Proiect Python

Authentication system using QR codes

Cuprins

Introducere

Inspiratie

Lucrarea are ca scop realizarea unui ecosistem capabil de generarea de coduri QR, stocarea, citirea si interpretarea lor. Codurile QR devin din ce in ce mai folosite in viata de zi cu zi. De la anunturi publicitare, la meniurile din restaurant, la accesul intr-o retea Wi-Fi, le regasim la tot pasul. Inspiratia pentru acest proiect sunt biletele de acces la evenimente – concerte, festivaluri, piese de teatru, etc – care adesea sunt sub forma de cod QR / cod de bare. Acestea contin un identificator unic care se regaseste intr-o baza de date, iar informatia propriu zisa este livrata clientului sub forma unui “desen” adesea format din mai multe patratele – albe si negre – ce stocheaza informatia. Conceptul este simplu, iar pentru scanare nu este nevoie de dispozitive speciale, de aceea este si foarte raspandit.

Ce sunt codurile QR si cum functioneaza?

Un cod QR – Quick Response code – este o alaturare de patrate albe si negre (sau orice alte culori cu un contrast suficient, recent au aparut diverse stilizari ale acestor coduri) care stocheaza informatii ce pot fi interpretate de catre un calculator.

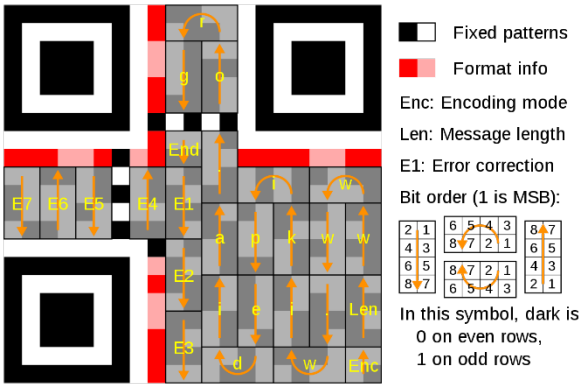
Codul QR nu este o inventie noua, acesta a fost inventat in anul 1994 de o companie japoneza pentru a tine evidenta produselor pe linia de asamblare. Acesta poate stoca mai multe tipuri de date, lucru specificat la inceputul codului. Tipul de caractere pe care il poate stoca codul, influenteaza direct marimea maxima a mesajului ce poate fi codificat.

- Numeric - ~7000 de carctere
- Alphanumeric - ~4000 de caractere
- Binar - ~3000 de caractere
- Kanji - ~2000 caractere

Uzual, codurile QR au o structura standard destul de simpla:

- Quiet zone – zona alba, ce nu contine nimic, ce inconjoara codul QR.
- Hinder pattern – cele 3 patrate negre din stanga-jos, stanga-sus, si dreapta-sus al codului
- Alignment pattern – patratul negru mic din coltul din dreapta-jos care asigura citirea codului din orice unghi
- Timing pattern – o linie in forma literei ‘L’ care ajuta la identificarea patratelor individuale, facand posibila citirea unui cod QR avariat.
- Informatii despre versiunea codului
- Celulele de date – restul codului QR

Generarea codului QR este mai complicata decar ar parea datorita modului de corectare al erorilor care asigura citirea codului chiar daca aproape un sfert din acesta nu exista, in orice unghi, din orice directie. Ca exemplu, in stanga avem un exemplu de 2 coduri QR total functionale desi o parte din cod nu este vizibil.



Pe scurt, generatorul evita forme ce ar putea induce scanner-ul in eroare – spre exemplu nu va produce alte patrate similare cu cel de aliniere pentru a stoca date. Pentru a face asta mesajul este impartit in mai multe blocuri, de la dreapta la stanga, incepand cu tipul de encoding si cu lungimea mesajului, urmand continutul acestuia. In cod sunt de asemenea prezente blockuri de corectie ale erorii, si modele fixe care nu se schimba niciodata.

