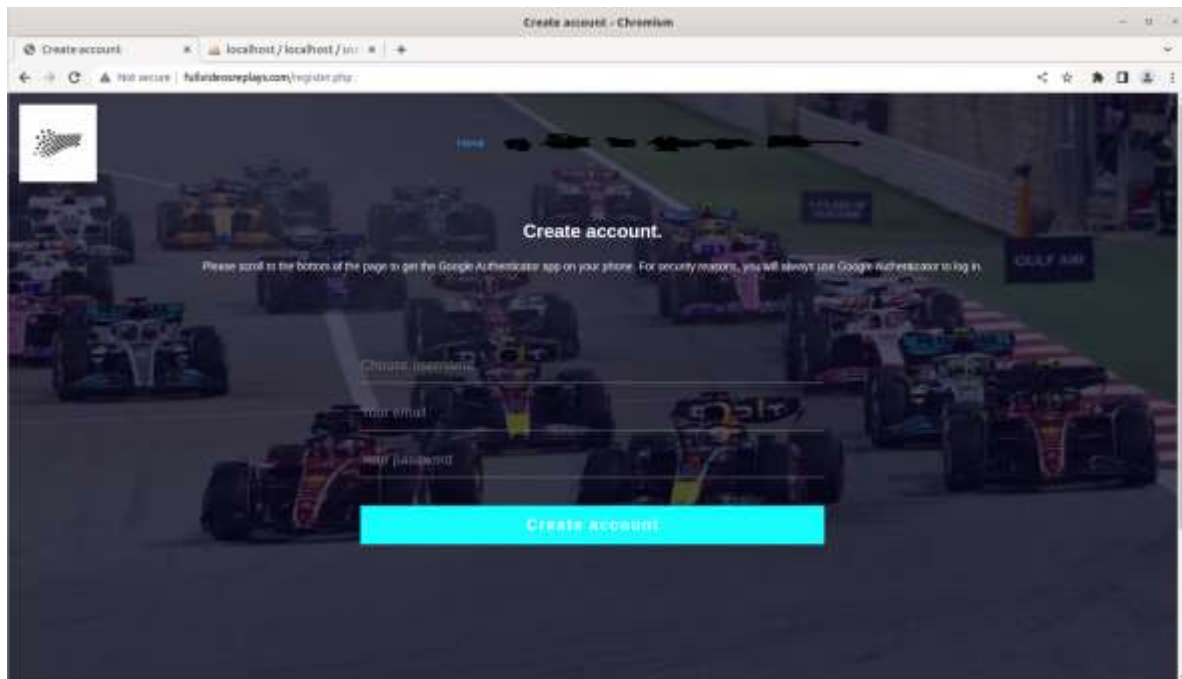


Autentificare two factor folosind cod QR si
aplicatia Google Authenticator

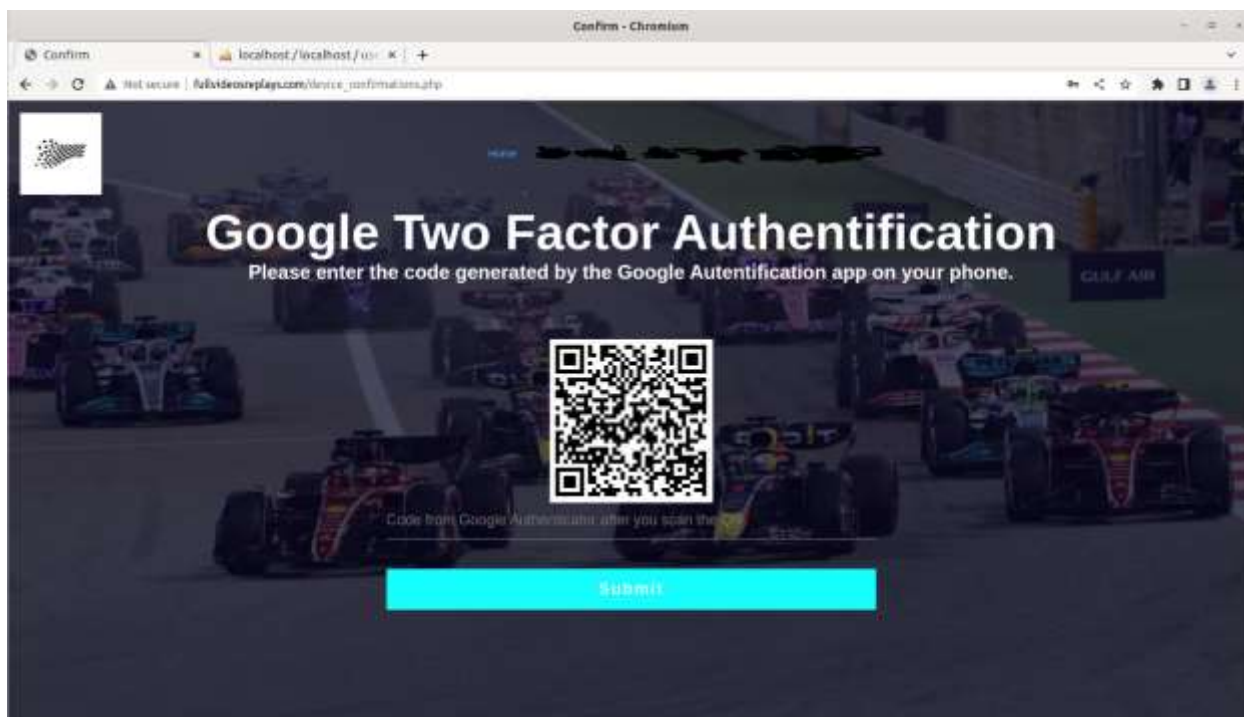
Codurile pentru paginile web si instructiunile pentru setarea LAMP pe Ubuntu sunt in Z:\19.TASK FORCE\1.Documente in lucru\3.Mihnea ANDREI\website\2_factor_google_QR_auth_PHP.

