# **CUPRINS**

[**CUPRINS** 1](#_Toc98424986)

[**1.Argument** 2](#_Toc98424987)

[**2.Proiectarea conținutului** 3](#_Toc98424988)

[**3.Proiectarea interfeței a aplicației** 13](#_Toc98424989)

[**3.1 Interfața Principala a aplicației** 16](#_Toc98424990)

[**3.2 Interfața de modelarea a amprentei digitale** 20](#_Toc98424991)

[**Pagina Proxy** 20](#_Toc98424992)

[**Pagina Geolocation** 22](#_Toc98424993)

[**Pagina WebRTC** 23](#_Toc98424994)

[**Pagina TimeZone** 24](#_Toc98424995)

[**Pagina Navigator** 25](#_Toc98424996)

[**Pagina Fonts** 26](#_Toc98424997)

[**Pagina Media devices** 27](#_Toc98424998)

[**Pagina Hardware** 28](#_Toc98424999)

[**Interfața de navigare intre pagini** 29](#_Toc98425000)

[**4.Arhitectura serverului si a interfeței** 30](#_Toc98425001)

# **1.Argument**

Website-urile moderne de obicei colectează anumite informații despre utilizator cum ar fi cookie-urile însă pot colecta si informați la nivel de hardware astfel pe baza unui algoritm aceste informații pot genera un id unic și static.

De obicei unele website-uri folosesc inclusiv algoritmi de publicitate a unor servici , persoane , videoclipuri , website-uri bazându-se pe respectivul trafic de utilizatori pe baza amprentei digitale astfel arătând impactul lor in mediul online altor utilizatori.

Acuratețea de a seta un anume id unic prin amprenta digitala a browserului de către un website este între 80%-90% (400.000 browsere desktop 89%) (sursa : <https://www.researchgate.net/publication/332873650_Browser_Fingerprinting_A_survey>) însă unele servici destinate către identificarea a informaților respective din browser confirma ca ar avea o rata de succes de peste 95% in a face unic un vizitator pe website.

Browserele normale nu ne oferă aceasta capabilitate de a putea sa ne stocam/editam/generam informațiile necesare pentru a fi putea identificați ca unici pe un anume site web in care sa avem reclame personalizate bazate pe preferințe/locație/regiune.

Acesta soluție software se orientează mai mult spre persoanele care vor sa aibă un anume rezultat din activitățile online cum ar fi:

-Posibilitatea de a face managementul conturilor online multiple (un exemplu ar fi promovarea unui produs pe platformele de tip social-media) fără a putea respectiva platforma sa știe exact ca conturile X Y Z au un deținător comun.

-Posibilitatea de a face putea urmări anumite campanii de marketing din competiție dintr-o anumita regiune/oraș/localitatea/tara.

-Posibilitatea de a distribui specific profilul & amprenta digitala altei persoane sub același cont in aplicație.

-Posibilitatea de a putea compara diferența de preț a unui produs/serviciu bazate pe geo locație.

-Testarea & Automatizarea folosind Selenium pentru o aplicație web folosind un profil & amprenta digitala din aplicație.

-Folosirea Framework-ului Selenium pentru web scraping in mod sigur eliminând toate informațiile in care se afla in mod deliberat in webdriver(fiind nucleul care pornește browserul) ca fiind un test automat inclusiv folosirea profilului & amprentei digitale specifice din server.

-Posibilitatea de a reduce costul aplicaților in Selenium destinate către colectare si procesoare de date in care companiile folosesc mai multe ‘masini-virutale’(servere) din cauza problemei existente cu amprenta digitala in browserul web, astfel aplicația reduce inclusiv costul achiziționări de servere si al energiei consumate fiindcă aplicația poate rula pe un singur calculator cu mai multe profele virtuale.

# **2.Proiectarea conținutului**

Ce este o amprenta digitala in browser?

O amprenta digitala a browser ului poate sa fie generata pe baza unor informații din browser unice.

Metodele pe care urmează sa le rezolvam

* **Time zone**

Orice browser folosește un ceas si o data calendaristica bazata pe o regiune de obicei acest set de informații vine direct de la sistemul de operare neavând posibilitatea in browser de a putea schimba data/ora la nivel de browser doar la nivel de sistem însă nu este deloc convenabil sa schimbam regiunea din sistemul de operare .

Atunci când folosim un VPN fie el cu o adresa IP de Germania/Estonia/Romania etc.. si intram pe un website care preia timezone-ul pe care browserul îl prezinta ca fiind cel de pe sistemul de operare , astfel o sa avem o adresa IP de Franța si cu Time zone de Romania ceea ce este deloc legit.

Ce putem face din aceasta soluție software?

* **Putem afișa valoare reala a browserului.**
* **Putem genera un alt id la nivel de browser pe care îl putem stoca/edita/șterge când dorim.**
* **Putem folosi informațiile corecte atunci când browserul este conectat la un server Proxy**
* **Geo locație**

Când vorbim de geo locația pe care adresa IP o oferă de la furnizorul de internet o arată aceasta nu poate fi cu acuratețe însă pot fi distribuite informații principale cum ar fi orașul in care ne aflam cu adresa IP

Însă când vorbim de geo locația prezenta pe unele placi de baza/laptopuri/telefoane aceasta poate avea o acuratețe foarte buna , evident daca acceptam aceasta funcție când ne este solicitata de browser , atunci când folosim un VPN longitudinea lactațiunea si acurateța pe care modulul GPS îl oferă este in contradictoriu cu ce anume spune adresa IP despre noi.

Browser ele normale nu ne oferă capabilitatea de a seta o geo locație preferabila.

Ce putem face din aceasta soluție software?

* **Putem afișa valoare reala a browserului.**
* **Putem genera un alt id la nivel de browser pe care îl putem stoca/edita/șterge când dorim.**
* **Putem folosi informațiile corecte atunci când browserul este conectat la un server Proxy**
* **Navigator**

Aici se afla o serie de mai multe informații in browser si ele ar fi :

* UserAgent = Versiunea browserului , tipul de browser , sistemul de operare , arhitectura CPU. (useragent-ul poate fi modificat si de o extensie & cod javascript însă ea poate fi scursa deoarece browserul își păstrează acest set de date original când browserul pornește)
* Limba standard a browserului. (romana,engleza,rusa,greaca,etc..) (atunci când folosim un VPN pentru unele platforme poate fi suspect ca avem o limba a browserului setata pe Romana pe când IP-ul nostru este din Germania)
* Rezoluția browserului.
* Rezolutia monitorului.
* Sistemul de operare (separat de useragent)

Ce putem face din aceasta soluție software?

