UNIVERSITATEA POLITEHNICA BUCUREȘTI
FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

LUCRARE DE LICENȚĂ

Aplicație WEB pentru gestionarea reviziilor și reînnoirea documentelor autovehiculelor

Coordonator
Conf.dr.ing. Cornel Popescu

Absolvent
Eusebiu Rizescu

BUCUREŞTI

2019

CUPRINS

CUPRINS 2
Sinopsis 4
Abstract 4
1 Introducere 5
1.1 Context 5
1.2 Problema 6
1.3 Obiective 7
1.4 Soluția propusă 7
1.5 Structura lucrării 8
2 Studiu de piață / Metode existente 9
3 Tehnologii software folosite 10
3.1 WWW 10
3.2 HTML 12
3.3 CSS 14
3.4 Bootstrap 16
3.5 Python 17
3.6 Flask si WTForms 19
3.7 Docker 20
3.8 MySQL si SQLAlchemy 20
4 Proiectarea aplicatiei 22
4.1 Descriere generala 22
4.2 Inregistrare si Autentificare 24
4.3 User Home Page 26
4.4 Actualizare si stergere cont 27
4.5 Adaugare, editare si stergere autovehicul 29
4.6 Adaugare, editare, stergere si reimprospatare scadent 31
5 Detalii implementare 35
5.1 Prezentare generala 35
5.2 Private Docker Registry 36
5.3 Flask 37
5.4 Emailer 39
5.5 MySQL 40

5.6 Securitate 42
6 Concluzii si dezvoltari ulterioare 45
6.1 Concluzii 45
6.2 Dezvoltari ulterioare 45
7 Bibliografie 47
8 Anexe 49
8.1 Baza de date 49
8.2 Script build si rulare 50

SINOPSIS

Traim in secolul vitezei. Totul in jurul nostru se misca cu rapiditate. Dezvoltarea mijloacelor de transport a facut ca planeta noastra sa para mica, orice persoana, din orice colt al lumii putand ajunge in orice alt loc in cateva ore. Insa nu tot transportul se efectueaza pe cale aeriana. In 2018, in Romania erau inmatriculate peste 8 milioane de autovehicule, la o populatie de aproape 20 milioane de locuitori, adica o masina la 2,5 persoane.

Aplicatia prezentata in aceasta lucrare are ca scop sa ajute proprietarii si utilizatorii de vehicule sa gestioneze cu usurinta autovehiculele. Fiecare autovehicul are nevoie de schimburi periodice variabile (durata timp, durata kilometri), dar totodata are nevoie si de documente valabile pentru a putea circula in legalitate (asigurare, inspectie tehnica periodica, taxa de drum).

Printr-un portal WEB utilizatorul poate sa isi actualizeze documentele autovehiculului, sa isi inregistreze reviziile, sa actualizeze kilometrii, dar si sa adauge, modifice si stearga autovehicule. De asemenea, utilizatorul este anuntat pe mail inainte de expirarea unui document / efectuarea unei revizii pentru a preveni circularea in nelegalitate si extinderea duratei de viata a autovehiculului.

ABSTRACT

We live in the century of speed. Everything around us is moving fast. The development of the means of transport has made our planet seem small, every person from every corner of the world can travel anywhere in a few hours. But not all transport is represented by air. In 2018, over 8 million vehicles were registered in Romania, with a population of almost 20 million, that means 1 car for 2.5 people.

The application presented in this paper aims to help owners and vehicle users easily manage their vehicles. Each vehicle needs variable periodic services (time limited, mileage limited), but it also needs valid documents to enable it to travel legally (insurance, periodic technical inspection, road tax).

Through a WEB portal, the user can update their vehicle documents, record their services, update mileage, and add, modify and delete vehicles. Also, the user is notified by mail before the expiration of a document / revision to prevent the unlawful circulation and expand the life span of the vehicle.

1 INTRODUCERE

1.1 Context

Primul autoturism propulsat de un motor cu combustie interna a fost proiectat in 1885 in Imperiul German de catre Karl Benz si brevetat la data de 29 ianuarie 1886 [1]. De atunci transportul atat de persoane cat si de marfa s-a schimbat radical. Automobilele, de la creare si pana in prezent si cu siguranta si in viitor, au suferit modificari importante. Au fost adaugate si imbunatatie sistemele de franare, mecanisme de siguranta (centuri in trei puncte, airbag-uri), numeroase sisteme de confort cum ar fi climatizarea, servodirectia, stergatoare de parbriz electrice si lista poate continua. Toate acestea au ca scop cresterea sigurantei si confortului pasagerilor.

Toate acestea vin cu un cost. Cu cat o masina este mai complexa si detine mai multe sisteme, cu atat aceasta are nevoie de mai multa intretinere. Vizitele la service sunt in principiu de doua feluri: prevenire si reparare. In cazul fericit o masina este dusa regulat la service pentru a preveni uzarea prematura a acesteia. Interventiile efectuate sunt schimburi ulei, filtre, lichide, verificarea sistemelor de franare si directie. Cealalta categorie mai nefericita de vizita la service este atunci cand masina se defecteaza, vizite care se pot reduce ca frecventa atunci cand masina este intretinuta corespunzator, adica vizite regulate la service, o conduita recomandata de producator.

Un automobil intretinut conform recomandarilor prodacatorului are o durata de viata mai ridicata si totodata fiind constant verificat de o persoana competenta, se pot prezice si schimba componente, astfel micsorandu-se sansele de a se defecta pe neasteptate.

Aceste revizii nu sunt reglementate si impuse de legislatia din Romania, ele ramanand la latitudinea proprietarului daca doreste sa le efectueze. Insa, pentru ca un autovehicul sa circule pe drumurile publice, este nevoie ca anumite documente sa fie valabile, cum ar fi asigurarea auto obligatorie, inspectia tehnica periodica si taxa de drum (vigneta).

1.2 Problema

In prezent, fiecare dintre noi avem nevoie sa tinem minte zeci sau poate chiar sute de lucruri, cum ar fi parole de la job sau personale (mail, retele socializare, conturi pentru platformele de hobby, etc), vizitele la doctor, plata intretinerii la locuinta, plata impzitelor, facturilor de utilitati, facturilor de telefonie si multe altele. Neglijarea si in final uitarea acestora au urmari mai mult sau mai putin costisitioare, atat material, cat si ca timp pierdut. Potrivit unui studiu realizat de Forbes, unul din trei americani este in urma cu plata unei facturi [2]. Bineinteles, nu toti sunt in aceasta situatie din cauza neglijarii, dar o parte cu siguranta au aceasta problema. Acelasi lucru se intampla si cu automobilele. Multe persoane uita sa verifice validitatea inspectiei tehnice periodice, a asigurarii oblicatorie sau a vignetei.

In anul 2017, in Romania au fost aplicate 570 000 de sanctiuni [3] pentru autoturisme care circulau pe drumuri publice fara taxa de drum. Raportat la numarul autovehiculelor din Romania de 8 milioane[4], inseamna ca aproximativ 7% din proprietarii masinilor inmatriculate in Romania au primit amenda pentru circulare fara taxa de drum platita. Insa in acest studiu este vorba doar de vigneta, dar aceasta este doar una din cele minim 3 documente obisnuite pentru a circula pe drumul public cu un autoturism. Pe langa acest considerent, circularea fara inspectia tehnica periodica valabila este extrem de periculoasa atat pentru pasagerii autoturismului, cat si pentru ceilalti participanti in trafic.

Asemanatoare cu documentele necesare circularii pe drumurile publice, sunt si reviziile (a nu se confunda cu cele reglementate de legislatie) autoturismului. Orice motor, fie el de motoscuter sau de avion, are nevoie de revizii tehnice periodice pentru a putea rezista in timp, datorita frecarii si caldurii. Astfel, pentru a putea prelungi viata autoturismului este necesar a se face vizite periodice la service pentru acestea. Insa, cum este precizat si mai sus, ritmul vietii din ziua de azi fiind atat de alert, tindem sa neglijam. Acest lucru are ca principal element impactat buzunarul proprietarului, deoarece piesele se uzeaza mai repede si necesitand schimbare timpurie a acestora.

1.3 Obiective

Obiectivul principal propus este realizarea unei platforme WEB prin care fiecare posesor sau conducator de autovehicul (masina personala, autoutilitara, transport persoane, camion) sa isi poata monitoriza autovehiculul sau flota de autovehicule atat din punct de vedere a reviziilor tehnice cat si a documentelor necesare circulatiei pe drumurile publice. Spre exemplu, o companie de transport marfa doreste sa stie in orice moment statusul reviziilor tehnice ale flotei, cat si a documentelor pentru a evita amenzi si eventuale defectionari timpurii are autovehiculelor.

Un obiectiv secundar, neacoperit in aceasta lucrare este monitorizarea in timp real, pin GPS al fiecarui automobil. Acest lucru, cel putin in companiile de transport si distributie este un factor crucial in eficientizarea companiei. Pe langa aceasta monitorizare in timp real, prezenta unui GPS poate calcula cu exactitate kilometri parcursi de autovehicul, introducerea regulata a kilometrajului pentru a determina urmatoarea vizita la service.

