД або перехід на іншу СУБД. Крім цього, дотримання стандартів дозволяє легше інтегрувати ІС з іншими інформаційними системами.

Масштабованість. Розробники СКБД велике значення надають масштабованості свого продукту. Це означає, що:

– при поліпшенні апаратної частини комп'ютера (збільшення обсягу оперативної пам'яті, кількості процесорів і т. д.) продуктивність системи повинна збільшуватися;

– збільшення обсягів оброблюваної інформації (обсяг баз даних, кількість запитів за певний проміжок часу) не повинно сильно позначатися на продуктивності системи.

Розробники СКБД вже багато років працюють над цією проблемою, покращуючи масштабованість своїх СКБД від версії до версії. Це дуже важлива перевага перед інформаційними системами, які розробляються без використання СКБД широкого застосування.

Показники продуктивності. Велике значення надають розробники СУБД показниками продуктивності. Це один з найбільш конкурентних показників будь-якої СУБД. Прикладний програміст може повністю покластися на високі показники продуктивності СУБД і зосередитися на інших властивостях розроблюваної системи.

Наявність засобів адміністрування даних. Найважливіша перевага використання СКБД загального призначення – це наявність в них всілякого інструментарію з адміністрування бази даних. Як правило, СКБД надає програми для візуальної розробки баз даних. Це першорядна перевага таких СКБД. Інтерактивна взаємодія з базою даних крім розроблюваної програми дає можливість розробником оперативно реагувати на мінливу ситуацію, поява нових вимог, виявлення помилок і т. д. Крім цього, для найбільш популярних СКБД є велика кількість програм адміністрування, створених сторонніми розробниками).

Недоліки СКБД.

Складність супроводу. Деякі СКБД досить складні для адміністрування. Для того щоб використовувати функціональність СКБД найбільш повно, потрібно, щоб адміністратор СКБД, адміністратори баз даних, кінцеві користувачі, розробники системи володіли певними знаннями в області баз даних, а також в можливостях конкретної СКБД. Безсумнівно, навчання персоналу або підбір кваліфікованих кадрів вимагає певних матеріальних витрат, які збільшуються з підвищенням складності програмного продукту.

Розмір СКБД. СУБД можуть вимагати значного дискового простору та інших ресурсів. Самі бази даних, у зв'язку зі складною їх структурою, також можуть зажадати значного дискового простору.

Вартість СУБД. Вартість сучасних СУБД може бути дуже суттєвою і досягати величезних сум (до мільйона доларів). Крім цього, потрібні великі витрати на супровід системи. Все це, зрозуміло, увійде у вартість розроблюваної інформаційної системи і у вартість її експлуатації.

2.2.2 Хмарні сховища

Хмарне сховище – це віртуалізація системи зберігання. У хмарі дані також зберігаються на якомусь обладнанні. Працюючи з хмарними сховищами користувач просто відправляєте дані в хмарне сховище через інтерфейс доступу до хмарного сховища. При цьому інтерфейс хмарної системи зберігання може бути схожий на звичайний файловий менеджер на комп'ютері.

Робота хмарного сховища. Незалежно від типу системи хмарної системи зберігання: внутрішньої (приватної, private) або зовнішньої (public, публічної) – принцип їх роботи наступний. Провайдер послуг хмарного сховища (приватного або публічного), надає свої IT-інфраструктуру, яка забезпечує надійне і безпечне управління потрібними серверами для зберігання даних.

Публічний хмарний сервіс – це віртуалізована система зберігання, послуги якої надає зовнішній провайдер. У його дата-центрі зберігаються дані багатьох клієнтів, на умовах "багатоарендності" (multi-tenancy), без взаємного впливу один на одного. За рахунок оптимального і централізованого використання ресурсів вдається досягти цінової ефективності.

Приватний хмарний сервіс – віртуалізована система зберігання, організована в масштабах підприємства. У ній є виділений дата-центр (ЦОД), у віртуалізованої інфраструктурі якого зберігаються дані і працюють додатки підприємства. У цьому випадку роль провайдера хмарних послуг найчастіше виконує IT-служба підприємства.

Сервери приватного або публічного хмарного сховища працюють не як незалежні системи всередині структури хмарного сховища, а як єдина група серверів. Для цієї мети дисковий простір разом з іншими компонентами сервера (наприклад, CPU або оперативною пам'яттю) віртуалізується з використанням гіпервізорів. Поверх гіпервізора будуть працювати вже не фізичні сервери, CPU і накопичувачі даних, а віртуальні сервери. А в них-віртуальні машини VM (Virtual Machine), які з точки зору функціоналу аналогічні фізичним пристроям. Але вони мають гарну властивість: можуть адаптуватися під конкретні вимоги, можуть швидко мігрувати між фізичними серверами і навіть дата-центрами.

При цьому між реальним обладнанням і віртуальними функціями зберігання (Virtual Storage) виникає якийсь рівень абстрагування, на якому працює монітор віртуальних машин VMM (Virtual Machine Monitor), який ще називають гіпервізором (Hypervisor).

Структура хмарних систем зберігання. Для доступу до віртуального сховища зазвичай потрібне відповідне програмне забезпечення. Послуги публічного хмарного сховища зазвичай містять не тільки веб-додаток, яким можна користуватися через звичайний браузер, але також драйвери доступу від різних пристроїв. З їх допомогою можна війти і отримати доступ до свого диска в хмарному сховищі. Збережені там файли можна витягти через різні пристрої (комп'ютер, планшет, смартфон та ін.), підтримка яких забезпечується хмарним провайдером.

Для приватного хмарного сховища зазвичай потрібне з'єднання з сервером VPN через відповідну корпоративну мережу (Інтранет), або за допомогою послуги віртуальної приватної мережі VPN (Virtual Private Network) через публічний Інтернет.

Хмарні провайдери, у своїй внутрішній інфраструктурі зберігання, крім звичайного файлового сховища (File Storage), можуть використовувати альтернативні види форматів: блочне сховище (Block Storage) і об'єктне сховище (Object Storage).

Незалежно від використовуваного формату зберігання даних (File/Block/Object Storage) хмарні провайдери можуть використовувати у фізичному обладнанні або жорсткі диски HDD, або твердотільні диски SSD. Останні характеризуються більш високою швидкістю запису і зчитування даних всередині інфраструктури хмарного провайдера. Але вони поки дорожчі, ніж класичні HDD.

Завдяки віртуалізації можна одночасно використовувати обидва типи дисків. Наприклад, так звані. «гарячі» дані – тобто ті, до яких частіше звертаються, можна розміщувати на SSD. Як тільки звернення до якихось даних стають рідше, їх можна перевести в розряд «холодних» і перенести в HDD, глибше в систему.

В цьому і складається одна з переваг хмарного зберігання: якщо те ж саме робити всередині власної системи зберігання підприємства, потрібно будувати відповідну архітектуру, застосовувати відповідні програмні менеджери, що займаються сортуванням даних і розміщенням їх або в SSD, або в HDD. Ефективно це можна зробити тільки в масштабах великого підприємства. Для невеликих компаній це не завжди вигідно, через що можливий нераціональний витрата ресурсів і неефективне зберігання даних.

Переваги хмарного сховища. Є багато причин, за якими доцільно організовувати зберігання даних у зовнішній хмарі. Перш за все, це економія коштів на придбання та обслуговування власного серверного обладнання для зберігання. Існує поширена думка, що хмарні послуги нітрохи не дешевше – або навіть дорожче, ніж власні системи зберігання (on-premise).

При виборі хмарного зберігання вся відповідальність за нижчу інфраструктуру лежить на хмарному провайдері – це важливо враховувати при оцінці вартості його послуг.

Перевагою хмарного зберігання є і те, що оригінал і резервна копія даних будуть перебувати в різних географічних місцях. Це захищає дані в разі різних непередбачених ситуацій, які зазвичай відбуваються в самий невідповідний момент: злом системи, пожежа, відмова обладнання.

Гнучкість. Це можливість споживати рівно такий обсяг сховища, який потрібен в даний момент. Якщо потрібно більше, провайдер надає більше, плата зростає, і навпаки.

Масштабування. Віртуалізація сховища дозволяє вибирати необхідний обсяг сховища за контрактом. У будь-який час можна збільшити або зменшити обсяг сховища без закупівель обладнання, його установки і налагодження.

Доступність: хмарне сховище доступне в будь-який час і з будь-якого пристрою (при наявності нормального Інтернету). Тому отримувати дані можна на ходу, там, де вони потрібні в той чи інший момент.

Недоліки хмарного сховища. Перш за все, це залежність від інтернет-з'єднання. Якщо воно порушується, файли в хмарі стають недоступними. Важливим фактором залишається доступна смуга пропускання: навіть при самому швидкодіючому сховищі доступ до даних буде повільним через низьку швидкість з'єднання. Залежність від провайдера. Безпека. Пересилання даних – це завжди ризик. Не всі провайдери надають послугу шифрування збережених даних. Незважаючи на те, що хороші провайдери завжди намагаються забезпечити вищий рівень безпеки своїх систем, інфраструктура провайдера – бажана мета для атак хакерів.

Захист даних. Як дані будуть захищені в інфраструктурі провайдера – основне питання, яке необхідно з'ясувати при виборі хмарного зберігання даних.

2.2.3 Локальні сховища

Локальні сховища. Локальні сховища є більш простими в установці і використанні рішенням, ніж всі інші варіанти, а найголовніше-недорогими.

Локальне сховище може являти собою локальне зберігання файлів безпосередньо на комп'ютері або на віддаленому nas-сервері, який знаходиться в локальній мережі або ж підключений як локальний диск до комп'ютера. Також існує гібридний варіант локального і хмарного сховища. Цей варіант має на увазі виділення якоїсь папки в файловій системі комп'ютера, яка буде синхронізуватися з хмарним сховищем, але це не просто резервна копія даної папки, що знаходиться в хмарному сервісі, так як синхронізація в даному випадку двостороння, що дозволяє поєднати переваги обох схем зберігання, при тому усунувши деякі недоліки.

До переваг локального сховища варто віднести:

- швидкість доступу, адже в даному випадку робота проводиться всередині системи, що і дозволяє досягти таких результатів;

- простоту у використанні, що важливо при використанні зі

сторонніми програмами, які не підтримують роботу з іншими способами зберігання даних;

– доступність незалежно від зовнішніх факторів, таких, як наявність інтернет з'єднання, або несправності на стороні провайдера послуг і погодних умов;

– безпека при використанні локального зберігання файлів мінімізується можливість здобути файли, що зберігаються, сторонніми особами.