* **Putem edita după bunul plac toate aceste informații**
* **Fonts & ClientRects**

Aceste doua metode sunt la fel de unice ca WebGL/Canvas/Audiocontext , metoda aceasta are 2 scopuri pentru a genera un id unic

1.Modul in care sunt desenate fonturile si preluarea valorilor in browser.

2.Enumerarea listei de fonturi de către browser

Ce putem face din aceasta soluție software?

* **Putem afișa valoare reala a browserului.**
* **Putem genera un alt id la nivel de browser pe care îl putem stoca/edita/șterge când dorim.**
* **WebGL**

Amprentarea WebGL este foarte similara ca mecanism cu Canvas însă ceea ce le diferă este faptul ca WebGL poate reda o imagine 3D ascunsa pe care placa video o înțelege într-un mod unic . Ceea ce are in plus WebGL fata de Canvas poate fi distribuirea modelului placi video de la producător.

Ce putem face din aceasta soluție software?

* **Putem afișa valoare reala a browserului.**
* **Putem genera un alt id la nivel de browser pe care îl putem stoca/edita/șterge când dorim.**
* **Putem edita numele modelului placi video.**
* **Canvas**

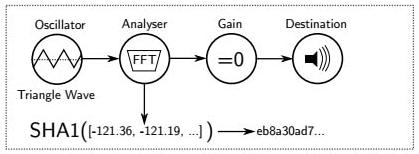
Amprentarea canvas este una dintre mai multe tehnici de amprentare a browserului pentru urmărirea utilizatorilor online, care permit site-urilor să identifice și să urmărească vizitatorii folosind elementul canvas HTML5 în loc de cookie-uri sau alte mijloace similare.

Aceasta metoda solicita browserului sa deseneze o imagine/animație 2D pe care fiecare placa video o generează mereu într-un mod unic

Ce putem face din aceasta soluție software?

* **Putem afișa valoare reala a browserului.**
* **Putem genera un alt id la nivel de browser pe care îl putem stoca/edita/șterge când dorim.**
* **Audiocontext**

Aceasta tehnică verifica modul in care sunetul primit in browser este perceput de anumite componente hardware cum ar fi placa audio ,oscilator etc… însă și diferența de arhitectura a procesorului.

Timpul de prelucrare a sunetului in unde de procesor/oscilator este foarte important pentru a putea genera o amprenta digitală.

Ce putem face din aceasta soluție software?

* **Putem afișa valoare reala a browserului.**
* **Putem genera un alt id la nivel de browser pe care îl putem stoca/edita/șterge când dorim.**
* **WebRTC**

Webrtc-ul este cel care oferă browserelor web și aplicațiilor mobile comunicare în timp real prin interfețe de programare a aplicațiilor.

Astfel webrtc-ul este un protocol care comunica cu sistemul de operare in redarea comunicărilor rapide si direct , teoretic atunci când folosim un server Proxy pe browser e posibil când navigam pe website-uri care folosesc tehnologia webrtc sa existe o scurgere de informații dintre ip-ul serverului Proxy si a adresei IP publica pe care o folosim. Atunci când instalam un VPN pe sistemul de operare si nu pe browser se face trecerea către tot traficul spre serverul vpn si de aceea nu este posibila scurgerea prin de ip prin webrtc , însă când vorbim de browser pentru a seta o anumita conexiune cu un server Proxy si nefiind globala(adică pe tot sistemul de operare) atunci daca nu dezactivam acest webrtc e posibil sa avem o scurgere din serverul proxy la ip-ul inițial.

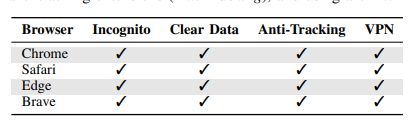
Ce putem face din aceasta soluție software?

* **Putem dezactiva webrtc**
* **Putem lasă scurgerea sa se întâmple si sa afișeze un ip real din serverul Proxy sau o adresa IP la alegere(de exemplu putem folosi o adresa IP 10.1.2.3 de scurgere pe când adresa serverului Proxy este 65.20.30.40)**
* **Favicon**

Aceasta metoda este relativ noua in care se bazează descărcarea fișierului favicon.ico prin mai multe servere atunci când se încarcă pagina accesata de către browserul web.

Metoda respectiva folosește o iconița de 32 biți pe care browserul o înțelege ca un fișier numit “**F-Cache**”

Conform acestui studiu [solomos-ndss21.pdf (uic.edu)](https://www.cs.uic.edu/~polakis/papers/solomos-ndss21.pdf) are o eficienta foarte mare in a putea genera un id unic vizitatorului , adică 4,300,000,000 browsere pe baza celor 32 de biți a iconiței.

****

| **Browser** | **Windows** | **MacOS** | **Linux** | **iOS** | **Android** | ***Info*** |
| --- | --- | --- | --- | --- | --- | --- |
| Chrome *(v 87.0)* | ✅ | ✅ | ✅ | ✅ | ✅ | - |
| Safari *(v 14.0)* | - | ✅ | - | ✅ | - | - |
| Edge *(v 87.0)* | ✅ | ✅ | ❌ | ❌ | ✅ | - |
| Firefox *(v 86.0)* | ✅ | ✅ | ❌ | ❌ | ❌ | Fingerprint different in incognito mode |

Ce putem face din aceasta soluție software?