1.4 Soluția propusă

Solutia propusa consta intr-o platforma WEB unde utilizatorii aplicatiei se pot loga si verifica statusul autovehiculelor inregistrate (documente + revizii). Acestia pot adauga, pe langa adaugare si stergere de automobile din baza de date, actualizarea numarului de kilometri, adaugarea intervalelor de service preferentiale (daca nu sunt dorite cele default pentru autovehiculul respectiv), adaugarea de documente suplimentare pe langa cele de baza. De exemplu, in cazul companiilor de transport pe langa asigurarea RCA oblicatorie, transportatorii sunt obligati sa detina si asigurare de marfa, aceasta avand bineinteles o data de expirare. Se mai pot adauga spre exemplu si documente apartinand conducatorului masinii, cum ar fi atestatul de transport marfa / persoane, atestatul ADR (pentru transportarea marfilor periculoase). Cu alte cuvinte, aplicatia nu se rezuma doar la revizii si documente strict apartinand autovehiculului. Aceasta se poate extine si la date despre sofer.

Toate aceste documente si revizii sunt de trei categorii: cu data de expirare, cu numar maxim de kilometri sau ambele. Cu cateva zile inainte de atingerea expirarii, utilizatorul este anuntat

prin email (se poate extinde aceasta aplicatia, prin a informa si prin SMS sau/si telefonic) cu privire la documentul / revizia care se apropie de expirare.

1.5 Structura lucrării

Capitolul 2 are ca scop prezentarea stadiului actual al domeniului, prin expunerea principalelor aplicatii software asemanatoare existente in Romania: software destinate tracking-ului detaliilor automobilelor

Capitolele 3, 4 si 5 urmaresc o abordare noua a problemei, arhitectura solutiei propuse si detalii legate de implementarea acesteia. Sunt detaliate componentele principale, descrise tehnologiile folosite. Limbajul utilizat este unul tehnic, cu multe tehnologii prezentate.

Capitolele 6 prezinta concluziile realizarii aplicatiei, dar si urmatorii pasi in ceea ce priveste dezvoltarea mai complexa si punerea in piata a aplicatiei.

2 STUDIU DE PIAȚĂ / METODE EXISTENTE

In prezent, in Romania, exista numeroase astfel de produse care se apropie de problema dezbatuta anterior. Majoritatea lor insa sunt axate pe monitorizarea GPS a flotelor de autovehicule si mai putin pe reaminitirea reinnoirii documentelor si reviziilor tehnice recomandate de producatorul autovehiculului.

MyCar Assistant[5] este o platforma atat WEB cat si mobila (iOS + Android) care are ca scop gestionarea eficienta a documentelor scadente pentru flote si vehicule. Aplicatia are un flow asemanator ca solutia propusa: utilizatorul isi face cont, isi adauga masinile pentru care doreste gestiunea, adauga deasemenea documentele scadente si apoi va fi instiintat atunci cand documentele sunt pe cale de expirare. Aceasta solutie are de altfel inca o scadenta la care eu personal nu m-am gnadit si anume trusa medicala si stingatorul de incendii. Prezenta ambelor este obligatorie in orice autovehicul care circula pe drumurile publice si totodata ambele au si data de expirare. Un mare plus al acestei solutii este ca are si aplicatie pentru telefon, atat Android cat si iOS, lucru care este de dorit atunci cand vrei sa verifici rapid statusul masinii.

AutoMinder[6] este alta varianta pe care un client o poate alege pentru gestiunea masinii personale sau a flotei de masini. Aceasta aplicatie insa, este orientata spre companiile mari, spre companiile de distributie sau de transport. Aceasta aplicatie este o aplicatie complexa de administrare a parcului auto, ea continant pe langa scadente si reparatii, inspectii si verificari, taxe si impozite, evidenta angajatilor si lista activitatii zilnice, evidenta licentelor de transport, istoricul autovehiculelor, mententanta utilajelor. Principala limitare pe care o vad la aceasta aplicatie, din punctul de vedere al unei persoane fizice, cu un autoturism de monitorizat este ca fiind atat de complexa, este destul de scumpa (1900 lei cu licenta pe viata).

AlerteMasina[7] este de asemenea o solutie care ajuta la gestionarea eficienta a documentelor masinilor, dar si ale conducatorilor acestora. Aceasta aplicatie este orientata atat catre persoanele fizice cat si catre companiile cu mai multe masini. Pretul este mic sau inexistent pentru persoanele fizice (2 masini gratuite, iar restul 1 euro pe an), dar totusi solutia ofera destule functii cat sa satisfaca nevoile companiilor cu flote de autovehicule din care amintim: management alerte, management revizii si reparatii si management alimentari. De altfel, aceasta solutie are si aplicatie mobila, atat pentru Android cat si pentru iOS.

3 TEHNOLOGII SOFTWARE FOLOSITE

3.1 WWW

Ce este WWW si de unde a inceput totul?

WWW a avut la baza ideea de a combina tehnologii informatice pentru a usura partajarea de informatie. Tehnologii precum retele de date, hyper text si protocoale de comunicatiecum ar fi TCP/IP au contribuit la dezvoltarea unui sistem global de care acum beneficiem cu totii.

Totul a inceput de la Tim Bernes-Lee, un om de stiinta englez care a inventat Word Wide Web in anul 1989 in timp ce lucra pentru firma CERN. Nevoia de a inparti informatia intre oamenii de stiinta i-a dat lui Tim ideea de a dezvolta un proiect care sa ajute la comunicarea usoara intr-o retea de calculatoare interconectate. Asa a luat nastere Internetul. Cu toate ca Tim a lucrat la acest proiect in timpul in care el era angajat la CERN, Web nu a fost niciodata un proiect oficial CERN. [8]

In martie 1989 Tim si-a impartasit ideea cu seful lui. Mike Sendall, insa, nu s-a prezentat asa de interesat de idee. Cu toate astea a recunoscut ca este un proiect interesant. In septembrie anul 1990 Mike i-a acordat lui Tim timp sa lucreze la ceea ce vor urma a fi bazele Internetului. Acesta a inceput sa lucreze la proiect pe un calculator NeXT nou achizitionat. Acesta a fost una dintre primele creeatii ale lui Steve Jobs.

Bernes-Lee a gasit un coleg care s-a aratat entuziasmat de ideea propusa de el. Acesta a fost Robert Cailliau, un inginer in informatica si calculatoare belgian. In septembrie 1990 cei doi au propus ideea Word Wide Web la Conferinta Europeana de Hypertext. Din pacate nu au gasit persoane/vanzatori/sponsori care sa aprecieze ideea impreunarii tehnologiei Hypertext cu Internetul.

In octombrie 1990 deja au fost dezvoltate tool-urile necesare si fundamentale pentru functionarea Web-ului. Tehnologii care stau, pana in prezent, la baza Internetului si de care inca avem nevoie pentru a dezvolta un browser web sau un site. Acestea sunt HyperText Transfer Protocol sau HTTP, HyperText Markup Language sau HTML, primul browser Web care s-a numit WordWideWeb.app, primul server HTTP mai tarziu cunoscut si ca CERN httpd si primul server web si primele pagini web care contineau descrierea proiectului in sine.[9]

In incercarea de a introduce sistemul dezvoltat de el, Tim si-a incurajat colegii sa se logheze pe site-ul dezvoltat de el. Pentru inceput informatia care era partajata si la care aveau acces

erau cateva pagini web care contineau detalii despre proiectul in sine si o carte de telefon. Asadar oamenii erau incurajati sa intre si sa isi caute numere de telefon direct pe platforma.

Pe durata timpului in care Bernes-Lee a lucrat la CERN a folosit un computer NeXT pentru a creea un server web. Asadar primul sever web a fost o masina care era pornita continuu. Cu atat mai mult, pentru a evita oprirea accidentala a masinii, Tim a lipit pe serverul sau o eticheta ce ii avertiza pe cei din jur ca masina rebuie sa ruleze continu.[10]

Problema portabilitatii platformei dezvoltate de Tim si colegii lui s-a pus in anul 1992. Initial browserele web puteau fi rulate numai pe sistemul de operare NeXT. Pentru inceput erau proiectul a fost compatibil numai cu computerele CERN. Cu toate astea aceasta problema a fost rezolvata de catre Paul Kunz si Louise Addis.

In 1993 a fost lansat o varianta de software care rula in mediul X Window System pentru Unix. Cu putin timp dupa a fost lansata o versiune user friendly bazata pe ferestre interactive. Proiectul a devenit open source si asta a dus la creearea de versiuni pentru masini precum PC si Macintosh. Posibilitatea rularii pe masini cunoscute si usurinta folosirii a dus la mediatizarea proiectului. La finalul anului 1993 au existanu peste 500 de servere web, iar WWW reprezenta 1% din traficul de Internet.

In urmatorul an a urmat o explozie a Web-ului, proiectul a devenit foarte mediatizat, iar numarul de useri a crescut considerabil. Pana la finalul anului 1994 au ajuns sa existe aproape 10 milioane de utilizatori. Se poate spune ca anul 1994 a fost “Anul Web”. Cresterea numarului de utilizatori a adus si multe beneficii. S-a dezvoltat un program de user bug report si utilizatorii puteau sa propuna noi functionalitati.

