BizyCards

Proiect IDP

Firuți Bogdan-Cristian
Grupa: 341C3

Cuprins

Descrierea temei

Odată cu trecerea timpului, cărțile de vizită devin o adevărată povară. Trebuie păstrate într-un loc sigur, ferit de prea multă lumină, dar un loc în care să fie accesibile mereu. Problema apare atunci când avem nevoie de un contact și trebuie să ne apucăm să scotocim prin ele până găsim persoana căutată.

Ar fi mult mai simplu dacă am avea toate contactele la un loc și am putea căuta rapid prin ele, fără a fi nevoie să stăm ore în șir în căutarea acelui cartonaș, care în final poate avea cerneala ștearsă și poate fi inutilizabil.

Tehnologia este în favoarea noastră, iar aplicația își propune să ajute utilizatorii, oferind o interfață intuitivă și ușor de folosit, o căutare rapidă în funcție de nume, job, companie, domeniul acesteia, oraș sau alte criterii.

Aplicația poate fi folosită în mediul online, fiind pusă la dispoziție de un server, datele fiind stocate într-o bază de date. Astfel, pe baza unui cont pe care utilizatorul îl va crea, va introduce datele sale și ale firmei (în cazul în care nu există deja) și va putea fi găsit foarte ușor de alți utilizatori.

Detalii tehnice

În implementarea aplicației au fost folosite tehnologii de tip **Docker** pentru a putea fi ușor de întreținut. Datorită acestei tehnologii, aplicația este foarte ușor de pornit și rulează în background. Simplitatea provine din faptul că aplicația este împărțită în mai multe servicii conectate între ele. Printre acestea se numără o imagine **mysql 5.7** pentru baza de date, o imagine **grafana** pentru monitorizare și trei imagini care au la baza script-uri **Python** pentru a oferi trei interfete vizuale utilizatorilor..

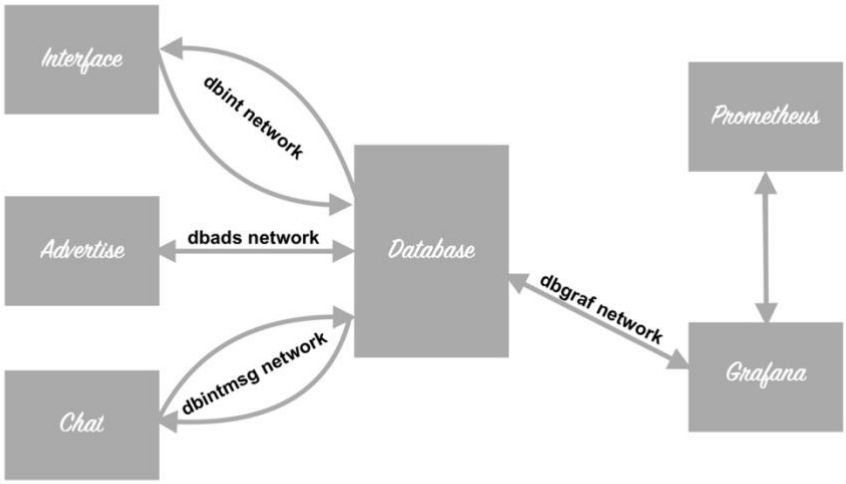
Imaginea mysql este folosită pentru a stoca baza de date. Este folosită o bază de date relațională pentru a putea corela tabelele între ele și a putea obține datele. Am ales o bază de date relațională pentru a nu stoca toate datele legate de companie, job, oraș, țară în fiecare intrare din baza de date. Astfel, mențin doar un identificator pe baza căruia pot realiza join-uri pentru a obține toate datele necesare.

Imaginea grafana este folosită pentru a putea prezenta câteva grafice pe care administratorul să le poată observa. Datele corelate **nu** sunt publice, ele reprezentând statistici precum procentul de persoane care lucrează în altă țară sau procentul de persoane care lucrează în alt oraș. Astfel, grafana este un mod mai plăcut de a realiza graficele.

Imaginea custom **interface** creată pe baza script-urilor Python reprezintă, de fapt, modul prin care utilizatorii pot accesa aplicația. Am folosit **Sanic** pentru a putea face aplicația publică în mediul online. Utilizatorul va trebui doar să acceseze o pagina web pentru a folosi aplicația. Astfel, am crescut nivelul de simplitate. Acesta va putea accesa baza de date și va putea căuta atât anumite persoane, cât și firme. De aici va avea acces si administratorul la log-uri și la partea de monitorizare.

Imaginea custom **interface_msg** a fost gândită pentru a oferi o modalitate mai rapidă de a comunica cu persoanele pe care le caută utilizatorul. Acest serviciu joacă rolul unui ”chat” online prin care putem contacta pe cineva folosind adresa de email.

Imaginea custom **advertise** foloseste tot **Sanic** pentru a pune la dispoziția utilizatorului o pagina unde poate vedea reclame de la firme, poate asculta radio și poate vedea vremea în București. În momentul în care utilizatorul apasă pe o imagine, i se deschide o noua fereastră în care apar datele firmei și locația.



Descrierea bazei de date

După cum am menționat mai sus, baza de date utilizată este de tip mysql. Datorită faptului că am folosit docker, am creat un serviciu care să ruleze această imagine. Aceasta are un script de inițializare în care se creează tabelele, funcțiile, procedurile și trigger-ele. Totodată, datele sunt persistente deoarece în secțiunea ”volumes”, pe lângă script-ul de inițializare, este trecut un folder unde se vor stoca datele. Folder-ul este ținut pe stația care rulează docker-ul.

Diagrama bazei de date

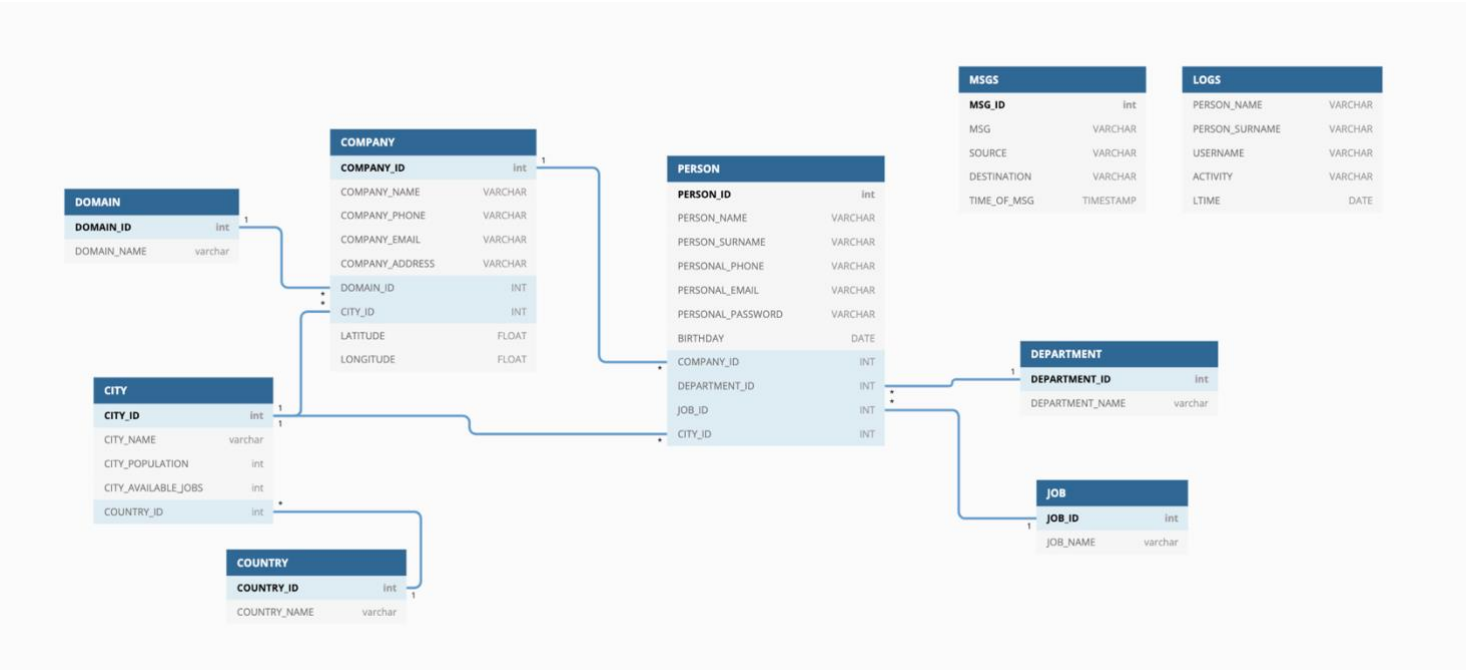


Diagrama poate fi obținută folosind site-ul https://dbdiagram.io/home, introducând datele din fișierul **diagram_generator.io**.