До недоліків локального сховища відносяться:

– є тільки локальний доступ, що обмежує області застосування;

– складності при масштабуванні сховища;

– зазвичай невеликий обсяг сховища при мінімальних витратах, і підвищення витрат пропорційно зростанню обсягу сховища і навантаженості сховища.

2.2.4 Вибір способу зберігання бази проектів

Після порівняльного аналізу способів зберігання інформації, були проаналізовані такі способи зберігання, як зберігання в базі даних, зберігання в сервісі хмарного зберігання і локальне зберігання файлів.

При аналізі бази даних, як способу зберігання в даному проекті, було виявлено, що використання бази даних було б не зручним, так як база даних більше підходить в проектах, де до додатка виробляються запити на отримання будь-якої інформації, а також оскільки програма повинна приймати велику кількість вхідних даних, доцільно передавати її у вигляді файлу, так само як і вихідні дані, які будуть корисні для подальшого використання. Таким чином, використання бази даних недоцільно для даних завдань.

При аналізі хмарного сховища в якості способу зберігання було визначено, що це дуже зручний з точки зору універсальності доступу до даних і надійності зберігання, але через можливі незручності і несумісності при роботі зі сторонніми програмами а також можливих збоїв при роботі з хмарними сховищами через їх недоступності через зовнішніх факторів, такий спосіб не підходить для даних завдань.

Таким чином, найбільш підходящим способом виявився спосіб локального зберігання даних, так як він задовольняє всім вимогам, необхідним при розробці і функціонуванні програми.

2.3 Вибір мови програмування для реалізації системи

Для реалізації система автоматизованого формування бази проектів для верифікації можна обрати багато мов програмування, але за ти чи інших причин вони не підходять. Для реалізації біль за все підійдуть такі мови, як:

– Perl;

– Python.

2.3.1 Огляд мови Perl

Perl – мова програмування загального призначення, яка була спочатку створена для маніпуляцій з текстом, але на даний момент використовується для виконання широкого спектру завдань, включаючи системне адміністрування, веб-розробку, мережеве програмування, ігри, біоінформатику, розробку графічних користувальницьких інтерфейсів. Згідно з даними webtech, Perl за поширеністю все ще займає почесне шосте місце серед мов на веб-серверах, програючи ColdFusion, але зате виграючи у Python.

Синтаксис Perl має багато спільного з синтаксисом мов C, AWK, sed і Bourne shell.

Мову можна охарактеризувати скоріше як практичну (легкість у використанні, ефективність, повнота), ніж гарну (елегантність, мінімалістичність). Головними перевагами мови є підтримка різних парадигм (процедурний, об'єктно-орієнтована і функціональний стилі програмування), контроль за пам'яттю (без збирача сміття, заснованого на циклах), вбудована підтримка обробки тексту, а також велика колекція модулів сторонніх розробників.

Загальна структура Perl в загальних рисах веде свій початок від мови C. Perl – процедурний за своєю природою, має змінні, вирази присвоювання, блоки коду, відокремлювані фігурними дужками, керуючі структури і функції.

Perl також запозичує ряд властивостей з мов програмування командних оболонок UNIX. Всі змінні маркуються провідними знаками, які точно виражають тип даних змінної в цьому контексті (наприклад, скаляр, масив, хеш). Важливо, що ці знаки дозволяють змінним бути інтерпольованими в рядки. Perl має безліч вбудованих функцій, які забезпечують інструментарій, часто використовуваний для програмування оболонки, наприклад сортування або виклик системних служб.

У Perl є підтримка складних типів даних, першокласних функцій (замикання як значення) і об'єктна модель. В останню входять посилання, пакети, виконання методів від класу, змінні з лексичним оголошенням області видимості, а також директиви компілятора (наприклад, strict). Найголовнішим удосконаленням, представленим в Perl 5, стала можливість поміщати код в «пакети» (package) в якості модулів для повторного використання.

Всі версії Perl виконують автоматичну типізацію даних і автоматичний контроль над пам'яттю. Інтерпретатор знає тип і запити пам'яті кожного об'єкта програми, він розподіляє і звільняє пам'ять, виробляючи підрахунок посилань. Переклад одного типу даних в інший - наприклад, числа в рядок -

відбувається автоматично під час виконання, неможливі для виконання переклади типів даних призводять до фатальної помилки.

2.3.2 Огляд мови Python

Python – це універсальна сучасна мова програмування високого рівня, до переваг якої відносять високу продуктивність програмних рішень і структурований, добре читається код. Синтаксис мови максимально полегшений, що дозволяє вивчити його за порівняно короткий час. Ядро має дуже зручну структуру, а широкий перелік вбудованих бібліотек дозволяє застосовувати значний набір корисних функцій і можливостей. Мова програмування може використовуватися для написання прикладних додатків, а також розробки WEB-сервісів.

Python може підтримувати широкий перелік стилів розробки додатків, в тому числі, дуже зручний для роботи з ООП і функціонального програмування.

Один з найпопулярніших інтерпретаторів мови – Python, написаний на C. Інтерпретатор підтримує більшість популярних платформ. Python підтримує практично всі поширені операційні системи. Він може прекрасно працювати на кишенькових комп'ютерах, так і на великих серверах. У разі, якщо платформа значно застаріває, вона виключається з підтримки ядра.

Python є інтерпретованим: вихідний код на Python не компілюється в машинний код, а виконується безпосередньо за допомогою спеціальної програми-інтерпретатора.

Всі бібліотеки на цій платформі прописуються як модулі. Перевагою такої концепції є можливість зібрати кілька модулів в пакет. Модуль може перебувати в архіві або безпосередньо в каталозі. Мова підтримує два види таких модулів-створені засобами пітона або вже перетворені в машинний код з будь-якої мови. Модулі оформляються як окремі файли і завантажуються пакетами кожен в свій каталог.

До особливостей мови можна віднести:

– легкий для навчання: у python відносно мало ключових слів, проста структура і чітко визначених синтаксис. завдяки цьому навчитися основам мови можна за досить короткий час;

– легко читається: блоки коду в python виділяються за допомогою відступів, що спільно з ключовими словами, взятими з англійської мови значно полегшують читання коду;

– легкий в обслуговуванні: однією з причин широкої популярності python'а є простота обслуговування коду написаного на цій мові;

– широка стандартна бібліотека: наявність широкої крос-платформної бібліотеки є ще однією сильною стороною цієї мови програмування;

– портативність: Python без проблем запускається на різних платформах, при цьому зберігає однаковий інтерфейс, незалежно від того на якому комп'ютері ви працюєте;

– розширюваність: при необхідності в Python можна впроваджувати низькорівневі модулі написані на інших мовах програмування для найбільш гнучкого вирішення поставлених завдань;

– робота з базами даних: у стандартній бібліотеці python можна знайти модулі для роботи з більшістю комерційних баз даних;

створення gui (графічного інтерфейсу користувача): на python можливе створення gui додатків, які будуть працювати незалежно від типу вашої операційної системи;

велика кількість готових бібліотек та гарна підтримка іншими розробниками.

2.3.3 Вибір мови програмування

Після порівняльного аналізу мов програмування, були виявлені такі гідності та недоліки обох мов.

До основних гідностей Perl можна віднести:

багато готових бібліотек - модулів;

підтримка роботи з регулярними виразами;

проста обробка великих обсягів даних.

З недоліків Perl можна виділити, що у мові зберігаються раніше популярні, але зараз вже застарілі підходи.

До основних переваг Python можна віднести:

актуальний на наш час;

багато готових бібліотек;

багато готових прикладів;

актуальність у наш час;

дуже обширна підтримка серед розробників;

простота в освоєнні.

До недоліків мови програмування Python можна віднести те, що не найшвидша мова програмування.

Таким чином через популярність Python, гарну підтримку розробників ПО та швидкість в освоєні програмування та написання програм варто обрати саме цю мову для розробки програми для вирішення поставленого завдання.

3 РОЗРОБКА ТА ОПИС АЛГОРИТМУ РОБОТИ ПРОГРАМИ

3.1 Розробка алгоритму роботи програми

Оскільки задачею роботи є автоматизоване формування бази проектів, а проекти для формування бази знаходяться у веб сервісі EDA Playground, програма має отримати посилання на проекти, які треба скачувати. Вхідними даними для роботи програми, крім переданих параметрів програми, є файл microsoft exel, з якого беруться прізвище, варіант та посилання на проект у EDAplayground. У вхідному файлі прізвища, варіант та посилання розташовані в декілька рядків.

На рисунку 3.1 показано вікно входу в EDA Playground.

