

UNIVERSITATEA POLITEHNICA BUCUREŞTI  
FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE  
DEPARTAMENTUL CALCULATOARE



## PROIECT DE DIPLOMĂ

Instrumente de securizare a informațiilor private pentru pentesting

Florian-Luis Micu

**Coordonator științific:**  
Conf. dr. ing. Daniel Rosner

BUCUREŞTI

2023

POLITEHNICA UNIVERSITY OF BUCHAREST  
FACULTY OF AUTOMATIC CONTROL AND COMPUTERS  
COMPUTER SCIENCE DEPARTMENT



## DIPLOMA PROJECT

Intelligence tools for pentesting

Florian-Luis Micu

**Thesis supervisor:**  
Conf. dr. ing. Daniel Rosner

BUCHAREST

2023

# CUPRINS

Sinopsis .....	5
Abstract.....	5
Mulțumiri .....	6
1    Introducere .....	7
1.1    Context .....	7
1.2    Problema .....	7
1.3    Obiective .....	8
1.4    Structura lucrării.....	9
2    Studiu de piață / Abordări existente.....	10
2.1    Analiză formular .....	10
2.2    Analiză soluții existente.....	16
2.2.1    Metasploit.....	17
2.2.2    Recon-ng .....	19
3    Analiza și specificarea cerințelor.....	22
3.1    Autentificare.....	22
3.2    Explorarea script-urilor de pe GitHub .....	22
3.3    Script-uri.....	23
3.4    Scenarii .....	24
3.5    Import/Export fișiere de configurare .....	25
3.6    Sincronizarea fișierelor de configurație și a rezultatelor cu serviciul cloud și actualizarea aplicației.....	25
4    Soluția propusă .....	27
4.1    Multi-platform.....	28
4.2    Front-End.....	29
4.3    Back-End.....	31
4.4    Servicii cloud.....	31
4.5    Actualizări.....	32
4.6    Împachetarea executabilelor .....	32
5    Detalii de implementare .....	33
5.1    Autentificare.....	33
5.2    Dashboard .....	35

5.3	Home .....	35
5.4	Import Scripts .....	37
5.5	Explore Scripts/Explore Scenarios.....	39
5.6	Scripts Status/Scenario Status .....	41
5.7	Create Scenario .....	43
5.8	Import/Export.....	45
5.9	Setări .....	47
5.10	About.....	49
5.11	Script-uri preconfigurate.....	49
5.12	Scenarii preconfigurate .....	54
6	Studiu de caz / Evaluarea rezultatelor.....	56
7	Concluzii .....	62
7.1	Dezvoltări ulterioare .....	62
8	Bibliografie .....	64

## **LISTĂ DE FIGURI**

Figura 2.1 Câteva categorii de unelte OSINT conform <i>OSINT Framework</i> .....	10
Figura 2.2 Întrebarea 1 – analiză chestionar .....	11
Figura 2.3 Întrebarea 2 – analiză chestionar .....	11
Figura 2.4 Întrebarea 3 – analiză chestionar .....	12
Figura 2.5 Întrebarea 4 – analiză chestionar .....	12
Figura 2.6 Întrebarea 5 – analiză chestionar .....	13
Figura 2.7 Întrebarea 6 – analiză chestionar .....	14
Figura 2.8 Întrebarea 7 – analiză chestionar .....	14
Figura 2.9 Întrebarea 8 – analiză chestionar .....	15
Figura 2.10 Clasament instrumente OSINT.....	16
Figura 2.11 Căutarea unui script în msfconsole.....	17
Figura 2.12 Configurarea și rularea unui script în Recon-ng din CLI.....	20
Figura 2.13 Vizualizarea datelor într-un tabel în Recon-ng prin GUI.....	20
Figura 3.1 Diagrama Use Case pentru autentificare .....	22
Figura 3.2 Diagrama Use Case pentru explorarea script-urilor de pe GitHub.....	23
Figura 3.3 Diagrama Use Case pentru script-uri .....	24
Figura 3.4 Diagrama Use Case pentru scenarii .....	25
Figura 3.5 Diagrama Use Case pentru import sau export.....	25
Figura 3.6 Diagrama Use Case pentru sincronizarea fișierelor cu serviciul cloud și actualizare a aplicației.....	26
Figura 4.1 Arhitectura generală a aplicației Valiant .....	27
Figura 4.2 Numărul total de request-uri procesate de diferite soluții back-end[16] .....	28
Figura 4.3 Arhitectura Electron.....	28
Figura 4.4 Clasament tehnologii web preferate 2023 (bullets - desired and admired) .....	30
Figura 4.5 Comportamentul unui DOM virtual în React .....	30
Figura 5.1 Folosirea modului React Router pentru rutarea cererilor HTTP .....	33
Figura 5.2 Componenta <i>PrivateRoute</i> care securizează rutele neautentificate .....	33
Figura 5.3 Pagina de logare.....	34
Figura 5.4 Pagina de înregistrare a unui utilizator.....	34
Figura 5.5 Pagina <i>Home</i> cu filtre pentru prima secțiune de recomandări .....	36
Figura 5.6 Pagina <i>Home</i> cu filtre pentru a doua secțiune de recomandări .....	36
Figura 5.7 Mecanismul de actualizare al script-urilor de GitHub recomandate .....	37
Figura 5.8 Primul pas de importare al unui script .....	37
Figura 5.9 Al doilea pas de importare al unui script.....	38
Figura 5.10 Al treilea pas de importare al unui script .....	38
Figura 5.11 Al patrulea pas de importare al unui script .....	39
Figura 5.12 Pagina de explorare a script-urilor.....	40
Figura 5.13 Pagina generată dinamic a unui script.....	40
Figura 5.14 Rularea unui script în back-end .....	41

Figura 5.15 Mecanismul de actualizarea a listei de script-uri rulate.....	41
Figura 5.16 Pagina de status a script-urilor .....	42
Figura 5.17 Tabel populat cu datele de ieșire al unui script.....	42
Figura 5.18 Exemplu Bar Chart populat cu date .....	42
Figura 5.19 Primul pas pentru crearea unui scenariu.....	43
Figura 5.20 Al doilea pas pentru crearea unui scenariu .....	44
Figura 5.21 Al treilea pas pentru crearea unui scenario de tip <i>agregare</i> .....	44
Figura 5.22 Al treilea pas pentru crearea unui scenario de tip <i>înlănțuire</i> .....	45
Figura 5.23 Al patrulea pas pentru crearea unui scenariu .....	45
Figura 5.24 Pagina de importare sau exportare a fișierelor de configurare .....	46
Figura 5.25 Mecanismul de importare al fișierelor de configurare .....	46
Figura 5.26 Mecanismul de exportare al fișierelor de configurare .....	47
Figura 5.27 Pagina de setări.....	48
Figura 5.28 Valiant temă cu fundal alb și culoare de accent rozalie .....	48
Figura 5.29 Valiant team cu fundal alb-gălbui și culoare de accent maronie .....	48
Figura 5.30 Pagina de informații .....	49
Figura 5.31 Rezultatul script-ului CrossLinked.....	50
Figura 5.32 Rezultatul script-ului Poastal .....	50
Figura 5.33 Rezultatul script-ului SocialScan .....	50
Figura 5.34 Rezultatul script-ului Have-I-Been-Pwned.....	51
Figura 5.35 Rezultatul script-ului MsDorkDump .....	51
Figura 5.36 Rezultatul script-ului Photon .....	52
Figura 5.37 Rezultatul script-ului Webenum .....	52
Figura 5.38 Rezultatul script-ului GitRekt .....	53
Figura 5.39 Rezultatul script-ului GitStalk .....	53
Figura 5.40 Rezultatul scenariului CrossPwned .....	54
Figura 5.41 Rezultatul scenariului CrossPoastal .....	54
Figura 5.42 Rezultatul scenariului FullSiteMap .....	55
Figura 5.43 Rezultatul scenariului CrossPhish .....	55
Figura 6.1 Întrebarea 1 – rezultate chestionar .....	56
Figura 6.2 Întrebarea 2 – rezultate chestionar .....	56
Figura 6.3 Întrebarea 3 – rezultate chestionar .....	57
Figura 6.4 Întrebarea 4 – rezultate chestionar .....	57
Figura 6.5 Întrebarea 5 - rezultate chestionar .....	58
Figura 6.6 Întrebarea 6 – rezultate chestionar .....	58
Figura 6.7 Întrebarea 7 – rezultate chestionar .....	58
Figura 6.8 Întrebarea 8 – rezultate chestionar .....	59
Figura 6.9 Întrebarea 9 – rezultate chestionar .....	59
Figura 6.10 Întrebarea 10 – rezultate chestionar .....	60
Figura 6.11 Întrebarea 11 – rezultate chestionar .....	60

## SINOPSIS

În contextul actual, există foarte multe date disponibile pe internet care pot fi folosite în scop malițios, deoarece acestea pot fi folosite pentru a eleva sisteme complexe de securitate. Pentru a asigura că datele publice nu conțin informații confidențiale, inginerii din domeniul cibernetic folosesc diverse soluții de extragere a datelor cu scopul de a interoga impactul acestora din perspectiva unui atacator.

Aplicația propusă, numită Valiant, are ca scop eficientizarea procesului de căutare și testare a soluțiilor de analiză a datelor publice existente, atât prin simplificarea etapei de identificare a acestor instrumente pentru utilizatorii neexperimentați cât și prin extinderea soluțiilor existente pentru utilizatorii familiarizați cu acest tip de audit. Soluția software va simula un coordonator pentru instrumente care detectează date confidențiale în mediul public, oferind recomandări, scenarii complexe de testare, integrarea unui serviciu cloud, posibilitatea de a partaja configurații personalizate și un utilitar pentru definirea unor script-uri și scenarii noi.

În această lucrare de licență sunt expuse arhitectura aplicației, pașii urmăți în dezvoltarea acesteia, deciziile tehnice și detaliile de implementare. În final, am expus prin imagini și rezultate aplicația și am propus câteva metode ulterioare de îmbunătățire.

## ABSTRACT

In today's context, there is a vast amount of data available on the internet that can be used for malicious purposes, as it can be utilized to bypass complex security systems. To ensure that public data does not contain confidential information, cybersecurity engineers use data mining solutions to interrogate their impact from an attacker's perspective.

The proposed application, named Valiant, aims to streamline the process of searching and testing existing intelligence tools solutions, both by simplifying the identification stage of these tools for inexperienced users and by expanding the existing solutions for users familiar with this type of audit. The software solution will simulate a coordinator for open-source intelligence tools, offering recommendations, complex test scenarios, cloud service integration, the ability to share custom configurations, and a utility for defining new scripts and scenarios.

The architecture of the application, the steps followed in its development, the technical decisions and the implementation details are exposed in this thesis. In the end, I uncovered the application through images and results, and I proposed some further improvement methods.

## **MULTUMIRI**

Doresc să îmi exprim recunoștință față de profesorul meu coordonator, Daniel Rosner, pentru îndrumare, timpul acordat și mai ales pentru răbdarea și atenția acordată fiecărei etape din realizarea acestui proiect de licență.

De asemenea, aş dori să mulțumesc companiei Adobe și a echipei din care fac parte pentru înțelegerea și timpul acordat în scopul finalizării acestui proiect oferind suport pe tot parcursul acestui proces.

În final, le mulțumesc familiei și persoanelor dragi pentru feedback-ul constant și pentru suportul moral și tehnic pe care mi l-au oferit necontenit.

# 1 INTRODUCERE

## 1.1 Context

Securitatea informațiilor reprezintă un pilon fundamental pentru crearea și susținerea unei întreprinderi indiferent de numărul de angajați sau de cifra de afaceri generată de aceasta. Posibilitatea unei scurgeri de informații poate simboliza catalizatorul pentru o pierdere masivă de fonduri generând lipsă de încredere, restructurări și în final chiar retragerea definitivă de pe piață a unei organizații.

În secolele precedente, lipsa de precauție dezvăluia adesea diverse informații confidențiale precum: locații nesupravegheate, informații personale care pot fi folosite pentru șantaj sau documente care atestă planurile viitoare ale conducerii. Pentru a procura aceste informații, un atacator trebuia să fie considerat un maestru al manipulării și să aibă o arie foarte mare de conexiuni cu alți oameni care îi pot procura informații vitale.

În prezent, datorită avansurilor majore și frecvențe în tehnologie, organizațiile au început să integreze cât mai multe sisteme software care au revitalizat și eficientizat procesele acestora, astfel crescând profitul. Datorită acestei reputații pozitive și a ajutorului oferit prin diverse pachete de sprijin oferite de guvern, numărul de business-uri lansate pe piață a crescut drastic, ajungând la aproximativ 305 milioane de startup-uri noi create anual<sup>[1]</sup>. Deși accesul la tehnologie a adus foarte multe avantaje economiei globale, atacatorii au reușit să se folosească de această răspândire în masă pentru a crea noi metode de exfiltrare a datelor. Mulțumită nevoii necontenite de a introduce sisteme software conectate la internet în fiecare dispozitiv, un atacator poate descoperii orice informație apăsând câteva butoane, astfel eliminând nevoia de atu-uri sociale, conexiuni interumane variate sau chiar nevoia de a se afla fizic în aceeași zonă cu cel păgubit.

În 2023 fondul pentru securitatea informației și al managementului de risc a crescut cu 11,3% ajungând la 188,3 miliarde de dolari<sup>[2]</sup>, cu toate acestea atacurile cibernetice sunt în continuă creștere, în 2022 aproximativ 53,35 de milioane de americani fiind victimele unui atac cibernetic<sup>[1]</sup>. Atacurile de tip Open Source Intelligence sunt cele mai răspândite tipuri de atacuri, vizând informațiile din domeniul public.

## 1.2 Problema

Inginerii care professează în domeniul securității cibernetice, numiți și *pentesteri* sau *red team pentesteri*, folosesc din abundență instrumente de tip open source situate în jurul etichetei OSINT (Open Source Intelligence Tools), deoarece există o comunitate dedicată cercetării în domeniul securității care pot oferi soluții vaste și de actualitate, însă pentru anumite situații se pot solicita și instrumente plătite.

Întrucât domeniul securității informației este foarte vast și în continuă schimbare, inginerii ajung să se confrunte cu problema găsirii unor unelte care pot acoperi cazurile specifice cerute de client. Găsirea unei unelte versatile care poate fi folosită pentru mai mulți clienți

este aproape imposibilă, deoarece metodele de extragere ale datelor confidențiale din domeniul public sunt foarte variate și greu de împachetat sub forma unui singur executabil.

Totodată, se preferă ca aceste instrumente să fie oferite în mod gratuit sau sub o singură licență, pentru a nu fi nevoie ca o întreagă echipă să gestioneze multiple licențe într-un mod securizat.

Odată găsit un instrument care acoperă vulnerabilitățile dorite, inginerul trebuie să se asigure că acesta poate rula și pe sistemul lui de operare. În majoritatea cazurilor, pentester-ii folosesc Linux sau macOS, însă există foarte mulți ingineri care folosesc sistemul de operare Windows cu toate că acesta este notoriu pentru incompatibilitatea lui cu majoritatea script-urilor online.

În unele cazuri, anumite script-uri necesită configurări complexe, ceea ce înseamnă că migrarea acestora în starea finală către un alt inginer devine mult mai grea, astfel creând o barieră în procesul de colaborare.

Totodată, majoritatea script-urilor prezintă informațiile obținute într-un format care nu este intuitiv, adesea text, ceea ce poate duce la îngreunarea procesului de identificare a vulnerabilităților. În consecință, generarea rapoartelor pe baza datelor obținute consumă mai mult timp decât în mod ușual, necesitând stilizări suplimentare asupra datelor, sub formă grafică, pentru o mai bună înțelegere asupra impactului vulnerabilităților găsite.

### **1.3 Obiective**

Soluția propusă în această lucrare, pentru atingerea cerințelor prezentate anterior, este o aplicație desktop nativă numită Valiant. Scopul principal al acestei aplicații este de a ușura gestionarea script-urilor de tip OSINT.

Aplicația oferă în mod standard script-uri care testează majoritatea scenariilor de preluare a informațiilor din domeniul public, fără să fie nevoie de altă licență decât cea oferită de aplicație la instalare. Pentru a oferi versatilitate, se pot importa în aplicație și script-uri externe, procesul de configurare fiind simplificat astfel încât utilizarea noilor script-uri să fie cât mai ușoară.

De asemenea, produsul pune la dispoziție și *scenarii* care reprezintă agregări sau înlănuiri de script-uri cu scopul de a obține mai multe date într-un timp mai scurt. Scenariile pot fi create de utilizator pe baza script-urilor prezente sau se pot folosi cele deja configurate de aplicație.

Datele capturate de script-uri sau scenarii vor putea fi vizionate cu ajutorul unui grafic ales de utilizator sau a unui tabel pentru a ușura procesul de documentație.

Pentru a încuraja importarea sau crearea de script-uri noi, aplicația pune la dispoziție un panou cu informații legate de cele mai noi script-uri de pe GitHub. Totodată, script-urile și scenariile create pot fi împărtășite cu un alt utilizator folosind o funcție automată de exportare sau importare a datelor.

Aplicația oferă posibilitatea de autentificare prin Google, GitHub sau mail pentru a securiza datele capturate de utilizator și pentru a oferi funcția de sincronizare a datelor capturate de pe un dispozitiv prin utilizarea spațiului de stocare din cloud.

Soluția software este disponibilă pe toate cele 3 sisteme de operare predominante: Windows, Linux și macOS, funcționalitățile aplicației fiind funcționale și prezente pe orice sistem.

## **1.4 Structura lucrării**

Această lucrare va conține o prezentare detaliată a etapelor de planificare, prototipare și construire a aplicației propuse. Structura documentului este împărțită după cum urmează:

- Capitolul 2: Studiu de piață / Abordări existente – prezentare a nevoilor pieței curente prin intermediul unui formular și a soluțiilor existente;
- Capitolul 3: Analiza și specificarea cerințelor – descriere sumară a funcționalităților propuse;
- Capitolul 4: Soluția propusă – prezentare a arhitecturii propuse și a tehnologiilor folosite;
- Capitolul 5: Detalii de implementare – descriere amănunțită a detaliilor de implementare la nivelul aplicației și prezentare a script-urilor și scenariilor incluse;
- Capitolul 6: Studiu de caz / Evaluarea rezultatelor – evaluare a aplicației prin intermediul unui formular;
- Capitolul 7: Concluzii – vedere de ansamblu a rezultatelor finale obținute și expunere a unor noi funcționalități care pot fi dezvoltate în viitor.

## 2 STUDIU DE PIAȚĂ / ABORDĂRI EXISTENTE

Datorită nevoilor vaste din domeniul OSINT, pentester-ii au creat diverse soluții care încearcă să înglobeze cât mai multe script-uri într-un singur executabil pentru a ușura procesul de identificare al vulnerabilităților. Totuși, deoarece domeniul OSINT este în continuă redefinire și în fiecare zi apar soluții noi, produsele existente de pe piață reușesc să acopere doar o ramură din ceea ce presupune securitatea informațiilor.

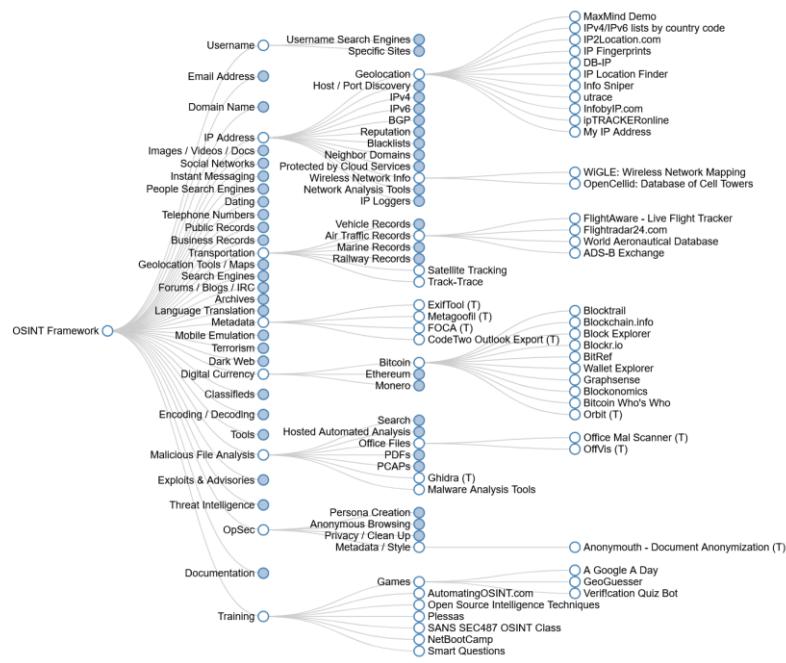


Figura 2.1 Câteva categorii de unelte OSINT conform *OSINT Framework*<sup>1</sup>

Pentru a putea identifica mai ușor nevoile pe care aplicațiile deja existente le acoperă, mă voi referi mai întâi la răspunsurile dintr-un formular creat pentru această lucrare, adresat în special inginerilor din IT și din securitate. După stabilirea cerințelor și nevoilor utilizatorilor, voi compara soluțiile existente cu acestea și cu funcționalitățile aplicației Valiant.

### 2.1 Analiză formular

Chestionarul conține opt întrebări care au ca scop analiza asupra modului în care utilizatorii de zi cu zi sau cei experimentați sunt diligenți în ceea ce privește amprenta lor digitală. Totodată, formularul conține întrebări care vor atesta nevoia unei aplicații generale pentru unele de tip OSINT, dar și nevoia de anumite funcționalități pentru a încuraja conștientizarea riscului oferite de datele cu caracter confidențial răspândite în domeniul public.

Chestionarul a fost completat de un număr de 34 de persoane. În continuare, voi prezenta rezultatele acestuia sub formă de grafice pentru fiecare întrebare.

<sup>1</sup> <https://osintframework.com/>

Ati incercat vreodata sa folositi script-uri care investigheaza amprenta voastră digitală?

34 de răspunsuri

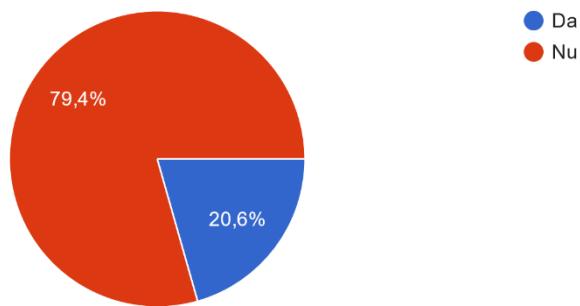


Figura 2.2 Întrebarea 1 – analiză chestionar

Aproape 80% din participanți nu au încercat să folosească instrumente software pentru a își vizualiza propriile date senzitive din mediul online. Aceste unelte sunt disponibile gratuit, iar expertiza participanților care sunt ingineri în domeniul IT, nu poate decât să confirme că majoritatea oamenilor nu înțeleg destul de clar riscul datelor publice, nu doresc să cerceteze metode de audiere a proprietății date sau în cazul celor mai puțini pricepuți, nu există o metodă palpabilă și rapidă pentru publicul larg care să ofere informații de acest tip.

Credeti ca ar fi util un dashboard care va ofera posibilitatea de a verifica amprenta voastră digitală sau a angajatilor unei firme? Credeti ca v-ar incura...punerea datelor voastre private în domeniul public?