CERN a hotarat in 1993 ca Web va fi un protocol accesibil pentru toti si codul va fi royality-free. Acest lucru a starnit interes si in 1996 Web a inceput sa fie comercializat. Multe companii au vazut o oportunitate. Usor, usor a luat nastere e-comertul. Partajarea de informatie gratuita a insemnat publicitate gratuia, lucru pe care multe firme nu l-au trecut cu vederea. Rezultatul a fost ca multe site-uri “.com” au inceut sa promoveze produse cu posibilitatea de a fi cumparate.

Comercializarea si mediatizarea web-ului a dus la aparitia companiilor tip “start up”. In 2001 a avut loc asa numitul “Dot-com boom”. Similar cu alte tehnologii din trecut Web a avut un moment in care facilita deschiderea cu usurinta al unui business. Multe companii inceput sa isi extinda aria de expertiza in domeniul Web. Tostusi nu toate au ajuns sa fie profitabile. Tot in

anul 2001 a avut loc o scadere in numarul de companii care profitau de mediul Web. Cele care au supravietuit s-au bucurat de success in secolul 21. Secretul succesului s-a dovedit a fi un plan bun de business.

In urmatorii ani au inceput sa se dezvolte tehnologii media precum Facebook, Instagram, Youtube au aparut creeatori de content pentru diferite domenii. E-comerce a ajuns foarte popular. De la bilete de avion, piese pentru masini, carti, pana si ghizi, tururi si excursii, toate se pot achizitiona de pe net. Companiile profita din plin de aceasta platforma. Amazon, e-bay sunt printre primele site-uri care au dus e-commerce la un alt nivel. Cu toate astea exista foarte multi retailer care ofera servicii si produse pietelor de nisa.

De la o nevoie interna a unei companii de cercetare, Web a ajuns sa fie un imens care ofera facilitate si oportunitati tuturor utilizatorilor. Este un loc in care se pot oferi produse si servicii, se poate face promovare si oamenii se pot exprima liber.

3.2 HTML

HTML sau Hypertext Markup Language este un limbaj universal pe care il poate interpreta orice Browser Web. Acest instrument ajuta la evidentierea limbajului si la structurarea unei pagini web. Acest lucru vine in ajutorul utilizatorilor. Ei vor fi capabili sa vada pagina asa cum a fost proiectata sa arate, fara a fi nevoiti sa isi instaleze alte tooluri pe propriul calculator. Cu cat o pagina este mai placuta din punct de vedere vizual si mai usor citit, cu atat creste posibilitatea ca utiizatorii sa se intoarca la ea. [11]

HTML respecta o structura arborescenta, ierarhica ce este cunoscuta sub numele de DOM. DOM este prescurtarea pentru Document Object Model si este un API pentru HTML si XML. Acesta are la baza idea de tree, fiecare component a documentului poatand avea un copil si un parinte. Nodul de inceput se mai numeste si obiectul document. Atunci cand userul deschide un document HTML in browser, acesta este parsat si transformat intr-un arbore, urmand ca apoi sa fie afisat. Un mare avantaj al DOM este faptul ca este o interfata orientate pe obiect, lucru care face mult mai usoara proiectarea si modificarea documentului.[12]

Pentru a evidentia si structura textul sunt folosite tag-uri. Tagurile in HTML macheaza elementele dintr-un astfel de document. Exista doua tipuri de tag-uri, tag-uri de start si tag-uri

de final. Un element este incadrat de doua astfel de taguri in felul urmator: <tag>element</tag>. De multe ori avem nevoie ca in interiorul unui element sa mai declaram un element. Aceste elemente se numesc nested elements, marcate si ele, la randul lor de taguri de start si de final.

Tag-urile unui document HTML macheaza ierarhia si tipul fiecarui element. De pilda, un document HTML este format din head si body. Inceputul head-ului este marcat de tagul <head> si sfarsitul head-ului este marcat de tagul </head>. Respectand aceeasi structura body-ul este marcat de tag-ul <body> la inceput precum si de tagul </body> la final. In interiorul head-ului si in interiorul body-ului pot aparea o multime de alte elemente marcate de taguri specifice care ajuta la structurarea paginii.[13]

Un tag foarte important este tag-ul <!DOCType HTML PUBLIC> care marcheaza inceputul unui document HTML si semnaleaza browserului ca are de aface cu un document de tip HTML. DOCTYPE specifica dealfel si versiunea de HTML folosita. HTML5 permite declarearea mai simpla a versiunii de html: <!doctype html>.

In html exista foarte multe tipuri de taguri, in functie de scopul lor si ce vor defini. Cateva tag-uri de baza sunt : <head>, <body>, <h1> pana la <h6> pentru heading , <p> pentru paragraf,<a> pentru link-uri, <img> pentru imagini, <button> pentru butoane, <ul> urmat de <li> pentru liste etc..

Un alt lucru de retinut despre HTML este ca acesta suporta atribute. Atributele sunt destul de des intalnite in documentele HTML. Ele tin de un element si au ca scop sa ofere informatii aditionale despre acesta . Atributele vin de obicei in forma de pereche cheie/valoare si sunt plasate in interiorul tagului de inceput ce marcheaza elementul. Atributele pot fi folosite pentru a adduce functionalitate unor elemente ale documentului cum ar fi introducerea unei imagini sau a unui link prin intermediul tagurilor <img> si <a>. Cateva exemple importante si foarte folosite de atribute sunt :“src” ce specifica numele fisierului introdus in cadrul elementului, “herf” care specifica adresa unui link, “width” si “height” care specifica dimensiunile elementului correspondent, “alt” care specifica o alternativa in caz ca elementul nu este incarcat corespunzator si “style” care specifica “stilul” in care elementul va fi afisat.[14]

Atributul “style” introduce in documentul HTML proprietati specific CSS (Cascading Style Sheets). De obicei cand proiecam o pagina web ne dorim ca aceasta sa fie cat mai placuta din punct de vedere esthetic si cat mai usor de urmarit. Ei bine, acest lucru este in mare parte

atributul “style”. Acesta introduce proprietati precum background-color, color, font-family, font-size, text-align care ajuta la infrumusetarea paginii si la asezarea elementelor in magina.

Un document HTML este origanizat in blocuri. Un exemplu foarte intalnit blocuri sunt cele marcate de tag-ul . Acest tag se comporta ca un container pentru elementul marcat. De obicei tag-ul nu are atribute. In practica insa, se intampla foarte des ca acest tag sa fie insotit de atribute precum “style”, “class” si “id”. De obicei blocurile au si un stil css propriu introdus de tagul “style” sau o casa CSS care sa specifice stilul blocului prin atributul “class”.[15]

Atribute precum “class” si “id” ajuta la imbunatatirea coduluil HTML. Atributul “class” introduce o clasa CSS ceea ce inseamna ca ajuta la minimizarea codului duplicat. In cazul in care avem mai multe elemente care impart acelasi stil ne vom folosi de atributul class pentru a nu rescrie stilul de mai multe ori. Atributul “id” este asociat unui element si marcheaza unicitatea acestuia. Prin id elementul poate fi identificat si modificat cu ajutorul codului JavaScript. In acelasi timp acest id poate sa injecteze si el stilul CSS, numai ca acesta va fi unic pentru elementul cu id-ul respectiv.[16]

In anul 2014 a fost introdus HTML5 care acum este foarte folosit. HTML5 a devinit foarte popular datorita atributelor si tag-urilor noi care pot fi folosite. In acelasi timp functionalitatile sunt mult mai usor de implementat, iar dinamica unei pagini scrisa cu ajutorul HTML5 e substantial inbunatatita. Odata cu trecerea la HTML5 au fost introduse noi elemente si atribute pentru elemente spre exemplu elemente multimedia (<audio> si <video>) sau atribute pentru elemente de tip formular(data, ora, calendar etc.).

Un alt pas in fata a fost introducerea de API-uri noi in HTML5. Cateva dintre cele mai interesante sunt HTML Geolocation, HTML Drag and Drop, HTML Local Storage, HTML Application Cache, HTML Web Workers si HTML SSE. HTML Locla Storage este o alternative foarte buna pentru Cookie-uri.[17]

3.3 CSS

CSS este un adjuvant al HTML-ului. HTML nu a fost gandit sa contina taguri pentru formatarea unei pagini web. Ulterior acestea au fost adaugate odata cu versiunea 3.2 a HTML-ului, dar nu au facut decat sa ingreuneze viata developerilor. Ca o alternativa, cei de la World Wide Web Consortium (W3C) au creeat CSS.

CSS specifica stilul documentului web, mai exact personalizeaza fiecare componenta. In relatie cu HTML, care este „scheletul” sau structura, CSS este estetica documentului. CSS a fost creat ca sa separe din punct de vedere vizual (si nu numai) elementele prezente intr-o pagina web. Acest lucru imbunatateste accesibilitatea si face pagina web mai placuta din punct de vedere vizual.