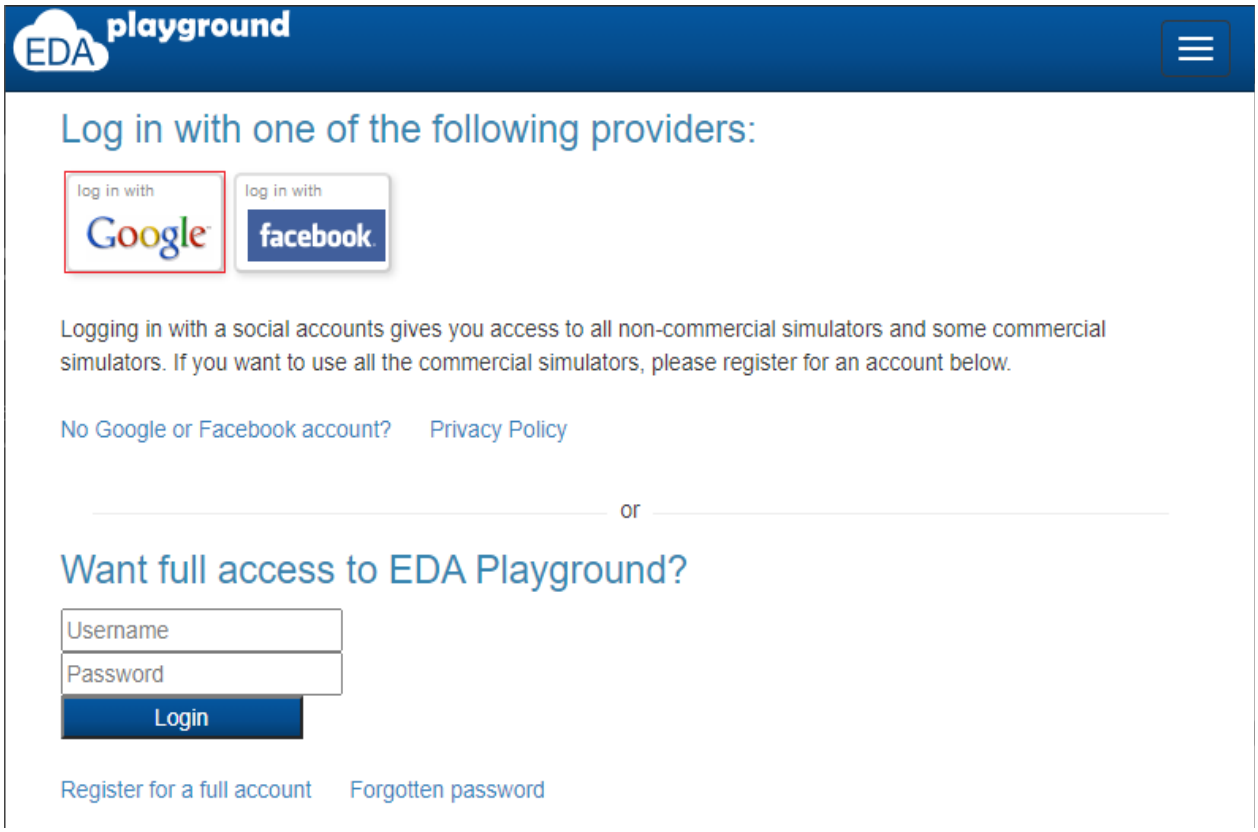


Рисунок 3.1 – Сторінка входу до EDA Playground

Далі за допомогою бібліотек відкривається браузер. Для того, щоб мати змогу запускати та скачувати проекти з EDA Playground, треба увійти під своїм логіном та паролем. Для цього спочатку завантажується головна сторінка сайту EDA Playground, далі автоматично обирається спосіб входу через систему google шляхом натискання на відповідна кнопка, яку обведено червоним на рисунку 3.1.

Далі у полі для логіну заноситься логін, переданий у якості одного з параметрів, після натисканні кнопки "далі"; цей етам можна побачити на рисунку 3.2.

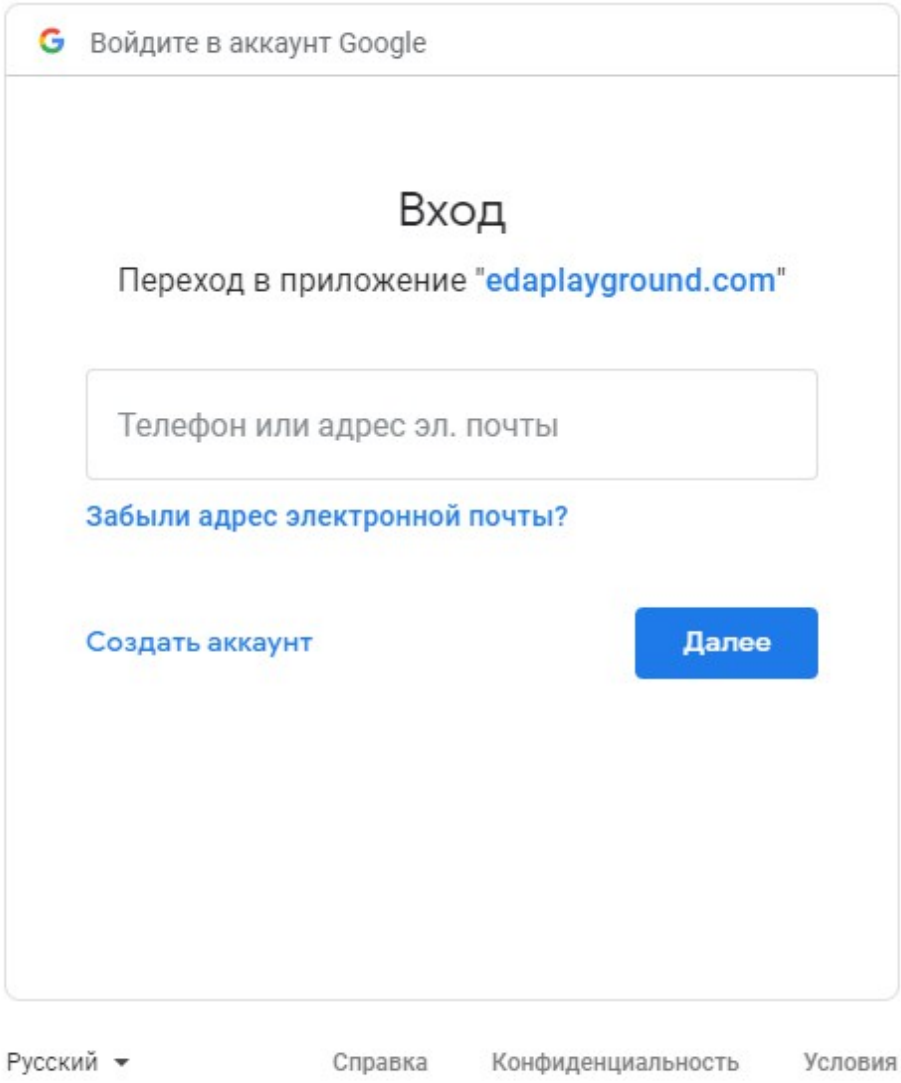


Рисунок 3.2 – Зображення віконця вводу логіна у сервісі Google

Після цього з’являється нове віконце,для введення паролю, яке можна побачити на рисунку 3.3. У поле для паролю заноситься пароль, також переданий у якості параметра та натискається кнопка кнопки "далі".

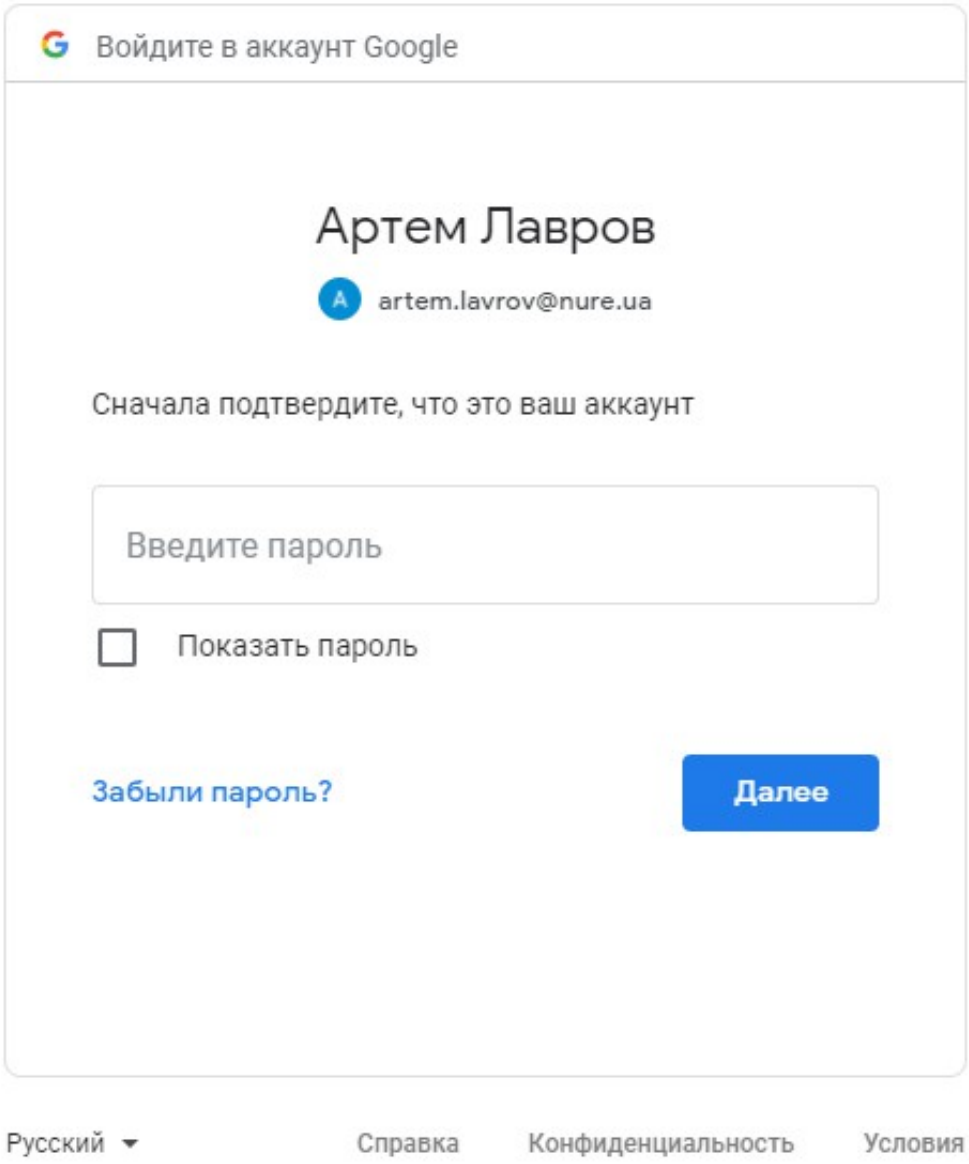


Рисунок 3.3 – Зображення віконця вводу пароля у сервісі Google

Далі програма переходить за кожним посиланням з вхідного файлу, далі, якщо чекбокс "Download after run" недоступний, буде натиснуте

"Tools$Simulators" для того, щоб чекбокс став доступним [16]. Після цього перевіряється чекбокс, та якщо він не відмічений, він відмічається, далі натискається кнопка "Run". На рисунку 3.4 зображений відкритий проект у EDA Playground, на цьому рисунку можна побачити чекбокс "Download after run" та кнопку "Run", які виділені червоним кольором.