* **Fiind vorba de o iconița a unui website care este stocat într-un fișier numit ca si “F-Cache” atunci fiecare profil are acest fișier separat de restul rezultând un ID unic static.**

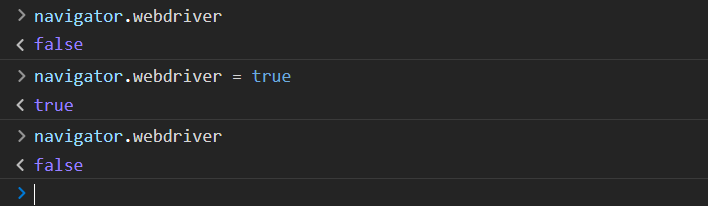
Majoritatea extensiilor pentru browser care se ocupa de a emula un id unic este total aleatorie si ea nu poate stoca/edita respectiva informație , astfel la fiecare accesare a unui link/website/buton avem cate o noua amprenta digitala ceea ce poate para suspect pentru un anume website pentru ca nici un utilizator nu își poate schimba amprenta digitala mai ales ea fiind dinamica la orice buton apăsat in pagina.

Însă când vine vorba de o extensie pentru browserul web pentru a putea împiedica generarea amprentei digitale unice in unele cazuri poate fi eronata si scurgerea de informații se poate face chiar prin respectiva extensie prin simplu motiv ca extensia executa bucăți de cod in browser prin JavaScript , astfel extensia doar ne ascunde un set de date din browser temporar însă aceasta rescriere poate sa fie incompleta/nelegitima si chiar ne poate face unici prin simplu fapt ca nimeni nu are o funcție standard modificata , inclusiv website-ul poate vedea ce fel de extensie folosim pe browser.

Soluția pe care urmează sa o prezint nu executa nici un cod javascript in browser sau o extensie , totul se face la baza browserului.

Atunci când folosim versiunea oficiala([ChromeDriver - WebDriver for Chrome (chromium.org)](https://chromedriver.chromium.org/home)) de Selenium trebui sa folosim chromedriver(nucleul principal pentru automatizare) el vine la pachet cu un set de date incomplete si specifice pentru a putea fi identificat (de exemplu navigator.webdriver fiind setat pe valoarea true când pornește automatizare ceea ce reprezintă ca browserul executa un test pe când browserul deschis normal are valoarea stabilita pe false ) însă exista o lista foarte mare de lipsuri atunci browserul este pornit cu Selenium.

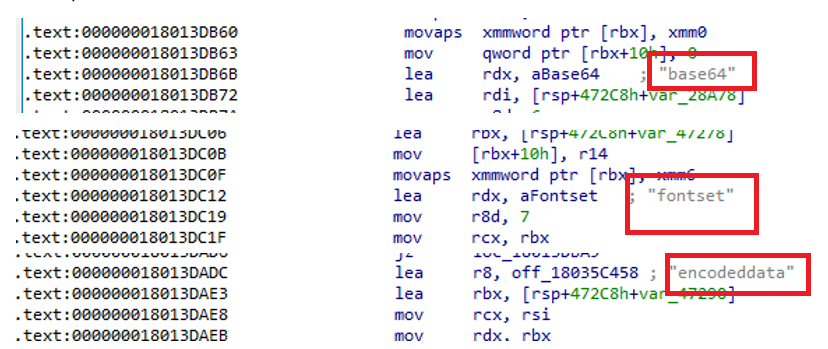
Astfel noi daca încercam forțat sa punem valoarea pe false nu avem nici un rezultat pentru ca browserul setează la valoarea inițiala cu care a fost pornit.



Astfel versiunea oficiala de ChromeDriver ne împiedica pentru a atinge scopurile din **Obiectivele proiectului** astfel o sa folosim o versiune neoficiala (ea se mai numește si **ungoogled chromium**), editata de o companie care are ca scop comercializarea in care inițial erau într-o strânsă legătura cu serverele lor pentru a putea genera o amprenta digitala pe serverele lor într-un mod criptat , însă după o analiza ampla a fișierelor din versiunea modificata am descoperit ca se poate face si într-un mod total diferit fără a interacționa cu api-ul lor.

Pentru a putea analiza fișierul de baza care se ocupa cu generarea de amprenta in browser ( fișierul se numește **net.dll**) am folosit programul **IDA Freeware** care are ca scop de a face o analiza a programului/modulului respectiv in pseudocod apropiata de cod sursa.

In fișierul net.dll avem următoarele argumente într-o funcție principala atunci când pornim browserul:

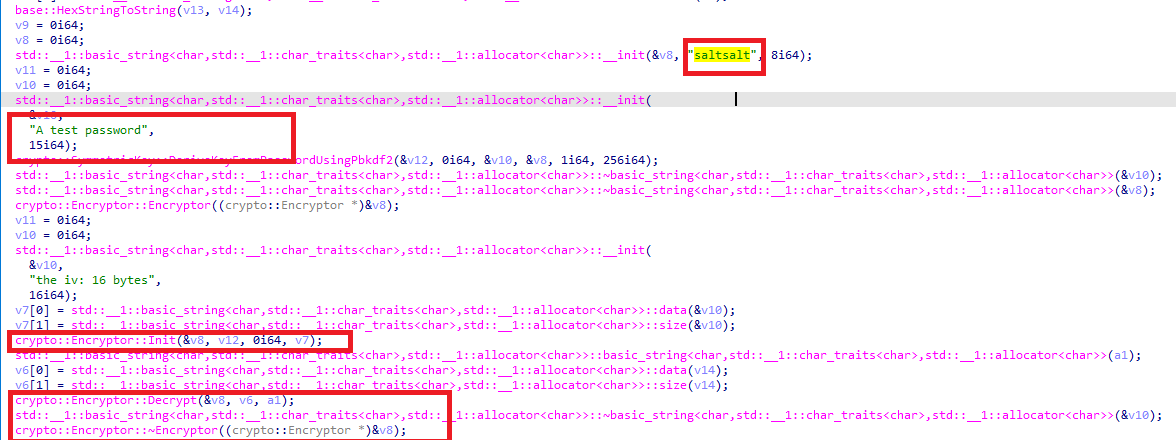


“**fontset**” - ne ajuta sa setam o lista de fonturi in browser fără a le instala

“**encodeddata”** – reprezintă funcția pe care nu o putem apela din cauza ca exista o criptare a informaților care se face pe un server la care nu avem acces , însuși funcția respectiva doar decodează acel set de informații primit de serverul lor .