Structura bazei de date

Baza de date conține 9 tabele (ilustrate și în imaginea de mai sus). Sunt marcate cheile primare (îngroșate și au denumiri de forma ***_ID**), iar prin liniile albastre sunt marcate cheile de tip **Foreign Key**. Astfel, pentru a putea adăuga un utilizator, trebuie să existe deja departamentul, job-ul, orașul și compania. Pentru a putea adăuga o companie, avem nevoie ca domeniul și orașul să existe.

Detalii legate de fiecare tabelă în parte:

- **LOGS** – tabela în care se vor păstra informații legate atât de noii utilizatori, cât și despre cei care și-au șters contul
- **MSGS** – tabela care păstrează mesajele utilizatorilor
- **DEPARTMENT** – tebela în care sunt stocate informații legate de departament (momentan e păstrat doar numele, dar se poate extinde)
- **JOB** – tabela în care sunt păstrate informații legate de job (momentan doar numele, dar se poate adăuga o scurtă descriere, numărul de ore sau alte informații)
- **COUNTRY** – tabela în care sunt stocate informații legate de țară (momentan e păstrat doar numele)
- **DOMAIN** – tabela în care sunt păstrate informații legate de domeniul de activitate al unei companii (momentan e păstrat doar numele)
- **CITY** – tabela în care sunt păstrate informații legate de oraș (nume, populație, locuri de muncă disponibile, țară)
- **COMPANY** – tabela care conține informații legate de companii (nume, telefon, email, adresă, id domeniu și id oraș). Putem folosi id-urile de domeniu și oraș pentru a corela datele cu tabelele **DOMAIN** și **CITY**. Astfel obținem toate informațiile de care avem nevoie
- **PERSON** – tabela principală, care conține datele utilizatorilor. Printre acestea se numără numele, prenumele, telefonul, adresa de email, parola, data nașterii și câte un id pentru companie, departament, job și oraș pentru a putea corela datele cu tabelele **COMPANY**, **DEPARTMENT**, **JOB**, **CITY**

Observație!

Parola nu este păstrată ”în clar”. În momentul în care un utilizator își face cont, script-ul Python calculează pe baza parolei o cheie de tip **SHA256**. Aceasta este stocată în baza de date, iar în momentul în care un utilizator încearcă să se conecteze, parola introdusă este transformată într-o cheie SHA256 și se verifică dacă aceasta este identică cu cea păstrată în baza de date corespunzătoare numelui de utilizator introdus.

Utilizatorii vor putea să acceseze aplicația prin pagina de log in folosind adresa de email și parola aferentă contului.

Descrierea constrângerilor de integritate

Detalii legate de fiecare tabelă în parte:

- **LOGS** – tabela are pe fiecare coloană constrângerea de tip **NOT NULL**
- **MSGS** – tabela are pe fiecare coloană constrângerea de tip **NOT NULL**
- **DEPARTMENT** – coloana DEPARTMENT_ID are constrângeri de tip **NOT NULL**, **AUTO_INCREMENT**, **PRIMARY KEY**, în timp ce DEPARTMENT_NAME are constrângeri de tip **NOT NULL, UNIQUE**
- **JOB** – coloana JOB_ID are constrângeri de tip **NOT NULL**, **AUTO_INCREMENT**, **PRIMARY KEY**, în timp ce coloana JOB_NAME are constrângeri de tip **NOT NULL, UNIQUE**
- **COUNTRY** – coloana COUNTRY_ID are constrângeri de tip **NOT NULL**, **AUTO_INCREMENT**, **PRIMARY KEY**, în timp ce coloana COUNTRY _NAME are constrângeri de tip **NOT NULL, UNIQUE**
- **DOMAIN** – coloana DOMAIN_ID are constrângeri de tip **NOT NULL**, **AUTO_INCREMENT**, **PRIMARY KEY**, în timp ce coloana DOMAIN_NAME are constrângeri de tip **NOT NULL, UNIQUE**
- **CITY** – coloana CITY_ID are constrângeri de tip **NOT NULL**, **AUTO_INCREMENT**, **PRIMARY KEY**, coloana CITY_NAME are constrângeri de tip **NOT NULL, UNIQUE**, în timp ce restul nu au constrângeri. Totodată, pe coloana COUNTRY_ID este pusă o constrângere de tip **FOREIGN KEY** pentru a asigura existența țării în momentul introducerii unui oraș
- **COMPANY** – coloana COMPANY_ID are constrângeri de tip **NOT NULL**, **AUTO_INCREMENT**, **PRIMARY KEY**, coloana COMPANY_NAME are constrângeri de tip **NOT NULL, UNIQUE**, iar COMPANY_PHONE, COMPANY_EMAIL și COMPANY_ADDRESS au constrângeri de tip **NOT NULL**. Totodată, pe coloanele DOMAIN_ID și CITY_ID sunt puse constrângeri de tip **FOREIGN KEY** pentru a asigura existența domeniului și a orașului în momentul introducerii unei companii
- **PERSON** – coloana PERSON_ID are constrângeri de tip **NOT NULL**, **AUTO_INCREMENT**, **PRIMARY KEY**, coloana PERSON_EMAIL are constrângeri de tip **NOT NULL, UNIQUE**, iar PERSON_NAME, PERSON_SURNAME, PERSONAL_PHONE și PERSONAL_PASSWORD au constrângeri de tip **NOT NULL**. Totodată, pe coloanele COMPANY_ID, DEPARTMENT_ID, JOB_ID, CITY_ID sunt puse constrângeri de tip **FOREIGN KEY** pentru a asigura existența companiei, departamentului, job-ului și a orașului în momentul introducerii unei persoane

Descrierea funcțiilor și a procedurilor

Pentru a putea implementa aplicația, a fost nevoie de mai multe funcții și proceduri.

Totodată am adăugat două trigger-e pentru a putea urmări cu ușurință ce se întâmplă cu conturile utilizatorilor (când sunt adăugați noi utilizatori sau când sunt șterse conturi).

Următoarele proceduri au fost implementate:

- **ADD_CITY** – are parametri numele orașului, numele țării și adaugă date în CITY
- **ADD_COUNTRY** – are parametru numele țării și adaugă date în COUNTRY
- **ADD_JOB** – are parametru numele job-ului și adaugă date în JOB
- **ADD_DEPARTMENT** - are parametru numele departamentului și adaugă date în DEPARTMENT

- **ADD_DOMAIN** – are parametru numele domeniului și adaugă date în DOMAIN
- **ADD_COMPANY** – are parametri numele companiei, telefonul companiei, email-ul, adresa, domeniul, țara și orașul și adaugă date în COMPANY
- **ADD_PERSON** – are parametri numele, prenumele, telefonul, email-ul, parola (SHA256), data nașterii, compania, departamentul, job-ul, țara și orașul și adaugă date în PERSON
- **UPDATE_PERSON** – are parametri user-ul, numele, prenumele, telefonul, email-ul, parola (SHA256), data nașterii, compania, departamentul, job-ul, țara și orașul și modifică datele din PERSON corespunzătoare user-ului
- **UPDATE_PERSON_NO_PASS** – la fel ca UPDATE_PERSON, doar că nu modifică parola, ci doar datele legate de user
- **UPDATE_COMPANY** - are parametri numele dinainte de schimbare, numele companiei, telefonul companiei, email-ul, adresa, domeniul, țara și orașul și modifică datele din COMPANY
- **DELETE_BY_EMAIL** – are ca parametru email-ul și șterge din PERSON înregistrarea corespunzătoare email-ului
- **DELETE_COMPANY_BY_NAME** – are ca parametru numele companiei și șterge din COMPANY înregistrarea corespunzătoare
- **CHECK_EMAIL** – are ca parametru un email și verifică dacă acesta există în baza de date
- **CHECK_COMPANY_NAME** – are ca parametru un nume de companie și verifică dacă acesta există în baza de date
- **SEARCH_BY_NAME** – are ca parametru câteva litere. Acestea vor fi căutate în numele sau prenumele din tabela PERSON folosind construcția
  LIKE '%LOWER(NAME)%' care semnifică faptul că putem avea orice înainte sau după literele primite ca parametru (nu e obligatoriu să facă match exact. De exemplu, putem căuta 'pop' și vom găsi inclusiv înregistrările unde numele e 'Popescu' sau 'Protopopescu')
- **SEARCH_BY_EMAIL** – are ca parametru o adresă de email și se caută în PERSON înregistrarea unde coloana PERSONAL_EMAIL este identică cu email-ul dat ca parametru
- **SEARCH_BY_COMPANY** – are ca parametru câteva litere. Acestea vor fi căutate în numele companiilor din tabela PERSON folosind construcția
  LIKE '%LOWER(NAME)%' care semnifică faptul că putem avea orice înainte sau după literele primite ca parametru (nu e obligatoriu să facă match exact. De exemplu, putem căuta 'ITs' și vom găsi inclusiv înregistrările unde numele companiei e 'ITSecurity' sau 'FitsBone')
- **SEARCH_BY_CITY** - are ca parametru câteva litere. Acestea vor fi căutate în numele orașelor din tabela PERSON folosind construcția
  LIKE '%LOWER(NAME)%' care semnifică faptul că putem avea orice înainte sau după literele primite ca parametru (nu e obligatoriu să facă match exact. De exemplu, putem căuta 'OR' și vom găsi inclusiv înregistrările unde numele companiei e 'Oradea' sau 'New York')
- **SEARCH_BY_JOB** – are ca parametru câteva litere. Acestea vor fi căutate în numele job-urilor din tabela PERSON folosind construcția
  LIKE '%LOWER(NAME)%' care semnifică faptul că putem avea orice înainte sau după literele primite ca parametru (nu e obligatoriu să facă match exact. De exemplu, putem căuta 'PrO' și vom găsi inclusiv înregistrările unde numele companiei e 'Programmer sau 'Junior Programmer')