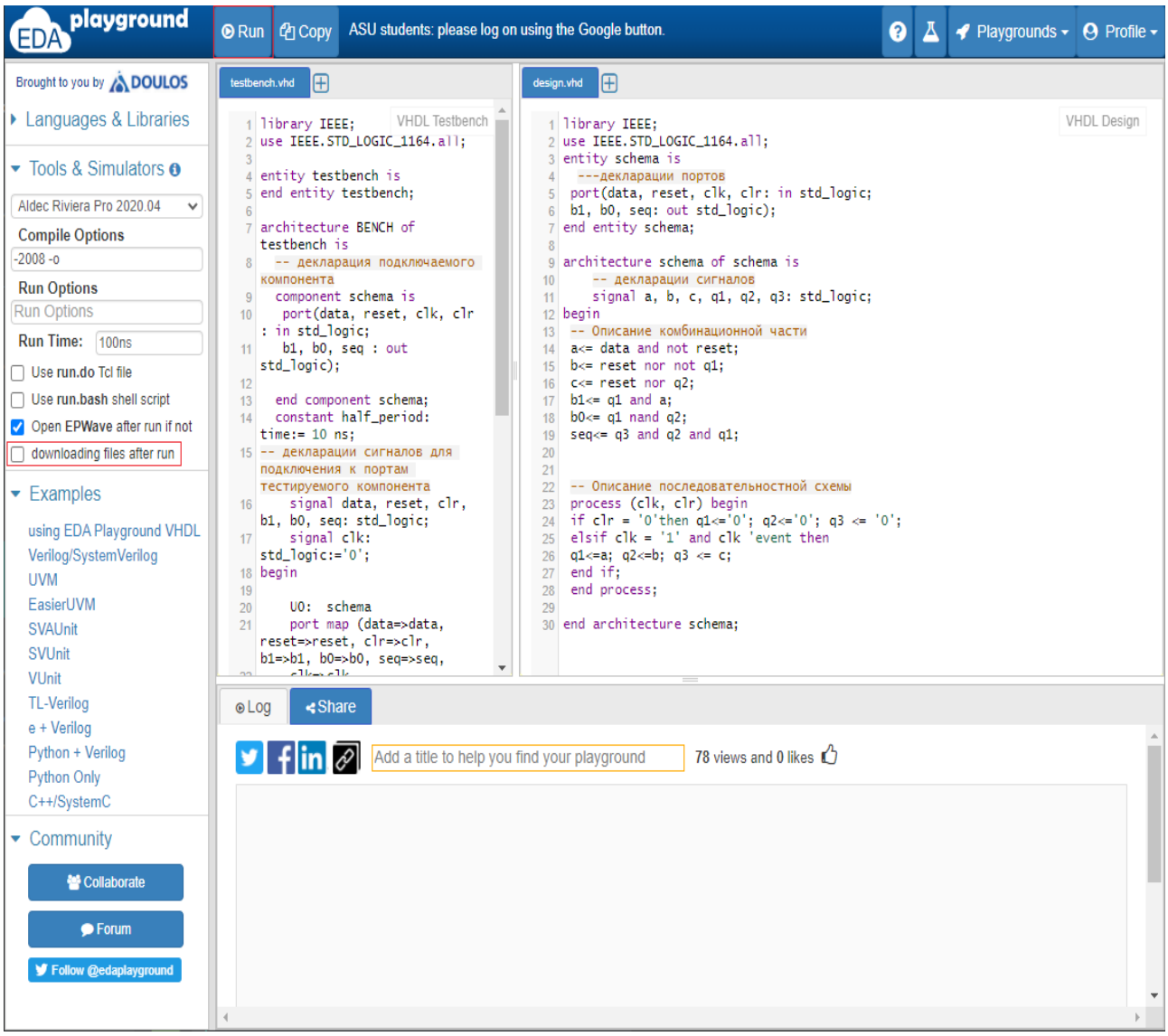


Рисунок 3.4 – Відкритий проект у EDA Playground

Схему-алгоритм роботи програми в цілому показано на рисунку 3.5.

Алгоритм скачування проекту з EDA Playground показано на рисунку 3.6. Після натискання на кнопку "Run" починається компіляція проекту та скачування архіву з VHDL проектом до заздалегідь заданого шляху, який передається у параметрах програми, та розпаковує зміст архіву у папку з назвою, яка складається з прізвища та номеру варіанта завдання.

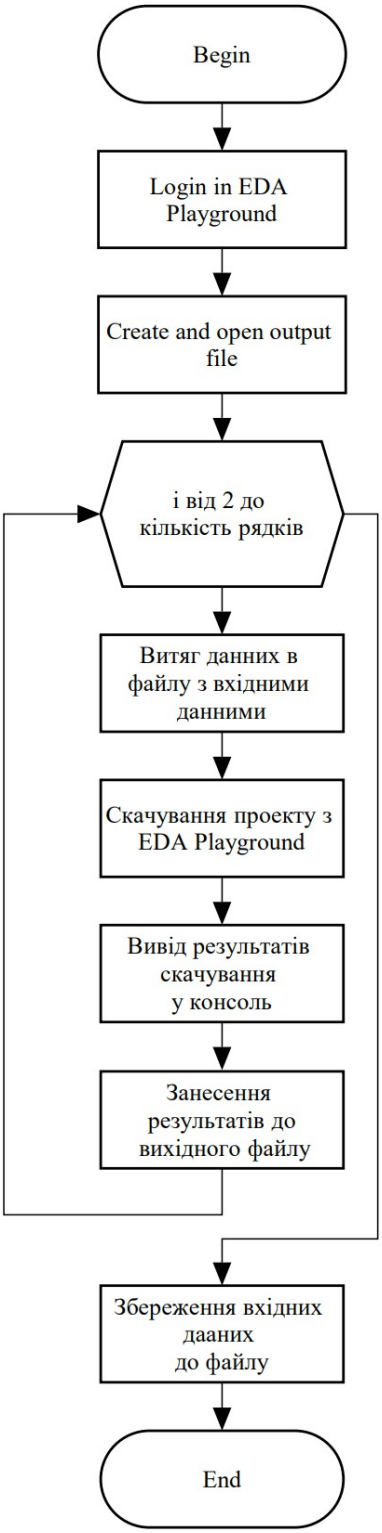


Рисунок 3.5 – Алгоритм роботи програми завантаження проектів

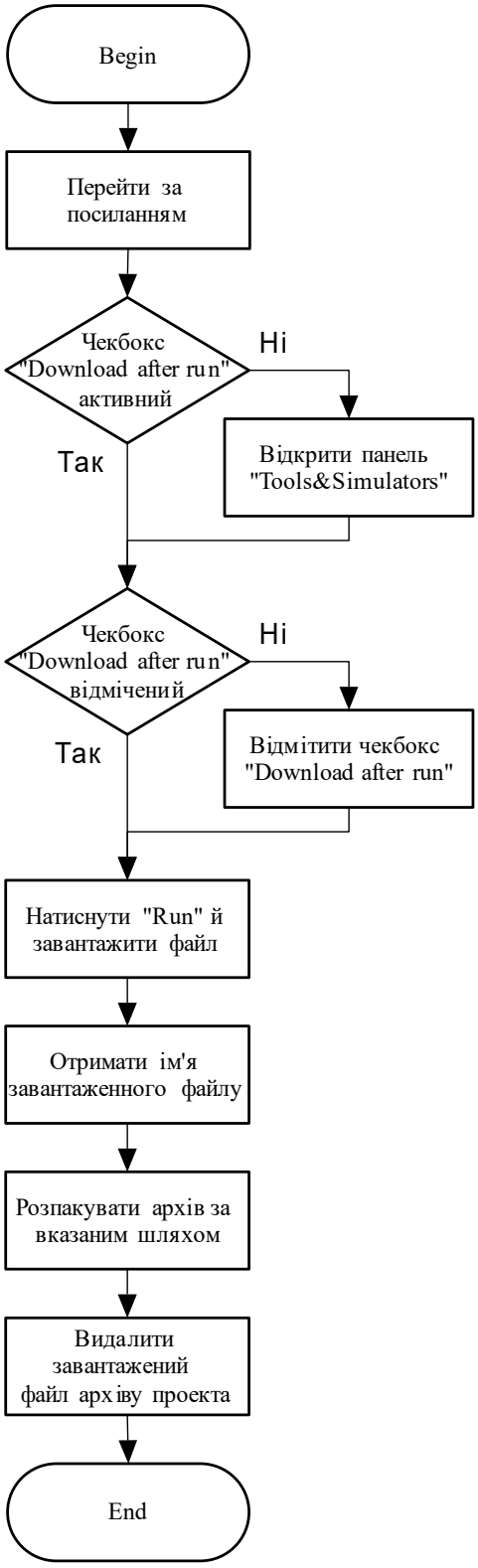


Рисунок 3.6 – Блок-схема алгоритму завантаження проекту з EDA Playground

Разом з цим у вихідний файл, назва якого співпадає з назвою вхідного але з доданням "_res" в кінець назви, заносяться такі данні:

– прізвище;

– номер варіанта;

– шлях до розпакованого проекту;

– результат завантаження.

Якщо компіляція проекту пройшла з помилками, тоді до поля результат завантаження буде занесений цей факт, інакше там буде записано «OK»

3.2 Опис розробленої програми

Програма складається з декількох модулів:

– основний модуль;

– модуль взаємодії з веб інтерфейсом;

– модуль перетворення символів кирилиці в символи латиниці.

При створенні вище перерахованих модулів був використаний різноманітний набір бібліотек і функцій для роботи з системними функціями, рядками, файлами, архівами, веб драйвером.

Основний модуль це частина програми, в якому зосереджена вся логіка роботи програми, виклики інших модулів. При розробці цього модуля були використані такі модулі та методи.

Системні функції використовуються для взаємодії з системою та системними об’єктами та виводу інформації у консоль.

Таблиця 3.1 – Системні функції

Назва функції	Опис функції
sys.argv	Це список аргументів командного рядка, які стосуються сценарію Python. Перший аргумент, argv[0], має аналогічне скрипту Python найменування.