34 de răspunsuri

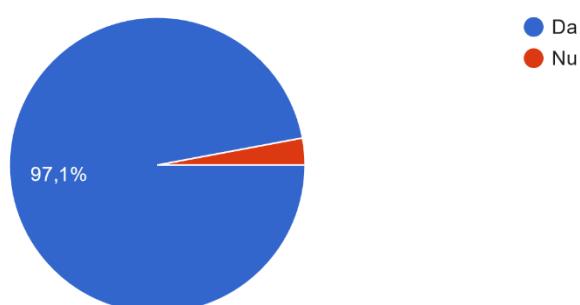


Figura 2.3 Întrebarea 2 – analiză chestionar

Pentru a demonstra nevoia de o unealtă simplificată pentru utilizatori care să ofere informații despre numărul de informații pe care acesta le are în spațiul public, am integrat întrebarea cu numărul doi în chestionar, care interoghează interesul participanților pentru o aplicație de acest tip. Aproape toți participanții au confirmat aprecierea pentru o aplicație cu acest scop, ceea ce permite integrarea următoarelor întrebări care au rol de aprofundare al soluției dorite.

Preferati rularea unor script-uri din CLI sau dintr-o interfata grafica (GUI)?

34 de răspunsuri

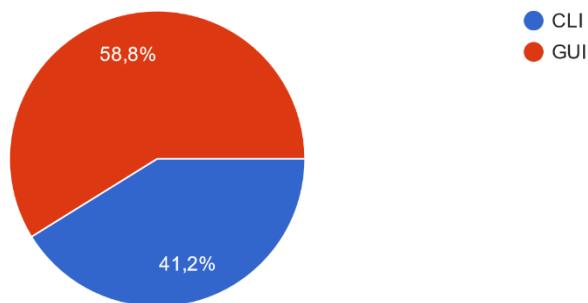


Figura 2.4 Întrebarea 3 – analiză chestionar

Întrebarea cu numărul trei are rolul de a afla preferințele utilizatorilor pentru gestionarea script-urilor dintr-o interfață grafică. Conform rezultatelor, majoritatea participanților preferă o interfață grafică, acestea fiind considerate mult mai intuitive și mai ușor de utilizat, dar o mare parte din participanți preferă rularea script-urilor din linia de comandă, datorită vitezei pe care aceasta o oferă și a versatilității în ciuda nevoii de memorare a parametrilor și formatului sau din pură obișnuință.

Din aceste două preferințe putem deduce că există avantaje pentru ambele soluții, deci utilizatorii ar beneficia de o aplicație care poate să ofere o interfață grafică care să ușureze gestionarea script-urilor, dar care să nu încetinească viteza de navigare a utilizatorului și care să nu eliminate din varietatea și simplitatea oferită de parametrii din linia de comandă.

Cat de usor va este să gasiti script-uri adecvate pe platforme open source precum GitHub?

34 de răspunsuri

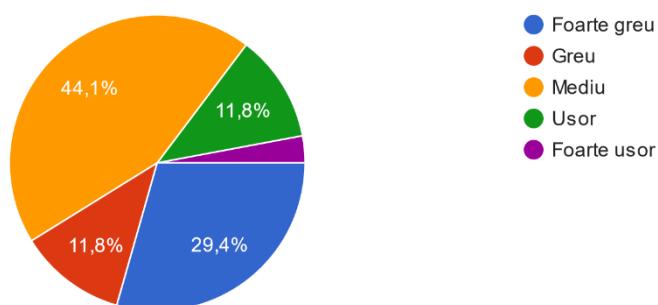


Figura 2.5 Întrebarea 4 – analiză chestionar

Peste 41% dintre participanți au asociat găsirea unor unelte potrivite pentru un uz specific ca fiind cel puțin dificilă, iar 44,1% au relaționat acest proces cu o dificultate medie.

Participanții în număr total de 85% se găsesc în aceste două tabere, deoarece majoritatea

script-urilor încărcate pe GitHub sau altă platformă de găzduire a instrumentelor open source, nu au calitățile unui produs software enterprise care să acopere majoritatea cazurilor limită sau uneori utilizatorii nu găsesc funcționalitatea pe care o caută. Totodată, folosirea acestor aplicații gratuite nu oferă o experiență completă fără probleme sau nevoi de intervenții la nivel de cod, ceea ce poate fi un impediment în găsirea unui script care funcționează fără modificări.

Credeti ca puteti gasi cu usurinta script-uri de tip OSINT (Open Source Intelligence Tools) care sa se potriveasca nevoilor voastre?

34 de răspunsuri

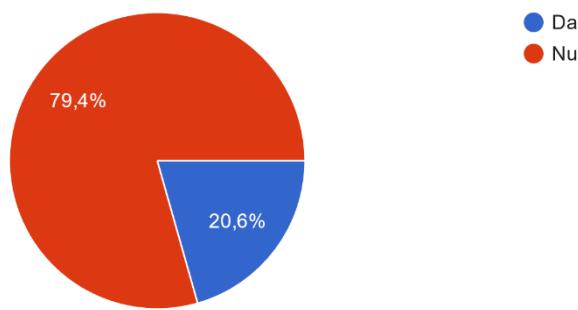


Figura 2.6 Întrebarea 5 – analiză chestionar

În continuare, întrebarea cu numărul cinci este mai specifică, fiind adresată domeniului OSINT pentru a prezenta dificultatea găsirii de instrumente potrivite pentru audierea informațiilor publice. Din punct de vedere al răspunsurilor, procentajul a rămas aproape neschimbat, confirmând din nou dificultățile de găsire a unui script potrivit.

Motoarele de căutare actuale și funcționalitățile de filtrare a rezultatelor puse la dispoziție de platforme precum GitHub pot fi suficiente pentru anumite cazuri, dar într-un domeniu care evoluează constant cu o viteză mare acestea nu pot fi mereu îndeajuns.

Conform unor statistici oficiale ale site-ului GitHub[5], în 2022 au fost făcute peste 3,5 miliarde de contribuții dintre care doar 20% au fost făcute în depozite publice, astfel putem înțelege de ce este greu ca un inginer să țină pasul cu cele mai noi script-uri datorită numărului foarte mare de date care sunt introduse zilnic pe site.

Credeti ca ar fi util sa existe o aplicatie care sa va ofere script-uri preselectate pentru OSINT?  
34 de răspunsuri

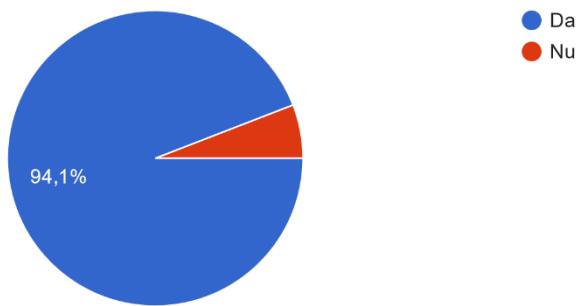


Figura 2.7 Întrebarea 6 – analiză chestionar

Din nou, întrebarea cu numărul şase califică nevoia de ghidaj pentru selectarea unor script-uri cu scop general care să poată fi reprezentative pentru majoritatea cazurilor. De asemenea, oferirea unor unele preselectate reprezintă o soluție de interes pentru aproape toți candidații, întrucât se poate începe audierea datelor private de la o bază solidă care poate fi mai apoi expandată conform cerințelor altor utilizatori.

Ati dori ca aplicatia sa poata sa fie rulata pe orice sistem de operare?  
34 de răspunsuri

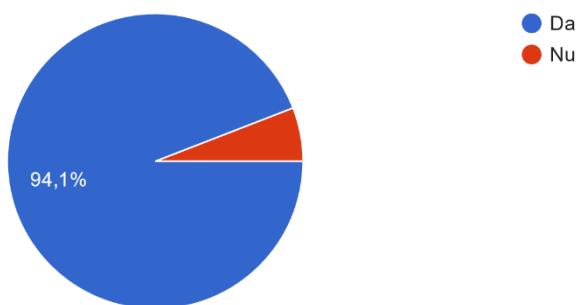


Figura 2.8 Întrebarea 7 – analiză chestionar

Majoritatea inginerilor aleg să folosească doar un sistem de operare pentru lucrul oficial cu programe datorită incompatibilităților provenite din diferențele de arhitectură. Conform unui studiu din 2022[6], peste 82% din utilizatori folosesc Windows, 15% folosesc macOS, aproximativ 2% folosesc Linux, iar restul folosesc alte sisteme de operare precum ChromeOS. Însă, un studiu din 2022 adresat programatorilor[7], prezintă o altă distribuție în care 61% din programatori folosesc Windows, 45% folosesc Linux și 46% folosesc macOS, iar 1% folosesc alte sisteme de operare. Deși majoritatea utilizatorilor se află pe Windows, scripturile sunt adresate în mare parte majoritatea sistemelor de operare bazate pe Unix, cum ar fi Linux și macOS.

Drept consecință, în statistica de mai sus putem observa cum 94% dintre participanți doresc ca aplicația pe care o folosesc să fie disponibilă pe orice sistem de operare tocmai pentru a nu exista probleme de compatibilitate și pentru a încuraja adopția unei astfel de aplicații.

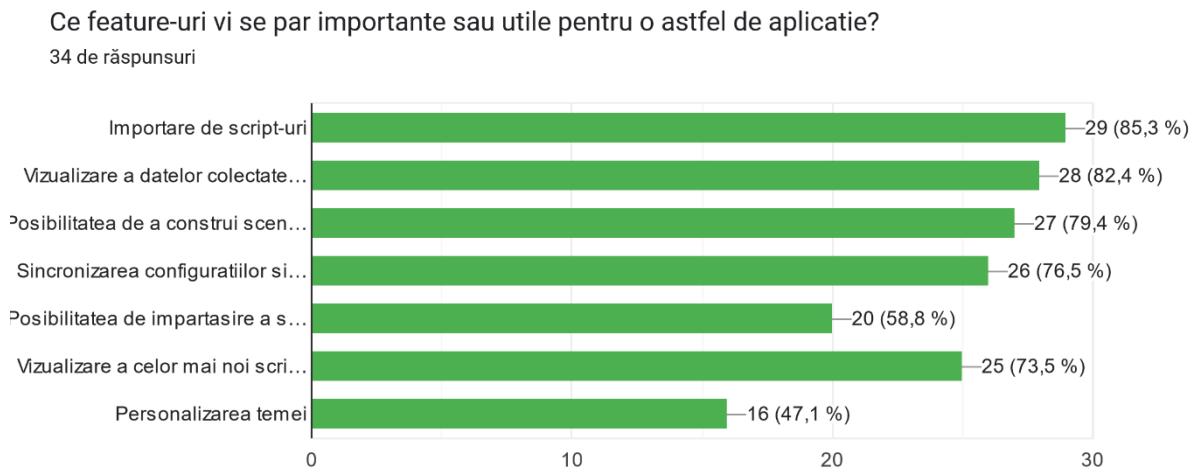


Figura 2.9 Întrebarea 8 – analiză chestionar

Ultima întrebare din chestionar, are rolul de a enumira precis nevoia anumitor funcționalități pentru o aplicație de tip panou de comandă care să gestioneze mai multe script-uri OSINT.

În ordinea descrescătoare a preferințelor, aproape toți participanții doresc să existe posibilitatea de importare a unor script-uri, fie ele personale sau preluate de la o altă sursă. Această preferință susține nevoia de varietate și personalizare a aplicației pentru variu scenarii.

Pe al doilea loc urmează cu un procentaj de 82,4%, nevoia de vizualizare a datelor colectate sub formă de grafice. Datele furnizate de uneltele care folosesc ca interfață cu utilizatorul linia de comandă nu oferă posibilitatea de vizionare într-un mod mai lizibil al datelor, astfel datele pot fi percepute ca fiind eronate sau prea complexe pentru scopul utilizatorului. Totodată, ilustrarea grafică a datelor ajută la procesul de documentare, unde clientul sau persoana în cauză are nevoie de o soluție minimală care să îi ofere o imagine de ansamblu asupra amprentei digitale pe care compania lui sau acesta o are.

Pe al treilea loc se află posibilitatea de construire a scenariilor pe baza mai multor script-uri, cu un procentaj de 79,4%. Această funcționalitate poate reduce drastic timpul de migrare al datelor dintr-un script în altul sau de agregare al acestora, întrucât aplicația ar putea automatiza acest proces. Drept urmare, pentru un singur set de intrări, se pot genera mult mai multe informații utile.

Apoi se clasează funcționalitatea de sincronizarea a rezultatelor și configurațiilor în cloud pentru a asigura disponibilitatea acestora pe mai multe dispozitive cu un procentaj de 76,5%. Datorită avansărilor în sfera IT, datele se pot transmite facil la un server extern,

disponibil oricând și oriunde, care poate să servească datele necesare pentru a nu exista probleme de coerentă între date sau nevoie de reconfigurare a aplicației în urma migrării pe alt dispozitiv. Acest utilitar se încadrează sub eticheta *Quality of Life Feature*, fiind o funcționalitate care scurtează din timpul de configurare al aplicației.

Pe locul cinci se află posibilitatea de vizionare a celor mai noi script-uri de pe GitHub, cu procentajul de 73,5%. Conform întrebărilor precedente, putem să concludem că există o nevoie de asistență în selectarea unor script-uri eficiente sau o asistență în nevoie utilizatorului de cunoaște cele mai noi script-uri care îi pot fi utile acum sau în viitor.

Pe următoarea poziție se află utilitarul de împărtășire al script-urilor și scenariilor create cu alți colegi, preferat de 58,8%. Acest utilitar permite fiecărui utilizator să folosească o configurație favorită de la alt utilizator cu ușurință, fără să fie nevoie de toți pașii de configurare. Această funcție are scopul de a favoriza colaborarea în echipe și de a încuraja comunitatea să creeze configurații proprii care pot fi utilizate de mai mulți utilizatori.

Pe ultimul rând se află posibilitatea de personalizare a temei, fiind favorizată de aproape 50% din participanți. Această opțiune a fost oferită pentru a pune accentul pe simularea creării unei aplicații proprii. Experiența utilizatorilor și elementele de *User Experience* și *User Interface*<sup>2</sup> sunt foarte importante într-o aplicație, deoarece acestea sunt elementele pe care utilizatorul le vede și cu care interacționează. Drept urmare, poate fi benefic ca aplicația să îi permită utilizatorului să își modifice aspectul aplicației după bunul plac.

## 2.2 Analiză soluții existente

În prezent, există puține soluții care încearcă să acopere cât mai multe scenarii de vulnerabilități. Conform preferințelor actuale care sunt ilustrate de clasamentul publicat pe pagina *Comparitech*, unelte care încearcă să acopere această arie sunt *Metasploit*[8] și *Recon-  
ng*[9].

Here's our list of the eight best OSINT tools:

1. **OSINT Framework** – a website directory of data discovery and gathering tools for almost any kind of source or platform.
2. **Babel X** This international search system uses AI to cross language barriers for any search term. This is a cloud-based service.
3. **Google Dorks** – OSINT data gathering method using clever Google search queries with advanced arguments.
4. **Shodan** – a search engine for online devices and a way to get insights into any weaknesses they may have.
5. **Maltego** – an OSINT tool for gathering information and bringing it all together for graphical correlation analysis.
6. **Metasploit** – a powerful penetration testing tool that can find network vulnerabilities and even be used to exploit them.
7. **Recon-  
ng** – an open-source web reconnaissance tool developed in Python and continues to grow as developers contribute to its capabilities.
8. **Aircrack-  
ng** – a wifi network security testing and cracking tool that can be used both defensively and offensively to find compromised networks.

Figura 2.10 Clasament instrumente OSINT<sup>3</sup>

<sup>2</sup> <https://careerfoundry.com/en/blog/ux-design/the-difference-between-ux-and-ui-design-a-laymans-guide/>

<sup>3</sup> <https://www.comparitech.com/net-admin/osint-tools/>

În continuare, voi prezenta fiecare produs, comparându-le cu datele rezultate din formular și cu trăsăturile stabilite ale aplicației Valiant.

### 2.2.1 Metasploit

Metasploit este un produs software disponibil ca variantă open-source, care oferă metode de atac ale vulnerabilităților, acesta fiind folosit în special în procesul de *pentesting*. Acest program este unul dintre cele mai populare utilizare pentru *pentesting*, fiind preinstalat pe Kali Linux<sup>4</sup>, datorită versatilității și popularității lui.

Conform documentației oficiale[10], programul rulează strict din linia de comandă, folosind o interfață text, care poate fi navigată folosind săgețile de la tastatură. Totodată, marcarea meniurilor importante și a *butoanelor* se face prin folosirea de culori, fontul folosit fiind uneori colorat conform unei acțiuni.

Aplicația este disponibilă pe toate cele trei sisteme de operare, dar în diferite moduri. Pe Linux aceasta poate fi preinstalată dacă se folosește distribuția Kali Linux sau poate fi instalată cu ușurință folosind utilitarul *apt*<sup>5</sup> din linia de comandă. Pentru sistemul de operare macOS se poate folosi utilitarul *brew*<sup>6</sup> din linia de comandă, în schimb pe Windows există un fișier executabil cu rol de *installer*.

Metasploit oferă vaste meniuuri din care se pot alege script-uri care conțin un *payload*, mai exact o metodă de a acapara sistemul unui utilizator. Mai multe script-uri pot fi înlănțuite pentru a crea un scenariu complex de atac, care are ca scop execuția de cod de la distanță, numită și RCE. Toate atacurile disponibile, fie ele pasive sau active, pot fi căutate din utilitarul msfconsole care servește drept sistem de navigare pentru toate opțiunile pe care produsul Metasploit le oferă. Gruparea utilitarelor este făcută în funcție de mai multe categorii, tocmai pentru a oferi o cale mai ușoară de găsire a tuturor vulnerabilităților pe care un utilizator le dorește. Script-urile sunt alese pe baza scorului CVE<sup>7</sup> pentru a asigura un impact cât mai mare, iar acestea sunt actualizate destul de frecvent pentru a ține pasul cu noile descoperiri publicate în catalogul oficial de vulnerabilități<sup>8</sup>.

```
msf > grep http search oracle
auxiliary/scanner/http/oracle_demantra_database_credentials_leak      2014-02-28
auxiliary/scanner/http/oracle_demantra_file_retrieval                  2014-02-28
auxiliary/scanner/http/oracle_llm_login
exploit/multi/http/glassfish_deployer                                     2011-08-04
exploit/multi/http/oracle_atg_file_upload                                 2016-01-28
exploit/multi/http/oracle_reports_rce                                    2014-01-15
exploit/windows/http/apache_chunked                                       2002-06-19
exploit/windows/http/bea_weblogic_post_bof                                2008-07-17
exploit/windows/http/oracle9i_xdb_pass                                     2003-08-18
exploit/windows/http/oracle_beehive_evaluation                            2010-06-09
exploit/windows/http/oracle_beehive_prepareaudioplay
exploit/windows/http/oracle_btms_writetofile                             2015-11-18
exploit/windows/http/oracle_endeca_exec                                  2012-08-07
exploit/windows/http/oracle_event_processing_upload                      2013-07-16
exploit/windows/http/osb_uname_jlist                                     2014-04-21
exploit/windows/http/osb_uname_jlist                                     2010-07-13
```

Figura 2.11 Căutarea unui script în msfconsole

<sup>4</sup> <https://www.kali.org/tools/>

<sup>5</sup> <https://linuxize.com/post/how-to-use-apt-command/>

<sup>6</sup> [https://brew.sh/index\\_ro](https://brew.sh/index_ro)

<sup>7</sup> [https://csrc.nist.gov/glossary/term/common\\_vulnerabilities\\_and\\_exposures](https://csrc.nist.gov/glossary/term/common_vulnerabilities_and_exposures)

<sup>8</sup> <https://www.exploit-db.com/>

Script-urile oferite pot fi configurate cu ajutorul msfconsole, fiecare argument fiind marcat ca fiind obligatoriu dacă este cazul. De asemenea, există argumente deja configurate cu valori uzuale pentru a ajuta la înțelegerea sintaxei și a formatului.

Rularea unui script se face tot din msfconsole, acesta putând fi pus în background pentru a îi permite utilizatorului să folosească alte utilitare cât timp acesta rulează. Starea lui poate fi verificată oricând, iar în cazul în care se stabilește o legătură directă la consola atacatului, aceasta se poate folosi direct din interfața msfconsole oferind și funcții de autocomplete.

Metasploit oferă o varietate mare în ceea ce privește metoda de atac și reușește cu siguranță să își atingă scopul, acela de a putea fi considerat o unealtă care poate fi folosită în orice caz.

Acest program reușește să acopere câteva nevoi generale, în principiu oferind script-uri preselectate care ușurează procesul de pentesting. Totuși există câteva probleme care fac pentester-ii să preferă în continuare rularea unui script manual sau folosirea altor soluții.

Interfața grafică simulată prin linia de comandă nu reușește să ofere utilizatorilor o experiență ușoară de navigare. Adesea utilizatorii au nevoie să apeleze funcția de *help* pentru a înțelege care sunt următorii pași. Deci este nevoie ca utilizatorul să se familiarizeze cu interfața și opțiunile aplicației înainte ca acesta să poată profita de viteza oferită de simplitatea liniei de comandă. Conform formularului din secțiunea precedentă, putem observa că programatorii din spatele produsului Metasploit au încercat să îmbine avantajele unei interfețe grafice tradiționale și ale liniei de comandă pentru a satisface cât mai mulți utilizatori, însă această abordare poate fi preferată de unii sau poate fi considerată ca fiind deloc intuitivă pentru alți utilizatori amatori.

Script-urile disponibile reprezintă o suiată bună pentru penetrarea unui sistem, însă nu există opțiunea de vizualizare a detaliilor acestor programe în masă. Un utilizator care nu este familiar cu vulnerabilitățile curente, nu poate selecta cu ușurință utilitarul necesar, astfel se ajunge la un scenariu în care utilizatorul caută manual fiecare cod CVE sau caută pagina oficială a script-ului sau cere sfatul altui inginer pentru a înțelege impactul soluției alese. În interfețele grafice tradiționale, această problemă este rezolvată folosind elemente grafice pentru a oferi informații despre fiecare element. Spre exemplu, pe site-urile de socializare există un paragraf sub numele fiecărui utilizator care are rolul de scurtă descriere.

Personalizarea interfeței sau a meniului de navigare nu poate fi efectuată fără intervenția utilizatorului în cod. Acest neajuns este accentuat și de faptul că aplicația nu a fost concepută pentru a oferi acest tip de funcționalitate, mai exact interfața de tip text și organizarea și modularizarea codului nu permit editarea unui singur fișier pentru a adera la soluția dorită.

De asemenea, aplicația nu dispune de posibilitatea de a crea grafice, deoarece toate script-urile oferite sunt folosite pentru găsirea de vulnerabilități în cod, astfel nu există posibilitatea

folosirii unui grafic. Totodată, lipsa de script-uri OSINT este un dezavantaj major pentru inginerii care doresc să facă cercetări pe datele disponibile în domeniul public. Drept consecință, aplicația nu poate fi folosită de ingineri din acest subdomeniu.

Mai mult, soluția propusă, Valiant, își propune să combată toate neajunsurile aplicației Metasploit prin folosirea unei interfețe grafice care poate oferi utilizatorului o experiență mult mai ușoară la navigare, prin introducerea graficelor și a script-urilor de tip OSINT, dar și prin vastele opțiuni de personalizare a aplicației. Pentru a rezolva problema impusă hibridizarea CLI-ului cu standardul GUI, Valiant va folosi o interfață grafică dinamică și configurabilă care servește drept intermediar între CLI și utilizator. Interfața și opțiunile propuse de autorul fiecărui script vor rămâne disponibile, dar acestea vor fi afișate sub format grafic pentru a putea ușura procesul de navigare, editare și rulare a script-ului fără să fie sacrificată viteza și eficiența CLI-ului. Scenariile și gestionarea script-urilor reprezintă un punct forte al aplicației discutate în această secțiune, iar acestea vor fi incluse în arhitectura soluției propuse.