- **SEARCH_ALL_COMPANIES** – nu are parametri. Va afișa toate companiile
- **SEARCH_COMPANY_BY_NAME** – la fel ca SEARCH_BY_COMPANY, doar că va căuta datele în tabela COMPANY, nu în PERSON (va lua datele legate de companii, nu cele legate de persoanele care lucrează la o anumită companie)
- **SEARCH_COMPANY_BY_DOMAIN** – primește câteva litere reprezentând un domeniu și caută companiile care au în denumirea domeniului acele litere
- **SEARCH_COMPANY_BY_CITY** – primește câteva litere reprezentând un oraș și caută companiile care au în denumirea orașului acele litere
- **SEARCH_ALL** – nu are parametri. Va afișa toți utilizatorii
- **SHOW_LOGS** – nu are parametri. Va afișa toate mesajele de log
- **LOGIN_USER** – procedură ce primește adresa de email și parola (SHA256) și verifică dacă acestea se află în baza de date

Următoarele proceduri sunt folosite împreună cu grafana:

- **GROUP_BY_CITY** – nu are parametri. Va afișa numărul de persoane grupate după denumirea orașului din care sunt
- **GROUP_BY_COUNTRY** – nu are parametri. Va afișa numărul de persoane grupate după denumirea țării din care sunt
- **GROUP_BY_JOB** – nu are parametri. Va afișa numărul de persoane grupate după denumirea job-ului
- **GROUP_BY_DEPARTMENT** – nu are parametri. Va afișa numărul de persoane grupate după denumirea departamentului din care fac parte
- **GROUP_BY_DOMAIN** – nu are parametri. Va afișa numărul de persoane grupate după domeniul firmei din care fac parte
- **GROUP_BY_COMPANY** – nu are parametri. Va afișa numărul de persoane grupate după denumirea firmei din care fac parte
- **SHOW_LAST_MSGS** – întoarce ultimele 20 de mesaje dintre două persoane
- **SHOW_ALL_MSGS** – întoarce toate mesajele dintre două persoane

Următoarele funcții au fost implementate (tot pentru grafana):

- **OTHER_CITY** – întoarce procentul de persoane care lucrează în alt oraș decât cel natal
- **OTHER_COUNTRY** – întoarce procentul de persoane care lucrează în altă țară
- **ADD_MSG** – adaugă un mesaj în baza de date

Următorii triggeri au fost implementați:

- **ADD_LOG_NEW_PERSON** – adaugă o nouă înregistrare în tabela LOGS în momentul în care cineva își creează cont (sunt inserate date în PERSON)
- **ADD_LOG_DEL_PERSON** – adaugă o nouă înregistrare în tabela LOGS în momentul în care cineva își șterge contul (sunt șterse date din PERSON)

Descrierea aplicației

Aplicația este construită folosind Docker. Sunt create cinci servicii (mysql, grafana și imaginile custom folosind script-uri Python). Ea a fost gândită pentru a fi folosită în mediul online.

Utilizatorului îi este pusă la dispoziție o interfață prietenoasă pentru a-și introduce datele, atât ale sale, cât și ale companiei la care lucrează. Odată creat contul, acesta va putea să vadă

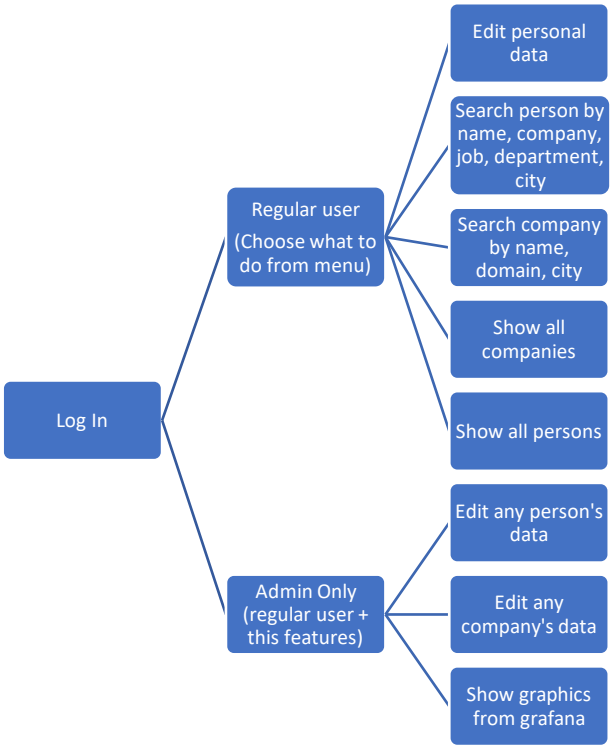
toate companiile și toți utilizatorii înscriși în sistem. Totodată, el va putea să caute printre aceste date în mod rapid, folosind meniul intuitiv.

Totul este accesibil prin intermediul unui site web.

Script-urile Python care formează imaginea custom a unuia din cele trei servicii care este construit pe baza Sanic, pentru a putea crea un server care să pună la dispoziție anumite pagini ce pot fi accesate printr-un URL. Acestea generează pagini folosind HTML și trimit cereri către baza de date. Cererile sunt simple apeluri de funcții sau proceduri, fiind interpretat rezultatul întors de acestea. Odată primit un răspuns de la baza de date (cel de-al doilea serviciu creat prin imaginea mysql 5.7), acesta este parsat, se creeaza pagina HTML și este trimis răspunsul către utilizator.

Grafana este folosită pentru a afișa câteva grafice atât în legătură cu datele din baza de date, cât și date de monitorizare. Acestea pot fi văzute și editate doar de către administrator. Graficele sunt create de grafana tot în urma apelurilor de funcții și proceduri.

Diagrama de stări și fluxul aplicației



Orice utilizator (fie el utilizator normal sau admin) trebuie să acceseze pagina de Log In și să introducă credențialele (email și parolă sau, pentru admin, user: admin, password: admin). Odată confirmată identitatea, acestuia i se prezintă un meniu intuitiv. Din meniu poate selecta să-și editeze datele personale, poate căuta o anumită persoană sau o anumită companie, poate vedea toate persoanele înscrise sau poate vedea toate companiile din sistem.

Un administrator are în plus față de aceste opțiuni să editeze datele oricărei persoane, să editeze datele oricărei companii sau să vizualizeze graficele disponibile în grafana. Totodată, acesta poate modifica graficele și crea altele noi.