	Залежно від використовуваної платформи, перший аргумент може містити повний шлях до скрипту або до

Продовження таблиці 3.1

Назва функції	Опис функції
	назви файлу. Системні аргументи всередині програми будуть представляти собою список рядків і числових значень аргументів, переданих в програму.
sys.exit	Ця функція дозволяє програмі завершити виконання примусово. Функція exit приймає необов'язковий аргумент, зазвичай ціле число, яке дає статус виходу. Нуль вважається як успішне завершення. Також слід звернути увагу на те, що при виклику exit, викликається виняток SystemExit, яке дозволяє функціям очищення очистити пам'ять.
isinstance (obj, classinfo)	Функція повертає True, якщо вказаний об'єкт є екземпляром зазначеного класу (класів), або успадковується від нього класу інакше False. obj – Об'єкт, що вимагає перевірки; classinfo – клас, або кортеж з класами, або рекурсивний кортеж кортежів. Інші типи послідовностей не підтримуються. Якщо аргумент не є класом, або кортежем з класами, збуджується TypeError.
print	print функція print Python виводить задані об'єкти на стандартний пристрій виведення (екран) або відправляє їх текстовим потоком у файл. Повний синтаксис функції : print (*objects, sep=' ', end=' \ n', file=sys.stdout, flush=False), де: objects – об'єкт, який потрібно вивести * позначає, що об'єктів може бути кілька; sep – розділяє об'єкти. Значення за замовчуванням:' '; end – ставиться після всіх об'єктів;

Продовження таблиці 3.1

Назва функції	Опис функції
	file – очікується об'єкт з методом write (string). Якщо значення не задано, для виведення об'єктів використовується файл sys.stdout; flush – якщо задано значення True, потік примусово скидається в файл, значення за замовчуванням – False.
time.sleep(sec)	Функція time.sleep дозволяє зупинити подальше виконання програми на вказане в параметрах (sec) час в секундах.

Функції класу str використовуються для роботи зі строками та форматування строк. Перелік функцій та їх опис можна побачити у таблиці 3.2.

Таблиця 3.2 – Функції роботи зі строками

Назва функції	Опис функції
str_src.replace (old, new[, count])	Функція str.replace(). Пошук і заміна підрядків у вихідному рядку. Функція str.replace () Повертає копію рядка, в який всі входження підрядка old замінені на рядок new. Загальна форма використання функції: str_res. str_res – результуюча рядок, в якій всі входження підрядка old замінені на рядок new; str_src – рядок-оригінал, в якій делаетс пошук і заміни підрядка old на рядок new; old – підрядок, який може бути

	замінений іншим підрядком new. Кількість символів в підрядку довільно. Якщо підрядок old не знайдено в рядку str_src, тоді функція повертає рядок str_src без змін;

Продовження таблиці 3.2

Назва функції	Опис функції
	new – підрядок, що замінює підрядок old в рядку str_src; count – кількість замін які можуть бути здійснені. Якщо count не задано, то заміни здійснюються у всіх можливих входженнях підрядка old в рядку str_src.
index = str.find (sub [, start [, end]])	str.find(). Пошук підрядка в рядку. Функція find() призначена для пошуку підрядка в рядку. Відповідно до документації Python загальна форма виклику функції наступна: index = str.find (sub [, start [, end]]), де index – цілочисельне значення, яке є індексом першого входження підрядка sub в рядку s. Якщо підрядка sub не знайдена в рядку s, то index=-1; str – рядок, в якому здійснюється пошук підрядка sub; start, end – позиції в рядку str. Ці позиції визначають межі зрізу s[start: end], що визначає оброблюваний пошуковий діапазон. Ці параметри є необов'язковими, і якщо не задавати параметри start, end, то пошук буде здійснюватися у всьому рядку.
Рядок f""	Рядок f"", також звані "форматованими рядковими літералами". F-рядки – це рядкові літерали, які мають “f” в початку і фігурні дужки, що містять вирази, які будуть замінені їх значеннями. Вирази оцінюються під час виконання, а потім форматуються. Для форматування даного типу рядків використовуються фігурні дужки, між яких поміщається змінна або вираз, значення якої

	необхідно помістити в тексті. Ці рядки виглядають більш читабельно, з огляду на те, що змінні та вирази розташовані всередині тексту, в який буде вставлені значення, а також такий вид рядків працює швидше.

Функції класу openpyxl використовується для роботи з файлами таблиць Microsoft excel. Використані функіці та опис функції можна знайти у таблиці 3.3.

Таблиця 3.3 – Функції класу openpyxl

Назва функції	Опис функції
load_workbook	Функція load_workbook призначена для відкриття документа. Загальна форма виклику функції наступна: openpyxl.load_workbook(filename [, read_only=False] [, keep_vba=KEEP_VBA] [, data_only=False] [, keep_links=True]) filename шлях до файлу, який необхідно відкрити, або файловий об'єкт. read_only – оптимізовано для читання, контент не можна редагувати. За замовчуванням False. keep_vba – зберегти вміст vba (це не означає, що його можна використовувати). data_only – визначає, чи містять комірки з формулами формулу (за замовчуванням) або значення, збережене при останньому читанні

	аркуша Excel. keep_links – чи слід зберігати посилання на зовнішні книги. За замовчуванням True.
active	Функція active призначена для доступу до поточного аркуша поточного документа. Загальна форма виклику функції наступна: openpyxl.active(), функція не має параметрів.
cell	Функція cell призначена для доступу до осередків поточного документа. Загальна форма виклику функції наступна: openpyxl.cell (row, column, [value]).

Продовження таблиці 3.3

Назва функції	Опис функції
	row номер рядка, до якого необхідно звернутися. column номер стовпця, до якого необхідно звернутися. value значення, яке необхідно записати в обрану комірку, необов'язковий параметр.
delete_cols	Функція delete_cols призначена для видалення колонок з документа. Загальна форма виклику функції наступна: openpyxl.delete_cols (idx, amount=1). idx номер колонки, починаючи з якої потрібно проводити видалення. amount це кількість, яку потрібно видалити починаючи з idx, якщо не задавати, буде видалена одна колонка.
save	Функція save служить для збереження раніше відкритого файлу. Є два варіанти використання функції, це зберегти і зберегти як для існуючих файлів. Загальна форма

	виклику функції наступна: openpyxl.save (path), де path це шлях по якому буде збережений файл. Для збереження файлу в функцію необхідно передати шлях, який був використаний при відкритті файлу, а для збереження як інший файл, необхідно передати в функцію шлях до файлу, куди потрібно зберегти файл.
close	Функція close служить для закриття раніше відкритого файлу, застосовується до об'єкта файлу, який відкритий. Функція не має параметрів. Після закриття файлу неможливо отримати до нього доступ через об'єкт файлу, поки файл не буде знову відкритий. Також важливо відзначити, що функція не зберігає вміст файлу.

Для взаємодії з веб інтерфейсом EDA Playground було обрано бібліотеку Selenium.

Selenium WebDriver – це набір бібліотек для різних мов програмування. Ці бібліотеки використовуються для відправки команд веб драйверу браузера, звідси і назва WebDriver. Відправка команд проводиться за допомогою протоколу JsonWireProtocol або за допомогою спеціальних API, які у кожного браузера свої. В командах зазначено дію, яку повинен зробити браузер в рамках поточної сесії. Прикладами таких команд можуть бути команди знаходження елементів по локатору, перехід за посиланнями, парсинг тексту сторінки/елемента, натискання кнопок або перехід по посиланнях на сторінці веб-сайту. Проектом Selenium і співтовариством підтримується робота з браузерами Microsoft Internet Explorer, Google Chrome, Mozilla Suite і Mozilla Firefox. Функції класу Selenium WebDriver використовується для роботи з веб браузером. Використані функції та опис функції можна знайти у таблиці 3.4.

Таблиця 3.4 – Функції класу Selenium WebDriver

Назва функції	Опис функції
webdriver.Chrome	Функція webdriver.Chrome служить для створення об'єкта веб драйвера. Загальна форма виклику функції наступна: webdriver.Chrome (self, executable_path= "chromedriver", port=0, options=None, service_args=None, desired_capabilities=None, service_log_path=None, chrome_options=None, keep_alive=True), де: executable_path – шлях до виконуваного файлу веб драйвера; Port-порт, на якому повинна запускатися

Продовження таблиці 3.4

Назва функції	Опис функції
	служба, якщо залишити 0, буде знайдений вільний порт; options-для цього потрібен екземпляр ChromeOptions; service_args – список аргументів для передачі в службу драйвера; desired_capabilities – об'єкт словника, не пов'язаний з браузером; тільки можливості, такі як» проксі " або " loggingPref»; service_log_path – куди записувати інформацію від драйвера; chrome_options – застарілий аргумент для параметрів; keep_alive – чи слід налаштувати

	ChromeRemoteConnection на використання перевірки активності HTTP.
get	Функція get служить для завантаження сторінки. Загальна форма виклику функції наступна: get (url), де url це посилання, на сторінку, яку потрібно завантажити.
find_element_by_id	Функція find_element_by_id служить для пошуку елемента на сторінці по заданнаму в праметрах id. Загальна форма виклику функції така: find_element_by_id (id), повертає елемент, якщо він знайдений, інакше викликає виняток NoSuchElementException.

Продовження таблиці 3.4

Назва функції	Опис функції
find_element_by_css_selector	Функція find_element_by_css_selector служить для пошуку елемента на сторінці по заданнаму в праметрах css_selector. Загальна форма виклику функції така: find_element_by_css_selector (css_selector), повертає елемент, якщо він знайдений, інакше викликає виняток NoSuchElementException.
get_attribute	Функція get_attribute спочатку спробує повернути значення властивості з заданим ім'ям. Якщо властивості з таким ім'ям не існує, воно повертає значення атрибута з тим же ім'ям. Якщо атрибута з таким ім'ям немає, повертається None. Загальна форма виклику функції наступна: get_attribute (name), name Назва властивості або атрибута.
is_displayed	Функція is_displayed повертає стан елемента, відображається він на екрані, чи ні. Загальна

	форма виклику функції така: is_displayed (), повертає True, якщо елемент відображається, інакше False.

Selenium WebDriver в основному використовується для автоматизації дій в браузері, імітації дій людини при роботі з браузером.

Функції класу os використовується для роботи з файлами, дозволяє відкривати, читати, записувати файл, а також різноманітні операції з каталогами. Функції класу os використовується для роботи з різними елементам через шар операційної системи. Використані функції та опис функції можна знайти у таблиці 3.5.

Таблиця 3.5 – Функції класу os

Назва функції	Опис функції
listdir	Функція listdir служить для отримання списку файлів. Загальна форма виклику функції наступна: os.listdir(path), де path це шлях до папки, список файлів потрібно отримати.
remove	Функція remove служить для вудлення файлу. Загальна форма виклику функції наступна: os.remove(path), де path це шлях до файлу, який необхідно видалити.