CSS interactioneazacu elemente HTML ca spre exemplu paragrafe <p>, butoane <button> s.a.m.d. Acestor elemente li se poate „aplica” cod CSS prin doua metode, fie prin atributul style adaugat tagului corespondent fie prin atributul class. In cazul atributului style, toate proprietatile CSS vor fi inserate in documentul HTML si vor duce la ingreunarea si marirea considerabila a dimensiunilor codului. Insa daca este folosit atributul class, care introduce selectorul clasei CSS, se vor evita supraincarcarea, repetitia, iar dimensiunea codului va fi substantial mai mica. Un alt avantaj este faptul ca prin modificarea unui singur fisier se poate schimba stilul intregului document HTML.

Un selector al unei clase CSS specifica elementul sau elementele carora li se va atribui stilul. Sunt mai multe tipuri de selectori: id selector, class selector, grouping selector, element selectror.

- Un selector in functie de id se foloseste de atributul id al unui element HTML. Un id este unic, asadar este mai usor un element consacrat al documentului. Pentru selectarea unui element in functie de id se foloseste ‚#id_element’
- Un selector in functie de element se foloseste de numele elementelor HTML. De exemplu <p>, o clasa de tipul p{ color:blue;} presupune aplicarea stilului (color: blue) pe toate elementele de tip <p> din documentul HTML.
- Un selector class presupune aplicarea stilului pe elementele HTML care au un atribut de tip ‚class’ specific. Sintaxa este .className{color: blue;}. In cazul in care ne dorim sa aplicam un stil pe o parte anume din elementele care au un atribut class specific, putem sa scriem li.className{color:blue;};
- Selectorii de grup presupun aplicarea stilului css pe toate elementele HTML de acelasi tip.

In general se prefera ca codul CSS sa fie separat de codul HTML pentru a se evita incarcarea fisierului si pentru a imbunatati lizibilitatea codului. Dar cum putem lega, totusi, codul CSS de codul HTML? Ei bine sunt doua metode. Fie prin tagul <link> impreuna cu atributele rel, type

si herf care specifica typul si calea fisierului css care se va importa, fie direct prin tagul <style> in care se va introduce direct codul css fara a fi nevoie de un fisier auxiliar. Orice metoda am folosi, este bine de stiut ca atat tagul <style> cat si tagul <link> impreuna cu fisierul .css, vor trebui puse in <head>. Explicatia pentru acest lucru este ca <head> este primul element html incarcat iar odata cu el se vor incarca si stilurile care vor fi ulterior folosite de celelalte elemente.

Numele Cascading vine de la ordinea in care style sheet-urile sunt prioritizate. Cel mai prioritar style sheet este cel Inline, cel din interiorul elementelor HTML. Dupa el urmeaza External si Internal style sheets, adica cele din sectiunea head care se incarca primele, iar ultimul ca prioritate este browserul default. Aceste prioritati trebuie respectate pentru a evita confuzii si suprarscrierea stilurilor intr-un document HTML.

3.4 Bootstrap

Bootstrap este un framework de CSS dezvoltat de Mark Otto si Jacob Thronton. Initial numele a fost Twitter Blueprint si a fost creeat pentru a evita inconsisteta in cod (de la un proiect la altul) si pentru a face mai usoara intretinerea codului.

Bootstrap suporta web design responsive si este compatibil si cu dispozitivele mobile. Un design web responsive se ajusteaza automat in functie de dispozitiv si de browser. Asa se garanteaza fuctionarea aplicatiei, interactivitatea si accesibilitatea. Pe langa functionalitate si accesibilitate Bootstrap ajuta la infrumusetarea aplicatiei.

Bootstrap contine in general HTML si CSS dar poate contine si JavaScript in anumite cazuri. Cu ajutorul Bootstrap-ului se pot dezvolta mai usor componente precum butoane, nav-bar-uri, etc. datorita faptului ca acesta din urma pune la dispozitie tamplate-uri (free) diversificate care pot fi refolosite de dezvoltatori. [18]

Pentru folosirea acestui framework (fara a-l descarca) este nevoie sa il includem in proiect din CDN (Content Delivery Network). Exista si MaxCDN care include support pentru CSS JavaScript si JQuerry. Includerea frameworkului in proiect se face prin intermediul tag-ului <script> impreuna cu atributul “src”. Daca vrem includerea CSS-ului in proiectul current se foloseste tagul <link>. Atat <link> cat si <script> vor fi puse in head-ul documentului. [19]

Avantajul major al acestui framework este viteza de randare. In general vizitatorii stie-ului vor avea deja Bootstrap 4 in cache-ul prowserului personal. Un alt lucru care aduce o imbunatatire din punct de vedere al vitezei este faptul ca CDN aduce fisierele de care are nevoie de la cel mai aproiat server. Toate aceste lucruri fac o mare diferenta si imbunatatesc interactivitatea site-ului. [20]

Pentru randarea cross-browser Bootstrap se foloseste Reboot. Reboot corecteaza inconsistentele care apar la schimbarea browserelor pe care randeaza site-ul. Inconsistentele de obicei apar la paginare: margins-top/bottom/left/right si la dimensionarea componentelor. Stilizarea se face numai cu ajutorul claselor CSS. Se foloseste ca unitate de masura “rem” in loc de “em” pentru scalabilitate mai buna, se evita margin-top si se foloseste in general margin pentru consistenta, iar majoritatea proprietatilor aplicate pe font se vor mosteni.

Bootstrap are conceptual de container. Un container este un element care inglobeaza continutul site-ului. Ele pot fi de doua feluri, fix si fluid[21]. Pe langa containere, Bootstrap introduce si un numar de taguri specifice, cum ar fi <mark>, <abbr>,<code>, <kbd>. Acest framework pune la dispozitie si un numar mare de clace css pentru diferite utilizari: text, background, tabele, aliniere componente si asa mai departe.

Bootstrap ajuta la dezvoltarea de componente interactive, ca spre exemple meniuri dropdown. Aceste tipuri de componente sunt gandite sa fie accesate prin muse-click-uri, touch, sau taste. De asemenea Componente sunt gandite sa fie interpretate de mecanisme de citire a ecranelor (special pentru persoanele cu probleme). Ele sunt in general generice, lasand la latitudinea programatorului felul in care vor fi adaptate si costumizate.

3.5 Python

Python a aparut in anul 1991 si a fost creat de Guido van Rossum. Este un limbaj interpretat, de nivel inalt care suporta mai multe paradigm de programare. Este in principal un limbaj de programare care ajuta la dezvoltatrea unui numar variat de aplicatii, in cazul de fata fiind folosit la dezvoltarea uni aplicatii web.[22]

Sintaxa Python este relative usor de inteles si de citit. Acest lucru este subliniat de filozofia limbajului in sine (The Zen of Python) care spune: “Beautiful is better than ugly; Explicit is better than implicit; Simple is better than complex; Complex is better than complicated;

Readability counts.” . Usurinta cu care este citit si interpretat de catre programator acest limbaj vine cu o serie de avantajul ca intretinerea codului ce poate face mult mai usor. Un alt avantaj ar fi acela ca mai multi dezvoltatori ar putea sa lucrze impreuna la un proiect fara a intalni bariere din punct de vedere al limbajului.

La fel ca alte limbaje Python are si el intructiuni care modifica fluxul de control al aplicatiei. Acestea sunt “if”-“else”-“elif”, “while”, “for”,”try”, etc. Acesteintructuni lucreaza fie cu variabile, fie cu obiecte.

Python este un limbaj orientat pe objiect. Nu este neaparat nevoie ca variabilele sa fie declarate inainte de utilizarea lor.[23] Variabilele vor fi create in momentul in care li se va atribui o valoare. Un alt detaliu ar fi acela ca tipul variabilei poate sa se schimbe in functie de ce valori ii sunt atribuite la un moment dat. Un exemplu clar ar fi : x=1 (unde x va fi un int) s=”Hello”(x isi va schimba tipul intr-un string)

Variabilele in Python pot avea mai multe tipuri. Aceste tipuri se inpart in: tipuri numerice, string-uri, Collections (liste) si obiecte. Tipurile numerice sunt int, float si complex. Stingurile sunt siruri de tip character care pot fi incadrate si intre <“ ”> (ghilimele) dar si intre <‘ ’> (apostrofi). Colectiile sunt si ele de mai multe tipuri:LIste, Tupl-uri, Set-uri si Dictionare. Ca orice limbaj orientat pe obiect Python suporta clase. Instantele claselor sunt obiectele. In mare parte cam tot ce exista in python se poate cataloga ca un obiect cu proprietati si metode specifice. O clasa este un “blueprint” al unui obiect. O clasa este create prin keyword-ul “class” si poate contine atat proprietati cat si functii. Un obiect al unei clase este creat printr-un constructor, fie cu parametrii, fie fara. Metodele unui obiect sunt functii specifice acestuia. Ele, la fel ca si proprietatile sunt marcate de un “.”.

Obiectele pot fi prelucrate si ne putem folosi de ele in diferite moduri. In Python, la fel ca si in Java, exista un keyword care refera instant clasei curente in care ne aflam, acesta este keyword-ul self. Self este folosit pentru a accesa cariabilele care apartin clasei. In cazul functiilor din clasa primul parametru din transmis va fi self.[24] Obiectele pot fi actualizate. Proprietatile obiectelor pot fi modificate de catre programator prin operatii simple de atribuire. De asemenea proprietatile unui obiect pot fi sterse cu ajutorul lui “del”. “del” poate fi folosit si pentru stergrea unui obiect cu totul.