Modul în care se face conexiunea cu baza de date

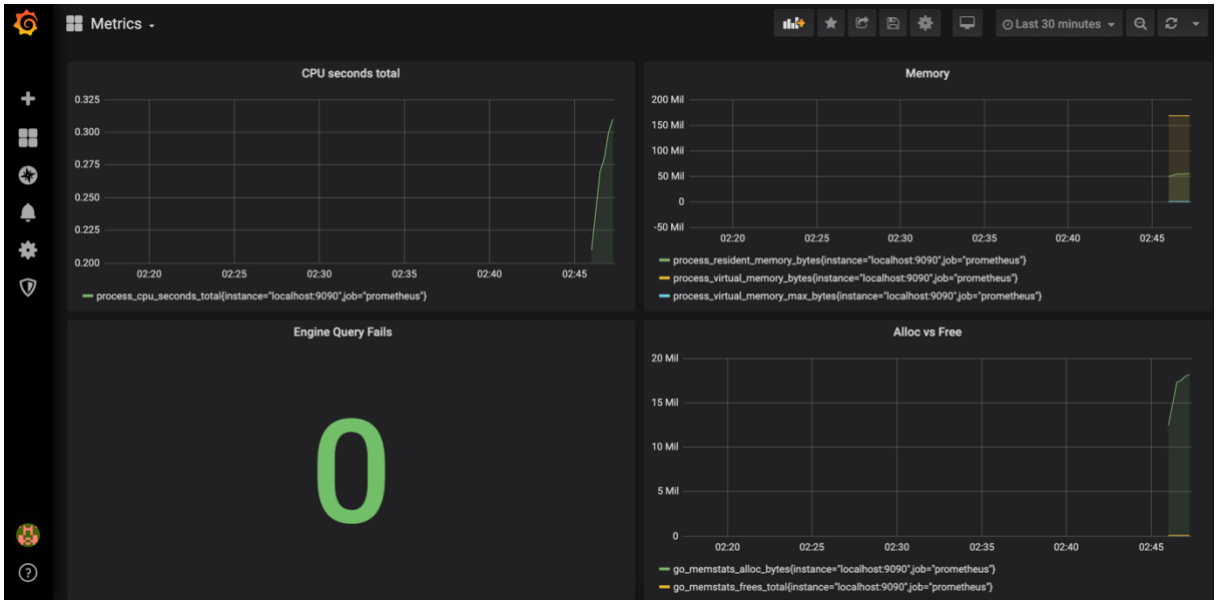
Datorită faptului că este folosit docker-compose, fiecare imagine devine un servicu de sine stătător și este creată o rețea care le leagă pe toate. Putem accesa direct servicul dintr-un al serviciu folosind drept hostname numele acestuia.

Astfel, imaginea custom create prin script-urile Python folosesc **mysql-connector** pentru a se conecta la baza de date. Serviciul mysql 5.7 pune la dispoziție un port (3306) prin care ne putem conecta la baza de date din alt serviciu. Efectiv vom crea o conexiune prin intermediul mysql-connector, având ca host numele serviciului mysql (”db”), user-ul și parola sunt cele setate prin parametri în momentul în care am creat serviciul mysql (user: ”root”, password: ”password”), portul este cel implicit (3306) pe care îl expune serviciul mysql, iar baza de date este cea pe care am creat-o prin script-ul de inițializare (”CARDS”).

Odată conectați la baza de date, putem apela proceduri și funcții stocate.

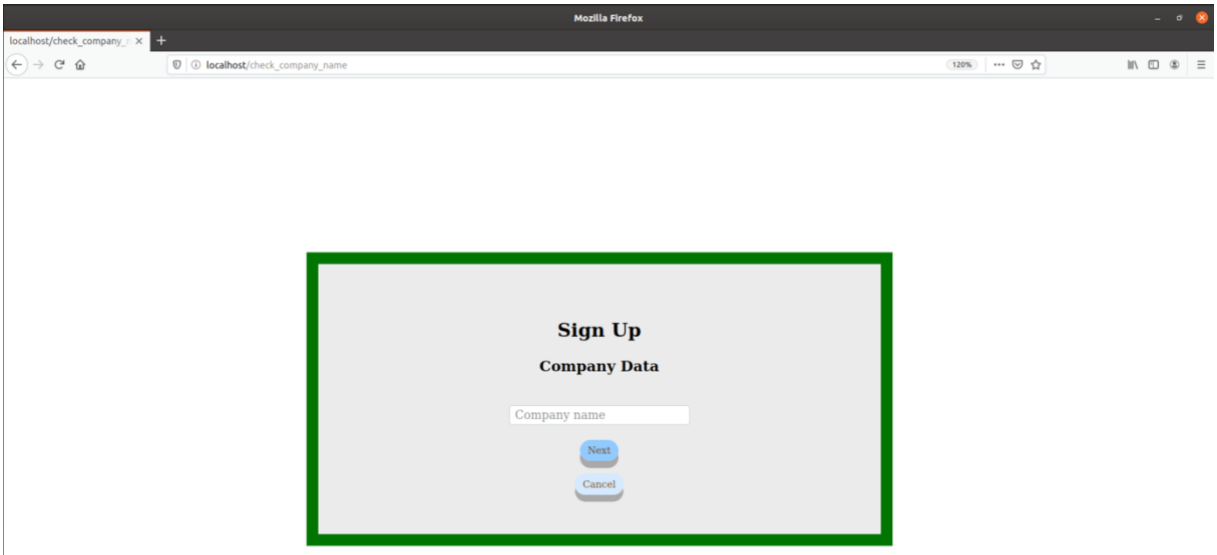
Monitorizare

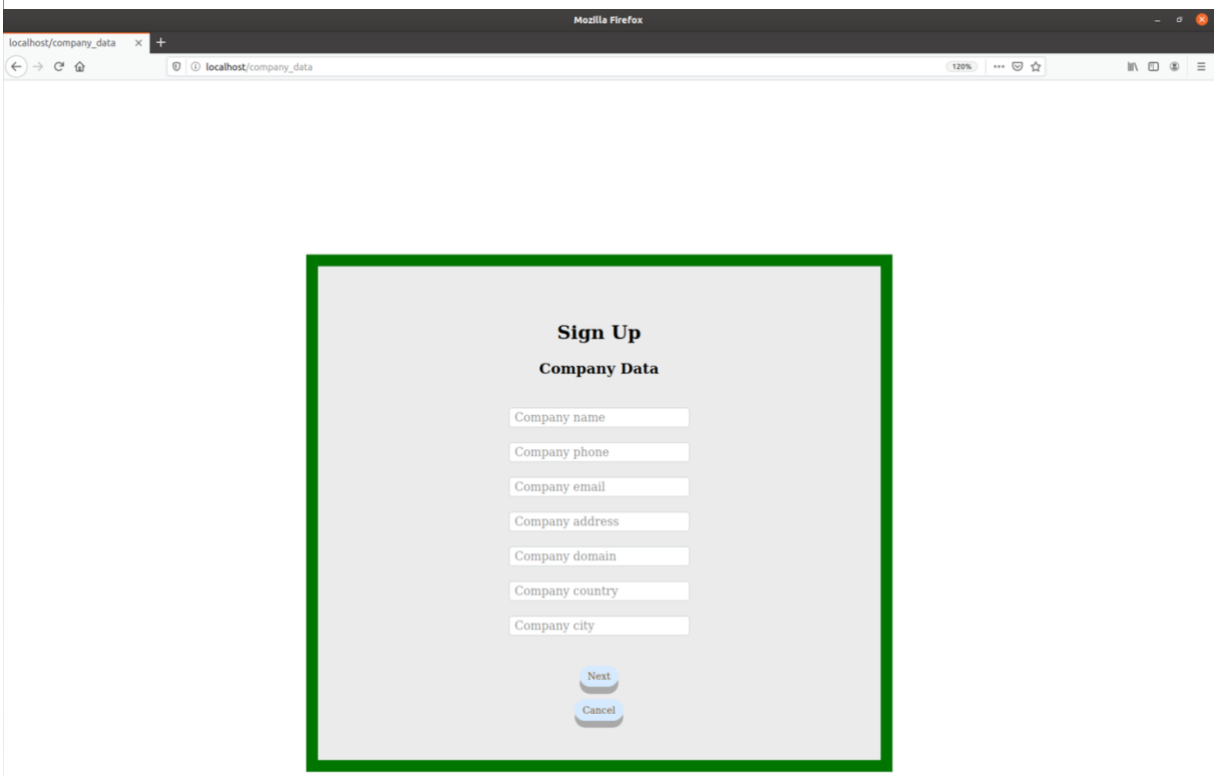
Pentru partea de monitorizare am folosit **Prometheus** împreună cu **Grafana**. Am realizat cateva query-uri simple pentru a demonstra funcționalitatea serviciului.