Функції класу zipfile використовується для роботи з файлами архівів, дозволяє відкривати, читати, записувати, розпаковувати файли архівів. Використані функції та опис функції можна знайти у таблиці 3.6.

Таблиця 3.6 – Функції класу zipfile

Назва функції	Опис функції
ZipFile	Функція ZipFile служить для відкриття файлу архіву. Загальна форма виклику функції наступна: ZipFile (self,

	file, mode= "r", compression=ZIP_STORED, allowZip64=True, compresslevel=None,*, strict_timestamps=True). параметр режиму відкриття архіву, повинен бути 'r'для читання існуючого файлу,' w'для усічення і запису нового файлу ,'a'для додавання до існуючого файлу або' x'для монопольного створення і запису нового файлу. За замовчуванням режим встановлений як 'r'; стиснення є методом Zip стиснення, може бути

Продовження таблиці 3.6

Назва функції	Опис функції
	ZIP_STORED, ZIP_DEFLATED, Zip_bzip2або ZIP_LZMA; якщо розмір Zip-файлу перевищує 4 ГиБ, то у файлу встановлюється прапор ZIP64, а параметр allowZip64 встановлений в True (за замовчуванням), буде дозволена робота з файлами з прапором ZIP64; параметр compresslevel управляє рівнем стиснення, використовуваним при записі файлів в архів; strict_timestamps аргумент, якщо задано значення False, дозволяє файли старше 1980-01-01.
extractall	Функція extractall служить для вилучення вмісту архіву. Загальна форма виклику функції наступна: ZipFile.extractall (path = None , members = None , pwd = None), де: path вказує інший каталог для вилучення; members не є обов'язковим і має бути підмножиною списку, що повертається namelist(); pwd – це пароль, який використовується для зашифрованих файлів.

Модуль перетворення символів кирилиці в символи латиниці служить для видалення можливих проблемних символів, а саме символи кирилиці і прогалини. Ці символи можуть перешкодити коректної роботі з шляхами, по яких будуть завантажуватися проекти. Цей модуль використовує словник для переводу кирилиці у латиницю.

Модуль складається з однієї функції latinizator(letter), яка у якості параметрів приймає строку, яку треба перетворити, та повертає вже перетворену строку.

3.3 Опис роботи програми

Так як програма написана на мові програмування Python, то для запуска програми необхідні вихідні файли програми або скомпільований виконуваний файл для операційної системи, в якій буде проводитися запуск програми. Також крім самої програми необхідний файл драйвера для браузера. Програма має консольний інтерфейс, тому перед запуском самої програми необхідно запустити консоль і перейти в папку, в якій міститься Програми, або, додати шлях до папки в оточення середовища для того, щоб запуск програми відбувався з будь-якого шляху командного рядка.

Після підготовки командного рядка до запуска програми, необхідно ввести назву програми в консоль і вказати необхідні параметри, такі, як:

- логін;
- пароль;
- шлях до WebDriver;
- шлях до папки, куди необхідно завантажити проекти;
- шлях до Excel файлу, в якому знаходяться прізвища та посилання на пректи;
- visible.

Параметр Login відповідає за адресу електронної пошти, з-під яким буде проведений вхід в систему, використовуючи авторизую через Google сервіси.

Параметр Password відповідає за пароль від аккаунта для входу в EDA Playground.

Параметр шлях до WebDriver це шлях до виконуваного файлу драйвера браузера Chrome. WebDriver – це інструмент з відкритим вихідним кодом для автоматичного тестування веб-додатків у багатьох браузерах. Він надає можливості для переходу на веб-сторінки, Введення даних Користувачем, виконання JavaScript і багато чого іншого. ChromeDriver – це автономний сервер, що реалізує стандарт W3C WebDriver. ChromeDriver доступний для Chrome на Android і Chrome на ПК (Mac, Linux, Windows і ChromeOS).

Шлях до папки, куди необхідно завантажити проекти це папка в яку при роботі програми будуть завантажені архіми проектів і розпаковані в підпапки, імена яких будуть сформовані з комбінації прізвища, яка, якщо це необхідно, буде перетворена латинськими символами, і номера варіанту, зазначеного у вхідному файлі.

Шлях до Excel файлу, в якому знаходяться прізвища та посилання на проекти це файл, в якому знаходяться стовпці:

- прізвище студента, якому належить робота;
- номер варіанту, який був виданий студенту викладачем при отриманні завдання;
- посилання на проект в EDA Playground, по якій знаходиться проект, зроблений студентом.

На рисунку 3.7 можна побачити, як виглядає вхідний файл.

	A	B	C
1	Surname	URI	Variant
2	Петров	https://www.edaplayground.com/x/UFGs	1
3	Иванов	https://www.edaplayground.com/x/bV3w	2
4	Крюков	https://www.edaplayground.com/x/rJdv	3
5	Сидоров Иван	https://www.edaplayground.com/x/QXYS	4
6	Сидоров Василий	https://www.edaplayground.com/x/TyvR	5
7	Пеньков	https://www.edaplayground.com/x/bkTt	6
8	Дроздов	https://www.edaplayground.com/x/SMtp	7
9	Галкин	https://www.edaplayground.com/x/9eFp	8
10	Голубев	https://www.edaplayground.com/x/kGAx	9
11	Воронова	https://www.edaplayground.com/x/PzQ_	10
12	Синицин	https://www.edaplayground.com/x/dkHb	11
13	Попугаев	https://www.edaplayground.com/x/udYU	12
14	Скворцов	https://www.edaplayground.com/x/6hWP	13
15	Воробьёв	https://www.edaplayground.com/x/cZbh	14
16	Дятлов	http://edaplayground.com/x/h48c	15
17	Чижова	https://www.edaplayground.com/x/BdJf	16
18			

Рисунок 3.7 – Приклад вигляду вхідного файлу

Параметр visible відповідає за те, чи буде вікно браузера мабуть у процесі виконання чи ні. Якщо цей параметр заданий як "visible", тоді вікно браузера буде видно, а якщо цей параметр заданий чимось іншим або зовсім не заданий, тоді вікно браузера сховається після входу в EDA Playground.

Після введення всіх необхідних параметрів, наприклад як показано на рисунку 3.8, і запуску програми на виконання шляхом натискання "enter" відкриється вікно браузера Google Chrome.

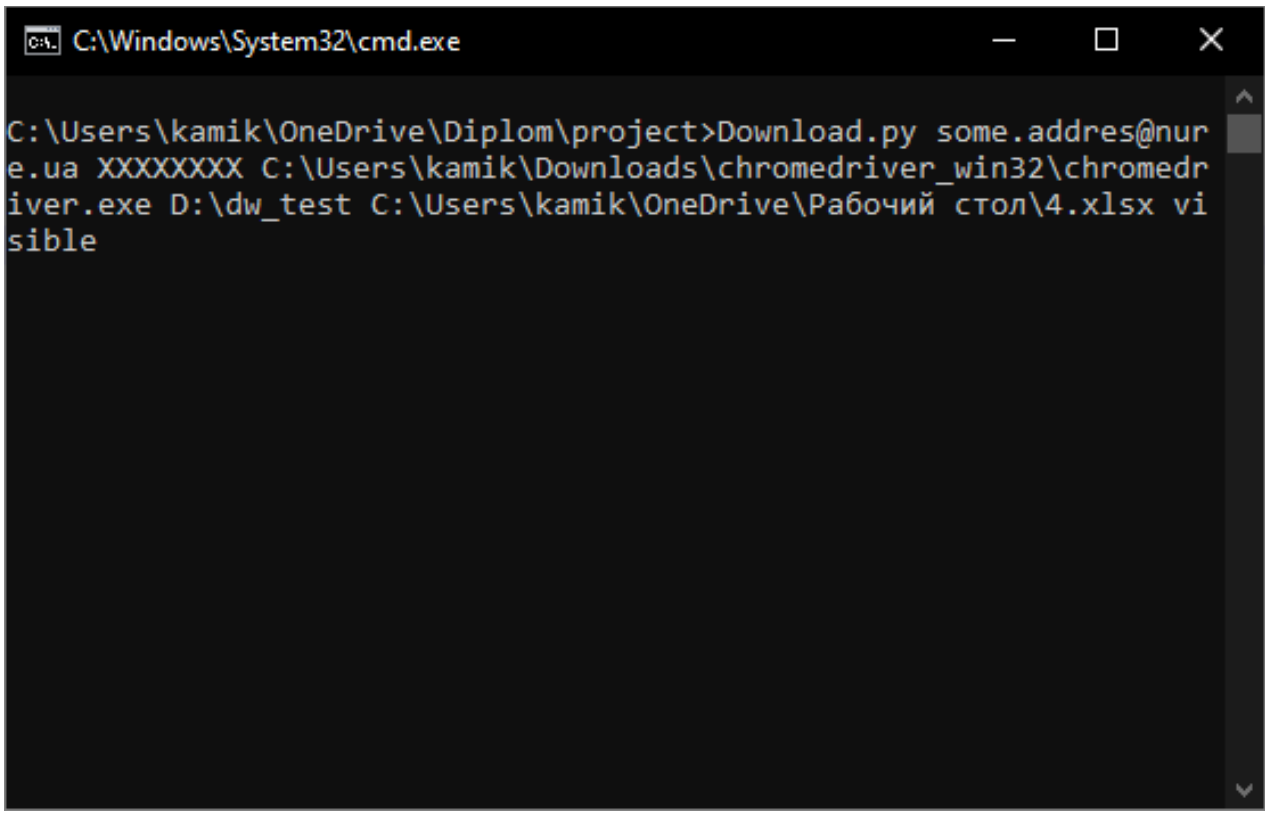


Рисунок 3.8 – Приклад вигляду запуску програми з усіма параметрами у консолі

Після запуску відкриється вікно браузера, як показано на рисунку 3.9.

Далі необхідно дочекатися, поки програма зробить авторизацію в EDA Playground через сервіси Google, а саме введе зазначений в параметрах логін в поле для введення, як на рисунку 3.10.

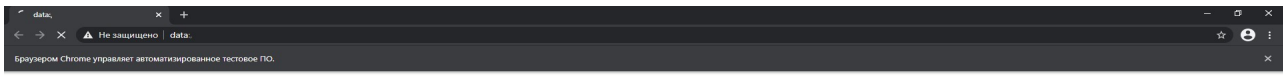


Рисунок 3.9 – Вікно браузера, відкрите програмою

Потім програма введе пароль, як на рисунку 3.11, після того, як це зроблено можна переходити до потрібної в даний момент програмі.