Tipuri de coduri QR

De-a lungul timpului s-au inventat mai multe tipuri de coduri QR, cu diverse atribute:

- Cel clasic – margini albe, arhicunoscut
- Aztec code – contine patrate de aliniere ce nu necesita “Quiet Zone”
- Micro QR – o versiune mai mica a codului QR normal
- iQR – au o forma dreptunghiulara, nu una patratica
- Secure QR – contine informatii criptate ce necesita o cheie pentru a fi descifrate
- Frame QR – codul este infasurat in forma unei ‘rame’, iar in mijlocul acesteia putem pune diverse poze sau elemente.

Riscurile folosirii codurilor QR.

Codurile QR functioneaza intr-un procedeu simplu deci nu aduce multe riscuri in sine. Singurele riscuri care pot surveni sunt:

- Spoofing – codul QR sa contina un link catre un website periculos, dar riscul nu vine din codul QR in sine, ci de ce poate face utilizatorul pe site-ul respectiv.
- Pozarea acestuia – in codurile QR nu ar trebui sa fie stocate date permanente sensibile. Chiar daca continutul poate fi criptat in interiorul acestuia, o simpla poza a codului poate garanta unei terte persoane acces la continutul codului. La acelasi risc se supun si cartelele magnetice care stocheaza date chiar pe acestea (nu prin pozare ci prin clonare, desigur). De aceea recomandarea este ca codurile QR ce sunt folosite pentru autorizatii de login-uri sau bilete de concerte sa aibe o durata de viata relativ scurta, si sa nu poata fi folosite decat o data.
- In functie de cititor, se pot realiza exploatari de tipul no-software in care atacatorul poate afla diverse date ale utilizatorului precum locatie, istoric de cautare, local storage si altele, desi nu se aplica la smartphone-uri moderne, este important de notat ca un dispozitiv ce nu este bine gandit poate fi la risc.

Proiectul propriu-zis

Ne-am propus sa facem un ecosistem intreg de generare al codurilor QR. Nu am dorit sa implementam toate normele de securitate ce ar fi necesare intr-un proiect real, dar dorim sa ilustram principiul de functionare si aplicatiile acestuia.

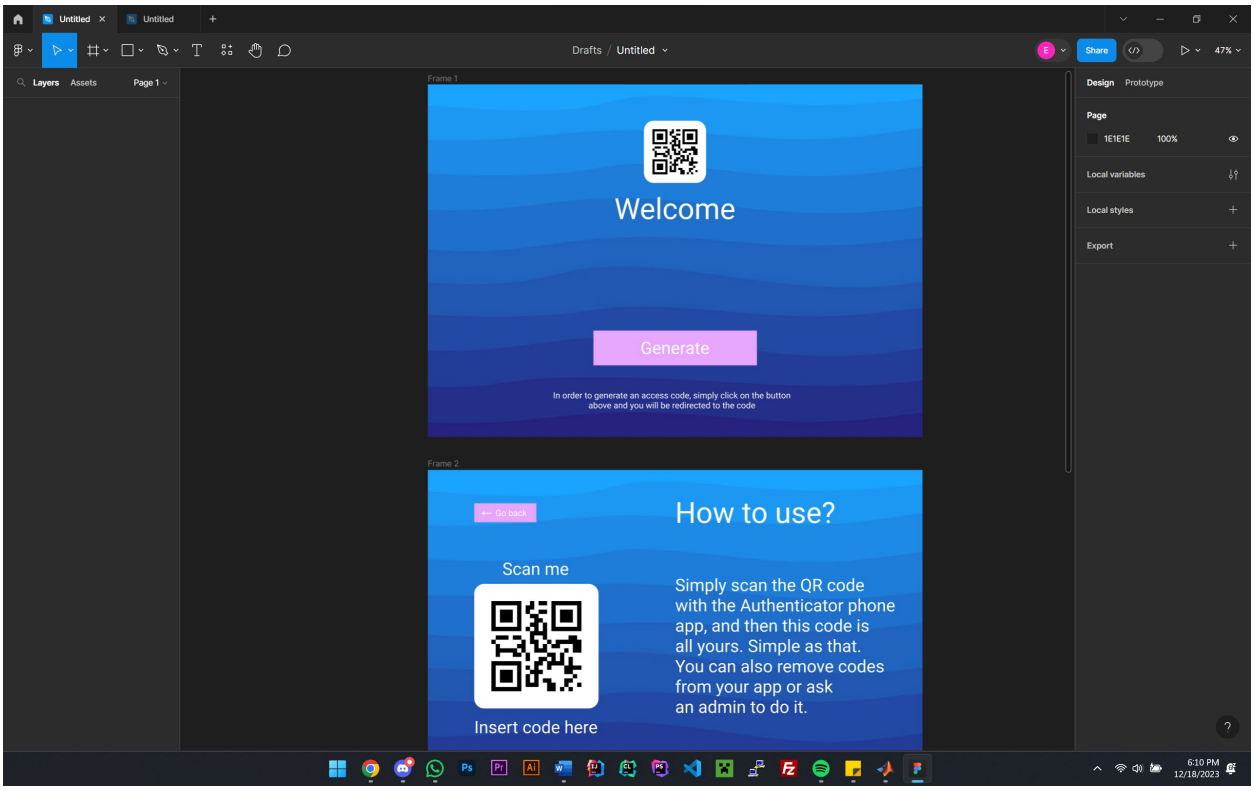
Proiectul nostru contine mai multe module:

- Interfata nativa - Python
    - Generarea de coduri QR
    - Citirea codurilor QR
    - Stocarea codurilor generate in vederea validarii lor
    - Comunicarea cu un dispozitiv embedded (ESP-32, in cazul nostru) care primeste semnal de la aplicatie cum ca autentificarea a avut succes.
- Embedded – C
    - Actionarea unui ecran LCD 16x2 la primirea semnalului de la aplicatie, prin comunicare seriala pentru a ilustra functionalitatea.
- Aplicatie Android – Java
    - O aplicatie pentru a stoca codurile QR

Abordare

Prima incercare

Pentru inceput am dorit sa avem o interfata user-friendly si moderna, cercetand documentatia tkinter am observat ca acesta avea sa fie un lucru complicat si anost, dar totusi am incercat, folosind libraria Tkinter-Designer (https://github.com/ParthJadhav/Tkinter-Designer), ce promite a genera cod python dintr-un proiect Figma, care la inceput a sunat foarte promitator.



Dupa transferul propriu zis catre python, folosind o singura comanda, respectiv `tkdesigner $FILE_URL $FIGMA_TOKEN` am observat ca programul promite mai mult decat livreaza. Desi design-ul era aproape functional, aplicatia nu era responsive, nu lasa o impresie placuta utilizatorului si nu era o optiune ideala.

Solutia

Dupa esecul cu tkdesigner, am dat de o alta librarie folosita de un numar mare de utilizatori, respectiv Eel (https://github.com/python-eel/Eel). Aceasta librarie promitea si mai mult decat, tkdesigner. Eel spune ca pentru a rula programul, creaza un webserver local, si poti scrie backend-ul in python, pe care il poti expune apoi pentru a fi folosit impreuna cu JavaScript si vice-versa. Desi aveam indoieli ca va functiona, am incercat si spre surprinderea mea a functionat. Dupa cateva ore in care am creat design-ul in HTML, CSS si JavaScript interfata grafica avea sa fie gata. Mai departe, urmeaza sa realizam restul programului.

Mod de functionare

Generarea codului

Pentru generare, am folosit o librarie cu o gama foarte larga de functionalitati, putand sa genereze coduri de toate tipurile – normale, micro, artistice, etc. Libraria se numeste segno, si documentatia acestuia poate fi regasita aici: https://segno.readthedocs.io/en/latest/. Astfel, cu cateva linii de cod in care am generat un identificator unic pe baza caruia sa generez codul si in care am reglat diversi parametrii precum marimea codului, formatul in care sa fie salvat, codul a fost generat. Libraria a generat astfel un fisier cu numele identificatorului unic de tipul 'png' in folderul 'web/img', pentru a fi mai usor de accesat prin JavaScript. Apoi, pur si simplu afisez respetiva poza folosind JavaScript.

Stocarea codului

In momentul generarii, acesta este stocat intr-un fisier json, manipulat prin prisma librariei 'json', alaturi de cateva alte date precum:

- Identificatorul unic
- Data si ora la care a fost creat
- Daca a fost validat pana acum sau nu

La fiecare cod generat, in fisierul database.json, este adaugat un nou obiect ce contine datele mentionate mai sus.