### 2.2.2 Recon-ng

Recon-ng este un produs open-source care orchestrează mai multe module independente de tip OSINT pentru platformele web. Față de programul anterior, acesta este adresat strict procesului de interogare al datelor publice.

Conform documentației oficiale[11], aplicația este bazată pe mai multe interfețe grafice, una dintre ele fiind inspirată de Metasploit, prin folosirea strictă a liniei de comandă și a ilustrațiilor de tip ASCII. Navigarea se face folosind săgețile de la tastatură, dar în acest caz nu există o metodă de evidențiere a interacțiunilor prin colorarea fontului. Există și posibilitatea de folosire a unei interfețe grafice prin browser, aceasta fiind minimală.

Varietatea script-urilor este destul de ridicată, dar nu la fel de cuprinzătoare ca cea a aplicației Metasploit, deoarece unele disponibile se adresează direct datelor confidențiale disponibile din domeniul web.

Sistemul de navigare, configurare, editare și rulare al script-urilor este identic cu cel din Metasploit, dar interfața grafică oferă posibilitatea de încadrare al datelor capturate sub forma unor tabele. Există de asemenea și opțiunea de exportare a datelor sub format PDF, JSON, CSV și XLSX sau de filtrare a rezultatelor, ceea ce poate fi benefic pentru procesul de documentare.

```
root@kali:~# recon-ng

          _/\_
         / \ \ \ \ \
        // \ \ \ \ \ \ \ \
       // \ \ \ \ \ \ \ \ \
      www.blackhillsinfosec.com

[recon-ng v4.9.4, Tim Tomes (@LaNMaSteR53)]]

[76] Recon modules
[8]  Reporting modules
[2]  Import modules
[2]  Exploitation modules
[2]  Discovery modules

[recon-ng][default] > use recon/domains-vulnerabilities/xssed
[recon-ng][default][xssed] > set SOURCE cisco.com
SOURCE => cisco.com
[recon-ng][default][xssed] > run
```

Figura 2.12 Configurarea și rularea unui script în Recon-ng din CLI

### [recon-ng] [ default ]

pushpin xlsx

Tables:	companies	contacts	credentials	dashboard	domains	hosts	leaks	locations	netblocks
Ports:	profiles	pushpins	repositories	vulnerabilities					
Fields:	host	ip_address	region	country	latitude	longitude	module	filter	
Export:	json	xml	csv	list	xlsx	proxy			
host	ip_address								
autodiscover.hsploit.com	104.18.43.9								
autodiscover.hsploit.com	104.18.42.9								
ftp.hsploit.com	104.18.42.9								
ftp.hsploit.com	104.18.43.9								
imap.hsploit.com	104.18.43.9								
imap.hsploit.com	104.18.42.9								
mail.hsploit.com	104.18.43.9								
mail.hsploit.com	104.18.42.9								
as.bbc.com	132.185.162.193								
autodiscover.bbc.com	57.73.56.178								

Figura 2.13 Vizualizarea datelor într-un tabel în Recon-ng prin GUI

Per ansamblu, aplicația Recon-ng și Metasploit sunt foarte asemănătoare, diferența venind din setul de script-uri pe care acestea le oferă și prin posibilitatea folosirii unei interfețe grafice în ceea ce privește Recon-ng.

Introducerea unei interfețe grafice reușește să reducă din problemele introduse de aplicația Metasploit, însă design-ul minimal și strict funcțional reprezintă doar o soluție rapidă care nu ia în considerare principiile moderne de UI și UX.

Totodată, posibilitatea de modificare a interfeței este posibilă doar din cod, restul de configurații vizuale provenind doar din opțiunile de filtrare ale datelor.

Funcționalitatea de reprezentare a datelor sub formă de grafice nu este disponibilă, utilizatorii fiind nevoiți să recurgă la tabele sau la graficele puse la dispoziție în *Microsoft Excel* în urma exportării datelor ca format XLSX.

Soluția propusă combate aceste probleme prin aderarea la un design modern care respectă ideologiile care favorizează experiența utilizatorului. Totodată, Valiant va permite afișarea datelor prin grafice variate sau prin utilizarea unui tabel care permite mai multe opțiuni, cum ar fi ștergerea de rânduri, căutare după celule, ordonare a coloanelor etc. Posibilitatea importării unui script în Valiant va oferi utilizatorului opțiunea de a crea o aplicație personalizată în care fiecare câmp de date va putea fi optional sau utilizat în funcție de tipul acestuia.

Aplicațiile comparate nu oferă funcționalitatea explorării script-urilor noi de pe GitHub, acestea oferind doar posibilitatea de importare a unor script-uri noi, însă procesul de configurare al acestora este mai complicat în comparație cu soluția propusă, mai exact folosirea unui formular de tip *multi-page*. Explorarea script-urilor a fost considerată în formularul din secțiunea precedentă ca fiind importantă pentru utilizatori, iar lipsa acestuia va fi acoperită de pagina dedicată de explorare a depozitelor de cod noi sau favorite de pe GitHub.

Mai mult, ambele aplicații nu oferă posibilitatea de sincronizare a datelor în cloud, astfel rezultatele obținute vor fi stocate doar pe dispozitivul original al utilizatorului. Împărtășirea de configurații noi nu poate fi făcută pe Metasploit sau Recon-ng decât prin schimbarea codului sursă și prin eventuala cerere de introducere a acestuia în ramura principală a produsului pe GitHub. Ambele probleme vor fi adresate prin realizarea unei conexiuni la un serviciu cloud securizată printr-o cheie unică de autentificare și prin crearea unui utilitar care automatizează procesul de importare și exportare a funcționalităților introduse de utilizatori.

Funcționalitate	Valiant	Metasploit	Recon-ng
<b>Importare de script-uri</b>	Multi-page form	Editare cod sursă	Editare cod sursă
<b>Grafice</b>	Da (tabele și grafice)	Nu	Doar tabele
<b>Scenarii complexe</b>	Da	Da	Da
<b>Sincronizare în cloud</b>	Da	Nu	Nu
<b>Import/Export configurații</b>	Da	Editare cod sursă	Editare cod sursă
<b>Recomandări de script-uri</b>	Da	Nu	Nu
<b>Interfață personalizabilă</b>	Da	Editare cod sursă	Editare cod sursă

Tabel 2.1 Comparație funcționalități Valiant, Metasploit și Recon-ng

În urma acestor comparații se observă faptul că Valiant reușește să preia din atuurile aplicațiilor deja existente pe piață, introducând elemente noi care ajută la conștientizarea și auditul datelor confidențiale din mediul public, coborând bariera de efort necesară și făcând totodată experiența utilizatorilor mult mai bună.

### 3 ANALIZA ȘI SPECIFICAREA CERINȚELOR

Pentru a oferi o plajă cât mai variată de posibilități care să permită utilizatorului să sustragă cât mai multe date confidențiale, aplicația a fost concepută pe baza mai multor piloni fundamentali care stabilesc funcționalități diferite.

#### 3.1 Autentificare

Serviciul de autentificare permite utilizatorului să folosească un cont existent de Google sau GitHub pentru a crea un cont, ușurând gestionarea conturilor pentru utilizator. Se pune la dispoziție și funcționalitatea de schimbare a datelor contului sau de resetare a parolei în cazul în care aceasta este uitată. Dacă utilizatorul dorește să folosească alt cont decât cele menționate anterior, se poate alege înregistrarea contului prin mail și parolă. Contul poate fi apoi șters pentru a asigura utilizatorul că datele nu vor rămâne în baza de date a aplicației dacă acesta nu dorește asta.

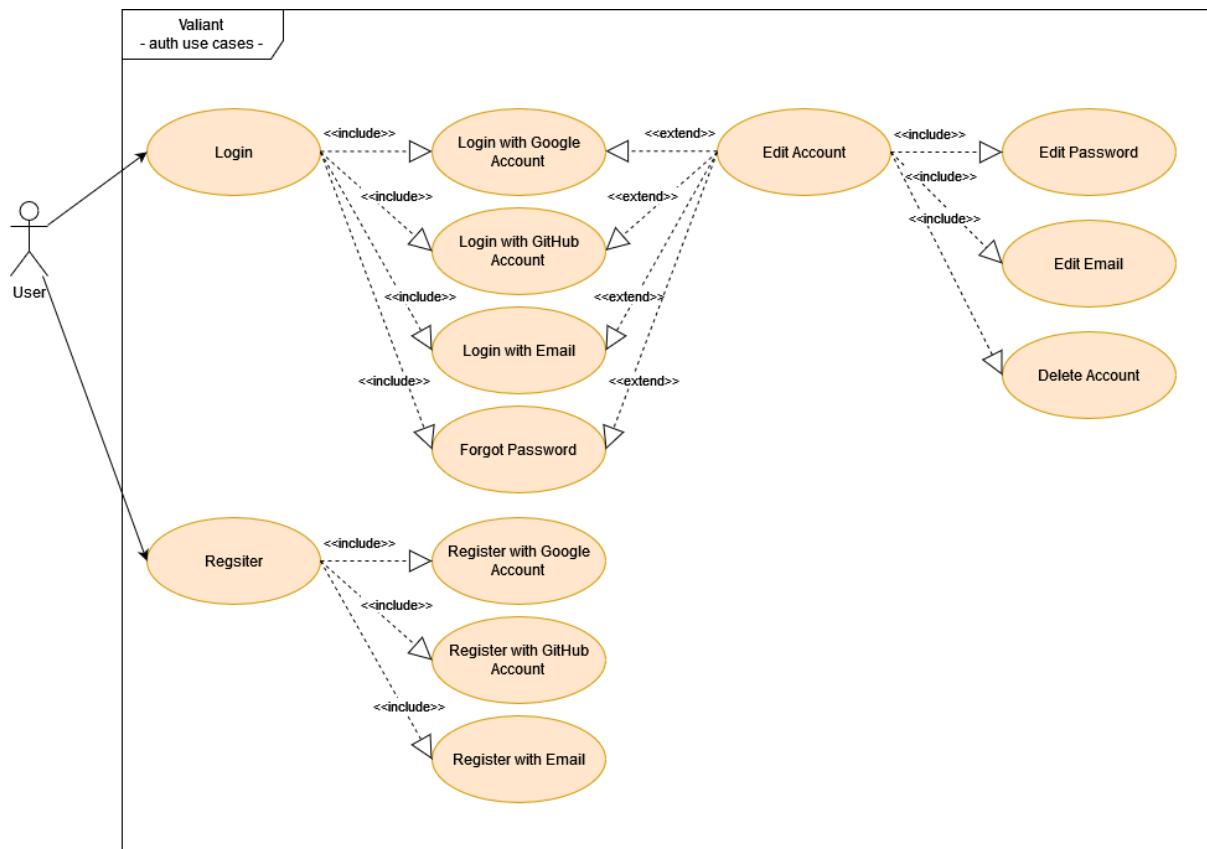


Figura 3.1 Diagrama Use Case pentru autentificare

#### 3.2 Explorarea script-urilor de pe GitHub

Pentru a putea avea o varietate cât mai mare a script-urilor și pentru a oferi metode noi de dobândire a datelor cu caracter confidențial, aplicația oferă funcționalitatea de vizualizare a script-urilor de pe GitHub într-un format concis. Există două tipuri de vizualizări pe care utilizatorul le poate selecta în funcție de filtrul principal: cele mai populare sau cele mai noi. Peste aceste două filtre, utilizatorul poate să rafineze căutarea folosind alte filtre precum perioada de apariție, limbajul folosit sau etichetele reprezentative.

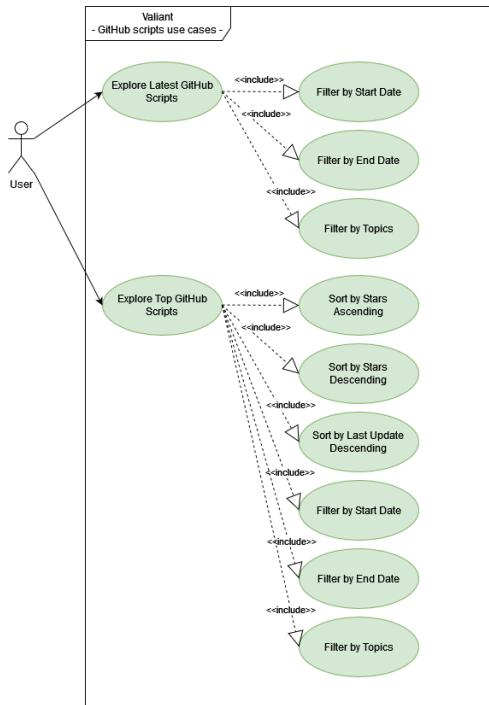


Figura 3.2 Diagrama Use Case pentru explorarea script-urilor de pe GitHub

### 3.3 Script-uri

Script-urile din aplicație pot fi rulate, importate sau vizualizate după ce execuția lor este terminată cu succes.

Un script poate fi importat folosind un formular care include mai multe pagini. Pe prima pagina, se completează datele elementare legate de script care vor fi vizualizate pe pagina de explorare a script-urilor, mai exact titlu, descriere, intrări, ieșiri etc. Pe a doua pagină, se introduc detalii legate de intrări cu scopul de a detalia nevoia lor și formatul în care vor fi folosite. Pe a treia pagină, utilizatorul trebuie să completeze formatul datelor de la ieșire din script, iar pe ultima pagină se introduc tipurile de grafice pe care utilizatorul le dorește în urma rulării script-ului.

Pentru a putea rula un script, utilizatorul poate viziona toate script-urile de pe pagina de *explorare*, fiind puse la dispoziție atât script-uri deja configurate sau cele importate în pasul precedent. După ce este ales instrumentul de atac, se introduc datele de intrare, iar statusul lui poate fi vizionat în secțiunea specială de status al script-urilor.

În secțiunea de status, se pot viziona script-urile în derulare sau cele care și-au terminat execuția. Dacă un script și-a terminat execuția, datele lui pot fi vizionate cu graficele alese conform pasului aferent de la importare.

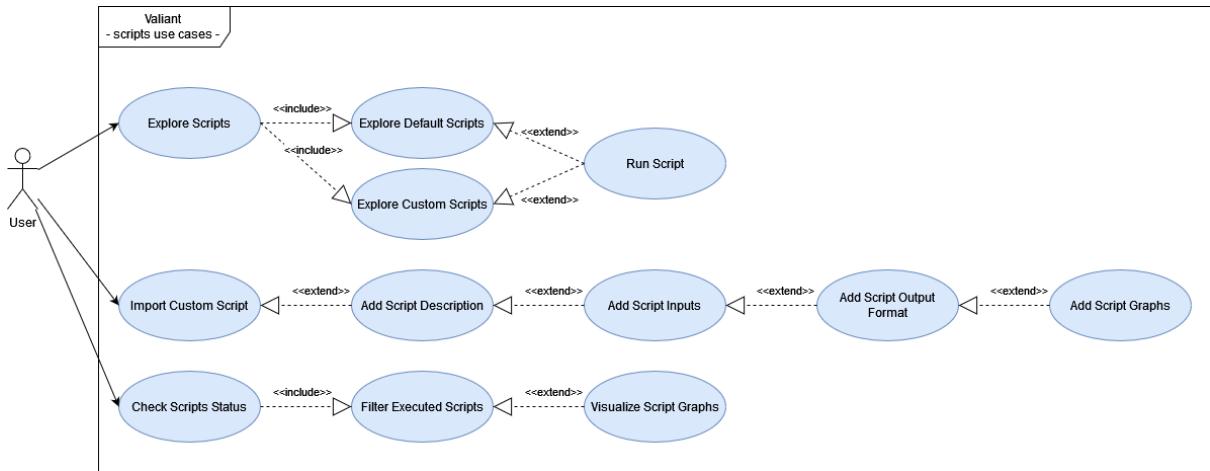


Figura 3.3 Diagrama Use Case pentru script-uri

### 3.4 Scenarii

Asemenei script-urilor, scenariile pot fi vizionate, rulate sau create. Acestea constituie o grupare de script-uri care sunt fie agregate, fie așezate sub formă de *pipe-uri*.

Un scenariu poate fi creat folosind un formular care include mai multe pagini. Pe prima pagina, se completează datele elementare legate de scenariu care vor fi vizualizate pe pagina de explorare a acestora, mai exact titlu, descriere, intrări, ieșiri etc. Pe a doua pagină, se introduc script-urile care participă la scenariu și tipul de scenariu care se dorește: agregare sau înlățuire. Agregarea presupune că toate ieșirile script-urilor vor fi suprapuse sau lipite în funcție de configurațiile utilizatorului, iar înlățuirea reprezintă așezarea în linie a script-urilor astfel încât ieșirea unui script să fie intrarea pentru alt script. Pe a treia pagină, utilizatorul trebuie să configureze intrările și ieșirile conform tipului de scenariu ales. Pe ultima pagină se introduc tipurile de grafice pe care utilizatorul le dorește în urma rulării scenariului.

Pentru a putea rula un scenariu, utilizatorul trebuie să selecteze scenariul dorit de pe pagina de *explorare*, fiind puse la dispoziție atât scenarii deja configurate sau cele create în pasul precedent. După ce alegerea este făcută, se introduc datele de intrare, iar statusul lui poate fi vizionat în secțiunea specială de status al scenariilor.

În secțiunea de status, se pot viziona scenariile în derulare sau cele care și-au terminat execuția. Dacă un scenariu și-a terminat execuția, datele lui pot fi vizionate cu graficele alese conform pasului aferent de la creare.

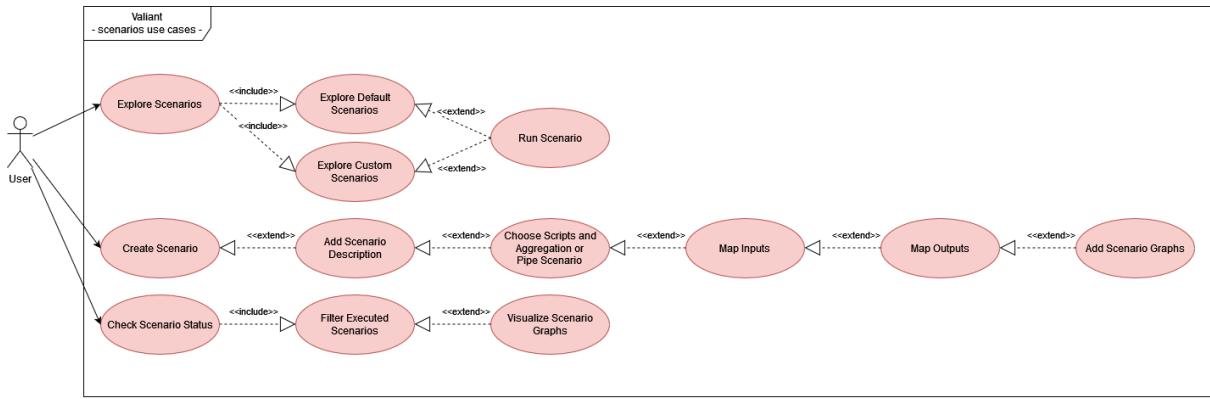


Figura 3.4 Diagrama Use Case pentru scenarii

### 3.5 Import/Export fișiere de configurare

Deoarece, utilizatorii au libertatea de a configura scenariile și script-urile conform cerințelor proiectelor pe care lucrează, aplicația introduce funcționalitatea de importare sau exportare a fișierelor de configurare create în urma pașilor de definire a script-urilor și scenariilor noi. Aplicația va prelua aceste fișiere, le va arhiva și le va cripta pentru ca utilizatorul să le poată trimite mai departe acestea urmând să fie dezarchivate și decriptate la import cu parola aprovizionată de utilizatorul sursă. Totodată, filtrele create pentru căutarea de script-uri noi vor fi și ele incluse în arhiva generată de aplicație. Drept urmare, configurațiile preferate vor putea fi copiate mai ușor de alți utilizatori, crescând eficiența colaborării în echipe de pentesteri.

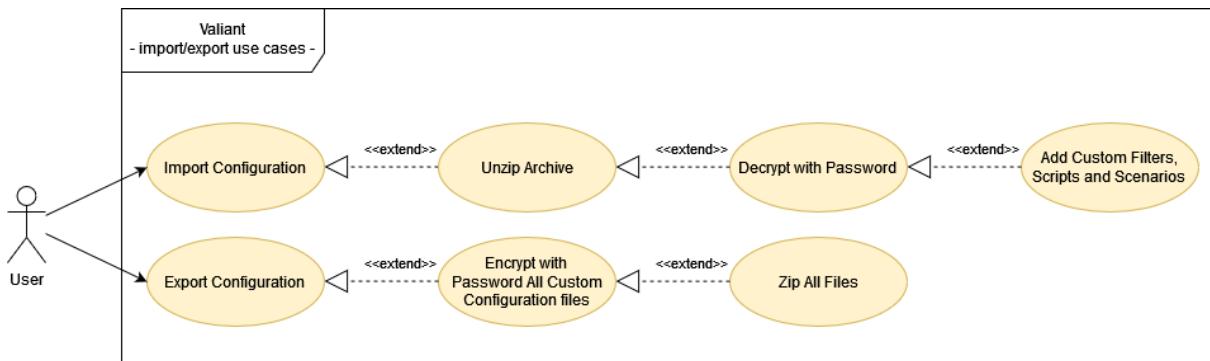


Figura 3.5 Diagrama Use Case pentru import sau export

### 3.6 Sincronizarea fișierelor de configurație și a rezultatelor cu serviciul cloud și actualizarea aplicației

Pentru a asigura disponibilitate pe mai multe device-uri, aplicația pune la dispoziție o funcționalitate de sincronizare cu un serviciu cloud, astfel fiecare utilizator își poate încărca sau descărca fișierele de configurație și rezultatele script-urilor sau scenariilor rulate. Fiecare utilizator are o zonă de memorie alocată în cloud în funcție de datele de autentificare pentru a asigura confidențialitatea datelor.

De asemenea, aplicația dispune de un sistem automatizat care verifică versiunea curentă a aplicației la pornire pentru a actualiza aplicația cu o versiune mai nouă dacă este cazul. Utilizatorul poate verifica și manual pentru noi versiuni folosind un buton special.

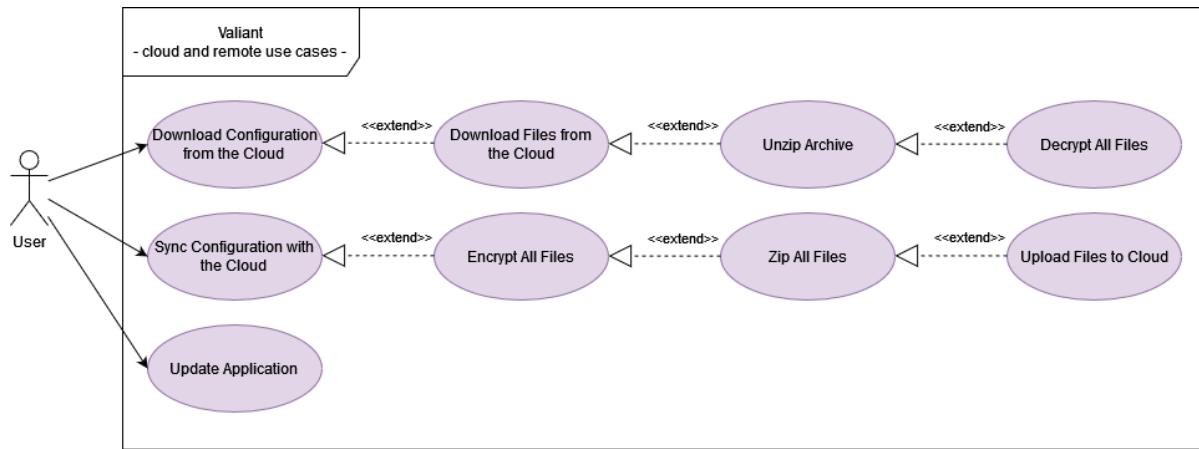


Figura 3.6 Diagrama Use Case pentru sincronizarea fișierelor cu serviciul cloud și actualizare a aplicației

## 4 SOLUȚIA PROPUȘĂ

Acest capitol prezintă arhitectura aleasă pentru aplicația Valiant, enumerând tehnologiile folosite în construcția, dezvoltarea și împachetarea aplicației, dar și motivația din spatele fiecărei decizii.