In Python se aplica conceptual de mostenire. Mostenirea este cand o clasa primeste proprietatile si metodele unei alte clase pe careo mosteneste. Cele doua clase sunt catalogate ca fiind clasa parinte si clasa copil care mosteneste clasa parinte. Clasa parinte are aceeasi sintaxa ca o clasa normala, in schimb clasa copil va primi ca parametru clasa parinte pe care o mosteneste.

3.6 Flask si WTForms

Flask este un framework web pentru Python bazat pe Werkzeug si Jinja. A fost creat in anul 2004 de catre un grup de entuziasti Python. Potrivit creatorului principal, ideea a fost orginial venita ca o gluma de 1 Aprilie, dar care s-a dovedit a fii o ideea foarte buna si astfel a fost pusa in practica. In 2016 Flask a fost cel mai folosit framwork de Python potrivit Github.[25]

Flask are multe valori de configurare configurabile, bineinteles cu valori defalut, astfel incat daca nu se doreste a se face fine-tuning Flask poate fi folosit usor. Prin conventie, template-urile si fisierele statice (imagini, CSS) sunt stocate in subdirectoare in cadrul sistemului de fisiere ale aplicatiei cu numele „templates” si „static”.

Flask este mai degraba un microframework deoarece este gandit sa fie simplu, dar extensibil.[26] Astfel, Flask nu impune folisirea unei anumite baze de date. Deciziile pe care totusi le ia, cum ar fi ce motor de template sa foloseasca, sunt usor configurabile. Flask suporta adaugarea de extensii in aplicatie ca si cum ar fi implementate direct de Flask. Astfel de extensii sunt, WTForms (pentru imbunatatirea formularelor clasice HTML), Flask-SQLAlchemy (pentru usurarea conexiunii cu baza de date), flask_login (pentru a usura logarea unui user in site).

WTForms este o librarie de validare si randare a formularelor clasice din HTML pentru dezvoltare web in Python.[27] WTForms poate functiona cu orice cadru web si motor de template, existand o multime de librarii opensource care ofera o integrale mai facila cu motoarele populare. Cu ajutorul WTForms se pot colecta mai usor datele din formularele din pagini, validate daca sunt corecte sau nu si trimise mai departe in aplicatie pentru a fi prelucrate.[28]

3.7 Docker

Docker este un instrument creat pentru a face mai usoara crearea, deploy-ul si rularea aplicatiilor utilizand containerele. Un container ii permite dezvoltatorului software sa impacheteze o aplicatie cu toate dependentele sale intr-o imagine pe care sa o poata rula oriunde, pe orice masina, indiferent se arhitectura host-ului, cu conditia ca acesta sa aiba docker-engine instalat.

O imagine Docker este o entitate standard de software care impacheteaza codul si toate depentetele sale pentru a rula aplicatia rapid de pe un host pe altul. Un container este o imagine care ruleaza. Dintr-o imagine se pot crea mai multe containere simultan.

Docker swarm este un instrument pentru gestionarea si programarea containerelor de Docker. Cu Swarm, administratorii si programatorii pot crea si administra un cluster de noduri de Docker ca fiind un singur sistem virtual[29]

3.8 MySQL si SQLAlchemy

MySQL este printre cele mai populare limbaje pentru accesarea, modificarea si mentenanta unei baze de date relationale. Este un software open source creat de Oracle. El poate rula pe mai multe platforme precum Unix, Linux, Windows si asa mai departe. MySQL gestioneaza o baza de date relationala. O baza de date relationala este un set de date organizat in tabele, iar fiecare table poate fi accesat, actualizat, sters si modificat. Aceste lucruri se pot face prin diferite comezi. [30]

MySQL are la baza limbajul SQL (Structure Query Language) si a fost scris in C respectiv C++. Prin intermediul SQL pentru a interactiona cu baza de date. Sunt patru mari tipuri de comenzi prin care se poate relationa cu baza de date si anume: Data Query, Data Manipulation, Data Identity, Data Access Control. Aceste comenzi ii spun serverului ce anume sa faca cu baza de date, in functie de caz.

Tipul de comunicare Client-Server este ceea ce sta la baza functionarii mai multor aplicatii. Un client (sau mai multi) se conecteazala server. Serverul asteapta, iar cand unul din clienti face un request acesta vine cu un response la requestul clientului. MySQL creeaza o baza de date relationala, bazata pe tabele si relatii intre tabele. Un client face un request SQL in baza de date MySQL, iar serverul ii raspunde cu datele cerute.[31]

SQLAlchemy este un toolkit si ORM (object-relational mapper) scris in Python. Acesta functioneaza ca un wrapper peste cele mai comune baze de date relationale. Functioneaza prin mapearea tabelelor bazei de date ca fiind clase in Python, astfel programatorul fiind usurat de grija conexiunii cu aceasta. Totodata SQLAlchemy are si mecanisme care previn SQL injection print traducerea comenzilor in limbajul bazei de date si nu executarea lor direct.[32]

4 PROIECTAREA APLICATIEI

4.1 Descriere generala

Pentru a incepe utilizarea aplicatiei, utilizatorul trebuie sa acceseze din browser adresa https://carplanner.ro. Aceasta adresa este mapata local catre IP-ul masinii virtuale pe care ruleaza aplicatia. A trebuit asociat IP-ul cu un nume de domeniu pentru a putea emite un certificat SSL pentru acel nume de domeniu pentru a putea avea comunicare criptata si astfel o grad de securitate mai ridicat

Diagrama site-ului, care va fi explicata in subcapitolele ce urmeaza, este urmatoarea:

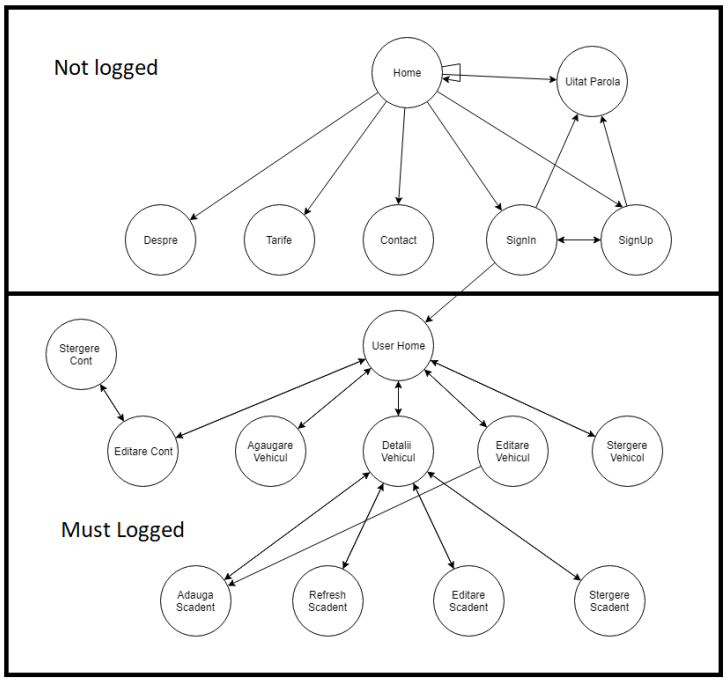

Fig. 1 - Harta Site

Paginile disponibile fara autentificare sunt Home, Despre, Tarife, Contact, Uitat Parola, SignUp, SingIn. Pagina de mai jos este cea care este redata utilizatorlui la accesarea link-ului aplicatiei. (local este carplanner.ro)

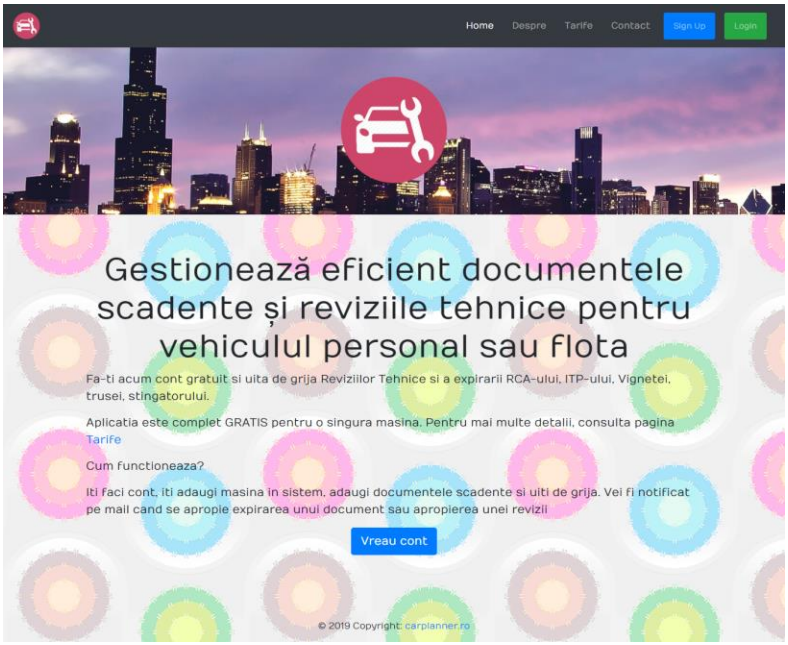

Fig. 2 – Home page

Paginile Despre si Tarife sunt doar pagini informative, fara impact asupra functionalitatii aplicatiei. Pagina Contact contine un formular pe care utilizatorii il pot completa daca au nelamuriri privitoare la aplicatie si doresc aplicatii suplimentare. Ca urmare a completarii acestui formular, se va trimite un mail la o adresa specificata in aplicatie. Acest formular are dublu rol; este mai usor si mai interactiv pentru utilizator sa adreseze intrebarile, dar, de asemenea, opreste si publicarea adresei de mail, astfel prevenindu-se spam-ul.