Vizualizarea codurilor generate

Codurile deja generate pot fi vizualizate navigand prin interfata grafica, unde putem vizualiza datele codului QR, precum identificatorul unic si data la care a fost creat, avand optiunea sa deschidem poza cu codul QR pentru a o vizualiza.

Citirea codurilor

Pentru citirea codurilor folosim libraria opencv (https://pypi.org/project/opencv-python/). Citirea functioneaza intr-un mod simplu, o noua fereastra este lansata, ce contine feed-ul camerei video. Daca aratam respectivei camere codul QR pe un ecran sau pe o foaie de hartie, acesta decodeaza continutul codului (in cazul in care este valid), il compara cu intrarile din baza de date si in cazul in care gaseste identificatorul in codul scanat, trimite un semnal prin conexiunea seriala ce reprezinta un singur bit, respectiv '1'. De asemenea scrie si in terminalul dispozitivului pe care este rulat programul 'Cod Valid'.

Embedded

Dispozitivul ESP-32 la care este legat laptop-ul pe care ruleaza programul principal, ruleaza in functia loop un program ce asteapta sa primeasca bit-ul '1' prin comunicatia seriala. Cand il primeste, afiseaza pentru 5 secunde textul 'Acces granted' pe ecranul LCD 16x2. Aceasta afisare are scopul de a arata functionalitatea proiectului. Poate fi interpretat ca un scanner de bilet la film / concert. In cazul in care nu este conectat un dispozitiv la portul de comunicatii COM4, va fi afisat doar mesajul din terminal.

Aplicatia Android

To be written

Instructiuni de utilizare

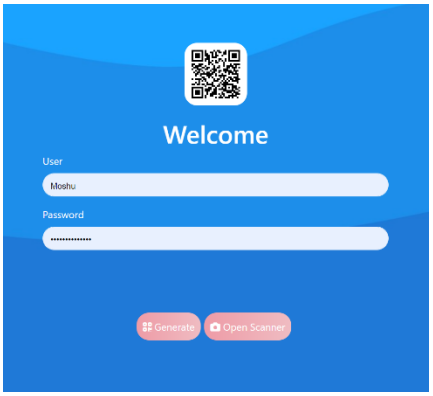
Lansarea programului

Pentru a lansa programul, putem aborda doua modalitati:

1. Executam comanda `python main.py` in terminal daca ne aflam in folderul in care se afla programul. (Daca nu ne aflam, ne putem reloca folosind comanda ‘cd’)
2. Daca folosim IntelliJ PyCharm, putem folosi butonul de ‘Run’ din partea de sus a aplicatiei.

Autentificarea in panoul de administrator

Dupa ce am lansat aplicatia, vom fi intampinati de pagina de autentificare a aplicatiei. Fiind doar un proiect ce doreste sa ilustreze conceptul, am creat o pagina pentru un asa-zis administrator care ar trebui sa creeze aceste coduri. Intr-o aplicatie reala, majoritatea codurilor ar fi create automat.

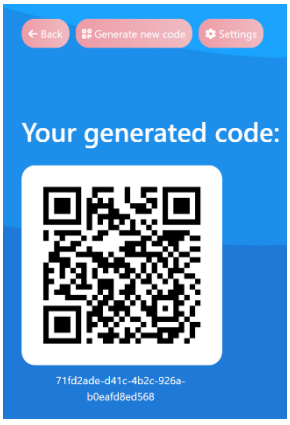


Pentru a continua mai departe trebuie sa introducem combinatia corecta de username si parola. Sistemul de autentificare nu este securizat, din nou, deoarece aceasta aplicatie este menita sa fie doar un demo.

Doua butoane de sub campurile de autentificare ne atrag atentia:

- Generate – genereaza un cod nou
- Open Scanner – deschide scannerul de coduri QR – nu necesita autentificarea.