Aplicația nu a fost dezvoltată ca platformă web, deoarece un site care execută cod intensiv din punct de vedere computațional are nevoie de un server care să poată să comunice cu mai mulți clienți simultan fără ca performanța acestuia să scadă, drept urmare soluția backend pentru această implementare va avea nevoie de sisteme costisitoare și avansate de scalare pe orizontală. Rularea codului din serviciul de front-end al browser-ului nu este posibilă, deoarece acesta este destul de limitat din punct de vedere al resurselor pe care le poate împrumuta. Drept urmare, aplicația va trebui dezvoltată ca executabil care rulează nativ pe sistemul de operare al utilizatorului. Această soluție rezolvă problema resurselor, deoarece aplicația va folosi eficient resursele proprii ale utilizatorului, dar pentru ca această aplicație să fie disponibilă pe fiecare sistem de operare va fi nevoie să creăm câte un executabil pentru fiecare sistem de operare sau să folosim framework-ul prezentat în secțiunea următoare.

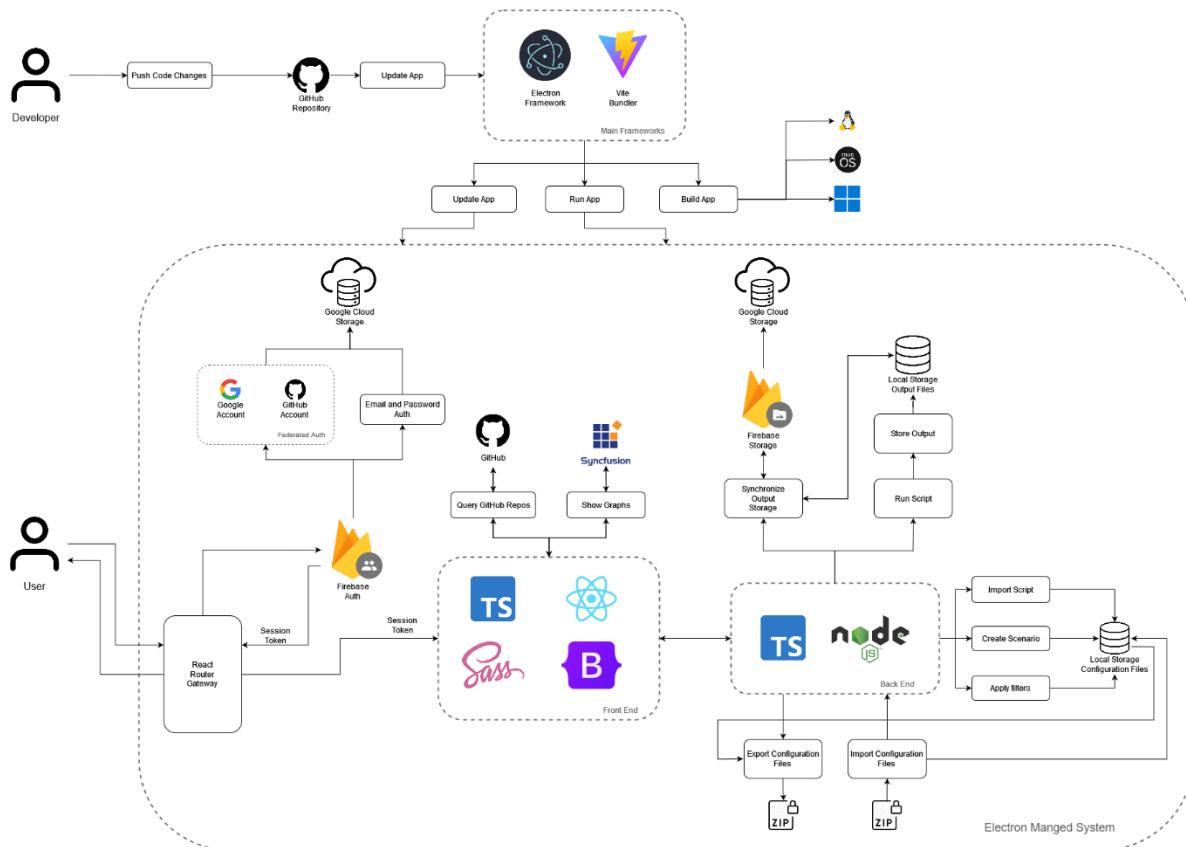


Figura 4.1 Arhitectura generală a aplicației Valiant

## 4.1 Multi-platform

Pentru a putea avea aplicația lansată și funcțională pe mai multe sisteme de operare, am decis că folosirea framework-ului Electron[12] va fi benefică, deoarece acesta permite integrarea unei singuri versiuni de cod pe mai multe platforme, în final generând executabile diferite pentru fiecare sistem de operare.

Electron se bazează pe includerea unui client de browser în executabil, astfel aplicația va fi redată identic pe orice sistem de operare, deoarece clientul de browser integrat numit *Chromium* va folosi cod HTML, CSS și JavaScript pentru crearea elementelor grafice și protocolul HTTP, la nivel local, pentru redarea paginilor specifice.

Totodată, acest framework pune la dispoziție un sistem de back-end în NodeJS care are funcții speciale introduse pentru accesul unor resurse specifice din sistem. NodeJS se bazează pe limbajul JavaScript, prioritatea acestuia fiind dezvoltarea rapidă de cod și apoi performanța, dar penalizarea de performanță poate fi ignorată, deoarece script-urile rulate în această aplicație nu au nevoie de resurse high-end, iar diferența percepță de utilizator nu va fi destul de mare[13] pentru a considera trecerea la un framework mai performant, dar care necesită mai mult timp de dezvoltare.

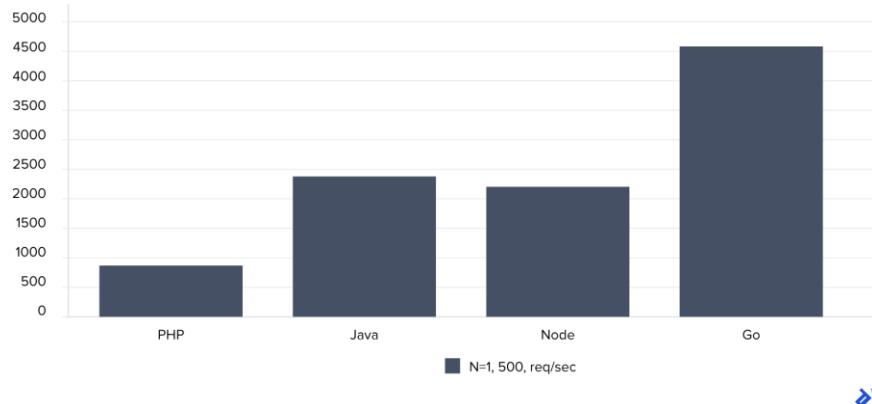


Figura 4.2 Numărul total de request-uri procesate de diferite soluții back-end[16]

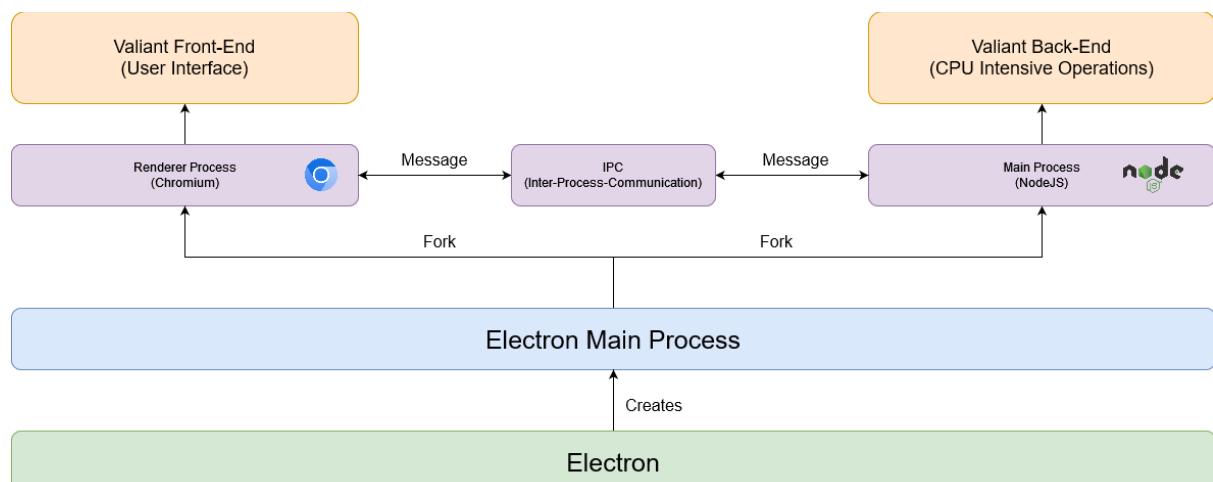


Figura 4.3 Arhitectura Electron

## 4.2 Front-End

Front-end-ul reprezintă un proces separat, dedicat pentru ilustrarea interfeței grafice cu care utilizatorul va interacționa. Tehnologiile necesare pentru front-end sunt HTML, CSS și JavaScript însă se pot folosi biblioteci sau framework-uri care au rolul de a simplifica experiența dezvoltatorului. HTML este un limbaj de programare pentru alcătuirea elementelor de bază ale paginii web, CSS este folosit în concordanță cu HTML pentru a stiliza elementele create, iar JavaScript este un limbaj de programare cu funcționalități extinse care permite crearea unor animații complexe sau efectuarea unor operații computaționale variate pentru a crea pagini care reacționează la diverse intrări aprovizionate de utilizator.

Pentru a spori funcționalitățile limbajului CSS, am folosit *Sass*[14], fiind definit ca limbaj de extensie aplicat peste CSS. Sass este un limbaj compilat care introduce funcționalități appropriate de paradigma OOP: variabile, posibilitatea selectării elementelor din HTML într-un mod imbricat, importarea altor fișiere Sass în fișierul curent și extinderea sau moștenirea de proprietăți CSS. Această extensie oferă o lizibilitate mult mai mare a codului, reducând totodată timpul de dezvoltare.

TypeScript[15] este un limbaj de programare compilat care a fost creat de Microsoft pentru a combate problemele limbajului JavaScript, mai exact erorile rezultate din tipizarea slabă și din metoda de parsare a codului prin interpretor. Aceste erori sunt eliminate prin forțarea unei tipizări tari, mai exact prin introducerea tiparelor în declararea variabilelor sau a metodelor/funcțiilor și prin generarea de erori de compilare, astfel programatorul observând mult mai rapid posibilele probleme din cod. Mulțumită limbajului TypeScript, programatorii pot depana mult mai ușor bug-uri fără să fie nevoie de executarea unor linii de cod care reprezintă un *edge case*.

Bootstrap 5[16] este o bibliotecă de elemente stilizate și animate care au rolul de acceleră viteza de dezvoltare a unui site. Dezvoltatorii pot folosi direct elementele oferite pentru a crea un prototip cu o interfață grafică modernă care respectă regulile de tip UX. Mai mult, Bootstrap oferă posibilitatea de a modifica aspectul fiecărui element prin editarea variabilelor CSS definite de aceștia.

React[17] este cel mai popular framework disponibil pentru front-end conform statisticilor oferite de Stackoverflow în 2023. Acesta oferă posibilitatea de creare a interfețelor prin componente care pot fi afișate pe ecran în mod dinamic. Reîncărcarea paginii este efectuată eficient prin folosirea unui DOM virtual care este clonat în DOM-ul real abia după ce se stabilește un *batch* de modificări. Prin gruparea schimbărilor înainte de reîncărcare, React reușește să mențină interfața web rapidă chiar dacă aceasta este alcătuită din mai multe elemente individuale dinamice. Pentru a afișa conținut dinamic, se folosesc variabile condiționale care sunt expuse programatorului prin funcții de *get* și *set*. Aceste variabile declanșează o reîncărcare mulțumită unui mecanism asemănător design pattern-ului Observer. Prin folosirea acestui framework, dezvoltarea de pagini poate fi mult mai ușor modularizată folosind conceptul de *componentă*, astfel crescând eficiența dezvoltatorilor.

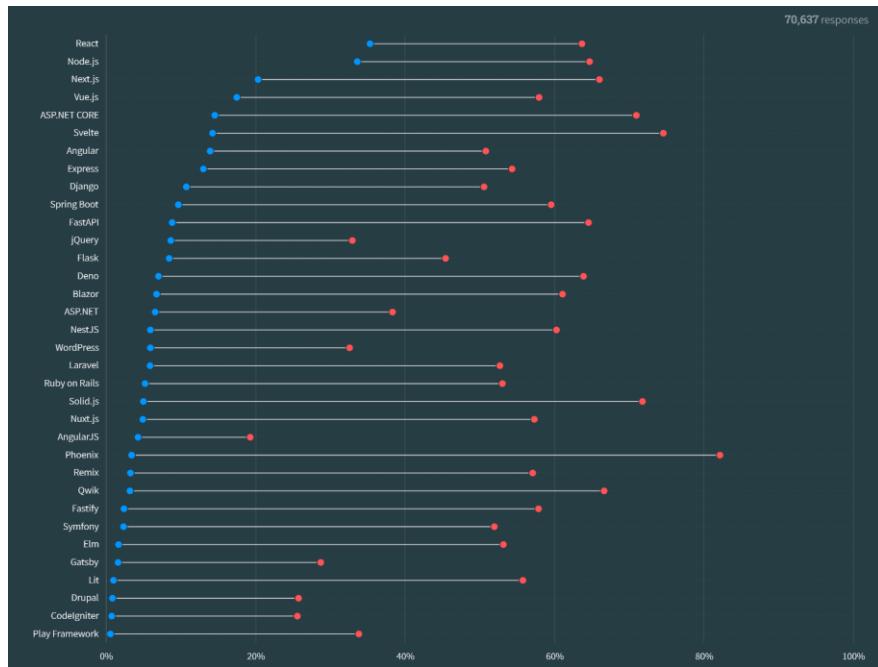


Figura 4.4 Clasament tehnologii web preferate 2023 (bullets - desired and admired)<sup>9</sup>

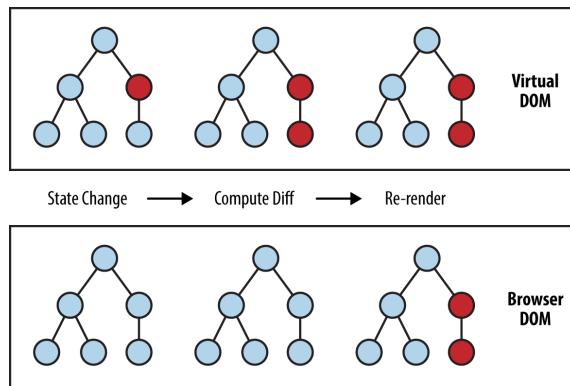


Figura 4.5 Comportamentul unui DOM virtual în React<sup>10</sup>

De asemenea, pentru generarea de grafice și tabele am folosit biblioteca *Essential JS 2 (EJ2)* oferită de Syncfusion[18]. Biblioteca oferă dezvoltatorilor componente deja create cum ar fi: grafice, tabele, PDF viewer, tooltip-uri etc. Aceste componente sunt compatibile cu React și oferă un grad foarte dezvoltat de personalizare, spre exemplu tabelele pot avea funcționalități precum: ordonare după coloană, căutare după celulă, ștergere a unui rând, filtrare de date etc. Complexitatea acestor componente este perfectă pentru soluția propusă, deoarece tabelele și graficele puse la dispoziție vor putea fi personalizate conform cerințelor utilizatorului. EJ2 este gratuit pentru organizațiile care fac anual sub 1 milion de dolari profit, urmând apoi ca dezvoltatorii să folosească o subscripție anuală pentru folosirea în scop comercial al acestei biblioteci.

<sup>9</sup><https://survey.stackoverflow.co/2023/#section-admired-and-desired-programming-scripting-and-markup-languages>

<sup>10</sup><https://programmingwithmosh.com/react/react-virtual-dom-explained/>

Stiva de tehnologii alese pentru aplicația Valiant va permite crearea unui cod foarte modular, eficient și foarte modern, ceea ce va accelera procesul de dezvoltare, dar și performanța aplicației.

### 4.3 Back-End

Back-end-ul reprezintă un proces separat, cu rolul de executare a operațiilor intensive, rezultatul lor urmând să fie comunicat procesului de front-end, în acest caz prin IPC. Soluțiile back-end existente sunt construite pe baza unui limbaj, astfel avem framework-uri precum Spring Boot care se bazează pe Java, NodeJS care folosește JavaScript, .NET care folosește C# etc.

Deoarece, Electron este bazat pe JavaScript, singura soluție de back-end validă este NodeJS. Pentru a crește robustețea codului, am ales să cuplez NodeJS cu TypeScript.

Toate apelurile de scriere sau ștergere în fișiere sau de rulare a script-urilor sunt făcute în back-end pentru a nu ocupa procesul responsabil de front-end cu sarcini care pot beneficia de mai multe resurse sau care nu se referă strict la elemente grafice.

Comunicarea cu procesul din front-end este făcută prin funcția IPC, pusă la dispoziție de Electron. Mai exact, se pot asculta și transmite mesaje pe anumite canale, urmând ca aceste mesaje să fie prelucrate.

De asemenea, setările ferestrei de afișare a aplicației sunt disponibile din procesul de back-end, putând fi selectate opțiuni precum: dimensiunea ferestrei, panou de debugging, notificări native sistemului de operare etc.

### 4.4 Servicii cloud

Pentru a oferi posibilitatea de sincronizare a datelor cu mai multe dispozitive este nevoie de utilizarea unui serviciu cloud care să gestioneze aceste fișiere și să fie disponibile în orice moment, în cea mai nouă versiune disponibilă.

Piața producătorilor de servicii cloud este foarte vastă, însă Google oferă posibilitatea utilizării platformei Firebase[19], care oferă un serviciu de autorizare a utilizatorilor dar și un spațiu disponibilă pentru stocarea datelor sau a fișierelor. Drept urmare, am decis să folosesc Firebase, deoarece ușurează drastic procesul de securizare al sistemului de autentificare, fiind necesar doar un apel cu datele utilizatorului din aplicație, sistemele de validare a datelor și a securității query-urilor fiind gestionate direct de Google în propria platformă. Totodată, aceștia oferă posibilitatea de autentificare prin servicii *federated* cum ar fi Google, Facebook, GitHub etc.

Serviciul Firebase este apelat din front-end, deoarece acesta permite crearea unor request-uri de tip HTTP, astfel se pot face query-uri directe cu datele necesare către serviciul aprovisionat de Google. Aceste request-uri pot fi făcute și din back-end, însă back-end-ul ar trebui folosit doar dacă trebuie făcute operații complexe din punct de vedere computațional,

iar Firebase face toate aceste calcule deja în propriul serviciu, transmitând direct datele necesare query-ului.

Pentru aplicația Valiant, am ales să folosesc funcționalitatea de autentificare prin parolă și email, dar și prin cont de Google sau GitHub. De asemenea, pentru sincronizarea fișierelor necesare voi folosi sistemul de storage destinat stocării fișierelor.

## 4.5 Actualizări

Actualizările aplicației vor fi făcute prin utilitarul *electron-updater*[20]. Utilitarul verifică semnătura aplicației stabilite la *build time* și creează o semnătură nouă asociată unei versiuni mai noi a aplicației.

Aplicația poate face un request HTTP la repository-ul asociat în semnătură pentru a verifica dacă există actualizări disponibile. În cazul în care acestea există, aplicația își va prelua noile date, actualizându-se automat la cea mai nouă versiune.

Pentru a putea declanșa procesul de actualizare, este nevoie doar de introducerea unui nou *commit* în repository-ul de GitHub pe branch-ul *main*. Acest utilitar funcționează atât cu repository-uri publice cât și private, dar se poate folosi și un server simplu care comunică prin protocolul HTTP.

## 4.6 Împachetarea executabilelor

Pentru generarea executabilelor, Electron pune la dispoziție un utilitar care creează fișiere unice pentru fiecare sistem de operare: Windows, macOS și Linux.

De asemenea, pentru optimizarea dependințelor și pentru integrarea unui sistem de *hot-reload* care reîncarcă doar componentele schimbate la fiecare modificare am folosit Vite[21]. Acest utilitar de împachetare oferă dezvoltatorilor o experiență mai bună prin optimizarea obiectelor statice, a dependințelor provenite din modulul *NPM*<sup>11</sup>, integrarea variabilelor de mediu și mai ales prin posibilitatea de a viziona rapid schimbările produse în cod.

Mai mult, Vite oferă funcționalități avansate precum *SSR rendering*<sup>12</sup>, preîmpachetare a dependințelor și multe altele. Acest utilitar ajută la eficientizarea aplicației atât în etapa de dezvoltare cât și la pasul de împachetare pentru lansarea în producție.

---

<sup>11</sup> <https://www.npmjs.com/>

<sup>12</sup> <https://www.heavy.ai/technical-glossary/server-side-rendering>

## 5 DETALII DE IMPLEMENTARE

Aplicația Valiant este organizată pe pagini, fiecare pagină având mai multe componente în funcție de complexitatea acestora. Se definesc astfel paginile principale, mai exact *Dashboard*, *Register* și *Login*.

Componentă	Rută	Metodă HTTP
Login	/login	GET
Register	/register	GET
Dashboard	/	GET

Tabel 5.1 Rutele HTTP ale aplicației Valiant

În momentul pornirii aplicației, se verifică dacă există un jeton de autentificare stocat în clientul browser, urmând ca apoi să se selecteze ruta potrivită. Prin folosirea modulului *React Router*[22], se simulează introducerea unui API Gateway care are rolul de a redirecționa o cerere HTTP către microserviciul potrivit, în acest caz React Router selectează componenta React care va fi afișată în funcție de ruta HTTP a cererii primite.

```
<Routes>
  <Route element={<PrivateRoute />}>
    <Route path='/' element={<Dashboard />} />
  </Route>
  <Route path='/register' element={<Register />} />
  <Route path='/login' element={<Login />} />
</Routes>
```

Figura 5.1 Folosirea modulului React Router pentru rutarea cererilor HTTP

Pentru a permite doar utilizatorilor autenticați folosirea aplicației, am creat o componentă care verifica, prin intermediul API-ului Firebase, dacă un utilizator este autentificat, caz în care se permite vizualizarea tuturor componentelor care au rang de copil, altfel se redirecționează utilizatorul către ruta */login*.

```
const PrivateRoute = () => {
  const { currentUser } = useAuth();

  return currentUser ? <Outlet /> : <Navigate to='/login' />;
};
```

Figura 5.2 Componenta *PrivateRoute* care securizează rutele neautentificate

Din punct de vedere al design-ului, toate paginile au fost create respectând principiul de *negative space* în care elementele din UI sunt așezate astfel încât să existe mereu spațiu liber pentru a nu copleși utilizatorul cu foarte multe detaliu într-un spațiu îngheșuit. Totodată, rata de contrast dintre elemente este de 3:1, astfel fiecare element este vizibil indiferent de fundalul pe care acesta se află. Tematica limbajului este minimalistă, cu accent pe elemente analitice și formulare conform unei aplicații de tip dashboard.