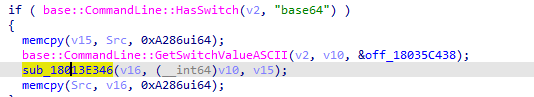


sub\_18013E5C4 = reprezintă funcția atunci când “**encodeddata”** este adăugat ca argument la pornirea browserului in care se ocupa cu decodarea informației respective care arată cam așa , deși într-un final daca decodarea este conforma atunci este apelata funcția din “base64” daca nu atunci nu funcționează.

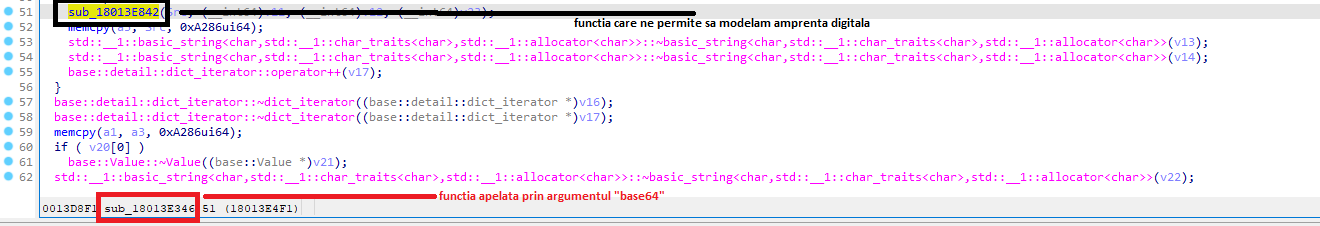


Unde ne este confirmata prezentei criptări prin o parola setata pe server si una in client când este generata, ea se numește Salt ([Salt (cryptography) - Wikipedia](https://en.wikipedia.org/wiki/Salt_(cryptography)))

“**base64**” – face exact ce ar trebui sa facă encodeddata cu excepția interacțiuni cu serverul extern si cu o criptare ușor înțeleasa , prin base64 se face codarea si decodarea informației respective , deci prin argumentul “base64” putem modela orice amprenta digitala in browser după un set de reguli stabilit de modul.



sub\_18013E346 = reprezintă funcția cheie pe care o putem folosi fără encriptia anterioara , care la rândul ei apelează funcția care ne interesează pentru a modela structura amprentei digitale in browser

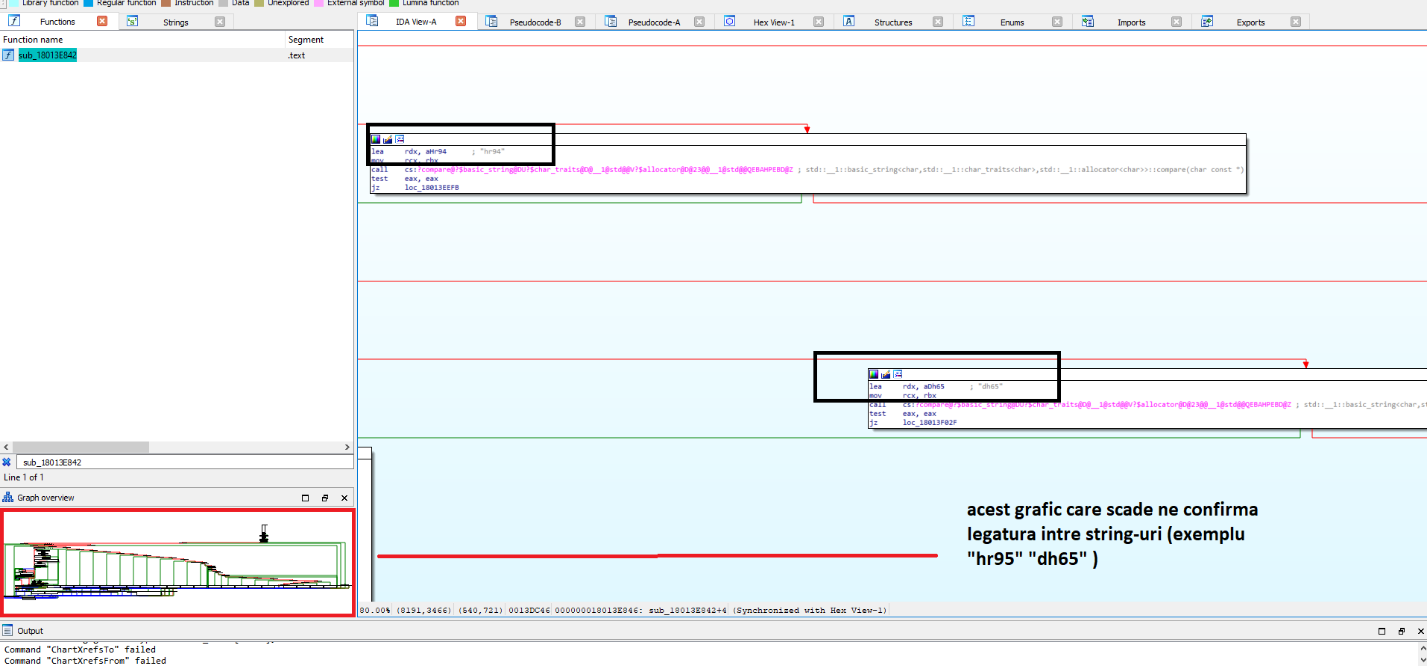


sub\_18013E842 = aceasta e funcția principala unde trebui sa respectam acel set de informații oferit care arată cam așa



Chenarul roșu reprezintă funcția(sub\_18013E842 ) din care parte sunt aceste string-uri pe care trebui sa le legam cu o anume valoare si apoi legate într-un string final care trece prin codarea lui prin base64 , atunci putem trimite prin argumentul “base64=” codul rezultat prin codare si el o sa fie automat citit si decodat de funcția respectiva.

Chenarul negru reprezintă argumentele pe care trebui sa le legam pe toate într-un string.