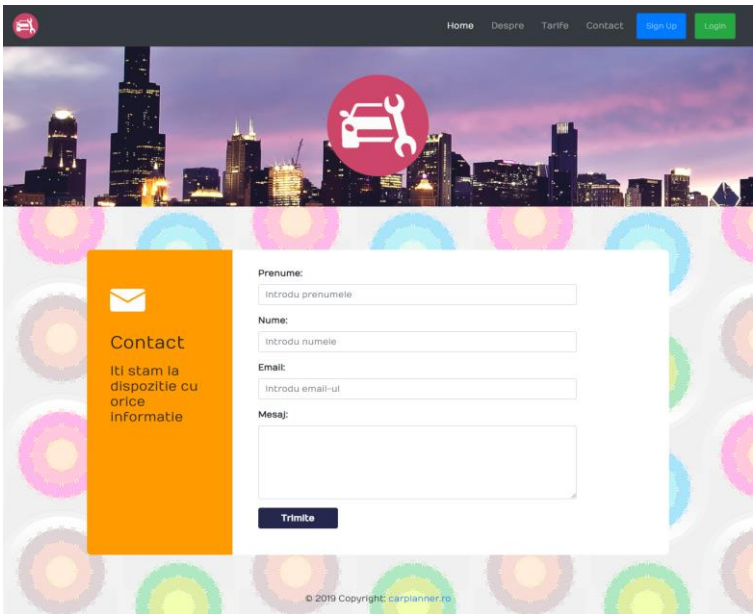

Fig. 3 - Contact

Principalele entitati pe care aplicatia este construita sunt:

Utilizator - mai multe detalii in subcapitolele 4.1, 4.2 si 4.3

Vehicul - mai multe detalii in subcapitolul 4.4

Scadent - mai multe detalii in subcapitolul 4.5

Pe scurt relatiile dintre aceste entitati sunt urmatoarele, relatiile complete fiind descrise in capitolul 5 unde este prezentata schema bazei de date.

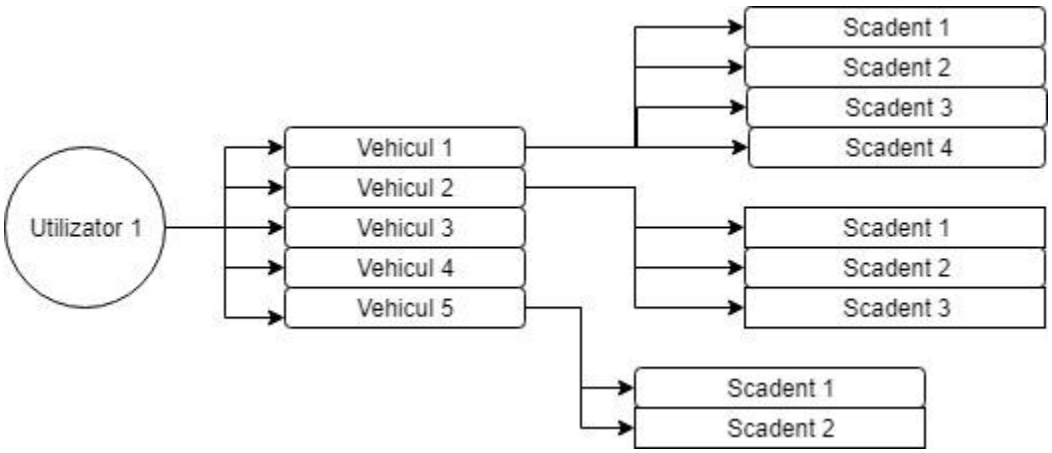

Fig. 4 - Structura entitati

4.2 Inregistrare si Autentificare

Aplicatia este orientata per utilizator, astfel necesita o autentificare pentru a exista o separare intre autovehicule. Pentru ca un utilizator sa poata adauga autovehicule si astfel sa poata fi alertat de apropierea scadentelor documentelor si a reviziilor tehnice, este nevoie sa isi creeze un cont. Pagina de creare cont consta intr-un formular, la care se poate ajunge prin butonul „Sign Up” din bara de navigare

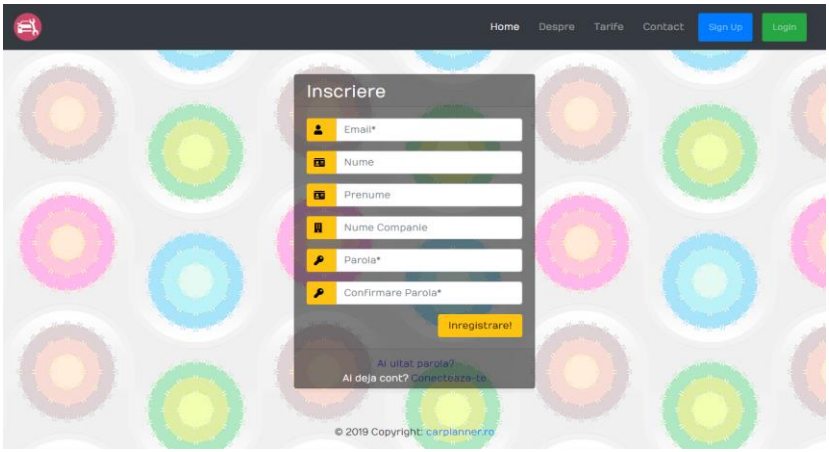

Fig. 5 - Signup

Dupa ce un utilizator completeaza formularul de inscriere, va primi un mail pentru a confirma ca respectivul utilizator este detinatorul de drept al acelui email. Dupa confirmarea prin accesarea link-ului primit pe emai, utilizatorul se va putea loga in aplicatie accesand pagina de conectare, pagina la care se poate ajunge, de asemenea, prin apasarea butonului „Login” din bara de navigare

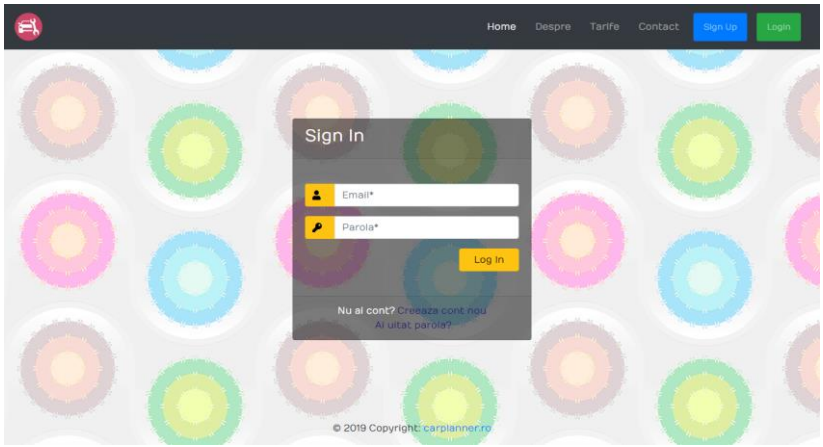

Fig. 6 - Login

Dupa ce un utilizator se logheaza, acesta va fi redirectat catre pagina UserHome, care va fi prezentata in subcapitolul urmator.

In cazul in care un utilizator deja inregistrat in prealabil isi uita parola asociata contului, aceasta poate cere resetarea parolei si trimiterea parolei noi pe email. Pentru a face acest lucru, utilizatorul trebuie sa acceseze link-ul „Ai uitat parola?”, link care se gaseste atat pe pagina „Sign Up” cat si pe pagina „Login”. Este redirectat catre o pagina in care este prezentat un

formular care contine un singur camp, si anume emailul, utilizatorul il intrtroduce, apasa butonul „Reseteaza parola”, iar daca exista un cont asociat acelui mail, se va trimite un mail cu un link pe care utilizatorul trebuie sa il acceseze pentru a reseta parola contului. Pagina pentru resetarea parolei este urmatoarea:

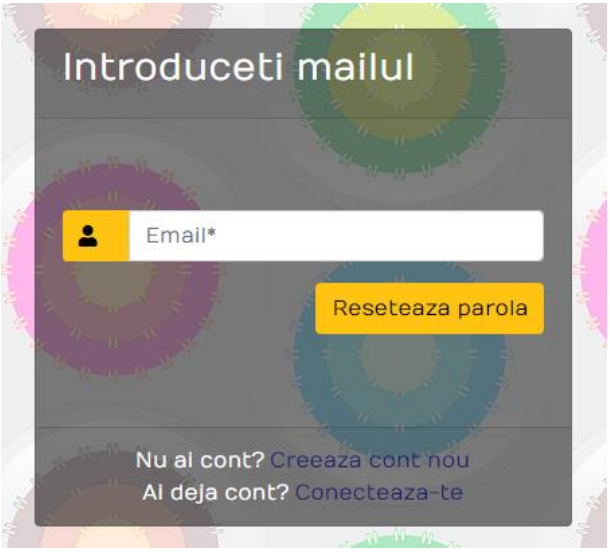

Fig. 7 - Resetare parola

4.3 User Home Page

Pagina UserHome este pagina default afisata dupa ce un user se logheaza. Aceasta contine un mesaj de bun venit, imaginea de profil a utilizatorului si lista de masini pe care o are adaugata in aplicatie.