### 5.1 Autentificare

Pagina de logare folosește două componente separate, una care conține un formular creat din componente HTTP de tip *input* și o componentă care introduce o animație continuă în

concordanță cu identitatea numelui aplicației care provine de la numele unui avion din secolul 20. Formularul îi permite utilizatorului să folosească autentificarea de tip federated prin Google sau GitHub sau prin folosirea unui email existent și a unei parole. Pagina de înregistrare a unui nou cont folosește aceleași principii, singura diferență din punct de vedere vizual fiind poziționarea celor două componente.

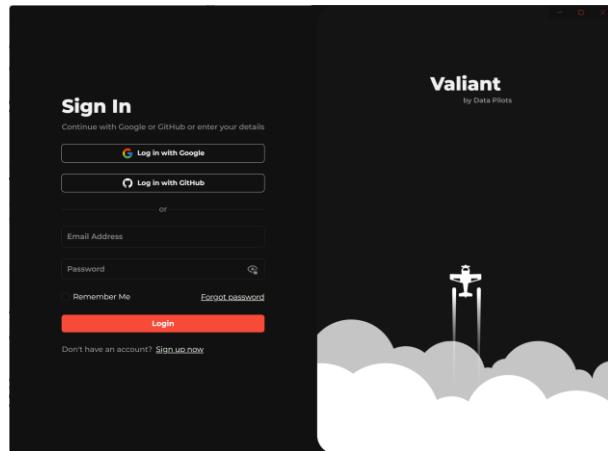


Figura 5.3 Pagina de logare

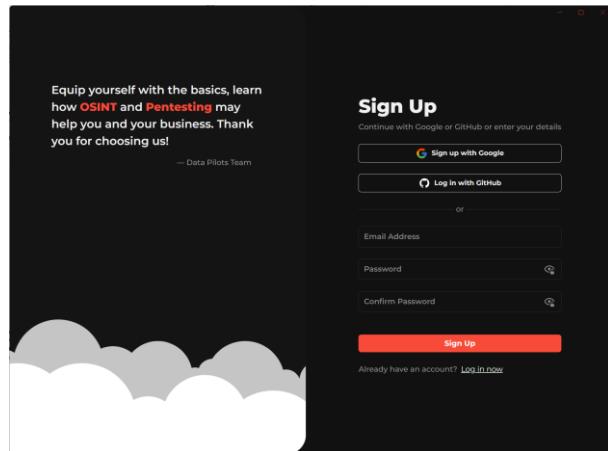


Figura 5.4 Pagina de înregistrare a unui utilizator

Autentificarea este realizată prin apelarea metodelor expuse de API-ul Firebase. Conectarea aplicației la serviciul creat în consola Google Firebase se face prin inițializarea unui obiect cu date private asociate proiectului, astfel fiecare cerere de folosire a serviciului Firebase va fi făcută la URL-ul corect cu drepturi depline.

Firebase Auth[23] pune la dispoziție câteva metode utile și abstractizate pentru procesul de autentificare, dezvoltatorul fiind nevoit doar să paseze datele introduse de utilizator către acestea, cum ar fi: emailul sau parola. Metoda de logare prin folosirea unui cont de Google sau GitHub este gestionată de Firebase, programatorul fiind desemnat să selecteze serviciul corect din cod.

Pentru scenariul de resetare a parolei, denumit în aplicație ca *Forgot your password?*, Firebase introduce o metodă care trimite un email de resetare pe contul asociat cu cel

introdus în câmpul de date. În email se află un link cu un domeniu gestionat de Firebase de unde utilizatorul își poate schimba parola.

Stadiul utilizatorului este monitorizat prin intermediul variabilelor de tip *State* din React, acestea având atașate un observer care permite implementarea unui sistem de notificare a componentelor care folosesc aceste variabile. Totuși, pentru a asigura gestionarea corectă a unei variabile și disponibilitatea acesteia din orice componentă, se poate folosi o componentă de tip *Context*. Această componentă este folosită pentru a semnala faptul că mai multe componente au nevoie să vizualizeze în timp real anumite variabile, soluția fiind crearea unui constructor numit *Provider*. Astfel, am creat un Provider pentru autentificare care conține toate metodele și variabilele asociate autentificării după care l-am introdus în aplicație sub formă unei componente de rang părinte, deasupra componentei principale a aplicației.

## 5.2 Dashboard

Pagina principală conține anumite componente statice precum un meniu de navigare vertical și o secțiune de tip *Header* care conține un meniu cu două butoane pentru editarea contului sau pentru ieșirea din cont. Totodată, în secțiunea definită de Header există și un sistem de notificări care se incrementează pe măsură ce anumite script-uri sau scenarii își termină execuția.

Din pagina de dashboard se poate naviga către alte pagini prin selectarea altor componente în funcție de iconița activă din meniul de pagini. Selectarea dinamică a componentelor ajută la reducerea liniilor de cod, deoarece componentele statice pot fi scrise în componenta principală acestea fiind afișate indiferent de pagina din dashboard pe care se află utilizatorul.

Paginile pe care se pot naviga din dashboard sunt: Home, Explore Scripts, Import Script, Scripts Status, Explore Scenarios, Create Scenario, Scenario Status, Import/Export, Settings și About. Pagina activă este marcată de un element vizual în meniul de navigare vertical.

## 5.3 Home

Înțial, pagina încărcată în dashboard este cea de *Home*. Pe această pagină se află un banner care explică rolul fiecărei secțiuni majore din aplicație și două subsecțiuni care au rolul de navigare al unor script-uri de GitHub.

Script-urile sunt preluate folosind API-ul public de GitHub. Fiecare căutare poate fi rafinată folosind parametrii de căutare din URL. Rezultatele returnate conțin date vaste despre autori și script-urile lor.

Prima secțiune folosește o căutare care returnează cele mai noi script-uri dintr-o anumită perioadă. Înțial, perioada de căutare este echivalentă cu luna curentă. Totodată, căutarea conține și un filtru pe etichetele disponibile, mai exact se caută doar depozitele de cod marcate cu eticheta *OSINT*.

Datele primite sunt introduse într-un element de tip *card* în care sunt include numele autorului, numele script-ului, limbajul script-ului, URL, numărul de aprecieri și etichetele

script-ului. Forma de cartonaș a fost aleasă pentru a favoriza o vizualizare rapidă a celor mai noi script-uri. Numărul de cartonașe afișate pe ecran este actualizat în funcție de mărimea ecranului. În modul ecran complet se pot viziona 4 cartonașe, dacă lățimea ferestrei este de 2/3 se pot viziona 3 cartonașe, iar dacă lățimea ferestrei este de ½ se pot viziona 2 cartonașe. Pentru a viziona mai multe script-uri se pot folosi butoanele de navigare orizontală, aceste butoane introducând câte un nou script per apăsare.

Utilizatorul poate folosi un filtru avansat de căutare pentru această secțiune, disponibil deasupra cartonașelor de unde se poate configura perioada de căutare sau etichetele reprezentative.

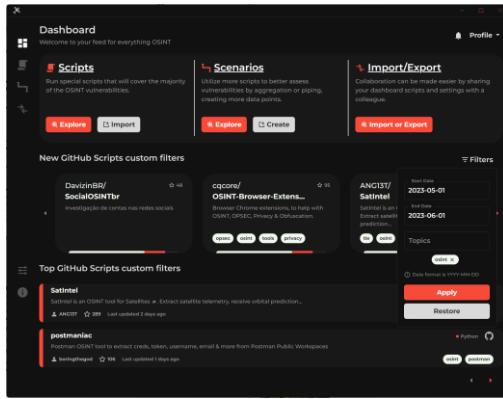


Figura 5.5 Pagina *Home* cu filtre pentru prima secțiune de recomandări

A doua secțiune folosește un tip de căutare care favorizează script-urile cu cele mai multe aprecieri. Inițial, se caută script-urile cu cele mai multe *stele* și cu eticheta OSINT.

Datele primite sunt introduse într-o listă verticală, fiecare element afișând date precum: numele script-ului, descriere, autor, număr de stele, ultima modificare, etichete, limbaj și URL. În această secțiune am ales să prezint datele sub formă de listă pentru a favoriza vizualizarea în masă mare a datelor, păstrând totuși datele relevante pentru utilizatori.

Numărul de elemente poate fi modificat din filtrul avansat disponibil deasupra listei, numărul standard fiind de 2 elemente per pagină. Utilizatorul poate folosi butoanele disponibile pentru a trece la următoarea pagină.

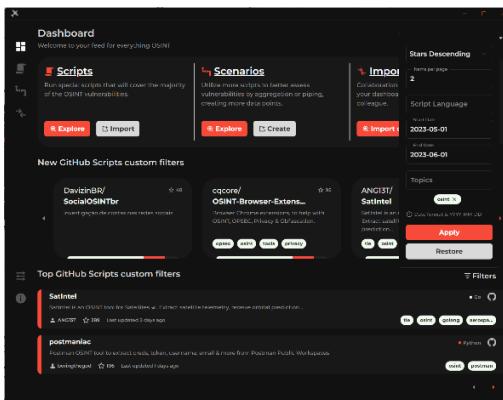


Figura 5.6 Pagina *Home* cu filtre pentru a doua secțiune de recomandări

Script-urile rezultate din căutare sunt salvate în două fișiere separate de tip *JSON*. Aplicația încearcă să citească cele două fișiere pentru a prelua datele deja găsite, dar dacă acestea nu sunt disponibile se fac două cereri noi cu filtre standard.

Aplicația folosește un interval aprovisionat de NodeJS care are rolul de a reinițializa căutările cu filtrele actuale pentru a asigura utilizatorul că datele afișate sunt actualizate. Intervalul de reîncărcare este de 24 de ore.

```
const useFetchOnThreshold = ({  
  threshold,  
  fetchFunction,  
  lastFetchTime,  
}: Props) => {  
  const timerRef = useRef<NodeJS.Timeout>(null);  
  
  useEffect(() => {  
    const checkThreshold = () => {  
      const currentTime = Date.now();  
  
      if (!lastFetchTime || currentTime - lastFetchTime > threshold) {  
        fetchFunction();  
      }  
    };  
  
    checkThreshold(); // Run immediately on mount  
  
    // Set up the interval to check the threshold periodically  
    timerRef.current = setInterval(checkThreshold, threshold);  
  
    // Clean up the interval on unmount  
    return () => {  
      clearInterval(timerRef.current);  
    };  
  }, [threshold, fetchFunction]);  
};
```

Figura 5.7 Mecanismul de actualizare al script-urilor de GitHub recomandate

## 5.4 Import Scripts

Pagina *Import Scripts* are rolul de a introduce script-uri noi în aplicație. Pentru a asigura că aplicația poate gestiona complet script-ul introdus, am introdus un formular cu mai multe pagini, fiecare pagină având un formular unic de care aplicația se poate folosi în diverse etape. Pentru a import un script, utilizatorul trebuie să parcurgă patru pași.

Primul pas reprezintă introducerea datelor sumare despre script care vor fi afișate în pagina de explorare a script-urilor încărcate. Se introduc date legate de: URL-ul paginii de unde script-ul a fost preluat, numele script-ului și o scurtă descriere a script-ului. Utilizatorul selectează folosind o scară de la 1 la 5 cât de rapid este script-ul și cât de des reușește să obțină rezultate. Totodată, se introduc cuvinte care vor sumariza intrările și ieșirile din script. Toate aceste date sunt folosite pentru a aproviza utilizatorul cu o scurtă descriere, dar semnificativă care să îl ajute în selecționarea script-ului corect.

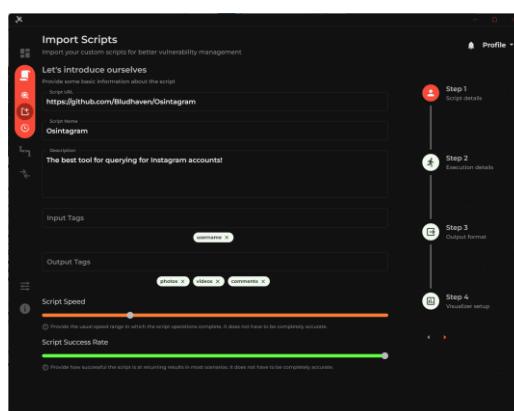


Figura 5.8 Primul pas de importare al unui script

Al doilea pas are rolul de a specifica calea către fișierul executabil și către utilitarul care va executa fișierul cum ar fi Python. De asemenea, se introduc intrările și tipul acestora. Un script poate folosi ca intrare un argument sau un argument care necesită un *flag* sau un *flag* care nu necesită alt argument. Fiecare argument poate fi marcat ca fiind necesar. Primele două intrări au rolul de a specifica exact cum se va executa script-ul, următoarele intrări fiind folosite în pagina dedicată a script-ului, fiecare argument având un mod de selectare unic conform tipului acestuia.

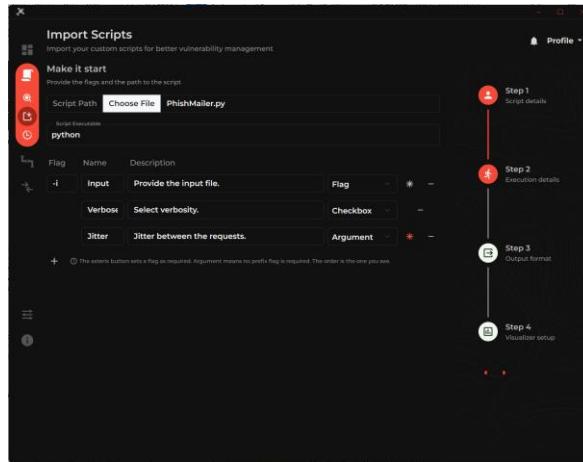


Figura 5.9 Al doilea pas de importare al unui script

Al treilea pas este folosit pentru a specifica ce date vor ieși din script. Există posibilitatea ca un număr de rânduri să fie ignorate la citirea fișierului de ieșire în cazul în care anumite rânduri nu sunt utile. Mai mult, se poate introduce o expresie *RegEx* pentru separarea textului de la ieșire, urmând ca apoi să se definească coloanele și tipul acestora în funcție de rezultatul expresiei introduse.

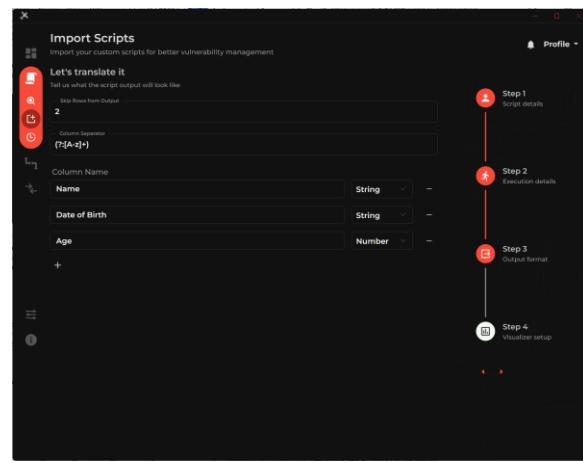


Figura 5.10 Al treilea pas de importare al unui script

Ultimul pas folosește coloanele introduse în pasul anterior pentru a selecta un tip de grafic. Acest grafic sau tabel va fi folosit pentru vizualizarea datelor. De asemenea, se specifică fișierul de unde se vor citi datele, cum ar fi *ieșirea standard*, un fișier sau se poate numele fișierului folosit într-un argument de intrare. Graficele disponibile sunt: Line Chart, Bar Chart,

Scatter Chart și Pie Chart. Aceste grafice au nevoie de cel puțin două coloane, iar în cazul graficului Scatter Chart este nevoie ca script-ul să aibă cel puțin trei coloane la ieșire. Dacă nu există suficiente coloane, anumite grafice nu vor putea fi selectate. Totodată, o eroare va fi afișată dacă nu este selectată o coloană care să reprezinte o valoare numerică. În cazul în care nu există o valoare numerică, se poate folosi o funcție de agregare care va număra intrările de pe prima coloană selectată. Se pot selecta maximum cinci grafice folosind un sistem asemănător sistemului de tab-uri dintr-un browser.

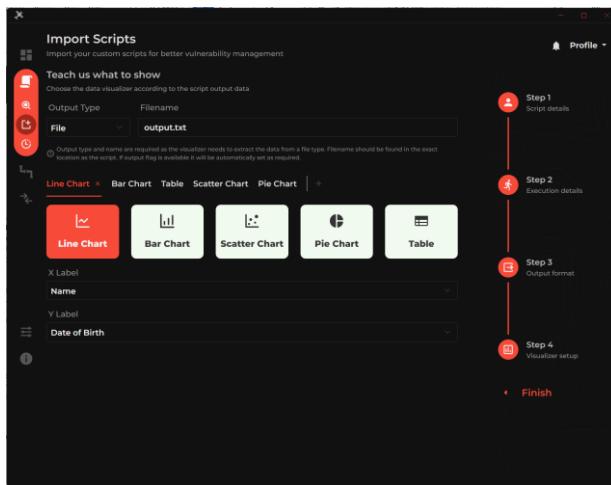


Figura 5.11 Al patrulea pas de importare al unui script

După ce ultimul pas este finalizat, utilizatorul va fi redirectat către pagina de explorare a script-urilor. Configurările introduse vor fi salvate într-un fișier JSON care va fi citit de fiecare dată când se va naviga spre pagina *Explore Scripts*.

## 5.5 Explore Scripts/Explore Scenarios

Script-urile oferite de aplicație sau importate pot fi vizualizate pe această pagină. Fiecare script poate fi vizualizat în propriul element afișat sub formă de card. Fiecare card conține datele introduse în primul pas din formularul de importare, mai exact: nume, descriere, viteză, rată de succes, intrări, ieșiri și URL.

Datele sunt citite din fișiere JSON, fiecare script fiind asociat unui cartonaș. Utilizatorul poate selecta ce tipuri de script-uri vrea să vizualizeze, standard sau importate, folosind un buton aflat deasupra cartonașelor.

Numărul de cartonașe aflate pe ecran este determinat în mod dinamic în funcție de lățimea ferestrei. Drept urmare, se pot afișa pe un rând cinci, patru, trei sau două cartonașe, pornind de la dimensiunea de *ecran complet* până la o lățime de  $\frac{1}{2}$ .

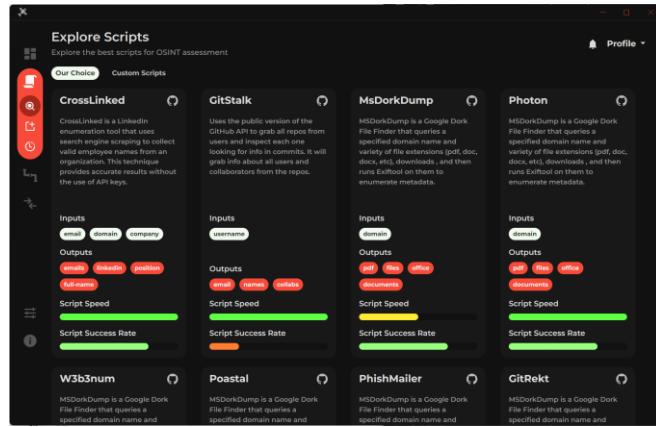


Figura 5.12 Pagina de explorare a script-urilor

Fiecare script are o pagină generată dinamic conform datelor inserate în formularul de importare al script-urilor. Pe pagină se află detalii generale despre script, dar și elemente de tip *input* pentru a completa intrările necesare din script. Unele argumente au nevoie de un parametru inserabil de utilizator, cele care au nevoie doar de introducerea unui simbol, cum ar fi introducerea flag-ului -v pentru verbozitate, sunt ilustrate sub formă de *checkbox*.

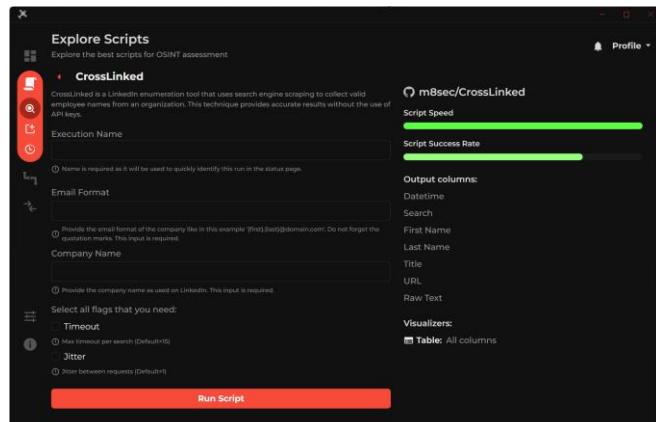


Figura 5.13 Pagina generată dinamic a unui script

Butonul de rulare al script-ului declanșează un proces complex în care parametrii introdusi pe pagină și detaliile introduse în formularul de import sunt transmiși prin protocolul IPC în serviciul de back-end de unde se rulează script-ul conform datelor introduse. După rulare, se aplică regula de RegEx introdusă și se face separarea datelor pe coloane și rânduri după care datele rezultate sunt trimise prin IPC către o componentă care ascultă continuu mesajele de pe acel canal. Când un mesaj este transmis, acesta îl procesează și îl adaugă în lista de script-uri rulate. Totodată, această listă este salvată într-un fișier JSON, conținând detalii precum: numele script-ului, numele execuției, data și ora pornirii script-ului, data și ora finalizării script-ului, status și datele de ieșire ale script-ului procesate.

```

const scriptDir = path.dirname(scriptPath);
process.chdir(scriptDir);

if (fs.existsSync(outputFilePath)) {
  fs.unlinkSync(outputFilePath);
}

const commandArgs: string[] = [];
args.forEach(arg: { name: string; value?: string }) => {
  if (arg.name) {
    commandArgs.push(arg.name);
  }
  if (arg.value) {
    commandArgs.push(arg.value);
  }
});

const command = [scriptPath, ...commandArgs];
console.log(`running command: ${command}`);

let scriptProcess;

if (useStdout) {
  scriptProcess = spawn(scriptExecutable, command, {
    cwd: scriptDir,
    stdio: ['inherit', 'pipe', 'ignore'],
  });
  const outputStream = fs.createWriteStream(outputFilePath);
  scriptProcess.stdout.pipe(outputStream);
} else {
  scriptProcess = spawn(scriptExecutable, command, { cwd: scriptDir });
}

scriptProcess.stdout.on('data', (data) => {
  console.log(`stdout from script: ${data}`);
});

```

Figura 5.14 Rularea unui script în back-end

```

const handleScriptStatusUpdate = async (scriptData: any) => {
  setData(prevData) => {
    const existingScript = prevData.find(
      (item) => item.startTime === scriptData.startTime
    );

    if (existingScript) {
      const updatedData = prevData.map((item) => {
        if (item.startTime === scriptData.startTime) {
          return {
            ...item,
            status: scriptData.isRunning ? 'Running' : 'Completed',
            endTime: scriptData.endTime,
            output: scriptData.output,
            outputColumns: scriptData.outputColumns,
          };
        }
        return item;
      });
      if (!scriptData.isRunning) {
        writeDataToFile(JSON.stringify(updatedData));
      }
      return updatedData;
    } else {
      return [
        ...prevData,
        {
          executionName: scriptData.executionName,
          scriptName: scriptData.scriptName,
          startTime: scriptData.startTime,
          endTime: scriptData.endTime,
          status: scriptData.isRunning ? 'Running' : 'Completed',
          output: scriptData.output,
          outputColumns: scriptData.outputColumns,
        },
      ];
    }
  );
};

const loadDataFromFile = async () => {
  const existsfile = await ipcRenderer.invoke('fs-exists-sync', {
    fileName: FILENAME,
  });
  if (existsfile) {
    const fileContent = await ipcRenderer.invoke('fs-readfile-sync', {
      fileName: FILENAME,
    });
    const parseddata = JSON.parse(fileContent);
    setData(parseddata);
  }
};

loadDataFromFile();
ipcRenderer.on('scripts-status', handleScriptStatusUpdate);

```

Figura 5.15 Mecanismul de actualizarea a listei de script-uri rulate

Pagina de explorare a script-urilor este identică cu pagina de explorare a scenariilor. Singura diferență provenind din fișierul din care se extrag date.