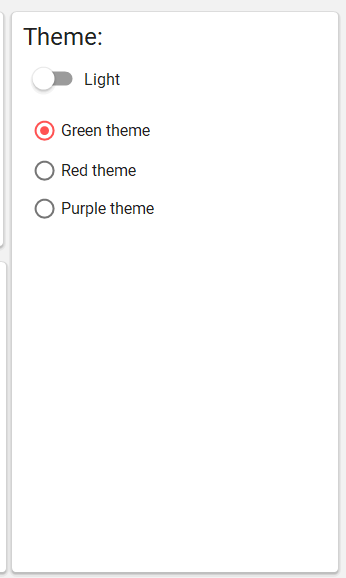
# **3.Proiectarea interfeței a aplicației**

Proiectarea interfeței a utilizatorului se face prin aplicația desktop in care exista :

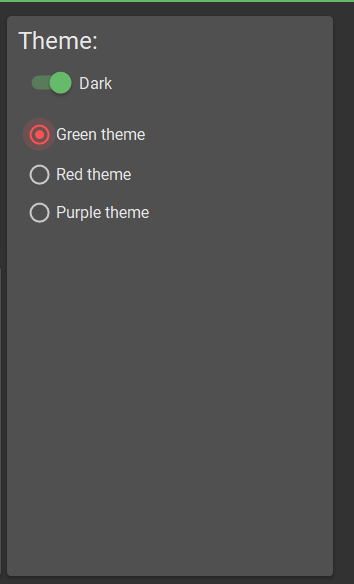
**Interfața de autentificare**

* Butonul de Register = înregistrează pe server id-ul/username-ul utilizatorului
* Butonul de “Connect to server” = permite utilizatorului sa folosească aplicația respectiva sincronizând datele de pe server.
* Textbox-ul unde utilizatorul își poate pune id-ul unic de conectare.
* Consola = permite afișarea unor mesaje de la server sau a aplicației in care utilizatorul este informat cu privința la acțiunea sa.
* Exemple : “Serverul nu este online” , “Acest id deja este înregistrat”
* Utilizatorul are posibilitatea de a își selecta tema respectiva din următoarele moduri :

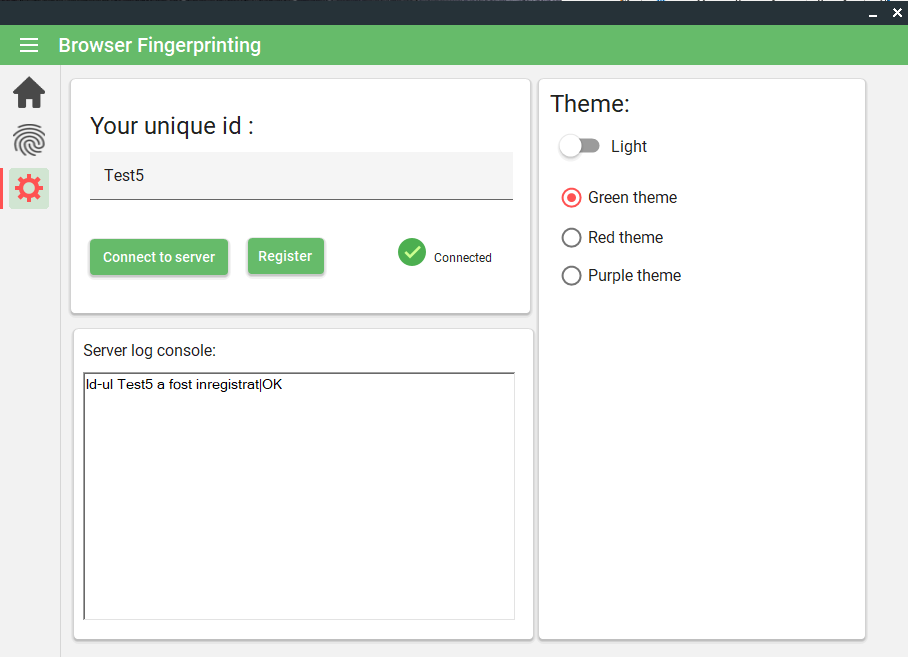
**-Light Mode**



**-Dark Mode**



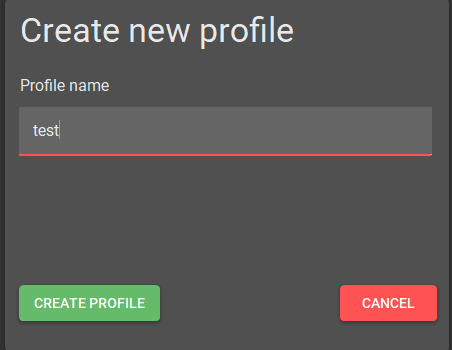
* Utilizatorul are posibilitatea de a își selecta designul aplicației in care exista 3 butoane de tip “radio”
  + Green theme
  + Red Theme
  + Purple Theme



# **3.1 Interfața Principala a aplicației**

Butonul de “Create new profile” are ca scop generarea amprentei digitale a browserului si stocarea ei pe server ( numele profilului selectat de utilizator + prima generare făcută de server)

* Utilizatorul își poate selecta numele profilului
* Excepție : daca utilizatorul are deja un profil creat cu același nume pe server atunci serverul nu creează din nou profilul.



Butonul de “Delete profile” are ca scop ștergerea din server a respectivului profil

Butonul de “Edit fingerprint” redirecționează utilizatorului către pagina principala de a putea modela amprenta digitala a browserului



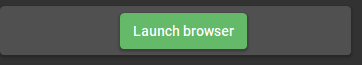
Butonul de “Test browser fingerprint” ruleaza un test automat al profilului selectat de utilizator in care pornește browserul chrome cu SDK-ul Selenium unde se afla mai multe pagini de testare a browserului in care generează un ID unic pentru fiecare vizitator

-Automatizarea Selenium se face automata

-La finalul testului automat aplicația scoate un raport de screenshot-uri a browserului cu website-urile respective pentru a putea compara amprenta digitala cu mai multe profile