Fig. 8 - User Home

Pentru fiecare autovehicul din baza de date, pe aceasta pagina este afisat urmatorul scadent in functie de data (daca exista) si urmatorul scadent in functie de data (daca exista). Pentru mai multe detalii despre aceste scadente, utilizatorul poate apasa pe link-ul „Detalii” pentru masina in cauza si acesta va fi redirectat la pagina „Detalii Autovehicul”:

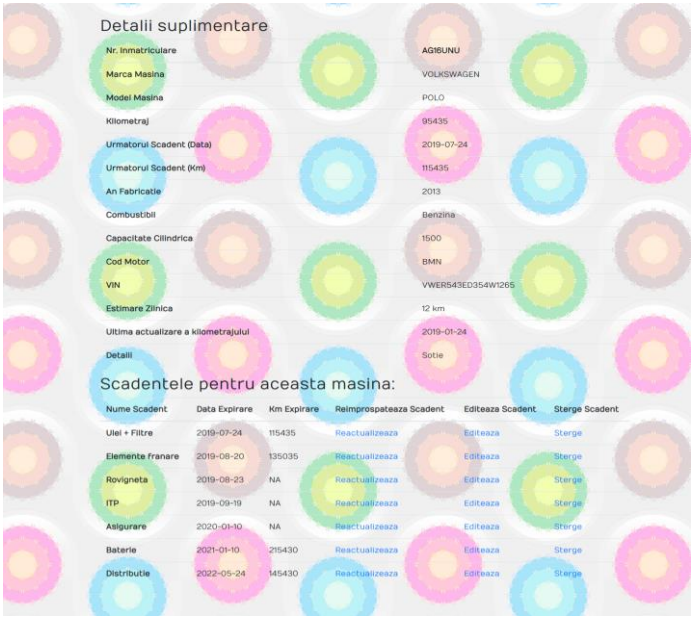

Fig. 9 - Detalii Masina

In aceasta pagina utilizatorul poate vedea toate informatiile stocate pentru masina selectata si anume: numar inmatriculare, marca, model, ultimul kilometraj inregistrat, urmatorul scadent (dupa data), urmatorul scadent (dupa kilometri), anul de fabricatie al autovehiculului, combustibilul, capacitatea cilindrica (daca este cazul), codul motor, seria de sasiu (VIN)m estimarea zilnica (estimarea numarului de kilometri pe care ii face autovehiculul, in functie de ultimele actualizari ale kilometrajului), ultima actualizare a kilometrajului si detaliile masinii.

4.4 Actualizare si stergere cont

Un utilizator care este logat, isi poate modifica datele contului si anume: email, nume, prenume, companie, parola, pagina de profil prin apasarea butonului „Editare Cont” din cadrul paginii „Contul meu” al utilizatorului.

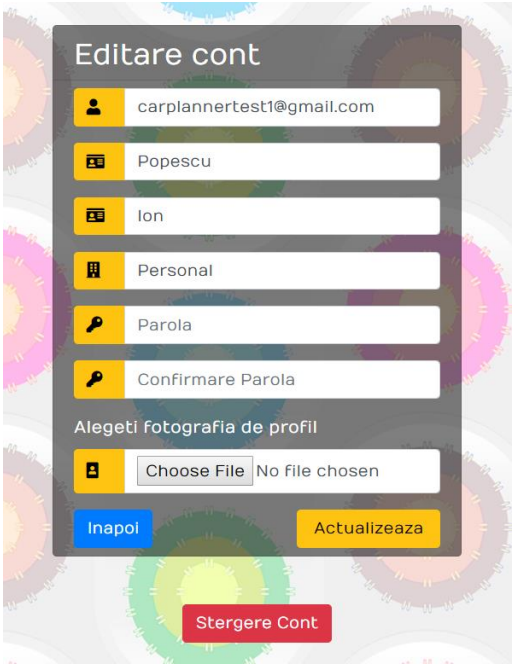

Fig. 10 - Editare cont

In cadrul acestei pagini, utilizatorul isi poate actualiza emailul, numele, prenumele, compania (unde este cazul), parola, si, de asemenea, isi poate incarca si o poza noua de profil. Tot din cadrul acestei pagini, utilizatorul isi poate sterge contul prin apasarea butonului „Stergere Cont", care vine cu o confirmare, bineinteles. Codul care se executa atunci cand un utilizator confirma ca doreste stergerea contului sau este urmatorul:

```
user = db.session.query(User).filter(User.email == email).first()
masini = db.session.query(Masina).filter(Masina.IDUser == user.IDUser).all()
for masina in masini:
 scadente = db.session.query(Scadent).filter(Scadent.IDMasina == masina.IDMasina).delete()
 masina = db.session.query(Masina).filter(Masina.IDMasina == masina.IDMasina).delete()
logout_user()
db.session.delete(user)
db.session.commit()
```

Fig. 11 - Stergere Cont

In primul pas, se gaseste userul in baza de date, iar apoi se gasesc toate vehiculele care apartin userului care se doreste a fi sters. Pentru fiecare vehicul in parte se sterg scadentele aferente iar apoi se sterge si masina. Dupa stergerea tuturor masinilor apartinand userului care se doreste a fis sters, se sterge si userul si se comite in baza de date

4.5 Adaugare, editare si stergere autovehicul

In ceea ce priveste opreatiile CRUD asupra unui autovehicul, utilizatorul poate face aceste operatii (adaugare autovehicul, editare autovehicul, stergere autovehicul) din pagina UserHome.

Pentru a adauga un anutovehicul, utilizatorul, din pagina „User Home" apasa pe butonul „Adauga Vehicul" si va fi redirectat catre o noua pagina. Aceasta pagina contine un formular in care utilizatorul indroduce marca masinii, modeulul masinii, kilometrajul, anul de fabricatie, combustibilul, capacitatea cilindrica, codul motor, seria de sasiu si in cele din urma eventualele detalii extra despre vehicul.

Campul combustibil este de tipul drop-down, static, cu optiunile salvate static in cod. Marca si modelul, insa, sunt stocate in baza de date in tabela „marci" (mai multe detalii despre structura de date vor fi prezentate in capitolele urmatoare). La fiecare accesare a paginii „Adauga autovehicul" se va interoga baza de date si obtine toate marcile. Utilizatorul alege marca dorita si ii sunt apoi afisate optinile de modele corespunzatoare marcii selectate. De mentionat este faptul ca exista o marca „Altul" in caz ca utilizatorul doreste sa adauge un vehicul a carui marca nu este prezenta in baza de date, si, de asemenea, fiecare marca din baza de date are un model numit „Altul" pe care utilizatorul il selecteaza in cazul in care doreste sa adauge un vehicul al carui marca este prezenta in baza de date, dar modelul nu. Pagina „Adauga vehicul" este urmatoarea:

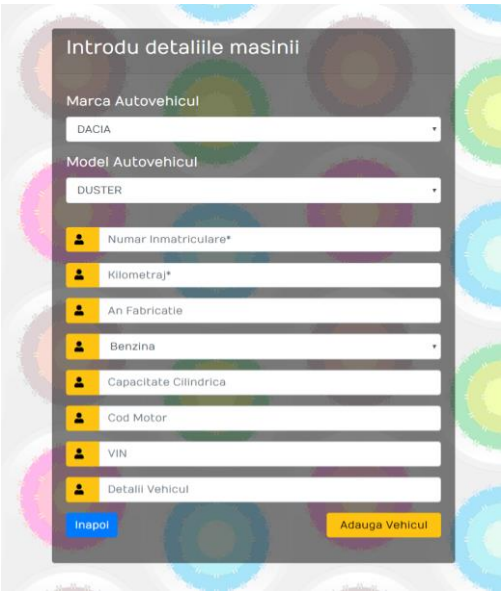

Fig. 12 - Adaugare vehicul

Din pagina User Home, prin accesarea link-ului „Editeaza” utilizatorul este redirectat la urmatoarea pagina, unde acesta poate sa editeze detaliile masinii respective. De precizat este faptul ca atunci cand se actualizeaza campul „Kilometraj”, in spate se actualizeaza campul „Ultima actualizare a kilometrajului” si se recalculeaza campul „Crestere Zilnica” pentru vehiculul respectiv dupa formula (kmNou – kmVechi) / days_between(now, lastUpdate).