## 5.6 Scripts Status/Scenario Status

Script-urile rulate sau în status de rulare sunt preluate din variabila de tip *state* actualizată activ de către componenta introdusă în secțiunea anterioară. Astfel, pagina de status afișează un tabel, creat prin folosirea unei componente din EJ2, populat cu datele script-urilor rulate.

The screenshot shows a software window titled 'Scripts Status'. At the top, there's a toolbar with icons for search, refresh, and export. Below the toolbar is a table with the following columns: Execution Name, Script Name, Start Time, End Time, Status, and Output. The table contains several rows of data, including 'Exec 1' and 'Exec 2' which are completed, and various runs and tests. The 'Output' column for each row has a link labeled 'View Output'. At the bottom of the table, there are navigation buttons for pages 1 through 5, and a note indicating '2 of 5 pages (50 items)'.

Execution Name	Script Name	Start Time	End Time	Status	Output
Exec 1	MoDxDump	22.06.2023, 01:59:25	22.06.2023, 02:02:11	Completed	
Exec 2	MoDxDump	22.06.2023, 01:55:52	22.06.2023, 01:55:59	Completed	<a href="#">View Output</a>
Run 1	CrossLinked	22.06.2023, 01:50:03	22.06.2023, 01:50:20	Completed	<a href="#">View Output</a>
Run 2	CrossLinked	22.06.2023, 01:47:35	22.06.2023, 01:47:52	Completed	
Run 3	CrossLinked	22.06.2023, 01:46:39	22.06.2023, 01:46:55	Completed	<a href="#">View Output</a>
Test	CrossLinked	22.06.2023, 01:41:15	22.06.2023, 01:41:32	Completed	
New Tests	CrossLinked	22.06.2023, 01:39:38	22.06.2023, 01:39:54	Completed	
Another Test	CrossLinked	22.06.2023, 01:38:48	22.06.2023, 01:39:08	Completed	
Demo	CrossLinked	22.06.2023, 01:36:02	22.06.2023, 01:36:19	Completed	

Figura 5.16 Pagina de status a script-urilor

Tabelul are funcționalități complexe cum ar fi ștergerea unui rând, căutarea unei celule, ordonare după coloană, paginare și posibilitatea de a selecta ce coloane vor fi afișate în tabel.

Fiecare intrare are o coloană denumită *Output*, unde se pot vizualiza datele obținute prin apăsarea acelei celule. Fiecare pagină conține graficele selectate în ultimul pas din formularul de importare al script-urilor, populate cu datele procesate de script. Tabelele au funcții descrise mai sus, iar în plus există funcționalitatea de exportare ca fișier PDF sau CSV.

This screenshot shows the same 'Scripts Status' interface as Figure 5.16, but the table is populated with data from a search for 'N + CrossLinked'. The table has columns: Datetime, Search, First Name, Last Name, Title, URL, and Raw Text. The data consists of 403 items across 41 pages, showing various LinkedIn profiles for people named Stelian, Adrian, Cristi, Radu, etc., with their titles like Senior Product Manager, Software Engineer, etc., and URLs to their LinkedIn profiles.

Datetime	Search	First Name	Last Name	Title	URL	Raw Text
06-22-2023 02:40:36	google	Stelian	Crisan	Senior Product Mana...	<a href="#">https://ro.linkedin.com...</a>	Stelian Crisan - Senio...
06-22-2023 02:40:36	google	Adrian	Aned	Senior Cloud Softwar...	<a href="#">https://ro.linkedin.com...</a>	Adrian Aned - Senio...
06-22-2023 02:40:36	google	Adrian	Lungu	Computer Scientist	<a href="#">https://ro.linkedin.com...</a>	Adrian Lungu - Comp...
06-22-2023 02:40:36	google	Cris	Radu	N/A	<a href="#">https://ro.linkedin.com...</a>	Cris Radu
06-22-2023 02:40:36	google	Cristina	Coman	University Talent S... Software Engineer	<a href="#">https://ro.linkedin.com...</a>	Cristina Coman - Univ...
06-22-2023 02:40:36	google	Raluca	Tanase	Senior Product Analyst	<a href="#">https://ro.linkedin.com...</a>	Raluca Tanase - Seni...
06-22-2023 02:40:36	google	Paul	Paul	Alexandru Chirita	<a href="#">https://ro.linkedin.com...</a>	Paul - Alexandru Chir...
06-22-2023 02:40:36	google	Evelina	Dumitrescu	Software Engineer	<a href="#">https://ro.linkedin.com...</a>	Evelina Dumitrescu - ...
06-22-2023 02:40:36	google	Andrei	Dragomir	Principal Scientifi...	<a href="#">https://ro.linkedin.com...</a>	Andrei Dragomir - Pri...
06-22-2023 02:40:36	google	Alexandru	Necula	Software Engineer	<a href="#">https://ro.linkedin.com...</a>	Alexandru Necula - S...

Figura 5.17 Tabel populat cu datele de ieșire al unui script

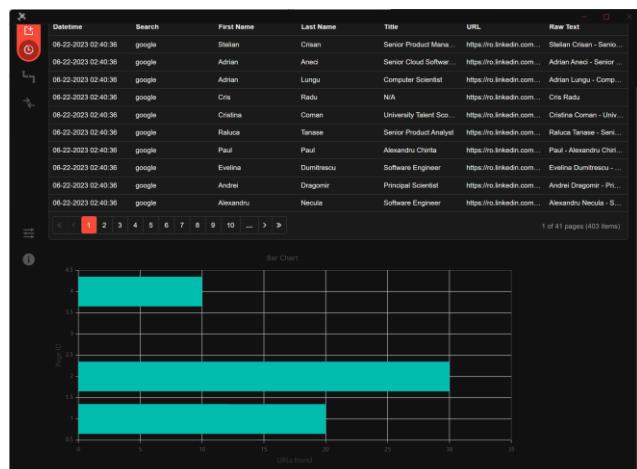


Figura 5.18 Exemplu Bar Chart populat cu date

Pagina de vizualizare a script-urilor finalizate este identică cu pagina de vizualizare a scenariilor finalizate. Singura diferență este că se folosește un fișier separat pentru fiecare pagină.

## 5.7 Create Scenario

Asemeni paginii *Import Script*, această pagină are rolul de crea scenarii pe baza script-urilor existente în aplicație. Pentru crearea unui scenariu utilizatorul trebuie să completeze un formular compus din patru pași.

Primul pas adresează datele sumarizate despre scenariu care pot fi vizualizate pe pagina *Explore Scenarios*. Utilizatorul trebuie să completeze următoarele date: nume, descriere, viteză, rata de succes, intrări și ieșiri. Această pagină este similară cu prima pagină din formular de importare a unui script, doar că în acest formular nu trebuie introdus un URL.

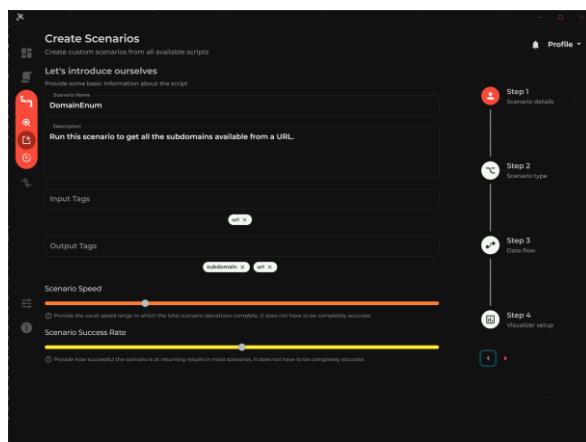


Figura 5.19 Primul pas pentru crearea unui scenariu

În al doilea pas se specifică script-urile care vor face parte din scenariu și tipul scenariului creat. Există două tipuri de scenarii pe care utilizatorul le poate alege: *agregare* și *înlănțuire*.

Agregarea presupune rularea mai multor script-uri în paralel în mod independent, după care se izolează și se grupează rezultatele obținute într-un singur rezultat. Acest scenariu este util dacă există mai multe script-uri din care rezultă același tip de date, spre exemplu pot exista două script-uri care identifică emailurile angajaților unei companii, iar aceste rezultate pot fi înglobate într-o singură listă folosind scenariul de tip *agregare*.

Înlănțuirea aranjează execuția script-urilor sub forma unei liste înlănțuite unde ieșirea unui script este intrarea următorului script. Acest scenariu poate fi folosit dacă există script-uri care pot genera rezultate pe baza rezultatelor altor script-uri, spre exemplu dacă un script generează email-uri și alt script verifică dacă există parole asociate pentru un email, putem înlănțui aceste două script-uri pentru a extrage date cât mai precise.

În funcție de tipul scenariului, următorii pași vor fi diferiți pentru a reflecta tipologia scenariului.

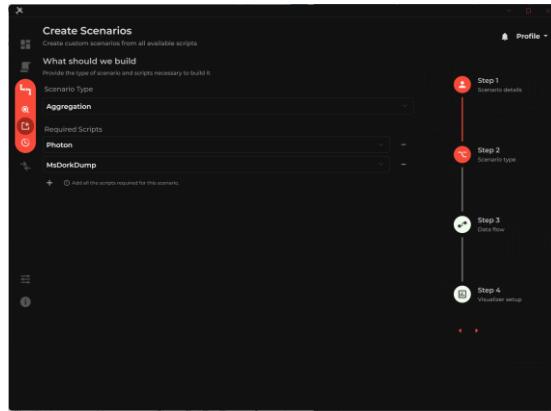


Figura 5.20 Al doilea pas pentru crearea unui scenariu

Al treilea pas este diferit în funcție de scenariul ales, mai exact al treilea pas adresează intrările și ieșirile scenariului, iar gestionarea acestora diferă datorită ierarhiei script-urilor generată de scenariu.

În cazul agregării, se specifică dacă există intrări sau ieșiri care sunt echivalente după care se selectează toate coloanele de ieșire care sunt de interes pentru acest scenariu. Funcția de echivalentă pentru intrări și ieșiri oferă utilizatorului o experiență mai bună, deoarece acesta nu o să fie nevoie să introducă aceleași date de două ori.

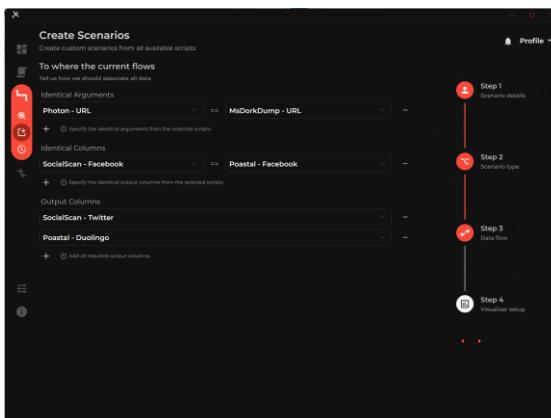


Figura 5.21 Al treilea pas pentru crearea unui scenario de tip *aggregare*

În schimb, în cazul înlățuirii se specifică ce coloane de ieșire vor fi folosite ca intrări pentru următorul script. Prin selectarea acestor parametri aplicația va putea înlățui automat toate script-urile fără să fie nevoie de intervenția utilizatorului.

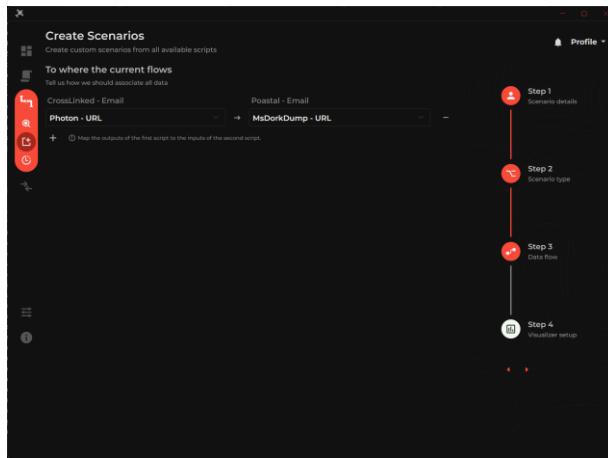


Figura 5.22 Al treilea pas pentru crearea unui scenario de tip *înlănțuire*

Ultimul pas presupune selectarea tuturor graficelor necesare și a coloanelor aferente. În cazul scenariilor, nu se va mai selecta fișierul de ieșire, deoarece acesta sau acestea vor fi preluate direct din configurațiile script-urilor.

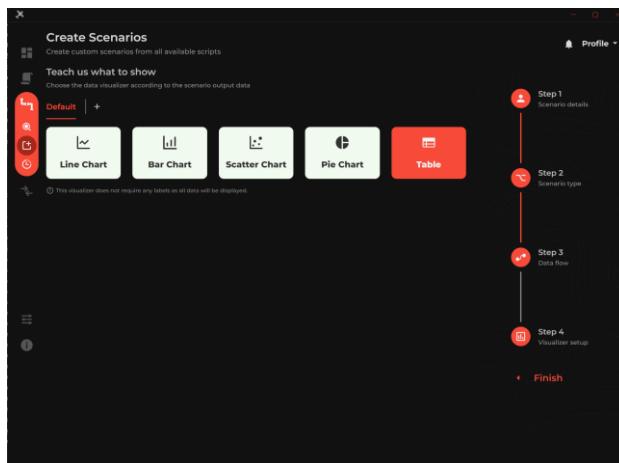


Figura 5.23 Al patrulea pas pentru crearea unui scenariu

După acest pas, se va actualiza fișierul JSON care conține datele de configurație ale scenariilor create de utilizator cu datele completate în acest formular.

## 5.8 Import/Export

În această pagină, utilizatorul poate selecta dacă dorește să importeze o arhivă cu fișiere de configurație sau dacă dorește să creeze o arhivă cu fișiere de configurație folosind funcția de exportare.

Exportarea presupune preluarea, criptarea și arhivarea tuturor fișierelor de tip JSON, în afară de cele care conțin rezultatele script-urilor sau ale scenariilor. Criptarea este făcută pentru fiecare fișier în parte folosind algoritmul de criptare AES-256<sup>13</sup> la care se adaugă o parolă introdusă de utilizator pentru generarea conținutului criptat. Tot în acest pas se selectează

<sup>13</sup> <https://www.ipswitch.com/blog/use-aes-256-encryption-secure-data>

locatia și numele arhivei folosind o funcționalitate pusă la dispoziție de Electron în procesul din back-end. Prin criptare se asigură că datele rămân confidențiale.

Funcționalitatea de importare reconstituie pașii exportării în sens invers. Utilizatorul selectează arhiva și introduce parola asociată arhivei, după care se dezarchivează și decriptează toate fișierele. Fiecare fișier din aplicație va fi suprascris de fișierele din arhivă.

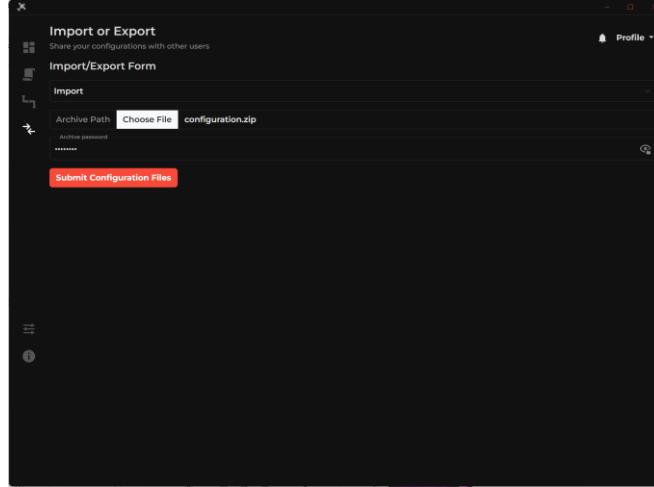


Figura 5.24 Pagina de importare sau exportare a fișierelor de configurare

```
try {
    // Extract the encrypted archive
    await fs
        .createReadStream(archivePath)
        .pipe(unzipper.Extract({ path: tempDir }))
        .promise();

    // Get the list of extracted files
    const files = fs.readdirSync(tempDir);

    // Decrypt and move each file to the data folder
    files.forEach(file) => {
        const filePath = join(tempDir, file);

        // Read the encrypted file content
        const encryptedContent = fs.readFileSync(filePath);

        // Decrypt the file content
        const algorithm = 'aes-256-cbc';
        const key = crypto.scryptSync(password, '', 32);
        const iv = encryptedContent.subarray(0, 16); // Extract IV from the encrypted content
        const decipher = crypto.createDecipher(algorithm, key, iv);
        const decryptedContent = Buffer.concat([
            decipher.update(encryptedContent.subarray(16)),
            decipher.final(),
        ]);

        // Write the decrypted content to the data folder
        const destinationPath = join(
            __dirname,
            './src/data',
            path.basename(file)
        );

        // Delete the existing file if it exists
        if (fs.existsSync(destinationPath)) {
            fs.unlinkSync(destinationPath);
        }

        fs.writeFileSync(destinationPath, decryptedContent);

        // Remove the temporary file
        fs.unlinkSync(filePath);
    };

    // Remove the temporary directory
    fs.rmdirSync(tempDir);

    return 'files imported successfully.';
} catch (error) {
    // Remove the temporary directory in case of error
    fs.rmdirSync(tempDir);

    return 'Error importing files. Invalid password or invalid archive format.';
}
```

Figura 5.25 Mecanismul de importare al fișierelor de configurare

```

const output = fs.createWriteStream(exportPath);
const archive = archiver('zip', {
  zlib: { level: 9 },
});
archive.pipe(output);
const dataFolderPath = join(__dirname, '../../src/data');
// Read the content of the data folder
const files = fs.readdirSync(dataFolderPath);
// Encrypt and add each file to the archive
files.forEach((file) => {
  if (
    file === 'scripts-status.json' ||
    file === 'scenarios-status.json' ||
    file === 'scenarios.json' ||
    file === 'scripts.json'
  ) {
    // skip encrypting and adding scripts-status.json
    return;
  }
  const filePath = join(dataFolderPath, file);
  const fileContent = fs.readFileSync(filePath);

  // Generate a random initialization vector
  const iv = crypto.randomBytes(16);

  // Encrypt the file content
  const algorithm = 'aes-256-cbc';
  const key = crypto.scryptSync(password, '', 32);
  const cipher = crypto.createCipheriv(algorithm, key, iv);
  const encryptedContent = buffer.concat([
    iv, // Add the initialization vector to the encrypted content
    cipher.update(fileContent),
    cipher.final(),
  ]);

  // Add the encrypted file to the archive
  archive.append(encryptedContent, { name: file });
});
archive.finalize();

```

Figura 5.26 Mecanismul de exportare al fișierelor de configurare

## 5.9 Setări

Pagina de setări conține mai multe funcționalități: sincronizarea datelor în cloud, schimbarea parolei utilizatorului, ștergerea contului și selectarea paletelor de culori a aplicației.

Datele complete ale utilizatorului pot fi sincronizate în Firestore Storage[24], permitându-i să aibă o soluție de back-up sau să își descarce datele pe orice dispozitiv. Fiecare utilizator are un folder dedicat în care se află o arhivă cu toate fișierele JSON din aplicație. Folder-ul este generat pe baza identificatorului unic al contului, identificator care este generat de Firebase la autentificare. La fel ca în cazul importării sau exportării, fișierele din arhivă sunt criptate cu algoritmul AES-256. Cheia folosită pentru criptare este salvată în spațiul privat de stocare al datelor din sistemul de operare. Fiecare sistem de operare dispune de un utilitar care poate stoca date private în mod securizat, spre exemplu pe Windows stocarea acestor date se face folosind modulul TPM<sup>14</sup>. Accesul la aceste utilitare se face folosind API-ul Electron care gestionează automat fiecare utilitar asociat fiecărui sistem de operare.

Sincronizarea cu platforma cloud se face în mod voit prin apăsarea butoanelor aferente de descărcare și încărcare a datelor. Am preferat ca sistemul să nu facă back-up automat al datelor, deoarece există utilizatori care vor să își păstreze datele doar pe propriul dispozitiv.

De asemenea, utilizatorul poate să își șteargă contul sau să își schimbe parola, în cazul în care folosește sistemul de autentificare prin email și parolă. Aceste scenarii sunt gestionate de Firebase ca la autentificare, fiind nevoie doar să se introducă datele corecte pentru fiecare scenariu.

---

<sup>14</sup> <https://support.microsoft.com/ro-ro/topic/ce-este-tpm-705f241d-025d-4470-80c5-4feeb24fa1ee>

În final utilizatorul poate selecta tema dorită pentru aplicație, mai exact o temă întunecată cu o culoare de accent roșie, o temă cu fundal alb și o culoare de accent rozalie sau o temă cu fundal alb-gălbui și o culoare de accent maronie. Aplicația inițial preselectează tema întunecată ca fiind standard.

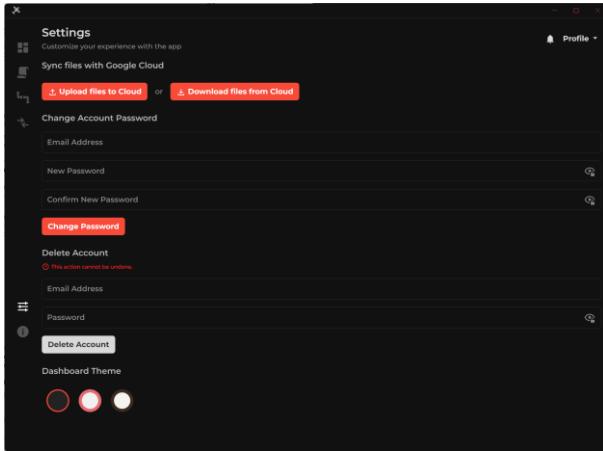


Figura 5.27 Pagina de setări

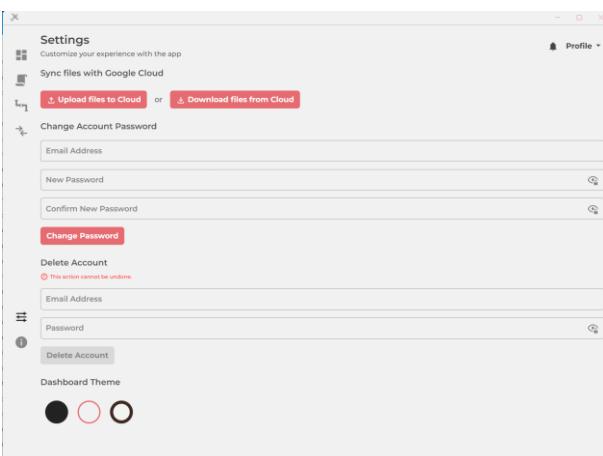


Figura 5.28 Valiant temă cu fundal alb și culoare de accent rozalie

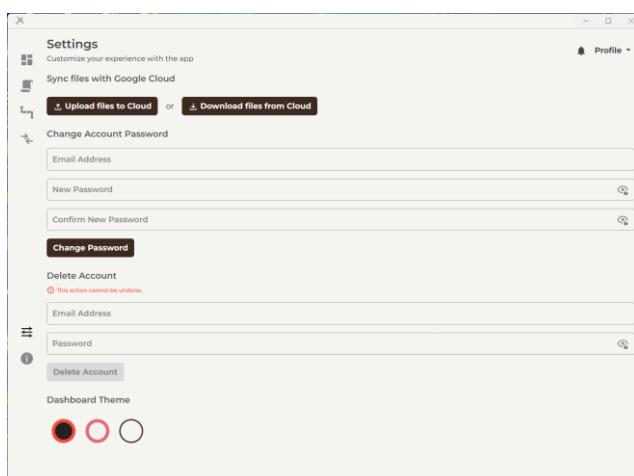


Figura 5.29 Valiant team cu fundal alb-gălbui și culoare de accent maronie

## 5.10 About

Aplicația conține o pagină de informații unde sunt specificate autorii tuturor resurselor preluate din surse externe. De asemenea, se specifică că aplicația este distribuită sub licența *GPL v3*<sup>15</sup>, deoarece unele script-uri cu care aplicația este împachetată sunt distribuite sub această licență care obligă utilizatorii să folosească și ei la rândul lor aceeași licență. Scripturile care nu folosesc această licență sunt distribuite sub formatul licenței *MIT*<sup>16</sup>.

