Aplicație de transferuri de jucători de fotbal

(mini Transfermarkt)

Marin Cristian-Alexandru

Pentru acest proiect am ales sa implementez un set de servicii care sa compună o aplicație care face management pe o baza de date cu jucători de fotbal și clubul lor sportiv la care activează în mod curent.

După cum știm, în funcție de modul în care evolueaza/involueaza în cariera și performanțe sau înaintează în varsta, jucatorilor le cresc, respectiv le scad prețurile pe piața globală a transferurilor. Pentru o echipa de prestigiu ce dorește performanța in mod constant, dar caruia ii si permite bugetul, tinta principala vor fi mereu cei mai buni jucători, cei mai tineri și cu cel mai mult talent.   
 Un alt aspect de trebuie luat în considerare este concurența între cluburi pentru anumiți jucători populari într-o perioada de transfer, aceștia sunt monitorizati de scouteri cu mai mult timp înainte și mai multe oferte sunt plasate sub forma de licitație pentru a îndeplini două condiții: un preț cel puțin cat cel minim stabilit intern de club, și un contract rezonabil pentru jucător. De aceea multe dintre transferuri se opresc la stadiul de zvon.

Aplicația propusă de mine va avea o instanța de client propriu pentru fiecare club pentru ca acesta sa confirme plecarea unui jucător de la clubul lui către un altul pentru ca au fost îndeplinite condițiile menționate anterior, dar, după propria dorinta, pentru a atrage și atenția altor cluburi de a-și înainte ofertele, sa publice un zvon despre o oferta primita la acel moment de unde licitația poate porni.

Toate transferurile sau zvonurile plasate de un club vor fi transferate către un server care le va procesa și le va stoca în baza de date pentru a fi aprobate de autoritatea supremă(adminul) și expuse în Internet(serviciu de monitorizare web).

În aplicația de web de final, diferite statistici vor putea fi analizate cu ajutorul mediului de monitorizare Grafana care va inspecta noile transferuri plasate în baza de date și fiecare utilizator ce o accesează poate vizualiza informații precum: cel mai scump transfer din acea perioada, o lista cu cei mai valoroși jucători “pe picior de plecare” și așa mai departe precum și multe altele.

Aplicația web de monitorizare are ca public țintă, pe langa managerii cluburilor pentru a fi instiintati, dar și suporteri la modul general, oameni neimplicați direct în procesele de transfer, astfel ca informațiile expuse trebuie sa respecte anumite politici de securitate ale cluburilor deci acestea vor fi simplificate oferindu-se clienților doar ceea ce poate fi publicat.

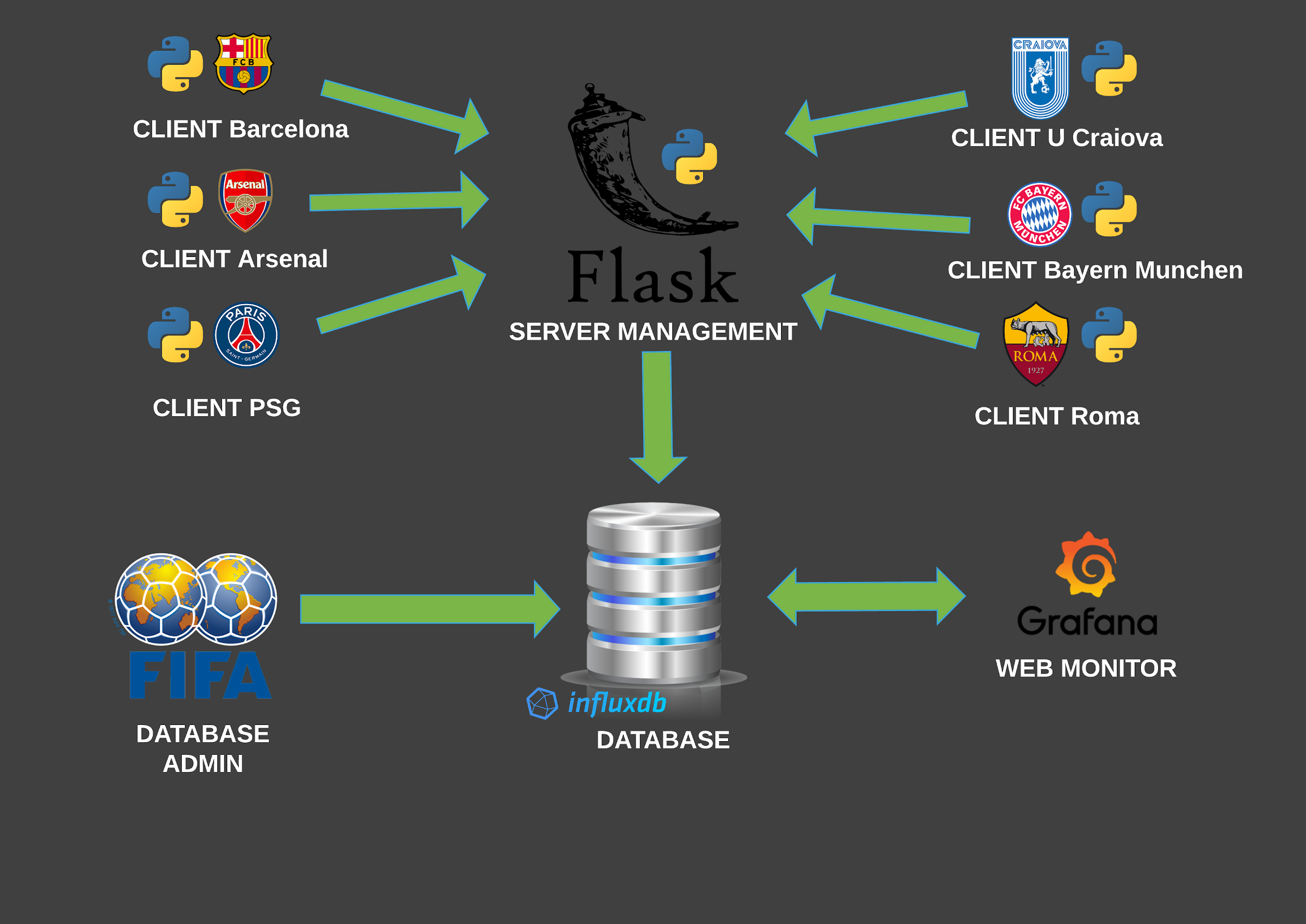
Baza de date va conține informații despre jucători precum: Nume, vârsta(in ani), club actual, club viitor(sau ofertant în cazul unui zvon), valoare jucător, preț oferit și alte informații relevante unui transfer și le va memora în ordinea plasării unui transfer (timestamp database) pentru a pastra veridicitatea și actualitatea informației.