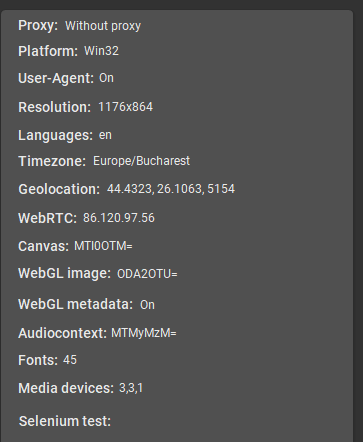


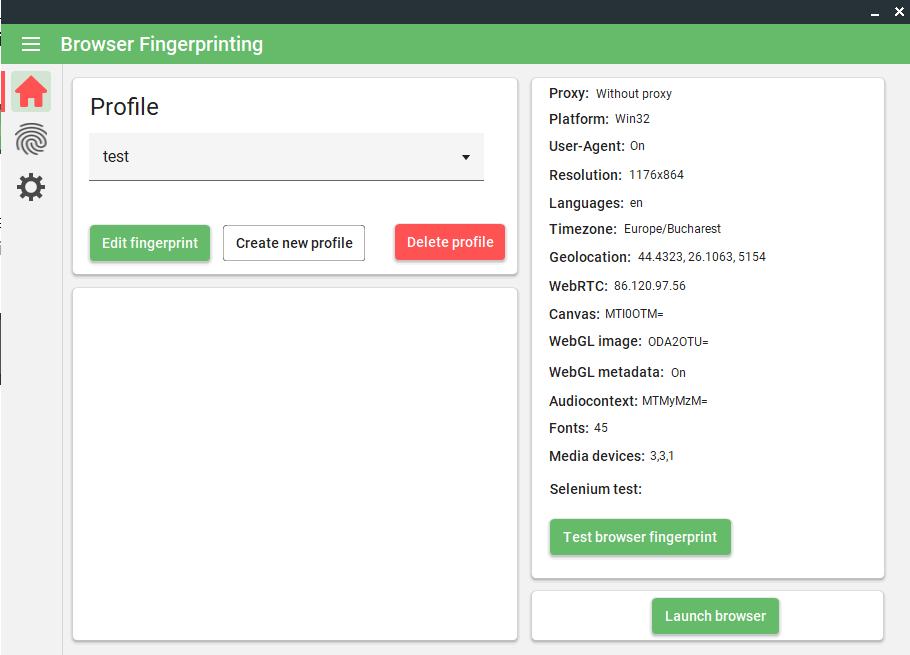
Butonul de “Launch Browser” se refera la pornirea browserului cu profilul specific si amprenta digitala generata/editata



Label-ul de deasupra butonului de “Test browser fingerprint” are ca scop abstractizarea informaților a amprentei digitale.

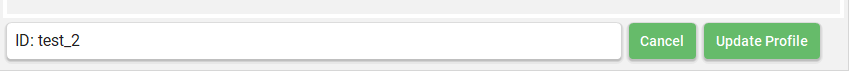
Exemplu :



Poza interfața completa: 

# **3.2 Interfața de modelarea a amprentei digitale**

* Utilizatorul are in dispoziție mai multe taburi in care are toate informațiile necesare pentru a își putea edita amprenta digitala a browseruluI specifica pe profilul creat.
* Utilizatorul are un “bottom bar” in partea de jos in care se afla numele profilului si un buton numit “Update profile” care face update la toate informațiile editate de utilizator din interfața , astfel serverul stochează fiecare modificare realizata de utilizator



## **Pagina Proxy**

* In pagina “Proxy” utilizatorul are posibilitatea de a își atașa un server Proxy la browser pentru a își ascunde identitatea pe o anumita regiune/locație

Protocoalele de servere Proxy care sunt permise de aplicație :

* + HTTP/S
  + SOCKS5

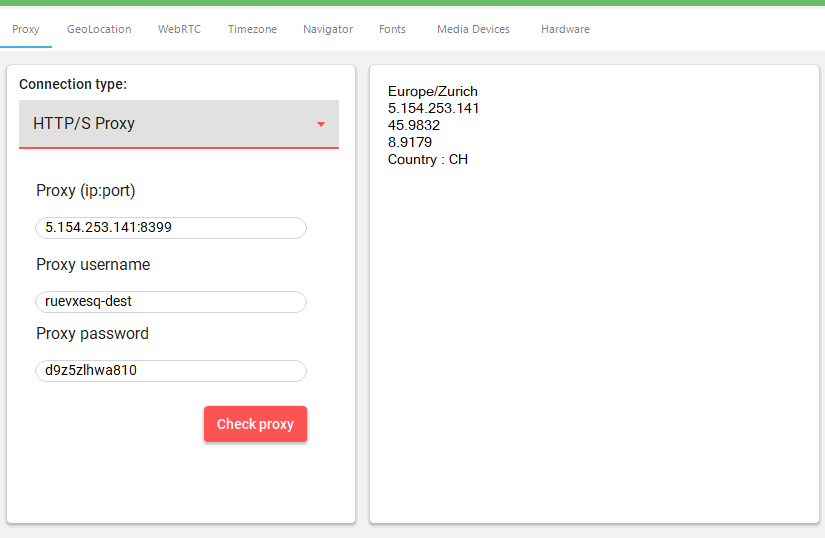
Autentificarea pe serverul Proxy propui se poate realiza prin :

* + IP/DNS
  + Username
  + Password

Butonul de “Check proxy” are ca scop de a trimite informațiile atașate de utilizator in cele 3 tabele către server-ul principal unde prelucrează informațiile oferite din serverul Proxy si sincronizează restul paginilor conform serverului Proxy.

Sincronizarea si interogarea serverului Proxy pe care o face serverul principal :

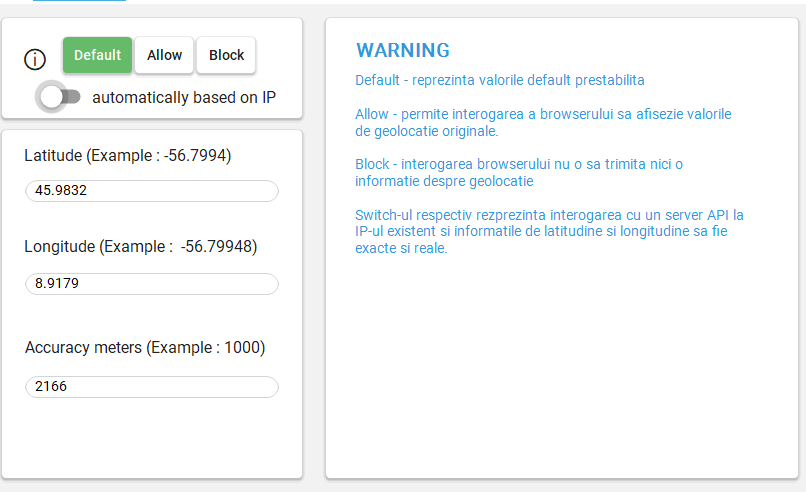
* + Adresa IP WebRTC
  + Geo locație (longitudine , latitudine si acuratetea e generata aleatoriu)
  + Time zone (ora exacata a regiuni in care se afla serverul Proxy si numele tari)