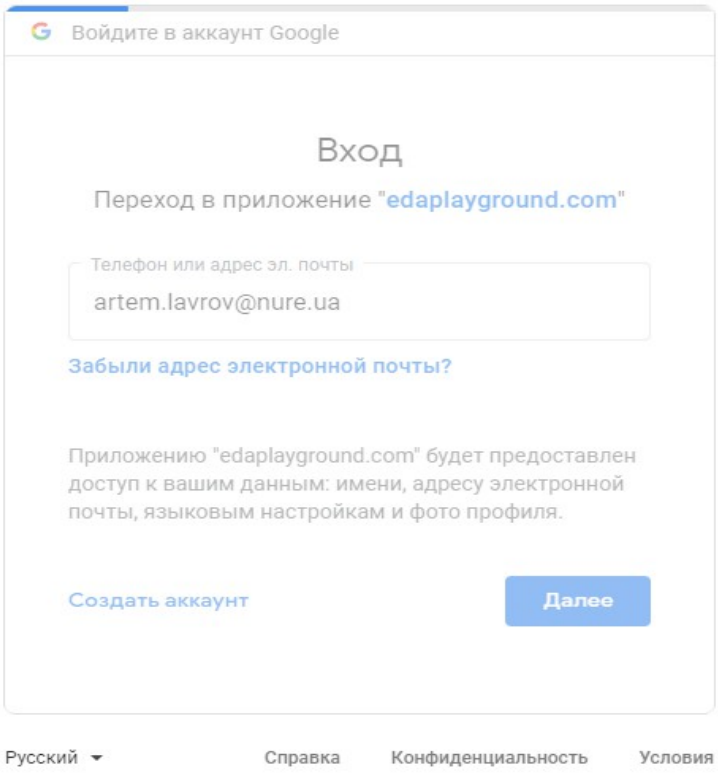


Рисунок 3.10 – Форма для вводу адреси електронної пошти

Це необхідно оскільки, як з'ясувалося, система входу Google перевіряє, чи відображається в момент введення елемент на екрані, чи ні. Якщо елемент не відображається або вікно згорнуто, тоді увійти використовуючи Google не вийде.

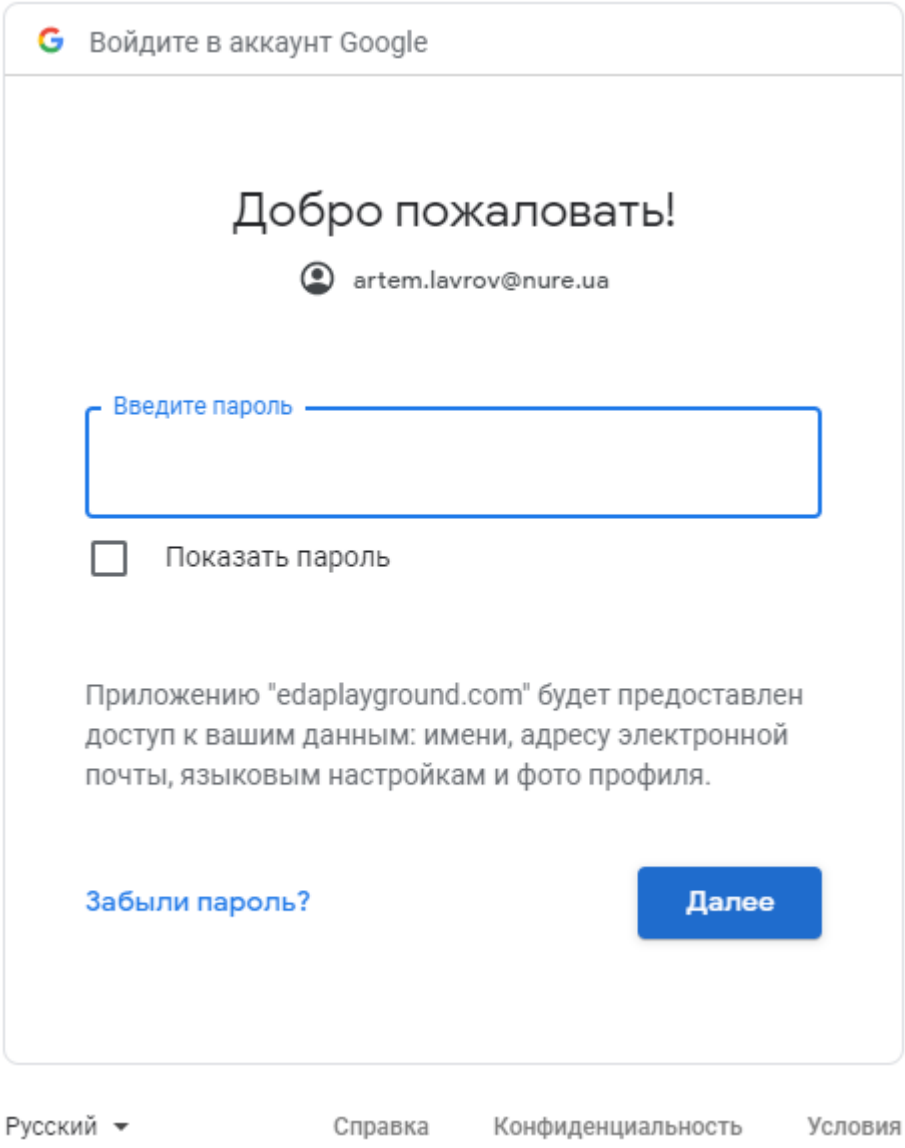


Рисунок 3.11 – Форма для вводу пароля від облікового запису Google

Якщо останній з параметрів був заданий не як "visible" або і зовсім не був заданий, тоді після входу в EDA Playground браузер перемістится за

зону видимості екрану, і буде продовжувати працювати там. Під час виконання Програми Користувач може продовжувати працювати зі сторонніми програмами на комп'ютері, при цьому виконання програми ніяк не позначиться на користуванні комп'ютером.

При скачуванні кожного проекту в вихідний файл і в консоль виводяться дані про результати скачування проектів. На рисунку 3.12 можна побачити, як виглядає консоль після виконання скачування всіх проектів зі списку.

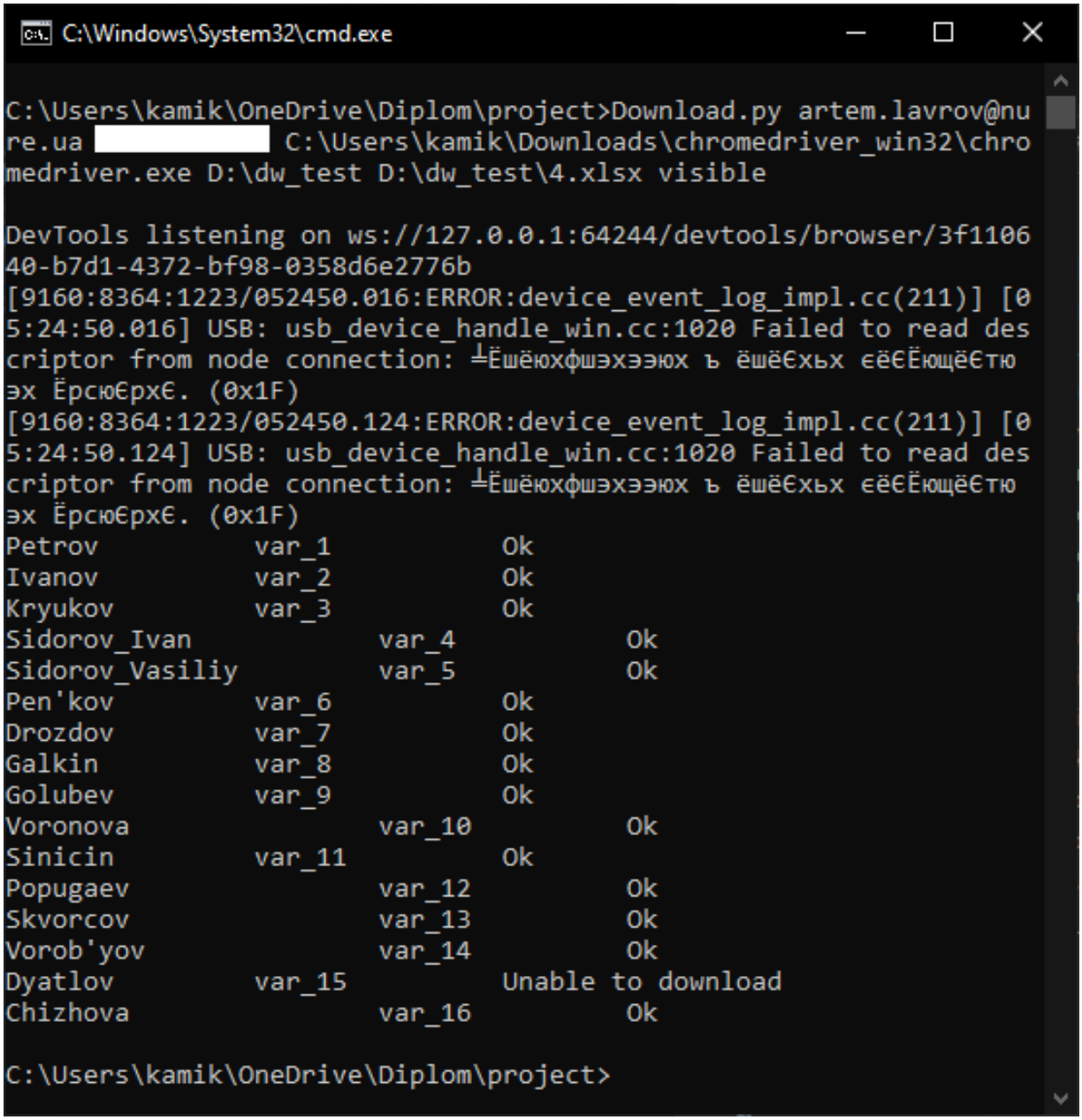


Рисунок 3.12 – Вигляд консолі після скачування всіх проектів

Після завершення скачування всіх проектів зі списку, програма закриє вікно браузера, закриє і збереже всі використовувані файли і завершить своє виконання.