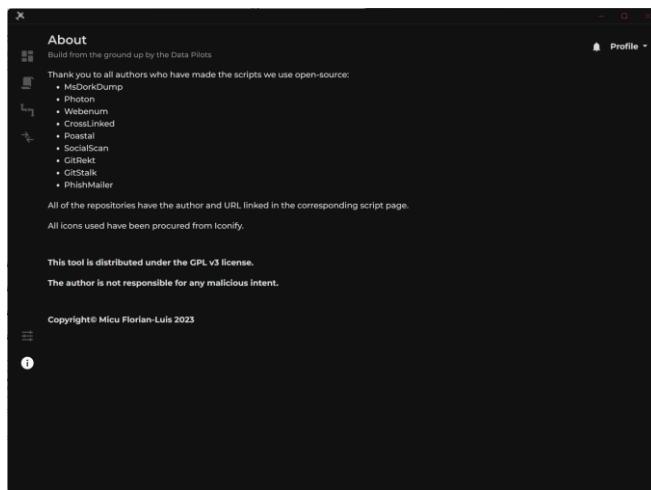


Figura 5.30 Pagina de informații

## 5.11 Script-uri preconfigurate

Aplicația are configurate mai multe script-uri care acoperă scenarii de analiză a rețelelor sociale, a site-urilor, a depozitelor de cod de pe GitHub și un ultimul rând există și o categorie de script-uri care testează scenariile de *phishing*<sup>17</sup>. Toate scripturile provin de pe platforma GitHub.

Analiza datelor de pe rețelele sociale este făcută cu scripturile *CrossLinked*[25], *Poastal*[26], *SocialScan*[27] și *have-i-been-pwned*[34].

*CrossLinked* este folosit pentru a afla detalii despre angajații unei companii: nume, prenume, poziție, pagină de LinkedIn și email. Script-ul face căutări pe platforma Google, Bing și LinkedIn folosind un format de email și numele companiei aprovizionate de utilizator.

<sup>15</sup> <https://www.gnu.org/licenses/gpl-3.0.html>

<sup>16</sup> <https://opensource.org/license/mit/>

<sup>17</sup> <https://www.microsoft.com/ro-ro/security/business/security-101/what-is-phishing>

Datetime	Search	First Name	Last Name	Title	URL	Raw Text	Email
06-22-2023 02:40:36	google	Emilia	U	EMEA Talent Acq...	https://ln.linkedin.c...	Emilia U - EMEA...	eu@adobe.com
06-22-2023 02:40:36	google	Carmen	Carmen	Luis Ghile	https://ln.linkedin.c...	Carmen Luis Ghile	cramen@adobe.c...
06-22-2023 02:40:36	google	Mihaela	Daniel	Senior Engineer...	https://ln.linkedin.c...	Mihaela Daniel - Se...	mdaniel@adobe.c...
06-22-2023 02:40:36	google	Cristina	Ionut	International Acco...	https://ln.linkedin.c...	Cristina Ionut - Int...	coionut@adobe.com
06-22-2023 02:40:36	google	Anca	Buratu	HR	https://ln.linkedin.c...	Anca Buratu - HR	aburatu@adobe.c...
06-22-2023 02:40:36	google	Ilinca	(Bordescu)	Customer Success...	https://ln.linkedin.c...	Ilinca Bordescu - CS...	ilinca.bordescu@adob...
06-22-2023 02:40:36	google	Andreea	Racovita	Software Develop...	https://ln.linkedin.c...	Andreea Racovita - SW...	aracovita@adobe.c...
06-22-2023 02:40:36	google	Razvan	Bratescu	Group Engineering...	https://ln.linkedin.c...	Razvan Bratescu - GE...	rbratescu@adobe.c...
06-22-2023 02:40:36	google	Ioan	Pop	Engineering Mana...	https://ln.linkedin.c...	Ioan Pop - Engine...	iopop@adobe.com
06-22-2023 02:40:36	google	Alexandru	Ivana	Product Manager	https://ln.linkedin.c...	Alexandru Ivana -	aviana@adobe.com

Figura 5.31 Rezultatul script-ului CrossLinked

Poastal verifică ce conturi sunt asociate unui email aprovisionat de utilizator. Verificarea se face prin interogarea disponibilității creării unui nou cont folosind adresa de email dată. Platformele sociale verificate sunt: Facebook, Twitter, Snapchat, Rumble, MeWe, Imgur, Adobe, Wordpress, Duolingo, Hulu, Rubmaps, GitHub, Gravatar și Discord.

User	Email	Face...	Twitter	Snap...	Rum...	MeWe	Imgur	Adobe	Wor...	Duol...	Hulu	Rub...	GitHub	Grav...
miculescu1	true	true	true	false	false	true	true	true	false	true	false	false	true	false

Figura 5.32 Rezultatul script-ului Poastal

SocialScan verifică aparteneța unui email sau nume de utilizator pe anumite platforme online. Platformele verificate sunt Firefox, GitHub, Pinterest, Tumblr, Twitter, Instagram și LastFM. Acest script folosește alte tehnici de analiză a conturilor și poate fi folosit împreună cu script-ul precedent pentru a valida anumite conturi.

User	Firefox	Github	Pinterest	Tumblr	Twitter	Instagram	LastFM
miculescu1	true	true	false	true	true	true	false

Figura 5.33 Rezultatul script-ului SocialScan

Have-I-Been-Pwned reprezintă o bază de date cu toate pierderile de date din urma unui atac cibernetic. Se pot face cereri HTTP către acest serviciu folosind API-ul disponibil, contra cost. Utilizatorul trebuie să introducă cheia generată de serviciu în urma achiziției unei licențe și adresa mail pe care dorește să o verifice.

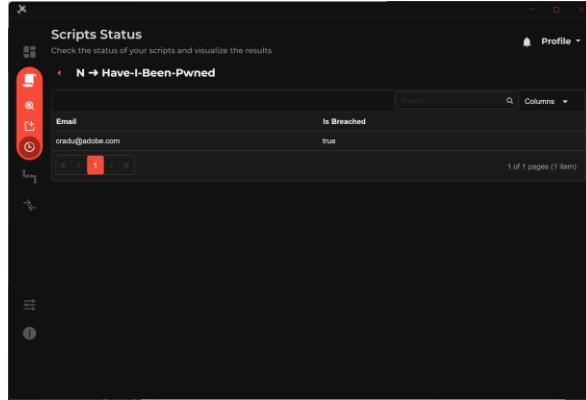


Figura 5.34 Rezultatul script-ului Have-I-Been-Pwned

Analiza site-urilor este realizată cu ajutorul script-urilor *MsDorkDump*[28], *Photon*[29] și *Webenum*[30].

*MsDorkDump* extrage documente de tip PDF, Office sau CSV dintr-un URL folosind un *crawler web*. Utilizatorul trebuie să introducă domeniul pe care dorește să îl analizeze, iar pentru rezultate mai bune se pot analiza separat și subdomenii.

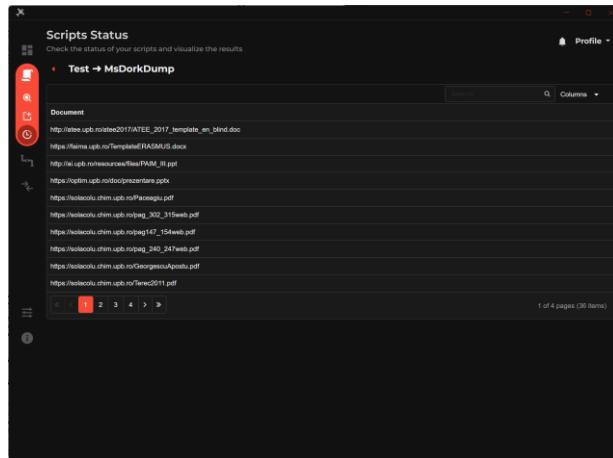


Figura 5.35 Rezultatul script-ului MsDorkDump

*Photon* cauță toate tipurile de link-uri și imagini disponibile pe domeniul aprovisionat de utilizator. Acest script poate fi configurat să ofere și date despre DNS prin aprovisionarea unui flag.

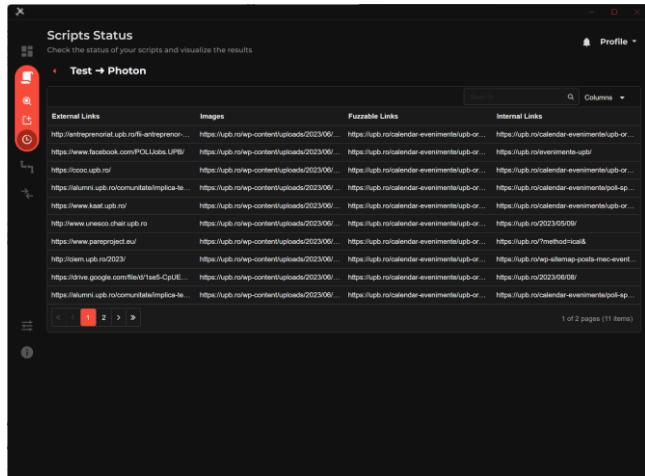


Figura 5.36 Rezultatul script-ului Photon

Webenum este un script care interoghează subdomenii, certificate, proprietăți de tipul *whois* și DNS. Analiza se face pe baza unui URL, datele rezultate fiind suficiente pentru crearea unui istoric sumar al site-ului.

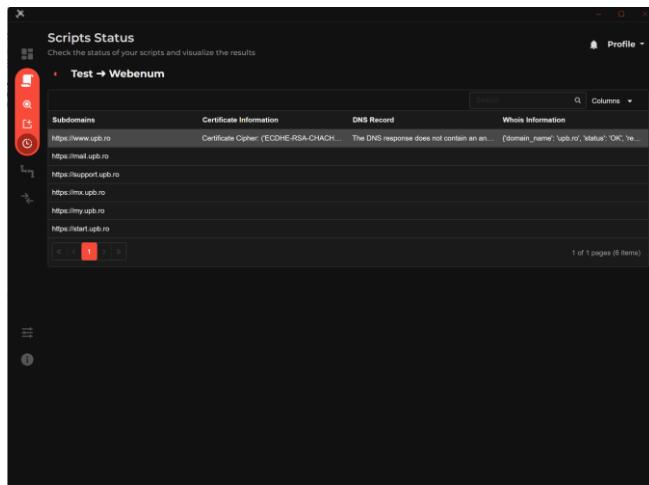


Figura 5.37 Rezultatul script-ului Webenum

Repository-urile de GitHub sunt analizate folosind script-urile *GitRekt*[31] și *GitStalk*[32].

*GitRekt* analizează folder-ul ascuns *.git* al unui site și extrage emailuri și URL-uri din acel folder. Majoritatea site-urilor folosesc un sistem de lansare integrat cu GitHub, uneori folder-ul *.git* nefiind ignorat sau șters la lansare. Utilizatorul trebuie să aprovizioneze numele site-ului pe care dorește să îl analizeze.

Scripts Status	
Check the status of your scripts and visualize the results	
Test → GitRekt	
Emails	URLs
apopecusa@gmail.com misiulea@gmail.com andrei82@gmail.com floremit99@gmail.com anamariaes@gmail.com elenabogdan@gmail.com moldovenescristina@gmail.com marian00@gmail.com cosminlesand@gmail.com forcamadelein97@gmail.com	https://www.instagram.com/upd1816/ https://upd1.ro/contul-de-administrator/ https://upd1.ro/conturi/ https://upd1.ro/nominal/ https://upd1.ro/nu-aveam-cont-updates/2022/01/file.jpg https://upd1.ro/noutatile-publice/ https://upd1.ro/executivul/facultatea-de-inginerie-medica/ https://upd1.ro/nouvorase profesionala/ https://posturamail.ro/pdf/jar/ https://upd1.ro/excelente-in-rezurse-umane-de-cercuri/
< < 1 2 > >	
1 of 2 pages (11 items)	

Figura 5.38 Rezultatul script-ului GitRekt

GitStalk analizează toate repository-urile publice ale unui utilizator pentru adrese de email. Pentru a folosi acest script este nevoie ca utilizatorul să introducă identificatorul unic al utilizatorului în câmpul aferent.

Scripts Status	
Check the status of your scripts and visualize the results	
N → GitStalk	
Full Name	Email
GfHub lued156 Luís Miu Florian Luis Miu Florian Luis Miu	noreply@github.com misiulea@gmail.com misiulea@gmail.com misiulea@gmail.com fmisu@192.168.0.104.csene.ro
< < 1 > >	
1 of 1 pages (5 items)	

Figura 5.39 Rezultatul script-ului GitStalk

Scenariul de phishing este testat folosind script-ul *PhishMailer*[33], care generează un template HTML care poate fi introdus în corpul unui email și un script care a fost creat de mine pentru a putea trimite mail-uri folosind protocolul *SMTP*.

Script-ul de trimitere a mail-urilor, numit *Bait*, permite introducerea datelor relevante pentru un scenariu de phishing, cum ar fi: subiect, mesaj, atașament, destinatari, port SMTP și server SMTP. Pentru ușurință, se poate folosi server-ul SMTP Google, disponibil gratuit, dar care necesită generarea unei chei speciale.

Aceste două script-uri au fost introduse pentru a crea un scenariu de phishing prin intermediul aplicației, rezultatul lor fiind analizat în secțiunea următoare sub formă de scenariu.

Anumite scenarii nu pot fi rulate continuu fără introducerea unui întârziere, deoarece se fac mai multe apeluri către anumite API-uri care vor fi limitate la un număr de cereri pe minut. În

acest caz, utilizatorul va trebui să ruleze script-ul în cauză din nou abia când se vor putea face din nou cereri.

## 5.12 Scenarii preconfigurate

Pe baza script-urilor deja configurate s-au putut crea anumite scenarii care expandează numărul de date obținute.

Script-ul CrossLinked a fost înlănțuit cu script-ul Have-I-Been-Pwned pentru a observa căte adrese de email ale unor angajați au făcut parte dintr-un atac cibernetic. Acest scenariu a fost denumit *CrossPwned* și necesită doar introducerea datelor pentru rularea primului script, dar și cheia privată a utilizatorului pentru serviciul Have-I-Been-Pwned.

Email	Is Breached
crstdu@adobe.com	true
fmcu@adobe.com	false
sgabor@adobe.com	false
opincar@adobe.com	false
ecalcari@adobe.com	false
cocoman@adobe.com	false
rstanescu@adobe.com	false
bgeorgescu@adobe.com	false
rrazvan@adobe.com	false
gnita@adobe.com	false

Figura 5.40 Rezultatul scenariului CrossPwned

De asemenea, script-ul CrossLinked și Poastal constituie un scenariu în care se verifică pe ce platforme mai sunt folosite email-urile angajaților. Scenariul se numește CrossPoastal și utilizatorul trebuie să introducă datele de intrare pentru primul script.

Email	Firefox	GitHub	Pinterest	Tumblr	Twitter	Instagram	LastFM
crstdu@adobe.com	false	true	false	false	false	false	false
fmcu@adobe.com	false	true	false	false	false	false	false
sgabor@adobe.com	false	true	false	false	false	false	false
opincar@adobe.c...	false	true	false	false	false	false	false
ecalcari@adobe.com	false	true	false	false	false	false	false
cocoman@adobe.com	false	true	false	false	false	false	false
rstanescu@adobe.com	false	true	false	false	false	false	false
bgeorgescu@adob...	false	true	false	false	false	false	false
rrazvan@adobe.com	false	true	false	false	false	false	false
gnita@adobe.com	false	true	false	false	false	false	false

Figura 5.41 Rezultatul scenariului CrossPoastal

Script-urile MsDorkDump, Photon și Webenum constituie un scenariu de tip agregare în care se pot afla cât mai multe date despre un site. Scenariul se numește FullSiteMap și necesită intrările specifice fiecărui script.

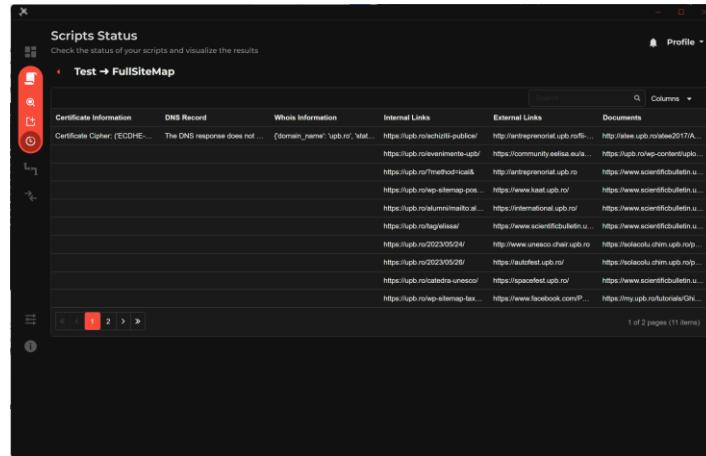


Figura 5.42 Rezultatul scenariului FullSiteMap

Ultimul scenariu disponibil conține script-urile CrossLinked, PhishMail și Bait. Mai întâi se găsesc toate adresele de mail ale angajaților, apoi se generează un template HTML de phishing, după care se trimit aceste mail-uri la adresele specificate. Acest scenariu este denumit CrossPhish și necesită doar intrările primului script.

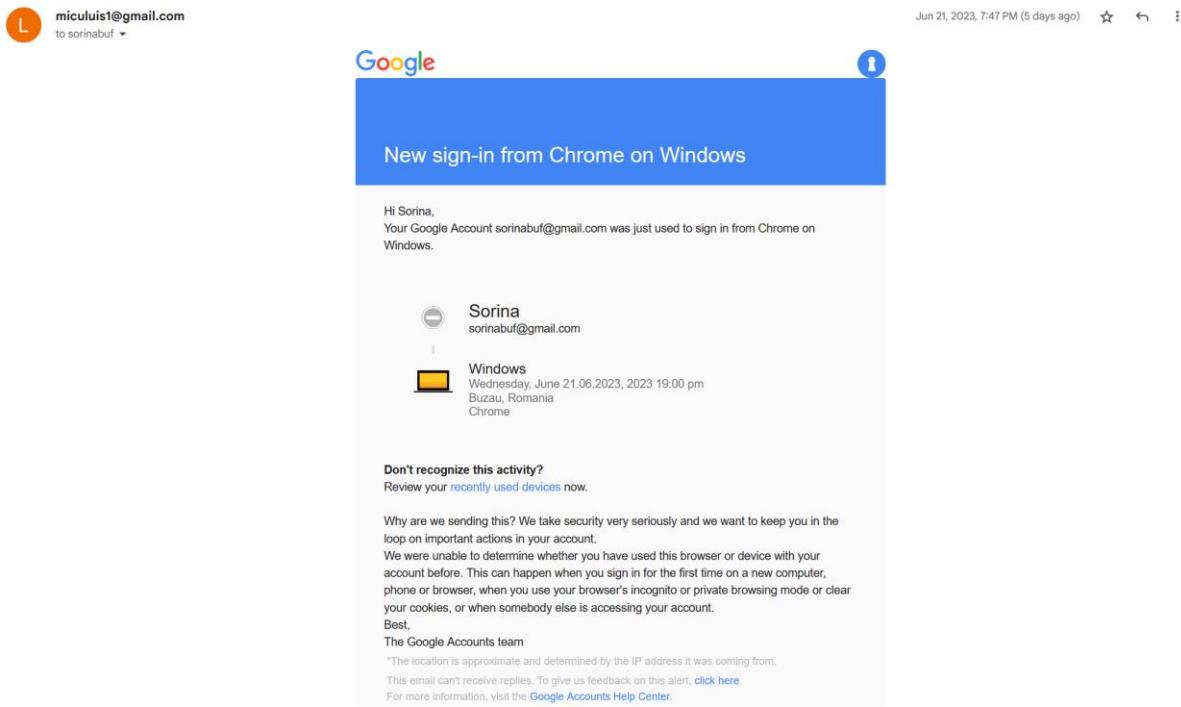


Figura 5.43 Rezultatul scenariului CrossPhish

## 6 STUDIU DE CAZ / EVALUAREA REZULTATELOR

Utilitatea aplicației poate fi demonstrată de rezultatele obținute în secțiunea anterioară, audierile fiind făcute pe organizațiile Adobe și Universitatea Politehnica București sau pe date personale în cazul audierii anumitor conturi sociale.

Mai mult, aplicația a fost testată pe Windows, Linux și macOS, comportamentul acesteia fiind identic, fără să existe probleme de compatibilitate.

Totodată pentru a confirma impactul pozitiv al aplicației, am creat un formular adresat unor ingineri din domeniul IT care are ca scop evaluarea concretă a performanței și experienței. Cuestionarul conține 11 întrebări și a fost completat de nouă persoane.

Cât de intuitivă vi se pare aplicația?

9 răspunsuri

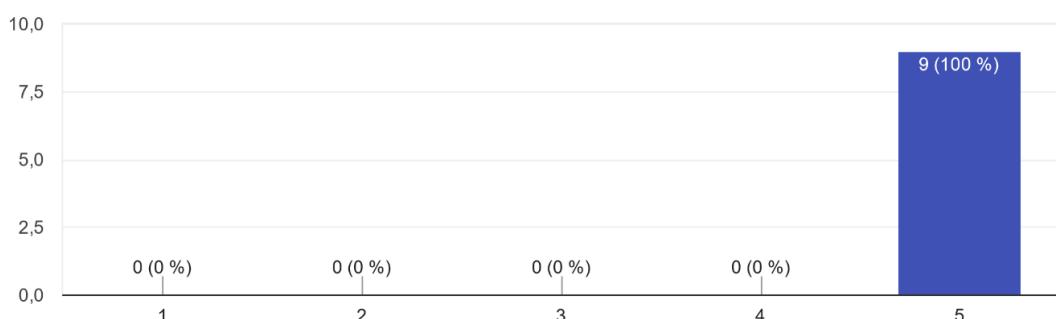


Figura 6.1 Întrebarea 1 – rezultate cuestionar

Prima întrebare marchează aprecierea utilizatorilor în ceea ce privește accentul pe UI și UX, 100% din utilizatori afirmând că aplicația este foarte intuitivă.

Cât de utile sunt script-urile preconfigurate?