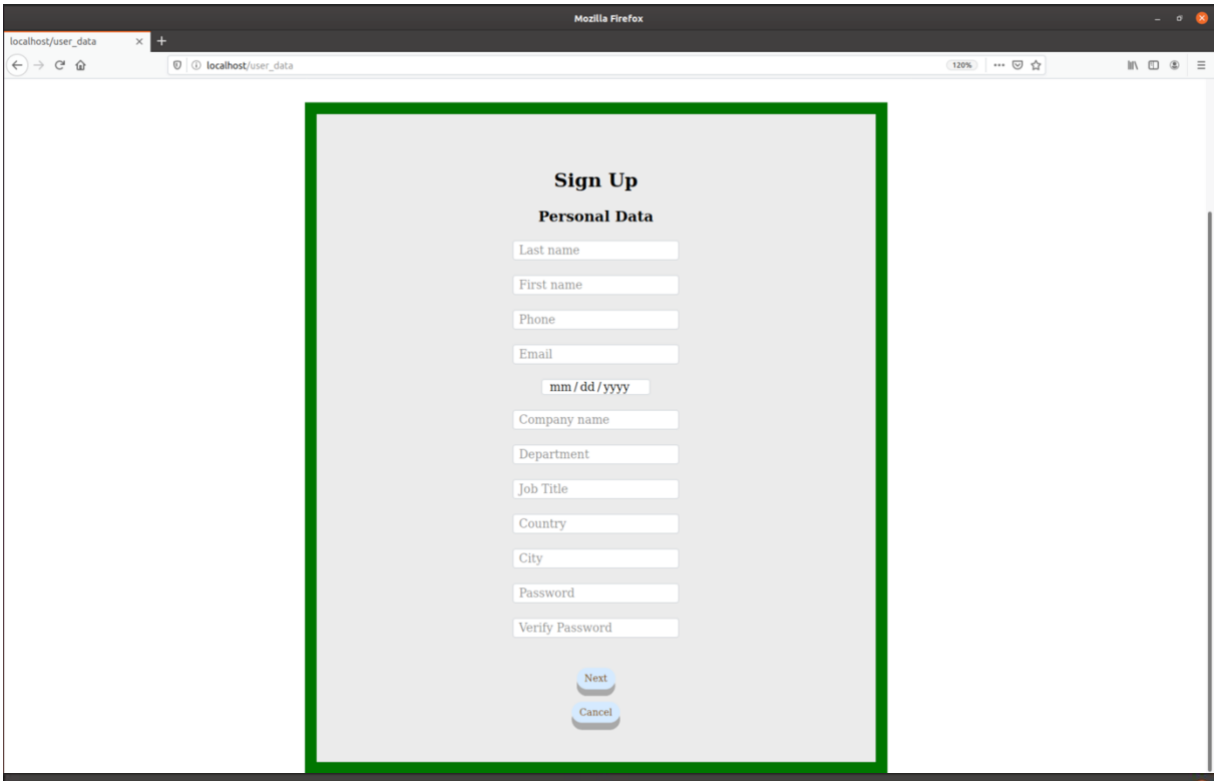


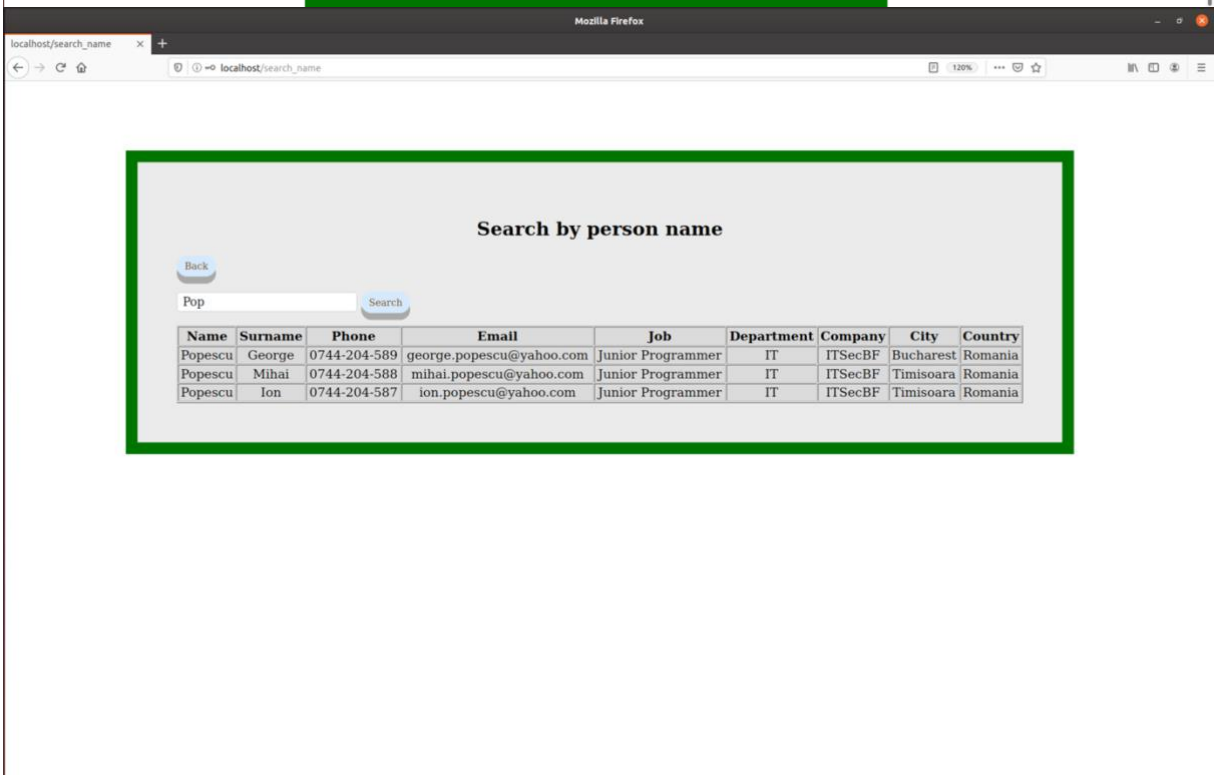
Capturi de ecran

Comun atât pentru un utilizatorul normal, cât și pentru un administrator:



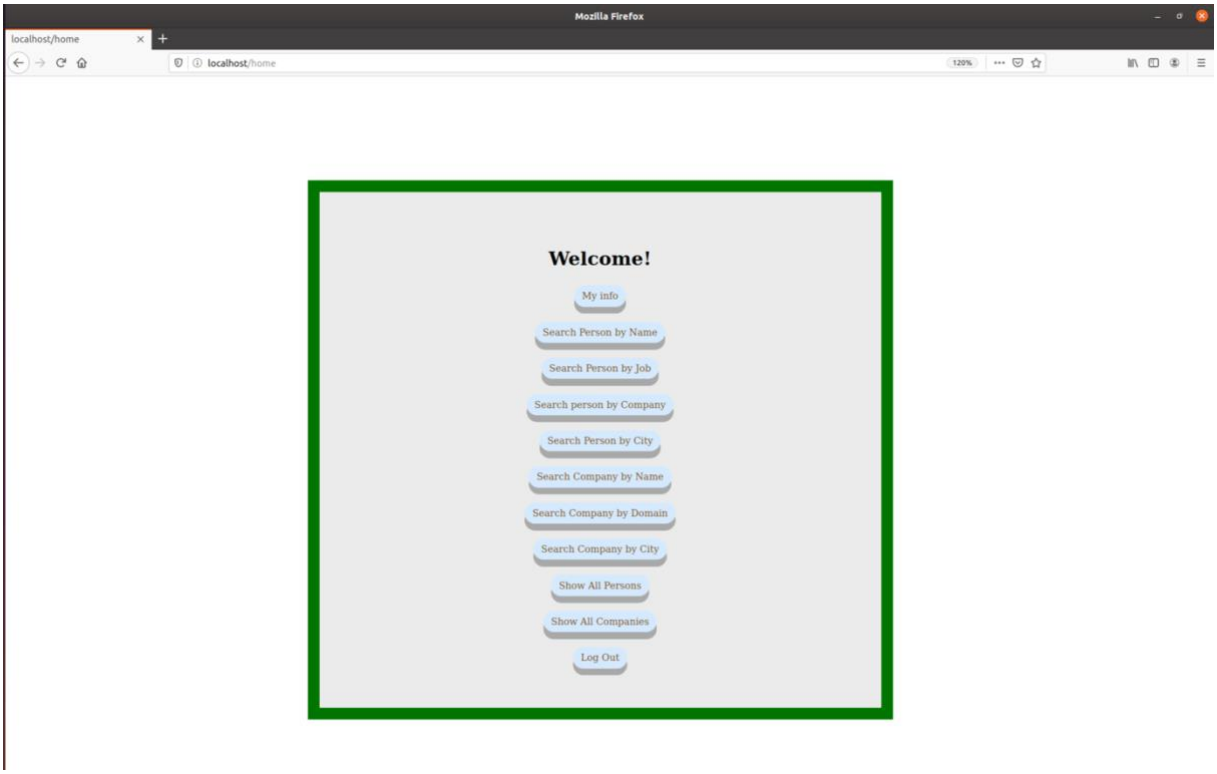




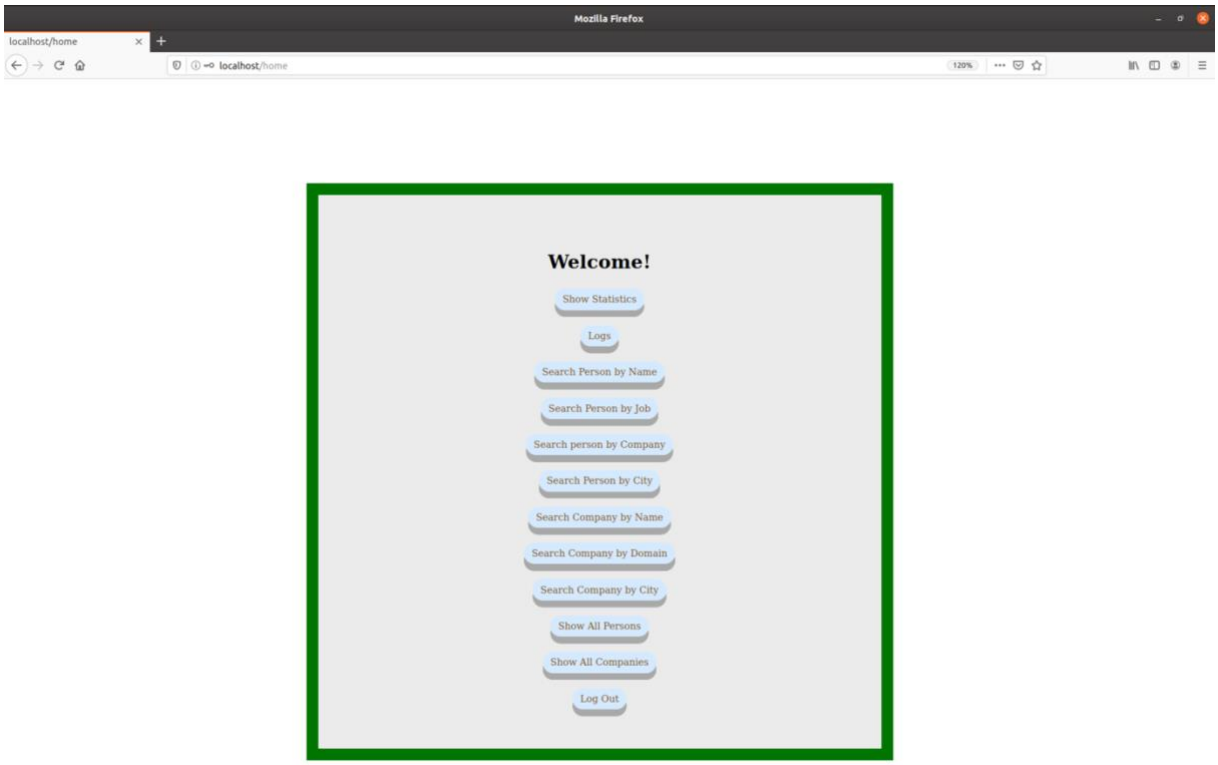


Vizualizari diferite pentru aceeași pagină:

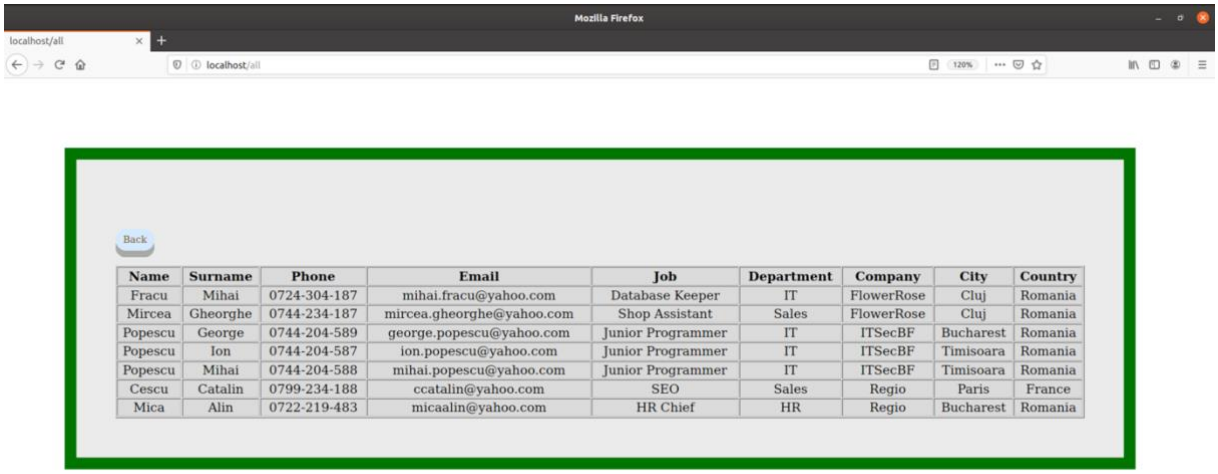
- Meniul utilizatorului normal:



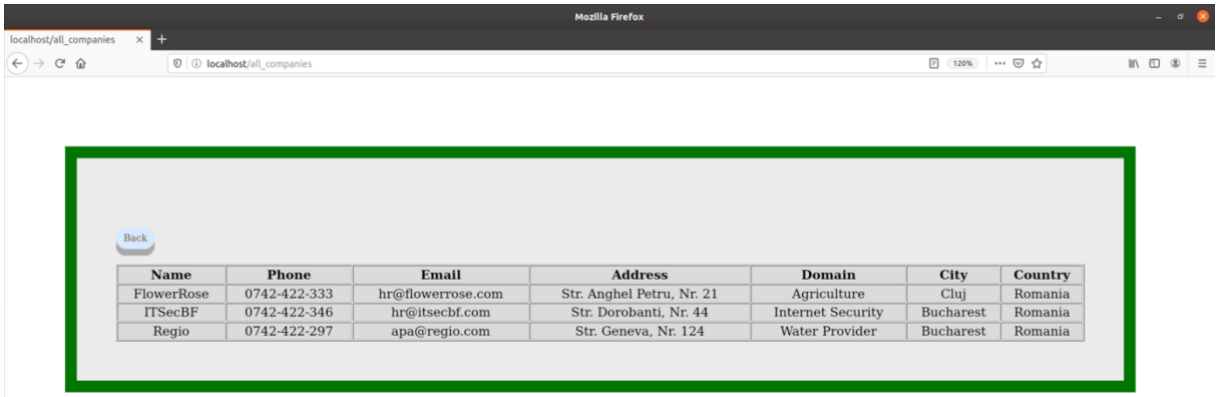
- Meniul administratorului:



- Vizualizare utilizator normal:

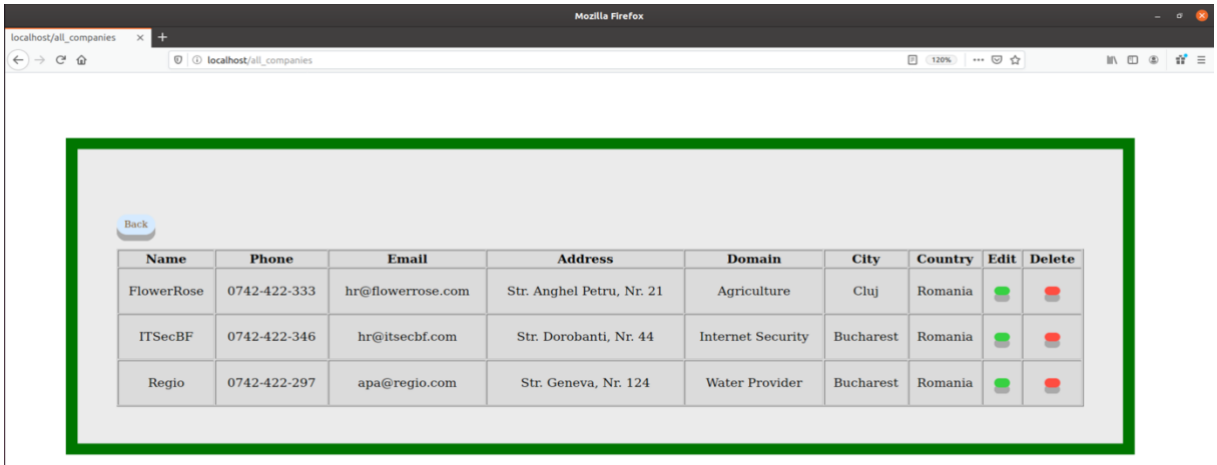


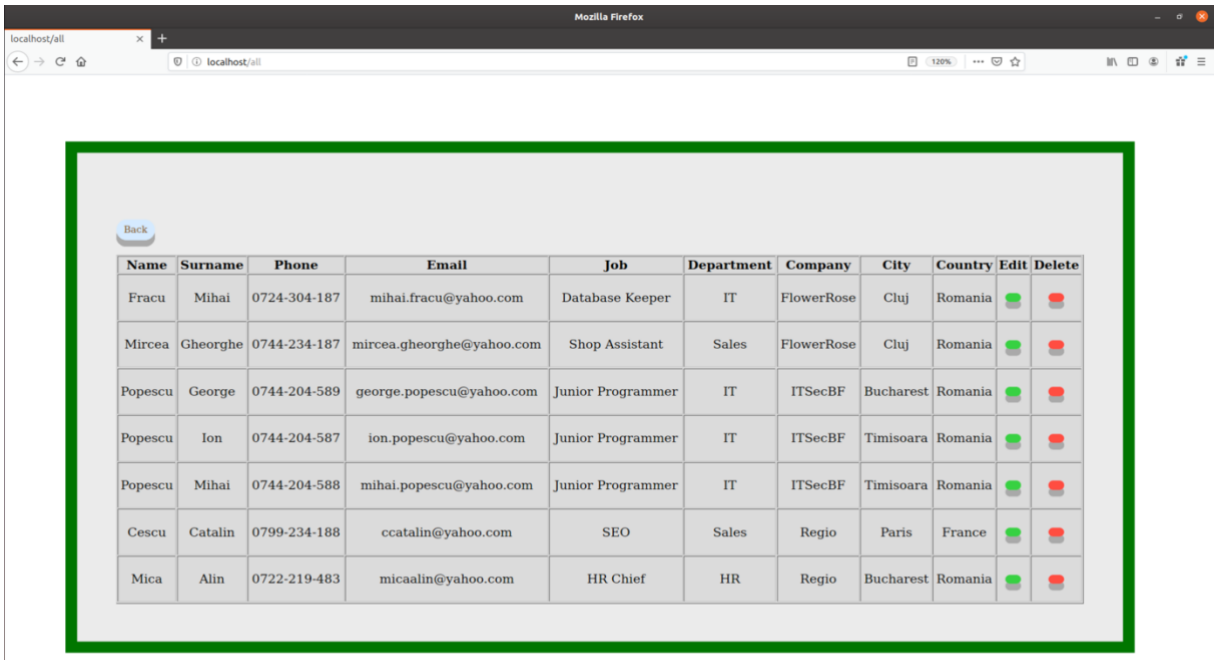
Name	Surname	Phone	Email	Job	Department	Company	City	Country
Fracu	Mihai	0724-304-187	mihai.fracu@yahoo.com	Database Keeper	IT	FlowerRose	Cluj	Romania
Mircea	Gheorghe	0744-234-187	mircea.gheorghe@yahoo.com	Shop Assistant	Sales	FlowerRose	Cluj	Romania
Popescu	George	0744-204-589	george.popescu@yahoo.com	Junior Programmer	IT	ITSecBF	Bucharest	Romania
Popescu	Ion	0744-204-587	ion.popescu@yahoo.com	Junior Programmer	IT	ITSecBF	Timisoara	Romania
Popescu	Mihai	0744-204-588	mihai.popescu@yahoo.com	Junior Programmer	IT	ITSecBF	Timisoara	Romania
Cescu	Catalin	0799-234-188	ccatalin@yahoo.com	SEO	Sales	Regio	Paris	France
Mica	Alin	0722-219-483	micaalin@yahoo.com	HR Chief	HR	Regio	Bucharest	Romania

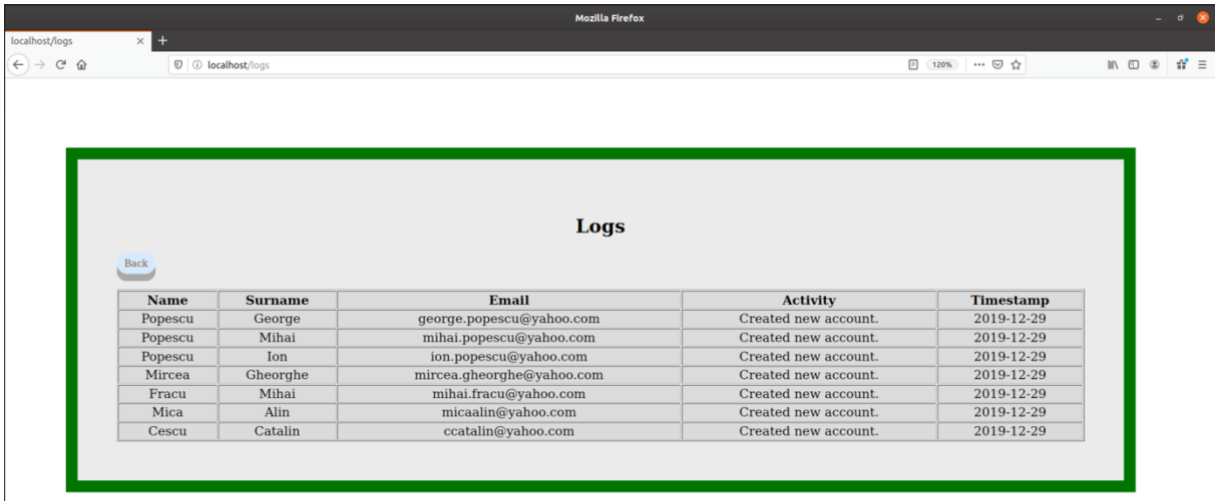


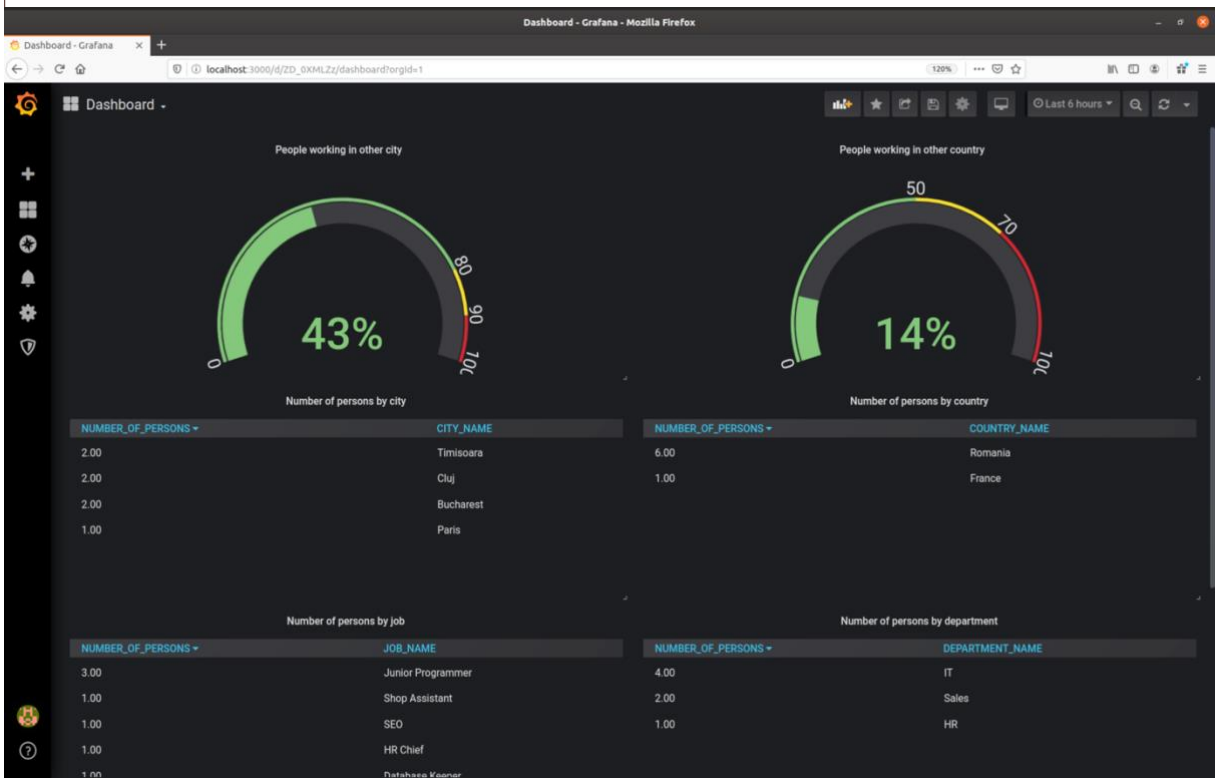
Name	Phone	Email	Address	Domain	City	Country
FlowerRose	0742-422-333	hr@flowerrose.com	Str. Anghel Petru, Nr. 21	Agriculture	Cluj	Romania
ITSecBF	0742-422-346	hr@itsecbf.com	Str. Dorobanti, Nr. 44	Internet Security	Bucharest	Romania
Regio	0742-422-297	apa@regio.com	Str. Geneva, Nr. 124	Water Provider	Bucharest	Romania