На рисунку 3.13 можна побачити, як виглядає зміст вихідного файлу, у вигляді таблиці, після скачування всіх проектів за посиланнями зі списку вхідного файлу.

	A	B	C	D
1	Surename	Variant	Path	Result
2	Petrov	1	D:\dw_test\Petrov_var1	Ok
3	Ivanov	2	D:\dw_test\Ivanov_var2	Ok
4	Kryukov	3	D:\dw_test\Kryukov_var3	Ok
5	Sidorov_Ivan	4	D:\dw_test\Sidorov_Ivan_var4	Ok
6	Sidorov_Vasiliy	5	D:\dw_test\Sidorov_Vasiliy_var5	Ok
7	Pen'kov	6	D:\dw_test\Pen'kov_var6	Ok
8	Drozdov	7	D:\dw_test\Drozdov_var7	Ok
9	Galkin	8	D:\dw_test\Galkin_var8	Ok
10	Golubev	9	D:\dw_test\Golubev_var9	Ok
11	Voronova	10	D:\dw_test\Voronova_var10	Ok
12	Sinicin	11	D:\dw_test\Sinicin_var11	Ok
13	Popugaev	12	D:\dw_test\Popugaev_var12	Ok
14	Skvorcov	13	D:\dw_test\Skvorcov_var13	Ok
15	Vorob'yov	14	D:\dw_test\Vorob'yov_var14	Ok
16	Dyatlov	15		Unable to download
17	Chizhova	16	D:\dw_test\Chizhova_var16	Ok
18				

Рисунок 3.13 – Вміст вихідної інформації у вигляді таблиці

В обох випадках серед успішно скачаних проектів з результатом «Ok» є один, зі статусом «». Це означає, що проект не вдалося завантажити з якоїсь причини. В такому випадку спробу можна повторити окремо з цим проектом та ввімкнути режим видимості браузера, якщо раніше вона була вимкнена.

Це надасть змогу власноруч побачити в чому проблема при скачуванні проекта.

Крім інформації виведеної в консоль, в файл також записується шлях до папки, в яку був викачаний проект.

На рисунку 3.14 показаний вміст каталогу, який був вказаний в якості одного з параметрів як папка для скачування проектів.

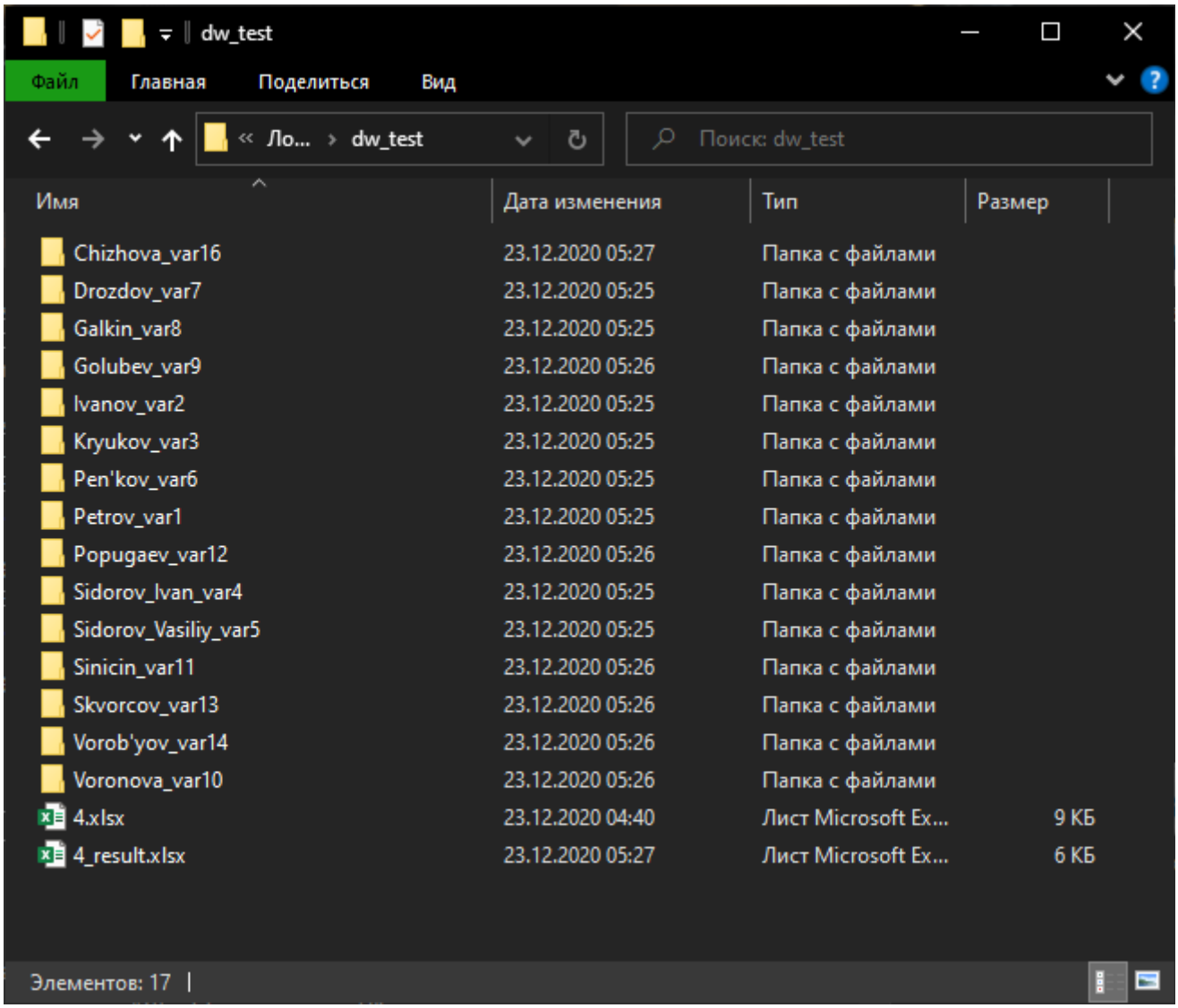


Рисунок 3.14 – Вміст каталогу, у який проводилося скачування проектів

ВИСНОВКИ

В результаті виконання атестаційної роботи проекту у першому розділі було оглянуто можливі галузі застосування, відзначена корисність розробки, оглянута система EDA Playground, зроблено огляд існуючих програмних продуктів для автоматизованої роботи з сайтами та зроблена постановка задачі.

Другий розділ містить огляд та вибір методів, засобів для реалізації системи. Так у другому розділі розглянуто та обрано спосіб взаємодії з веб сервісом, способи зберігання бази проектів, вибір мови програмування для реалізації системи автоматизованого формування бази проектів.

У третьому розділі було розроблено алгоритм роботи системи, опис бібліотек, які використовуються і їх функціоналу та опис роботи програми.

За поставленими вимогами та проведеними дослідженнями була розроблена системи автоматизованого формування бази проектів, яка дозволяє скачувати проекти з EDA Playground, що дозволяє економити час викладача. Система може бути налаштована на інші сервіси, для скачування з них проектів.

Також розроблена система має такі переваги:

зменшення часу на формування бази проектів;

підвищення швидкості та точності обробки проектів завдяки автоматизації рутинних дій, які доводилось робити до цього власноруч, що призводило до втомленості вже на початку.

Систему можна поділити на три етапи. Перший етап – це вхід до онлайн сервісу, другий – скачування проекту, третій – занесення результатів до вихідної таблиці.

Розроблена система задовольняє всім поставленим вимогам технічного завдання і готова до застосування на практиці.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Лавров А.А., Лопатина А.А., Хаханова И.В., Разработка системы автоматизации тестирования большого объема проектов [Текст] // XIII Всеукраїнська науково-практична WEB конференція аспірантів, студентів та молодих вчених «Комп'ютерні інтелектуальні системи та мережі», 24 – 26 березня 2020 р. Харків: 2020. – С. 32-35

2. Белова Е.В. Автоматизация организационно-управленческой деятельности как один из компонентов создания информационно-образовательной среды учебного заведения // Вестник МГПУ. Серия: Информатика и информатизация образования. – 2007. – № 9. – С. 152-155.

3. Архипова Е.Н., Кононова О.В., Крюков В.В., Шахгельдян К.И. Автоматизация рейтинговой оценки деятельности преподавателей // Университетское управление: практика и анализ. – 2010. – № 5. – С. 51-62.

4. Белова Е.В., Яникова З.М., Хожаева Т.С. Автоматизация организационно-управленческой деятельности на разных уровнях системы общего образования // Вестник МГПУ. Серия: Информатика и информатизация образования. – 2008. – № 15. – С. 22-24.

5. Бова В.В. Процессо-ориентированный подход к автоматизации деятельности образовательных учреждений // Перспективные информационные технологии и интеллектуальные системы. – 2007. – № 1. – С. 50-55.

6. Денисова А.Б. Средства автоматизации организационно-управленческого процесса внеучебной деятельности // Вестник РУДН. Серия: Информатизация образования. – 2013. – № 1. – С. 126-132.

7. Malavida [Електронний ресурс] – Режим доступу: https://www.malavida.com/ru/soft/winhttrack/#gref

8. INFOSTART [Електронний ресурс] – Режим доступу: https://infostart.ru/public/360421/

9. INFOSTART [Електронний ресурс] – Режим доступу: https://vistatask.en.softonic.com/

10. LWP - The World-Wide Web library for Perl [Електронний ресурс] – Режим доступу: https://metacpan.org/pod/LWP

11. AnyEvent::HTTP [Електронний ресурс] – Режим доступу: https://metacpan.org/pod/AnyEvent::HTTP

11. Selenium with Python [Електронний ресурс] – Режим доступу: https://selenium-python.readthedocs.io/

13. Свейгарт, Эл. Автоматизация рутинных задач с помощью Python: практическое руководство для начинающих. – М.: ООО «И.Д. Вильямс», 2017. – 592с.

14. WWW::Mechanize::Chrome [Електронний ресурс] – Режим доступу: https://metacpan.org/pod/WWW::Mechanize::Chrome

15. Win32::GuiTest [Електронний ресурс] – Режим доступу: https://metacpan.org/pod/Win32::GuiTest

16. Лавров А.А., Лопатина А.А., Хаханова И.В., Разработка автоматизированной системы тестирования большого объема проектов [Текст] // XXIV Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI столітті», 7 – 9 квітня 2020 р. Харків: 2020. – С. 40-41