9 răspunsuri

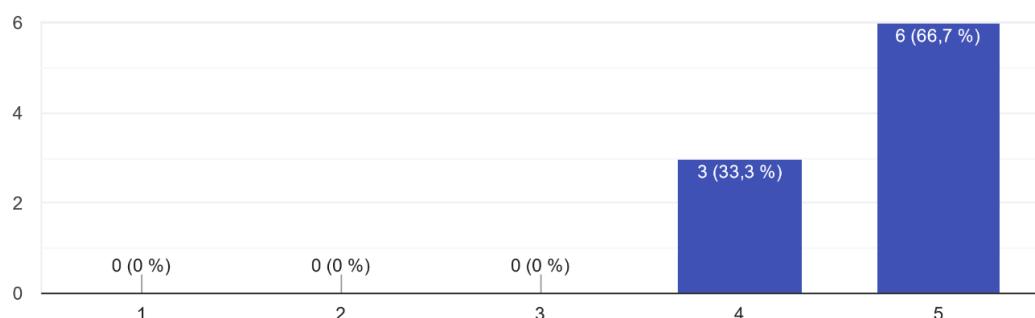


Figura 6.2 Întrebarea 2 – rezultate cuestionar

În a doua întrebare se observă gradul de mulțumire al utilizatorilor în ceea ce privește script-urile preconfigurate. Conform datelor din figură, 2/3 din utilizatori consideră script-urile foarte utile, restul utilizatorilor fiind satisfăcuți cu alegerile făcute. Pentru a crește gradul de mulțumire, în viitor s-ar putea introduce mai multe script-uri cu scop variat sau cu o rata de succes mai mare.

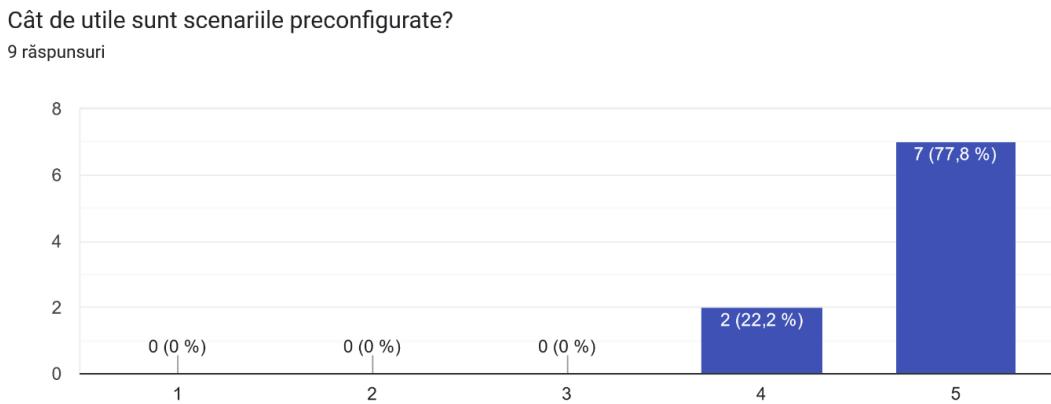


Figura 6.3 Întrebarea 3 – rezultate chestionar

În continuare, utilizatorii au apreciat mai mult selecția de scenarii preconfigurate. Acest rezultat poate fi datorat faptului că scenariile oferă în general o rată de succes mai mare pentru cazurile de agregare sau datorită faptului că un scenariu oferă date mult mai concrete și variate.

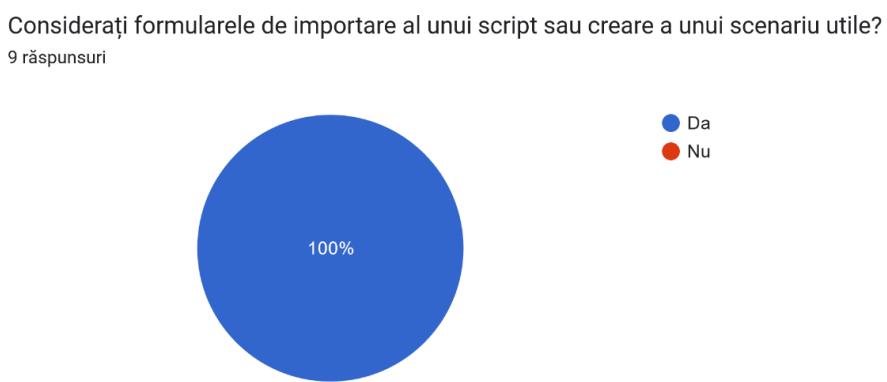


Figura 6.4 Întrebarea 4 – rezultate chestionar

Cât de intuitive sunt formularele de importare al unui script sau creare a unui scenariu?  
9 răspunsuri

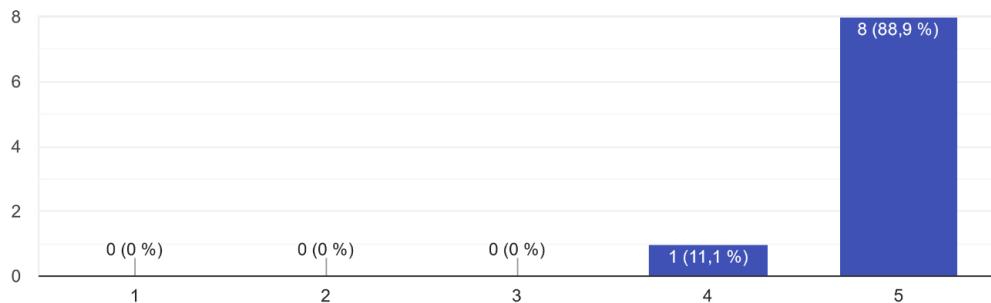


Figura 6.5 Întrebarea 5 - rezultate chestionar

Următoarele două întrebări adreseză utilitarele de definire a unor scenarii sau script-uri. Satisfacția utilizatorilor este foarte mare, marcând nevoie de versatilitate și personalizare a utilizatorilor.

Considerați funcționalitatea de sincronizare a datelor cu serviciul cloud utilă?  
9 răspunsuri

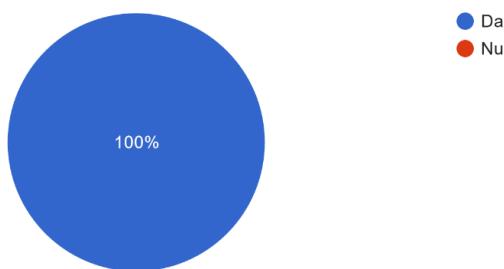


Figura 6.6 Întrebarea 6 – rezultate chestionar

Considerați funcționalitatea de împărtășire a fișierelor cu un coleg utilă?  
9 răspunsuri

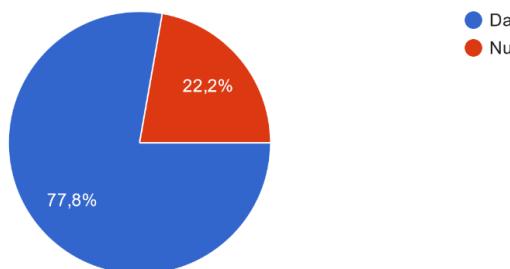


Figura 6.7 Întrebarea 7 – rezultate chestionar

Întrebările şase și şapte fac referire la mecanismele de partajare a fișierelor de configurare. Se poate observa din graficele de mai sus preferința utilizatorilor pentru mecanismul de sincronizare a datelor cu serviciul de cloud inclus. Rata scăzută de apreciere a mecanismului de împărtășire a fișierelor cu un coleg poate fi datorată faptului că aplicația nu a fost folosită într-un mediu colaborativ, astfel s-a favorizat munca individuală.

Ați reușit să extrageți date de interes folosind utilitarele disponibile?  
9 răspunsuri

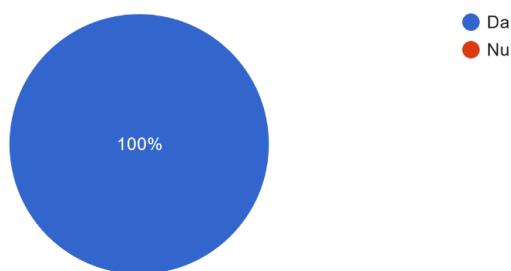


Figura 6.8 Întrebarea 8 – rezultate chestionar

Următoarea întrebare verifică rata de succes a script-urilor și scenariilor în ceea ce privește audierea datelor publice. Conform graficului, toți utilizatorii au reușit să extragă date de interes, astfel validând intenția aplicației de ajuta la audierea datelor publice cu caracter personal.

Cât de utile vi s-au părut graficele și tabelele disponibile?  
9 răspunsuri

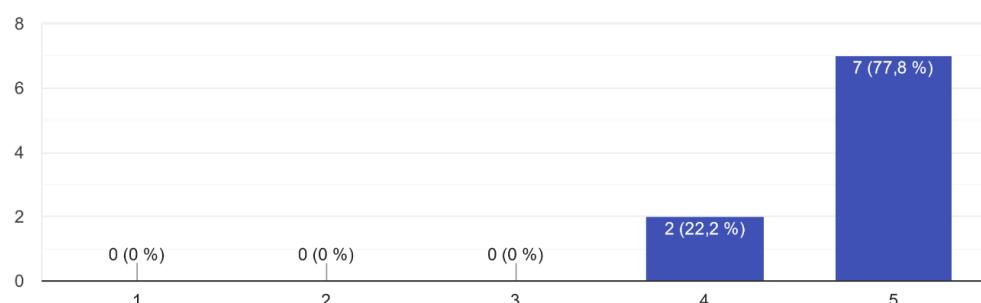


Figura 6.9 Întrebarea 9 – rezultate chestionar

Întrebarea nouă, verifică calitatea graficelor și a tabelelor disponibile, utilizatorii calificându-le ca fiind foarte bune în proporție de 80%, restul considerându-le satisfăcătoare. În viitor, se pot introduce mai multe funcții de compunere a coloanelor, cum ar fi *sum* sau *count distinct*, pentru a încuraja folosirea graficelor. În stadiul curent, execuția script-urilor de tip OSINT favorizează folosirea tabelelor, deoarece execuția acestora rezultă în generarea datelor de tip

șir de caractere. Drept urmare, prin introducerea mai multor funcții de compunere se pot introduce și coloane care reprezintă un număr, favorizând graficele.

Prefer să folosesc aplicația Valiant pentru folosirea script-urilor de tip OSINT. Sunteți de acord cu această afirmație?

9 răspunsuri

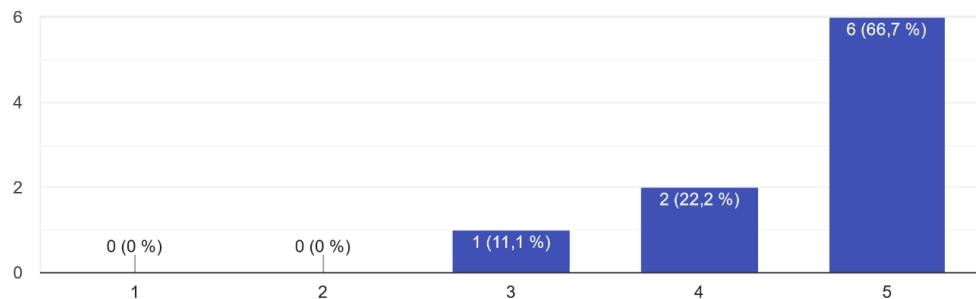


Figura 6.10 Întrebarea 10 – rezultate chestionar

Întrebarea zece verifică gradul total de satisfacție al utilizatorului față de aplicație, aproape 90% din utilizatori preferând să folosească aplicația Valiant față de soluțiile existente. Preferința pentru script-urile tradiționale poate proveni din simplitatea și rapiditatea folosirii unui singur script din linia de comandă. Însă, soluția creată excelează în cazul gestionării unui număr mai mare de script-uri, utilizatorul folosind doar elemente grafice pentru a lansa, vizualiza sau crea script-uri. Totuși pentru a asigura un grad de satisfacție mai mare, se pot introduce mai multe script-uri și scenarii standard, astfel reducând timpul de set-up necesar pentru proiecte care necesită script-uri diferite.

Există o funcționalitate care v-ar fi îmbunătățit experiența, dar care nu este prezentă în aplicație?

3 răspunsuri

Aș fi dorit să se poată folosi și scenarii atunci când se crează unul, nu doar script-uri.

Mi-ar fi placut să pot să adaug notite în rubrica de output a unui script

ar fi fost util să se poată importa direct în aplicație script-urile recomandate prin apasarea unui buton

Figura 6.11 Întrebarea 11 – rezultate chestionar

Ultima întrebare are rolul de a identifica posibilele surse de îmbunătățiri. Conform figurii de mai sus, utilizatorii au identificat trei posibile funcționalități noi.

Prima funcționalitate introduce posibilitatea de creare a unor scenarii mai complexe care se bazează pe alte scenarii. Această funcționalitate va putea fi implementată cu siguranță în viitor, singura dificultate provenind din management-ul scenariilor în timpul execuției.

A doua funcționalitate propune augmentarea procesului de documentare al datelor obținute prin introducerea unei rubrici text pe pagina de vizualizare a rezultatelor pentru fiecare script sau scenariu. Introducerea unei astfel de rubrici nu este dificilă, dar presupune o schimbare minimală în structura datelor JSON care sunt salvate în fișierul aferent rezultatelor obținute.

Ultima funcționalitate presupune autoconfigurarea unor script-uri pe baza recomandărilor făcute pe pagina *Home*. Această funcționalitate necesită un utilitar special, asemănător cu cel de importare al unui script, datorită parametrilor foarte diferiți de la un script la altul. Aplicația în stadiul curent, încurajează utilizatorul să își procure manual script-urile, execuția fiind orchestrată de către acesta prin completarea formularului de importare.

Considerând de asemenea specificațiile propuse de utilizatori în urma analizei cerințelor, aplicația Valiant reușește să satisfacă toate aceste nevoi, rezultatele încurajând folosirea soluției create pentru gestionarea script-urilor OSINT. Experiența utilizatorilor a fost pozitivă, demonstrând utilitatea unei aplicații de acest tip în domeniul securității cibernetice.

## **7 CONCLUZII**

În această lucrare s-a prezentat procesul de dezvoltare a unei aplicații care are ca scop gestionarea și recomandarea script-urilor de tip OSINT pentru a facilita experiența utilizatorilor cu domeniul OSINT indiferent de experiența acestora.

Pentru început, am creat un formular adresat inginerilor din domeniul IT care a semnalat nevoia de creare a unei aplicații de tip dashboard care să poată simplifica procesul de analiză a vulnerabilității datelor publice prin oferirea unor script-uri standard. De asemenea, participanții au fost interesați de posibilitatea de a crea scenarii din mai multe script-uri, aceste scenarii fiind orchestrate de către aplicație. Totodată, în formular au fost prezentate și alte funcționalități care au fost apreciate de către participanți cum ar fi: utilizarea graficelor pentru vizualizarea datelor, recomandări de script-uri de pe GitHub, sincronizarea fișierelor de configurare cu un serviciu cloud și posibilitatea de împărtășire a fișierelor de configurare cu alți utilizatori.

Au fost apoi comparate soluțiile existente de pe piață, mai exact Metasploit și Recon-ng. Ambele soluții reprezintă utilitare care fac management-ul mai multor script-uri, interfața grafică fiind implementată folosind doar funcționalitățile expuse de linia de comandă. În urma unor comparații mai amănunțite, s-a ajuns la concluzia că aceste aplicații nu reușesc să fie destul de intuitive pentru utilizatori, deoarece nu există o interfață grafică modernă care să nu utilizeze linia de comandă. Mai mult, ambele aplicații nu reușesc să simplifice sau să augmenteze acțiuni elementare ale pentester-ilor cum ar fi: vizualizarea datelor obținute sau importarea unor noi script-uri.

Arhitectura aplicației se bazează pe utilizarea resurselor utilizatorului, astfel s-au folosit tehnologii care permit crearea unui executabil care poate fi rulat pe orice sistem de operare fără probleme de compatibilitate. Totodată, s-a introdus un serviciu de cloud care poate gestiona complet procesul de autentificare și stocare a datelor în cloud.

Implementarea soluției propuse detaliază soluțiile alese pentru a îmbunătății experiența utilizatorului prin crearea unor sisteme automatizate care necesită indicații minime de la utilizator. Totodată, au fost prezentate paginile variante ale aplicației care izolează perfect funcționalitățile preferate de participanții formularului prezentat în primele capitole.

Produsul final este prezentat în ultimele capitulo prin rularea tuturor scenariilor și script-urilor oferite pe organizații cunoscute. Totodată, experiența utilizatorilor a fost analizată într-un nou formular în care mai mulți ingineri din domeniul IT au folosit aplicația Valiant pentru a extrage date publice care pot prezenta un risc de vulnerabilitate. În urma analizei formularului, s-a constat că 90% din participanți ar prefera să folosească exclusiv aplicația Valiant, restul fiind indiferenți.

### **7.1 Dezvoltări ulterioare**

Pe lângă recomandările făcute de utilizatori în capitolul 6, aplicația ar putea beneficia de posibilitatea rulării script-urilor sau a scenariilor într-un serviciu cloud, în cazul în care

dispozitivul curent rulează lent majoritatea script-urilor. Pentru a implementa această soluție ar fi nevoie de introducerea unui serviciu cloud care poate executa comenzi complexe cum ar fi AWS<sup>18</sup>.

Mai mult, pentru că unele script-uri nu pot rula continuu datorită limitării de cereri HTTP pe minut, s-ar putea introduce un utilitar care are rol de *VPN*<sup>19</sup> pentru a ocoli acest obstacol.

---

<sup>18</sup> <https://aws.amazon.com/>

<sup>19</sup> <https://nordvpn.com/ro/what-is-a-vpn/nord-site/>

## 8 BIBLIOGRAFIE

- [1] S. Morgan, „2023 Cybersecurity Almanac: 100 Facts, Figures, Predictions, And Statistics,” 2023. [Interactiv]. Available: <https://cybersecurityventures.com/cybersecurity-almanac-2023/>. [Accesat 26 06 2023].
- [2] C. Griffiths, „The Latest 2023 Cyber Crime Statistics,” 2023. [Interactiv]. Available: <https://aag-it.com/the-latest-cyber-crime-statistics/>. [Accesat 26 06 2023].
- [3] J. Howarth, „50+ Password Statistics: The State of Password Security in 2023,” 2023. [Interactiv]. Available: <https://explodingtopics.com/blog/password-stats>. [Accesat 26 06 2023].
- [4] R. Kelly, „Why You Need to Stop Using Your Pets’ Names in Passwords,” 2021. [Interactiv]. Available: <https://www.digit.fyi/why-you-need-to-stop-using-your-pets-name-in-passwords/>. [Accesat 26 06 2023].
- [5] J. Holcombe, „Key GitHub Statistics in 2023,” 2023. [Interactiv]. Available: <https://kinsta.com/blog/github-statistics/>. [Accesat 26 06 2023].
- [6] S. Singla, „What is the Operating System Market Share?,” 2022. [Interactiv]. Available: <https://www.scaler.com/topics/operating-system-market-share/>. [Accesat 26 06 2023].
- [7] Statista, „PC operating system distribution for software development worldwide in 2018 to 2022,” 2023. [Interactiv]. Available: <https://www.statista.com/statistics/869211/worldwide-software-development-operating-system/>. [Accesat 26 06 2023].
- [8] rapid7, „Metasploit Framework,” 2023. [Interactiv]. Available: <https://github.com/rapid7/metasploit-framework>. [Accesat 26 06 2023].
- [9] lanmaster53, „Recon-ng,” 2021. [Interactiv]. Available: <https://github.com/lanmaster53/recon-ng>. [Accesat 26 06 2023].
- [10] Offsec, „MSFconsole Commands,” 2023. [Interactiv]. Available: <https://www.offsec.com/metasploit-unleashed/msfconsole-commands/>. [Accesat 26 06 2023].
- [11] Kali, „Recon-ng,” 2023. [Interactiv]. Available: <https://www.kali.org/tools/recon-ng/>. [Accesat 26 06 2023].

- [12] ElectronJs, „Electron Docs,” 2023. [Interactiv]. Available: <https://www.electronjs.org/docs/latest/>. [Accesat 26 06 2023].
- [13] B. Peabody, „Server-side I/O Performance: Node vs. PHP vs. Java vs. Go,” 2021. [Interactiv]. Available: <https://www.toptal.com/back-end/server-side-io-performance-node-php-java-go>. [Accesat 26 06 2023].
- [14] SassLang, „Sass Language Documentation,” 2023. [Interactiv]. Available: <https://sass-lang.com/documentation/>. [Accesat 26 06 2023].
- [15] TypeScriptLang, „TypeScript Documentation,” 2023. [Interactiv]. Available: <https://www.typescriptlang.org/docs/>. [Accesat 26 06 2023].
- [16] Bootstrap, „Bootstrap Documentation,” 2023. [Interactiv]. Available: <https://getbootstrap.com/docs/5.3/getting-started/introduction/>. [Accesat 26 06 2023].
- [17] React, „React API Reference,” 2023. [Interactiv]. Available: <https://react.dev/reference/react>. [Accesat 26 06 2023].
- [18] Syncfusion, „Syncfusion React UI Components demos,” 2023. [Interactiv]. Available: <https://ej2.syncfusion.com/home/react.html#platform>. [Accesat 26 06 2023].
- [19] Google, „Firebase,” 2023. [Interactiv]. Available: <https://firebase.google.com/>. [Accesat 26 06 2023].
- [20] iffy, „Electron Updater Example,” 2021. [Interactiv]. Available: <https://github.com/iffy/electron-updater-example>. [Accesat 26 06 2023].
- [21] ViteJs, „Vite Features,” 2023. [Interactiv]. Available: <https://vitejs.dev/guide/features.html>. [Accesat 26 06 2023].
- [22] ReactRouter, „React Router Feature Overview,” 2023. [Interactiv]. Available: <https://reactrouter.com/en/main/start/overview>. [Accesat 26 06 2023].
- [23] Google, „Get Started with Firebase Authentication on Websites,” 2023. [Interactiv]. Available: <https://firebase.google.com/docs/auth/web/start>. [Accesat 26 06 2023].
- [24] Google, „Get started with Cloud Storage on Web,” 2023. [Interactiv]. Available: <https://firebase.google.com/docs/storage/web/start>. [Accesat 26 06 2023].
- [25] m8sec, „CrossLinked,” 2023. [Interactiv]. Available: <https://github.com/m8sec/CrossLinked>. [Accesat 26 06 2023].

- [26] jakecreps, „Poastal,” 2023. [Interactiv]. Available: <https://github.com/jakecreps/poastal>. [Accesat 26 06 2023].
- [27] iojw, „SocialScan,” 2023. [Interactiv]. Available: <https://github.com/iojw/socialscan>. [Accesat 26 06 2023].
- [28] dievus, „MsDorkDump,” 2022. [Interactiv]. Available: <https://github.com/dievus/msdorkdump>. [Accesat 26 06 2023].
- [29] s0md3v, „Photon,” 2022. [Interactiv]. Available: <https://github.com/s0md3v/Photon>. [Accesat 26 06 2023].
- [30] xeroxxhah, „W3b3num,” 2021. [Interactiv]. Available: <https://github.com/Xeroxxhah/W3b3num>. [Accesat 26 06 2023].
- [31] Ekultek, „GitRekt,” 2021. [Interactiv]. Available: <https://github.com/Ekultek/GitRekt>. [Accesat 26 06 2023].
- [32] un1kOn, „GitStalk,” 2019. [Interactiv]. Available: <https://github.com/un1kOn/GitStalk>. [Accesat 26 06 2023].
- [33] BiZken, „PhishMailer,” 2021. [Interactiv]. Available: <https://github.com/BiZken/PhishMailer>. [Accesat 26 06 2023].
- [34] Have-i-been-pwned, „Have I Been Pwned API Overview,” 2023. [Interactiv]. Available: <https://haveibeenpwned.com/API/v3>. [Accesat 26 06 2023].