## **Pagina Geolocation**

Butoane :

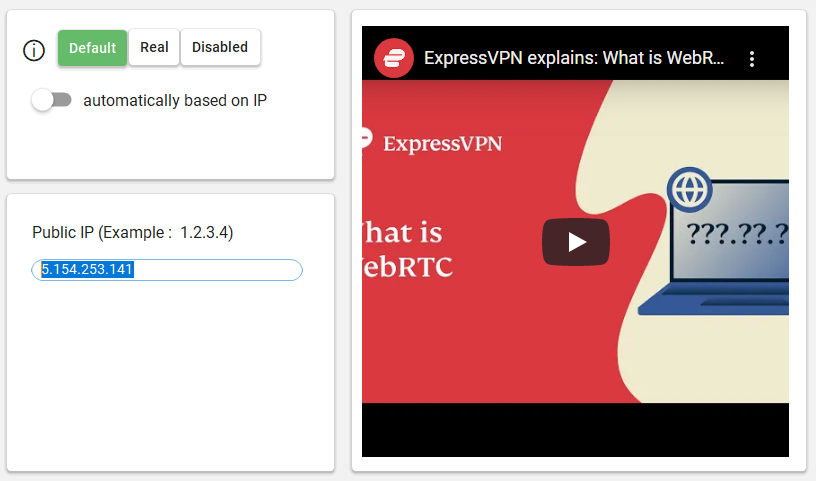
* + Default = Afișează valoarea default generata de server
  + Allow = permite browserului sa afișează latitudinea longitudinea si acuratețea reala.
  + Block = nu permite browserului sa afișeze nici o informație legata de geo locație
  + automatically based on IP = sincronizează datele conform serverului Proxy
  + = afișează in dreapta programului fiecare funcționalitatea a fiecărui buton
* -Accesibilitatea butonului Default cu selectarea butonului a switch-ului
  + Longitudine proprie
  + Latitudine proprie
  + Acuratețe proprie



## **Pagina WebRTC**

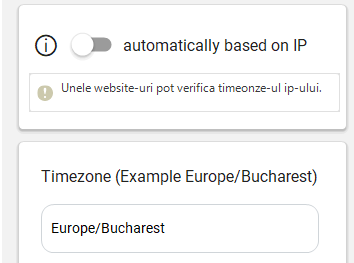
Butoane :

* + Default = Afișează valoarea default generata de server
  + Real = permite browserului sa afișează scurgerea a IP-ului real a utilizatorului
  + Block = nu permite browserului sa afișeze nici o informație legata de scurgerea IP-ului a utilizatorului
  + automatically based on IP = sincronizează datele conform serverului Proxy
  + = afișează in dreapta programului un videoclip in care este explicata importanță WebRTC-ului
* Accesibilitatea butonului Default cu selectarea butonului a switch-ului
  + Posibilitatea de a selecta un IP de scurgere fals care este afișat atunci când webrtc-ul pe un website este activ.



## **Pagina TimeZone**

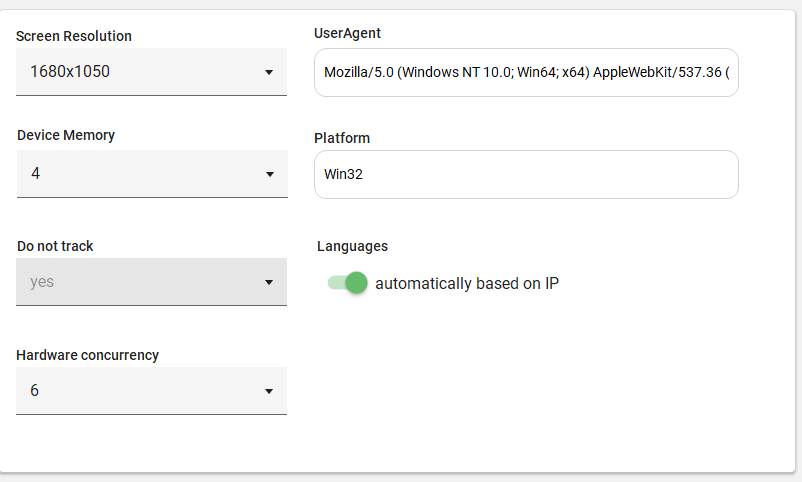
Butoane :

* + automatically based on IP = sincronizează datele conform serverului Proxy
  + = redirecționează utilizatorului către ultima pagina a programului prezentând un videoclip detaliat.
* Accesibilitatea butonului Default cu selectarea butonului a switch-ului
  + Posibilitatea de a selecta un timezone specifica unei regiuni/tari. 

## **Pagina Navigator**

Butoane :

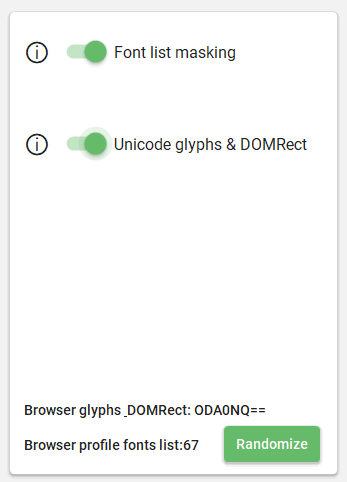
* + UserAgent = Versiunea browserului , tipul de browser , sistemul de operare , arhitectura CPU.
  + Platform = Denumirea sistemului de operare (exemplu : Win32)
  + Device Memory = Memoria RAM pe care browserul o afișează atunci când este interogata
  + Hardware concurrency = Afișează numărul nucleelor a procesorului
  + Screen resolution = Afișează o lista de rezoluți in care utilizatorul poate selecta valoare default cu care browserul sa “minta” valoarea originala a utilizatorului.
* Languages = limba default atunci când browserul pornește (exemplu romana,engleza,germana etc..)
  + automatically based on IP = sincronizează datele conform serverului Proxy



## **Pagina Fonts**

Butoane :

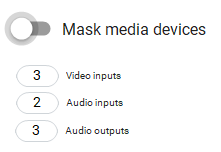
* + Font list masking OFF = browserul afișează valoarea lista de fonturi a calculatorului originala.
  + Font list masking ON = browserul afișează valoarea lista de fonturi generata de server.
  + Unicode glyphs DOMRect OFF = browserul afișează lista de emoticoane/simboluri a calculatorului originala.
  + Unicode glyphs DOMRect ON = browserul afișează valoarea lista de emoticoane/simboluri generata de server.
  + Randomize = Serverul generează un nou set de date (font-uri si simboluri) si îl transmite aplicației de interfața .
  + = redirecționează utilizatorului către ultima pagina a programului prezentând un videoclip detaliat.