Odata ce am introdus combinatia corecta, urmatorul meniu va fi lansat:

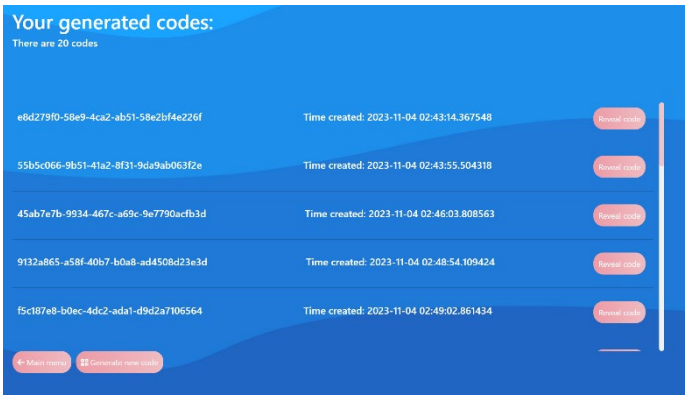


Putem recunoaste cu usurinta codul QR generat, iar sub el este identificatorul unic dupa care a fost generat codul. Urmatorul pas ar fi sa facem o poza codului QR, sa il scanam cu aplicatia Android, sau sa ne transferam poza generata altundeva.

Putem de asemenea, remarca 3 butoane in partea de sus a ecranului:

- Back – inapoi la fereastra principala (de autentificare, un fel de logout)
- Generate new code – genereaza un cod nou (cel generat deja va ramane salvat, desigur)
- Settings – listeaza toate codurile QR generate.

Lista de coduri generate

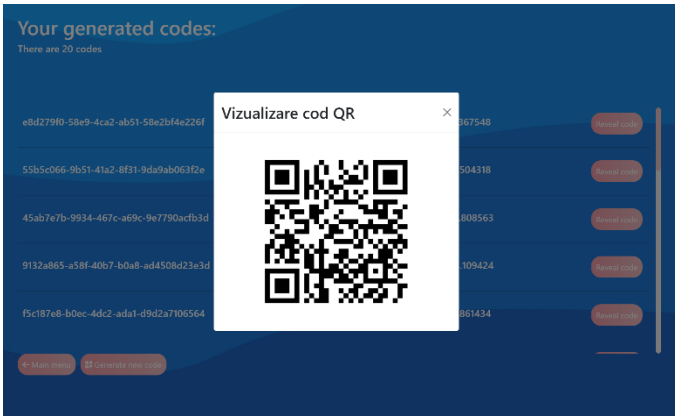


Daca apasam butonul 'Settings' din pagina de generare a codurilor, vom fi redirectionati catre o pagina pe care regasim lista tuturor codurilor generate pana acum, cu urmatorii parametri:

- Identificatorul unic
- Data si ora la care a fost creat
- Butonul pentru a vedea respectivul cod

Odata ce apasam butonul, din dreptul codului pe care dorim sa il vizualizam, va fi afisat un pop-up ce contine respectivul cod, pentru o vizualizare facila si rapida.

Dupa ce terminam de vizualizat codul. Cele doua butoane din partea de jos a ecranului, ne ofera posibilitatea de a naviga la un meniu anterior, respectiv pagina de autentificare sau pagina pentru generarea codurilor QR.



Prerequisites

Pentru a rula programul este nevoie de urmatoarele:

- Programul ruleaza folosind python 3.9
- Librarii:
  - cv2
  - time
  - serial
  - win10toast
  - eel
  - segno
  - uuid
  - json
  - os
  - datetime

Concluzii

Tinta acestui proiect a fost sa construim un ecosistem functional (nu din punctul de vedere al unui release pe piata, ci doar pentru demonstratie) pentru un sistem de autentificare folosind coduri QR, ce se ocupa de la partea de generare si stocare pana la partea de scanare si interactiune cu utilizatorul.

A fost un proiect relaxant, ce nu a avut o dificultate sporita din niciun punct de vedere si a fost distractiv de realizat un proiect scris intr-un limbaj cu care nu eram familiari pana acum.

Nota

Codul poate fi regasit in urmatorul repository: https://github.com/Moshulika/python-project, pe durata desfasurarii proiectului, dupa care vizibilitatea acestuia va fi privata.

Bibliografie

- https://github.com/ParthJadhav/Tkinter-Designer
- https://en.wikipedia.org/wiki/QR_code
- https://blog.hubspot.com/blog/tabid/6307/bid/16088/everything-a-marketer-should-know-about-qr-codes.aspx
- https://github.com/python-eel/Eel
- https://segno.readthedocs.io/en/latest/
- https://pypi.org/project/opencv-python/