Poze ale site-ului

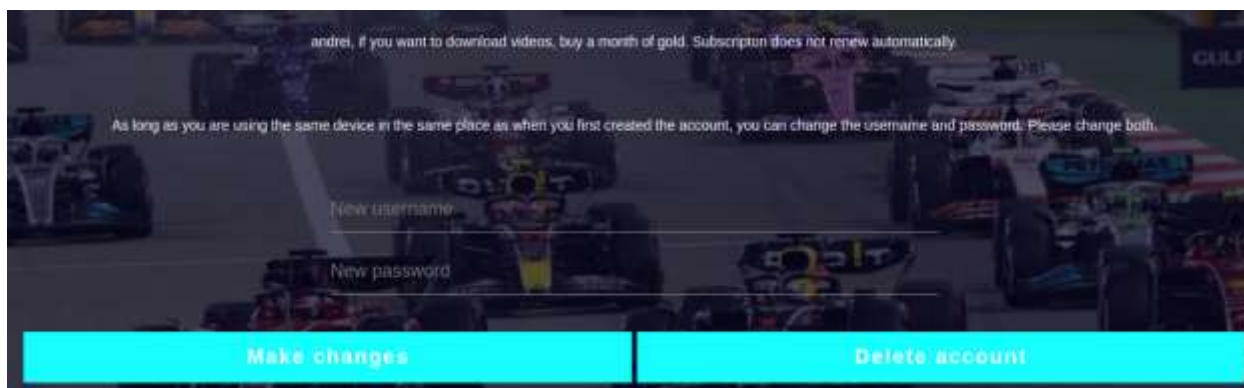
Pagina de creare de cont numita *register.php* arata ceva de genul:



In josul paginii sunt si linkuri la apple store si google store pentru aplicatia Google Authenticator. O data ce completati formularul de mai sus, sunteti redirectionati la urmatoarea pagina numita *device_confirmation.php*:



Daca contul este creat, utilizatorul trebuie sa scaneze codul QR cu aplicatia Google Authenticator si sa introduca in campul de dedesupt codul ce apare in aplicatie. Daca contul este creat si utilizatorul vrea sa schimbe username-ul si parola, nu trebuie decat sa scrie codul din aplicatia Google Authenticator. Pagina pentru schimbarea username-ului si parolei sau stergerea contului arata astfel:

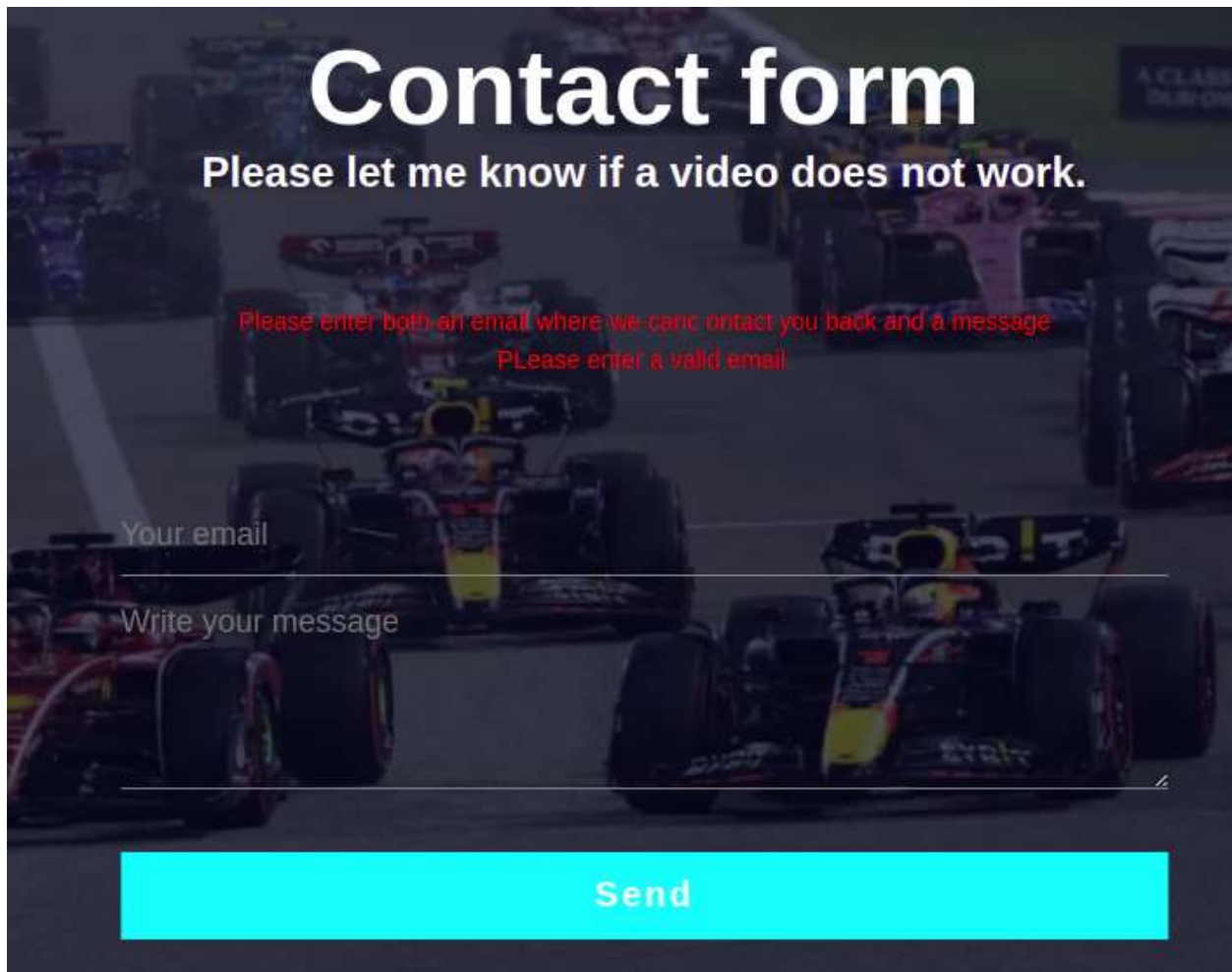


[Instructiuni autentificare folosind Google Authenticator](#)

Eu nu sunt programator, dar exista foarte multi oameni in lumea asta foarte buna asta si unii chiar pun lectii online pe gratis. Normal ca unele nu sunt bune, dar cu rabdare gasesti ceva care chiar merge. Asa am invatat si chiar am facut sa mearga aceasta metoda de autentificare care foloseste QR code si aplicatia pe telefon numita Google Authenticator.

Folder-ul Z:\19.TASK FORCE\1.Documente in lucru\3.Mihnea ANDREI\website\2_factor_google_QR_auth_PHP contine urmatoarele fisiere:

- *LAMP_and_Kubernetes.txt* –instrucțiuni de instalare și setare a PHP, MySQL, apache2, directorul root pentru site și configurări mai ales în apache2. Totul este în Ubuntu. Nu știu dacă merge să faceți asta pe windows care are un emulator de Ubuntu instalat. Eu am făcut direct pe Ubuntu și totul merge. Am scris acolo și câteva instrucțiuni pentru a crea noduri, servicii, etc în kubernetes. Scopul acestui README este însă de a prezenta metoda de autentificare.
- *register.php* – este un fișier ce conține codul PHP pentru a crea un nou cont
- *device_confirmations.php* – o dată ce este completat formularul din *register.php* utilizatorul este redirectionat la această pagină, unde vede un cod QR, pe care trebuie să îl scaneze cu aplicația Google Authenticator. Această aplicație creează un cod de 6 cifre la fiecare 30 de secunde, pe care utilizatorul trebuie să îl introducă în această pagină în cel mult 1 minut (de la vederea acestei pagini). Dacă totul este corect, contul este creat pe pagina web și apare și în Google Authenticator, așa ca utilizatorul nu mai trebuie să scaneze codul QR din nou, ci doar să introducă codul de 6 cifre din aplicație.
- *login.php* – dacă utilizatorul are deja un cont, apasă pe un buton de login din pagina principală (pe care nu am pus-o aici, dar dacă sunteți interesați, o pot arăta și pe aceea). În pagina *login.php*, utilizatorul introduce username și parola (poate fi email în loc de username foarte ușor), este retrimis la pagina *device_confirmations.php*. Aici repetă procedura de la punctul anterior, numai că dacă are deja contul creat în Google Authenticator nu mai trebuie să scaneze codul QR. Dacă cumva a sters contul din Google Authenticator sau a sters aplicația, trebuie să o reinstaleze și să scaneze codul QR care apare pe pagină. Contul este recreat în Google Authenticator automat și utilizatorul introduce codul de 6 cifre din aplicație.
- *myaccount.php* – este pagina unde utilizatorul deja logat pe site își poate crea un nou username și parola sau poate șterge contul. Dacă vrea un nou username și parola, este trimis la pagina *device_confirmations.php*, unde trebuie să folosească Google Authenticator din nou. Pentru ștergerea contului, nu este nevoie de o astfel de confirmare.
- *forgot_account.php* – dacă utilizatorul și-a uitat usernameul și parola, le poate schimba aici, este redirectionat la pagina cu codul QR, unde introduce codul din aplicația Google Authenticator de pe telefon și modificările sunt înregistrate în baza de date
- *logout.php* – pur și simplu se șterg variabilele de sesiune ale utilizatorului.
- *googleLib/GoogleAuthenticator.php* – clasa scrisă de google pentru generarea codului QR, conexiunea cu aplicația pe mobil și verificarea codului introdus de utilizator.
- *stuff.php* – metoda de criptare și decriptare a oricărei informații pe care o bagăm în baza de date. Ați putea chiar crea chei noi la fiecare (ora, zi, săptămână), dar este computationally intensive deoarece atunci când o cheie nouă este creată, trebuie folosită cheia veche pentru a decripta toate datele din baza de date și după aceea trecute prin cheia nouă ca să fie criptate cu cheia nouă. Numai că așa crește și mai mult siguranța informațiilor cetățenilor. Aveți voi specialiști mult mai buni care pot îmbunătăți acest schelet și știu exact cum să folosească asta.
- *contact.php* este un formular care poate fi folosit pentru pagina MCID în loc de a lista emailuri, se poate crea un câmp pe care oamenii îl completează și mailul este transmis automat:



Contact form

Please let me know if a video does not work.

Please enter both an email where we can contact you back and a message
Please enter a valid email

Your email

Write your message

Send

Cod autentificare folosind Google Authenticator

O sa prezint pe scurt codul. Acesta este doar un schelet, din moment ce poate fi imbunatatit cu mult din punct de vedere al securitatii, pentru care aveti specialisti (de exemplu, eu memorez in baza de date username, email, parola si secretul pentru generarea codului QR, dar voi probabil ca doriti sa incryptati toate astea si dupa aia sa le salvati in baza de date, dar voi stiti mai bine).

Ideea cu Google Authenticator este ca cei de la Google au scris o clasa, pentru care codul este in *googleLib/GoogleAuthenticator.php*. Oricum, noi nu o sa folosim decat vreo 2 functii de acolo.