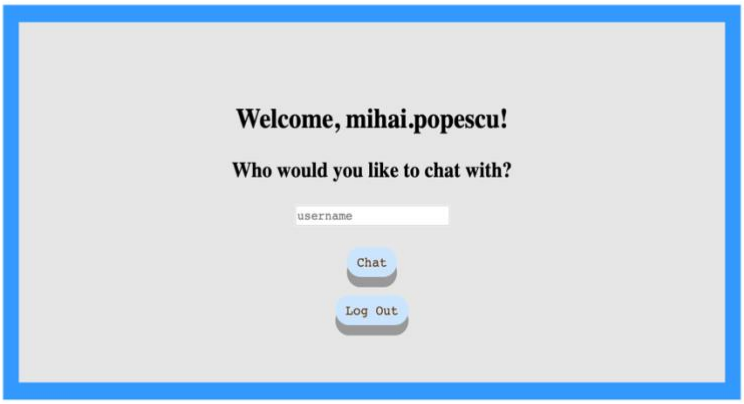
- Vizualizare administrator:


Mozilla Firefox
localhost/all_companies
localhost/all_companies
Back
Name
Phone
Email
Address
Domain
City
Country
Edit
Delete
FlowerRose
0742-422-333
hr@flowerrose.com
Str. Anghel Petru, Nr. 21
Agriculture
Cluj
Romania
ITSecBF
0742-422-346
hr@itsecbf.com
Str. Dorobanti, Nr. 44
Internet Security
Bucharest
Romania
Regio
0742-422-297
apa@regio.com
Str. Geneva, Nr. 124
Water Provider
Bucharest
Romania

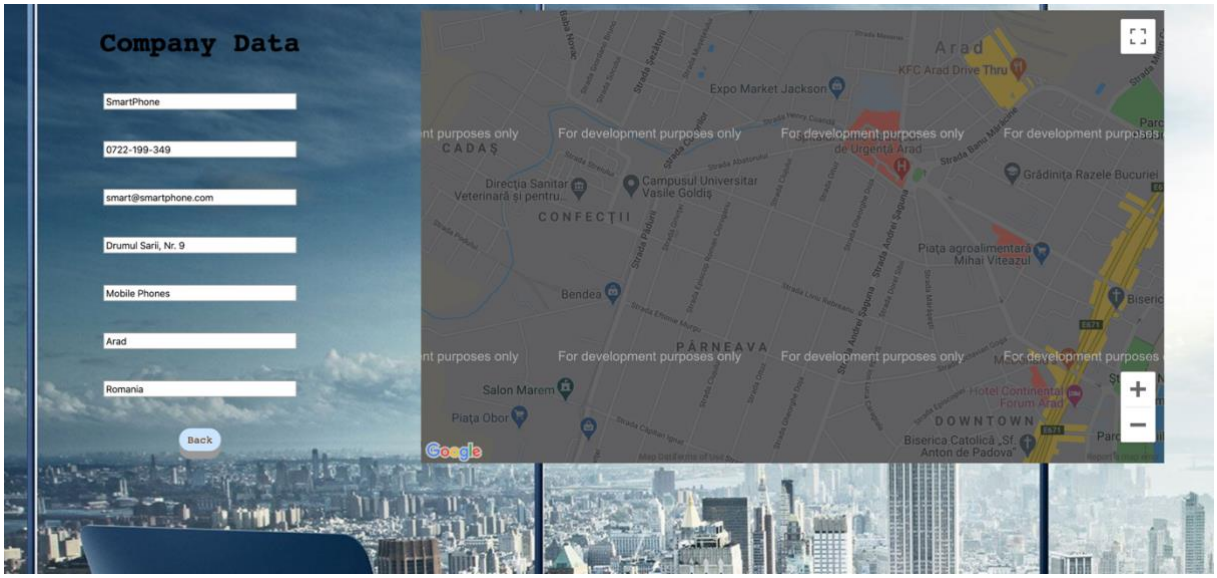

Mozilla Firefox
localhost/all
localhost/all
Back
Name
Surname
Phone
Email
Job
Department
Company
City
Country
Edit
Delete
Fracu
Mihai
0724-304-187
mihai.fracu@yahoo.com
Database Keeper
IT
FlowerRose
Cluj
Romania
Mircea
Gheorghe
0744-234-187
mircea.gheorghe@yahoo.com
Shop Assistant
Sales
FlowerRose
Cluj
Romania
Popescu
George
0744-204-589
george.popescu@yahoo.com
Junior Programmer
IT
ITSecBF
Bucharest
Romania
Popescu
Ion
0744-204-587
ion.popescu@yahoo.com
Junior Programmer
IT
ITSecBF
Timisoara
Romania
Popescu
Mihai
0744-204-588
mihai.popescu@yahoo.com
Junior Programmer
IT
ITSecBF
Timisoara
Romania
Cescu
Catalin
0799-234-188
ccatalin@yahoo.com
SEO
Sales
Regio
Paris
France
Mica
Alin
0722-219-483
micaalin@yahoo.com
HR Chief
HR
Regio
Bucharest
Romania


Mozilla Firefox
localhost/logs
localhost/logs
Logs
Back
Name Surname Email Activity Timestamp
Popescu George george.popescu@yahoo.com Created new account. 2019-12-29
Popescu Mihai mihai.popescu@yahoo.com Created new account. 2019-12-29
Popescu Ion ion.popescu@yahoo.com Created new account. 2019-12-29
Mircea Gheorghe mircea.gheorghe@yahoo.com Created new account. 2019-12-29
Fracu Mihai mihai.fracu@yahoo.com Created new account. 2019-12-29
Mica Alin micaalin@yahoo.com Created new account. 2019-12-29
Cescu Catalin ccatalin@yahoo.com Created new account. 2019-12-29


Dashboard - Grafana - Mozilla Firefox
Dashboard - Grafana
Dashboard
Last 6 hours
People working in other city
43%
People working in other country
14%
Number of persons by city
NUMBER_OF_PERSONS CITY_NAME
2.00 Timisoara
2.00 Cluj
2.00 Bucharest
1.00 Paris
Number of persons by country
NUMBER_OF_PERSONS COUNTRY_NAME
6.00 Romania
1.00 France
Number of persons by job
NUMBER_OF_PERSONS JOB_NAME
3.00 Junior Programmer
1.00 Shop Assistant
1.00 SEO
1.00 HR Chief
Number of persons by department
NUMBER_OF_PERSONS DEPARTMENT_NAME
4.00 IT
2.00 Sales
1.00 HR


Welcome, mihai.popescu!
Who would you like to chat with?
username
Chat
Log Out


dadelina: Hello! Pe la munca. Tu?
You: Buna! Ce mai faci?
Message
Send
Go Back



Concluzii

Proiectul este abia în stagiul inițial. El mai poate fi dezvoltat și poate fi chiar o idee de afacere.

Nu este chiar atât de complicat de lucrat cu docker și microservicii, implicit mysql sau un alt tip de bază de date. Se poate conecta foarte ușor la ea din Python și graficele se pot realiza rapid folosind grafana.

Toate aceste tehnologii aduc un plus aplicației și devine mai simplu de utilizat.

Cel mai probabil se pot face optimizări pentru a reduce timpul de căutare sau timpul de răspuns pentru o cerere, dar, după cum am menționat mai sus, aplicația este în stagiul de dezvoltare.

Bibliografie

- https://acs.curs.pub.ro/2019/course/view.php?id=679
- https://ocw.cs.pub.ro/courses/bd2
- https://www.w3schools.com/python/python_mysql_getstarted.asp
- https://www.w3schools.com/html/
- http://www.mysqltutorial.org/
- https://dev.mysql.com/
- https://hub.docker.com/_/mysql
- https://hub.docker.com/r/grafana/grafana/
- https://docs.python.org/
- https://docs.python.org/2/library/hashlib.html