Un alt serviciu îl va reprezenta un administrator de baza de date care va fi instanța unei autoritati supreme (FIFA spre exemplu) care va putea interzice anumite transferuri chiar și dacă acestea au fost în prima faza acceptate de ambele parti, din varii motive (un jucător deja a fost transferat în aceeași perioada de transferuri și nu poate sa mai fie transferat, fairplay financiar: o echipa ce are surse de venit incomparabile cu altele, trebuie limitată prin realizarea unor transferuri pentru a putea păstra farmecul jocului).

Technologies & TOOLS

* DOCKER
* FLASK
* PYTHON
* GRAFANA
* INFLUXDB
* LUCIDCHART
* GIT
* KONG
* CI/CD
* ADMINER

FLOWCHART BUSINESS\_LOGIC (pastrat de la proiectul IDP) la care au fost adaugate elementele specifice pentru Proiectul CC:



Conturi:

1. Docker HUB

https://hub.docker.com/repository/docker/alexmarin88/proiect-cc

1. GITHUB

https://github.com/marin-cristian-alexandru/ProiectCC

Servicii:

* Clientul

Ce va reprezenta un script de python în care se vor prelua datele de la user și se vor pune folosind un request la una din căile la care asculta serverul. Interfața acestuia va fi una minimala realizata în terminal.

* Serverul

Un serviciu ce are și rolul de adaptor intre clienti si serviciul de persistentă. Acesta va fi implementat cu Flask pentru a asculta la mai multe endpointuri pentru a primi date de la clienți și a le adăuga la baza de date cu amprenta de timp curentă

* Baza de date

Va fi un TSDB(cheia de memorare va fi un timestamp) pentru a facilita scopul de a reține ordinea transferurilor. Va stoca într-o tabela cu jucători datele transferului acestora. Pentru aceasta voi folosi o imagine deja existentă de InfluxDB.

* Administrator baza de date

Acesta va fi o autoritate supremă și va fi cel de-al treilea serviciu creat de mine, dupa server și client și va avea dreptul de a analiza transferurile realizate și în caz ca anumite reguli nu sunt respectate, sa nu le permita publicarea, deci o citire si o stergere directă a bazei de date.

* Serviciu de monitorizare

Un serviciu pentru care voi folosi Grafana pentru monitorizare și pe care îl voi lega la baza de date pentru a prelua informațiile și pentru a urmări statistici și informații generale despre transferurile realizate.

* KONG – serviciu API Gateway care mapeaza pentru testare endpointurile din client la /client si din admin la /admin

Fiecare imagine creata de mine va fi creata cu un dockerfile si se va putea porni împreună cu stiva de servicii.

Pentru Grafana se va realiza autentificare personalizată și un set de template-uri de grafice prestabilite. Dashboardul preexistent consta intr-un tabel dupa care pot fi sortati jucatorii listati la transferuri dupa fiecare criteriu care ii clasifica.

In tabela de jucatori, tagul este dat de cele patru tipuri de transferuri:

transfer\_listed (jucator plasat spre vanzare), rumour(zvon de transfer, oferta trimisa neprocesata), confirmed(transfer acceptat de ambele parti) si loaned(cerere de imprumut de jucator).

Datele sunt retinute in TSDB pentru a fi retinut un istoric precis al jucatorilor in ordine cronologica.

Credentialele pentru Grafana sunt admin:proiect idp si portul de acces este 3000.

Portul pentru baza de date este 8086, iar requesturile pentru server se vor face pe portul default pentru Flask, 5000.

Interfetele pentru admin si client sunt construite in python si ofera mai multe operatii specifice fiecaruia dintre instante.

In dockerhub, am creat buildurile automate pentru cele 3 imagini create de mine.

Swarmul este multinod, creat cu ajutorul utilitarului docker-machine, precum in tutorialul din laborator.

In repo poate fi gasit si un document STEPS.txt in care sunt enumerati mai multi pasi si comenzile aferente pentru testare si deploy.