La inregistrarea unui nou cont (*register.php*) ne conectam la baza de date (si ma rog voi puteti verifica daca este conexiune etc) si se creeaza un secret:

```
require_once("googleLib/GoogleAuthenticator.php");  
$ga = new GoogleAuthenticator();  
  
#The secret is some random choice of characters and  
$secret=$ga->createSecret();
```

Acest secret impreuna cu email-ul introdus de utilizator sunt folosite pentru a genera codul QR. Partea de cod care face acest lucru este in *device_confirmations.php*. Secretul este retinut in baza de date impreuna cu username, email, parola, IP, etc. Asta este deoarece atunci cand utilizatorul o sa se

logheze din nou, trebuie sa folosim acelasi secret si aceeași adresa de email pentru a crea același cod QR. Practic codul QR devine unic pentru fiecare utilizator (secret si cu email creeaza codul QR). Daca utilizatorul o sa isi schimbe adresa de email, o sa trebuiasca sa scaneze codul QR din nou si un nou cont o sa apara in Google Authenticator care genereaza alte combinatii de 6 cifre la fiecare 30 de secunde. Revenind la codul *device_confirmations.php*, generarea codului QR se face tot cu clasa definita de Google:

```
$secret=$_SESSION["secret"];
$email=$_SESSION["email"];

require_once("googleLib/GoogleAuthenticator.php");
$ga = new GoogleAuthenticator();
$qrcodeUrl=$ga->getQRCodeGoogleUrl($email,$secret,"fullvideosreplays");
```

Verificarea se face pur si simplu astfel:

```
$code=$_POST["code"];
$checkResult=$ga->verifyCode($secret,$code,2);
if($checkResult){
    $_SESSION["secret"]=$secret;
    header("Location:index.php");
}
```

Daca verificarea este cu success, utilizatorul este redirectionat la pagina principala.

Atunci cand utilizatorul se logheaza cu username si parola, ne uitam in baza de date sa vedem ce secret si email are utilizatorul cu username-ul, parola si adresa IP respective si este redirectionat la *device_confirmations.php* in caz ca username-ul si parola sunt corecte. Acolo, generam codul QR unic pentru acest utilizator (in caz ca trebuie sa il scaneze din nou daca a sters aplicatia Google Authenticator de pe mobil din greseala de exemplu), utilizatorul introduce cele 6 cifre din aplicatie, si daca sunt corecte, este redirectionat la pagina principala:

```
$username=$_POST["username"];
$password=$_POST["password"];
$ip=$_SERVER['REMOTE_ADDR'];

$query=mysql_query($connection,"SELECT * FROM google_auth WHERE username='".$username.'" AND password='".$password.'" AND userip='".$ip.'"");
$numrows=mysql_num_rows($query);
if($numrows>0){
    $rows=mysql_fetch_array($query);
    $_SESSION["email"]=$rows["email"];
    $_SESSION["secret"]=$rows["secret"];
    $_SESSION["secretcode"]=$rows["secretcode"];
    header("Location:device_confirmations.php");
}
```

Observatii autentificare folosind Google Authenticator

Acesta este doar scheletul, dupa cum am mai zis. Voi puteti implementa metode de securitate pe care le stiti foarte bine pe asta). De asemenea puteti fi mai grijulii la verificarea utilizatorului care se logheaza. De exemplu, poate va intrebati de ce am fortat ca IP-ul de la care se logheaza utilizatorul sa fie același cu cel de la care s-a creat contul. Este pentru ca m-am gandit ca din moment ce acest site ce contine date confidentiale utilizatorului ar trebui sa fie accesat numai de acasa, acolo unde nimeni altcineva care nu este cunoscut (ca de exemplu un necunoscut care sta langa utilizator in metrou) nu poate vedea tragand cu ochiul la datele personale ale utilizatorului. De asemenea, acasa este mai probabil ca utilizatorul sa fie inconjurat de persoane de incredere.

În plus, o dată cu inter-operabilitatea, se poate vedea adresa fizică ce corespunde adresei respective de IP. Dacă adresa fizică ce corespunde adresei de IP este aceeași cu cea declarată în bulletin, atunci probabilitatea ca persoana care a făcut teste să fie cea din bulletin crește și mai mult.

Dar voi vă puteți gândi la tot felul de idei. Există multe posibilități. Oricum, recunoașterea facială ar aduce un surplus de securitate, care nu ar putea fi depășit de nimic. Am început să lucrez și la asta și puteți să îmi citiți README-urile:

- *Z:\19.TASK FORCE\1.Documente in lucru\3.Mihnea ANDREI\python_scripts\facial_detection_and_recognition_final_scripts* – care este un model deja antrenat și pare să meargă destul de bine

Z:\19.TASK FORCE\1.Documente in lucru\3.Mihnea ANDREI\python_scripts\resnet_andrei – care arată cum să alterezi modelul creat de Google, numai că antrenarea durează mult probabil și datorită imaginilor proaste pe care le am (imagini downloadate de pe Wikipedia care includ și picture în unele cazuri – vă rog citiți README-urile despre asta de asemenea).