## **Pagina Media devices**

Butoane :

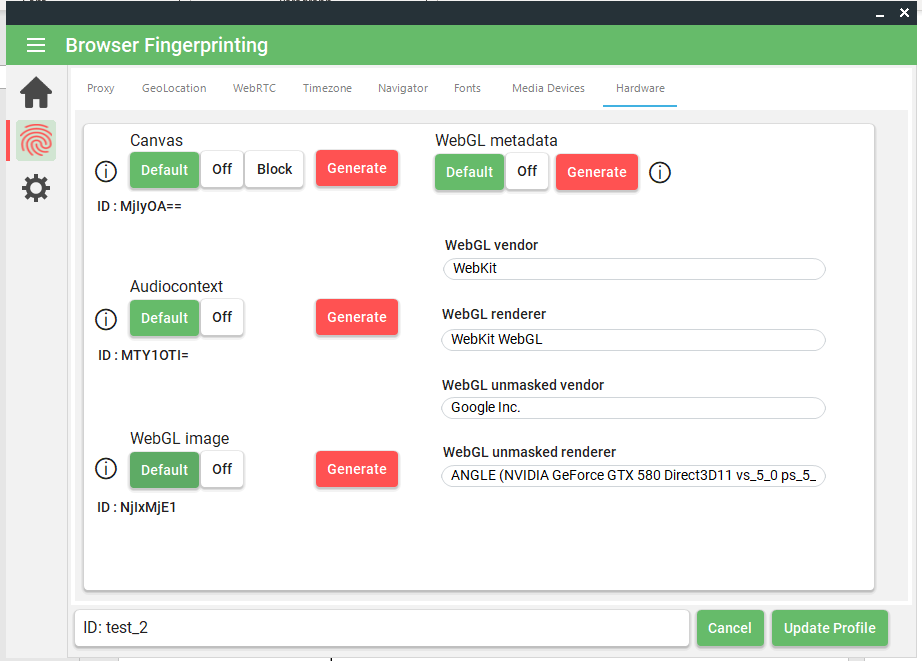
* + Mask media devices OFF = afișează numărul de dispozitive real al calculatorului pe browser (cum ar fi Camera Web, Tastatura wireless , Microfon etc..)
  + - utilizatorul poate scrie singur numărul in cele 3 tabele.
  + Mask media devices ON = afișează numărul de dispozitive generat de către server



## **Pagina Hardware**

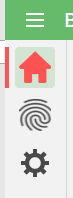
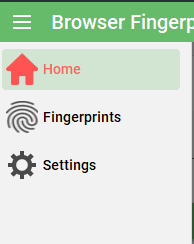
Butoane :

* + Default = Afișează valoarea default generata de server
  + Off = permite browserului sa afișează valoarea default a browserului specifica fiecărui funcții
  + Block = nu permite browserului sa afișeze nici o informație legata de Canvas
  + Generate = Serverul primește o cerere de a genera un nou ID unic pentru funcția specifica si interfața preludează ce a generat serverul.
  + = afișează un videoclip detaliat despre toate informațiile.
* Accesibilitate
  + WebGL vendor = WebKit (default)
  + WebGL render = WebKit WebGL(default)
  + WebGL unmasked vendor = Google Inc.(default)
  + WebGL unmasked renderer = denumirea producătorului si a modelului de placi grafice



## **Interfața de navigare intre pagini**

Navigarea intre pagini se realizează prin butonul partea de sus a programului



# **4.Arhitectura serverului si a interfeței**

Interfața prezentata in punctul 3 are la baza un server cu care comunica constant atunci când sunt salvate/generate/șterse date din interfața.

Arhitectura intre client-server se face prin protocolul UDP.

Serverul are ca scop prelucrarea , procesare de date , stocarea de date a utilizatorului interfeței .

Astfel interfața prezentata in punctul 3 nu are nici o importanta daca serverul este offline.

Interfața aplicației si al serverului a fost realizata in c# .net core

Serverul poate fi numit si ca un “API” având in vedere posibilitatea de a putea scrie un program fără a avea o interfața sau o automatizare selenium fără a putea depinde de interfața respectiva , astfel spus orice dezvoltator poate își poate scrie singur o aplicație (indiferent ca e vorba de c# c++ , c , php , javascript, java , etc… ) bazata acest set de reguli impus de server .

Arhitectura comenzilor a serverului

* Cerințe:
  + Serverul este pornit pe portul 5000 , astfel dezvoltatorul este nevoit sa folosească ca port de destinație 5000.
  + Serverul răspunde doar atunci când este folosit protocolul UDP ca transmise a datelor.
  + Mesajul trimis către server trebui sa aibă un început pentru fiecare comanda numita ca “angine”
* Comenzile recunoscute de server :
  + **Font**
  + **Static\_font**
  + **Screen**
  + **Canvas**
  + **Audiocontext**
  + **Wid\_webgl**
  + **Webgl\_metadata**
  + **Rects**
  + **Async\_proxy**
* Astfel la fiecare comanda executata prin protocolul UDP care se afla in setul de reguli de mai sus reprezintă abstractizarea funcțiilor aflate in server si le transmite către server (principiul de cerere si răspuns)
* Un exemplu de aplicație realizata fără interfața prin ajutorului serverului este automatizarea Selenium pentru testarea amprentei digitale.
* Funcție extrasa din codul sursa al automatizări
  + **async\_data(Encoding.Default.GetString(buffer));**