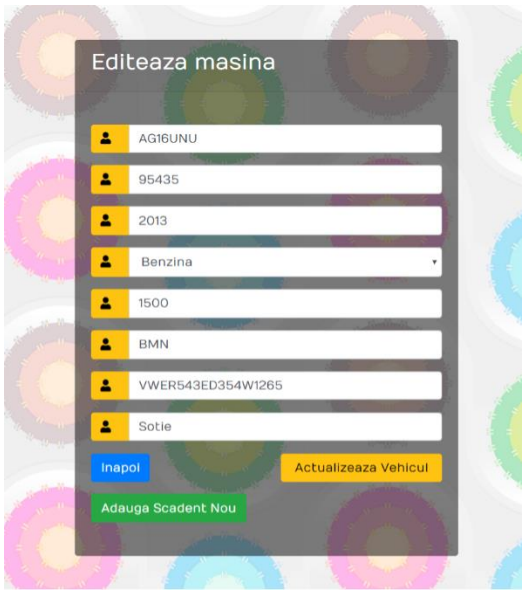

Fig. 13 - Editare vehicul

Pentru stergerea unui vehicul, utlizatorul trebuie sa acceseze link-ul „sterge” din pagina „User Home” de pe linia cu masina pe care doreste sa o stearga. Ulterior se va afisa un mesaj de confirmare, prin care utilizatorul poate alege sa se intoarca la pagina „User Home” sau sa stearga autovehiculul alaturi de toate scadentele asociate acestuia.

4.6 Adaugare, editare, stergere si reimprospatare scadent

Pentru ca aplicatia sa isi indeplineasca scupul si anume acela de a trimite notificari cu privire la scadentele unui autovehicul, fie ele documente sau revizii, utilizatorul trebuie sa adauge pentru fiecare vehicul scadentele dorite.

Un scadent poate fi de doua feluri:

- cu data: acest tip de scadent are doar data de expirare, fara numar de kilometri
- cu data si cu kilometri – acest tip de scadent se considera depasit atunci cand prima din cele doua metrici este depasita; data sau kilometri

Atunci cand utilizatorul adauga un autovehicul nou, este redirectat automat catre o pagina de unde poate selecta care dintre scadentele default pentru marca si modelul vehiculului nou adaugat doreste sa le asocieze cu acesta. Reviziile default pentru fiecare marca si model in parte sunt tinute in tabela „reviziiDefault”. Daca aceasta aplicatie este pusa in piata, aceasta tabela trebuie completata corespunzator cu recomandarile date de producatorul masinii. In prezent, pentru testare, toate modelele de masini au asociate patru schimburi (cu daca si cu kilometri), si anume:

- Ulei si Filtre: 365 zile, 15000 kilometri
- Distributie: 1825 zile, 60000 kilometri
- Elemente franare: 1095 zile, 40000 kilometri
- Baterie: 1825 zile, 60000 kilometri

Pagina in care utilizatorul este intrebat care dintre scadentele default doreste sa le asocieze cu autovehiculul nou adaugat este urmatoarea:

Fig. 14 - Revizii Default

Bineinteles, utilizatorul poate adauga oricand un scadent nou la masinile sale prin apasarea butonului „Adauga Scadent Nou”, buton aflat atat in pagina „Detalii Vehicul” cat si in pagina „Editeaza Vehicul”, ambele aferente vehiculului pentru care se doreste adaugarea scadentului. Formularul pentru adaugarea unui scadent este alcatuit din campuri:

- Nume scadent: numele scadentului
- Viata zile: durata de viata a scadentului; trebuie obligatoriu adaugata
- Viata kilometri: durata de viata (in kilometri) a scadentului; deoarece exista si scadente care nu limita de kilometr (cum ar fi asigurarea RCA), aceasta informatie nu este obligatorie, utilizatorul introducant „0” daca scadentul nu are limita de kilometri

Pagina pentru adaugarea unui scadent este urmatoarea:

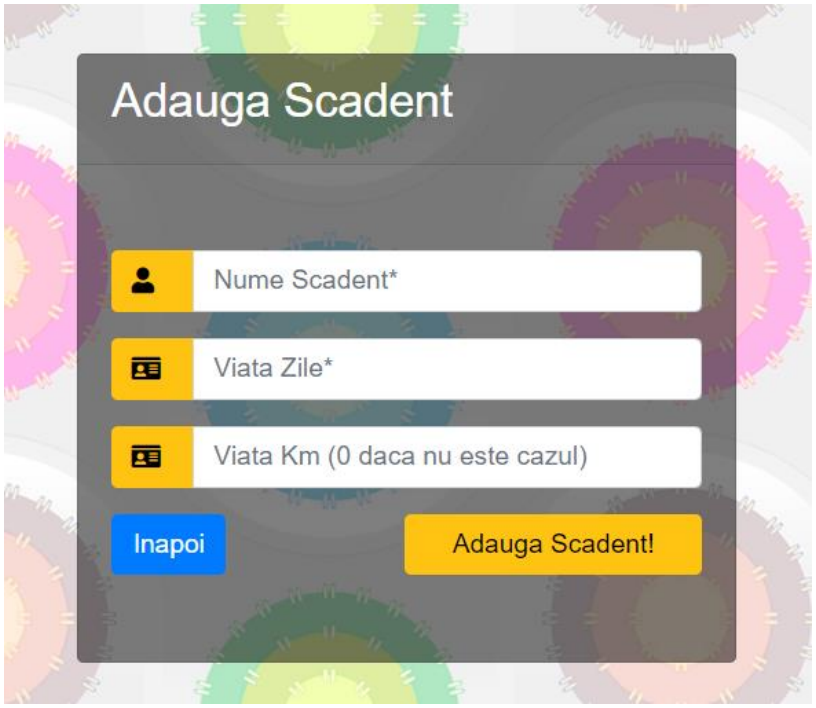

Fig. 15 - Adaugare scadent

Pentru a edita un scadent utilizatorul trebuie sa acceseze pagina „Detalii” a masinii pentru care doreste sa editeze scadentul, iar apoi sa caute link-ul „Editeaza” aferent scadentului pe care doreste sa il editeze. Formularul este identic cu cel adaugare a unui scadent. Un aspect important este faptul ca la editarea unui scadent, nu se actualizeaza datele de expirare actuale ale scadentului, insa se va tine cod de la urmatorul scadent.

Un aspect importat este atunci cand un scadent a fost actualizat de utilizator. Pentru a se modifica si in baza de date, utilizatorul trebuie sa acceseze link-ul „Reimprospateaza” corespunzator scadentului care se doreste a fi actualizat.

Se poate observa in secventa de cod de mai jos ca atunci cand utilizatorul acceseaza link-ul pentru reimprospatarea unui scadent, data de expirare este calculata ca fiind data curenta adunat cu durata de viata a scadentului din baza de date. In plus, daca scadentul are si limita de kilometri, este actualizat si aces parametru.

```
masina = db.session.query(Masina).filter(Masina.IDMasina == IDMasina).first()

scadent = db.session.query(Scadent).filter(Scadent.IDScadent == IDScadent).first()

now = datetime.datetime.now()

scadent.dataExp = now + datetime.timedelta(int(scadent.viataZile))

if scadent.areKM == True:

    scadent.kmExp = int(masina.kilometraj) + int(scadent.viataKm)

db.session.commit()
```

Fig. 16 - Reimprospatare scadent

Pentru stergerea unui scadent, utilizatorul trebuie sa acceseze link-ul „Sterge” din dreptul scadentului care se doreste a fi sters. Ulterior se va afisa un mesaj de confirmare, prin care utilizatorul poate alege sa se intoarca la pagina „Vehicle Details” sau sa stearga scadentul, fara a se sterge si autovehiculul din baza de date

5 DETALII IMPLEMENTARE

5.1 Prezentare generala

Solutia propusa consta intr-o aplicatie web, bazata pe microservicii. Microserviciile, de asemenea cunoscute si ca arhitectura de microservicii, este un stil arhitectural care structureaza o aplicatie ca o colectie de servicii care:

- se pot intretine si testa usor
- sunt slab cuplate
- se pot implementa independent

Structura aplicatiei este urmatoarea, fiecare componenta fiind descrisa in detaliu in subcapitolele ce urmeaza:

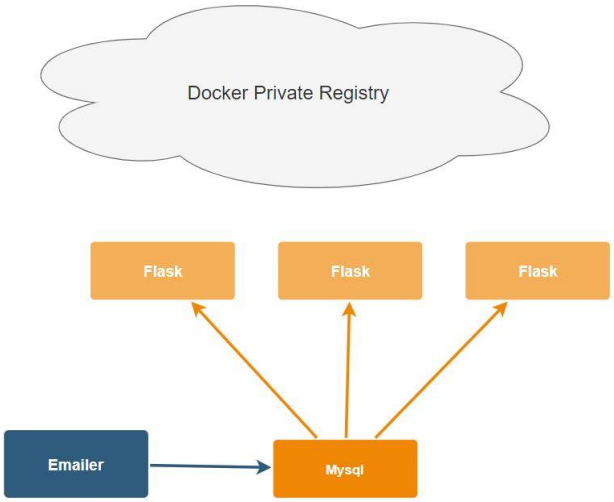

Fig. 17 - Structura aplicatiei

Aplicatia prezentata ruleaza pe un singur host. In cazul in care aplicatia este intens folosita, datorita faptului ca este implementata folsind docker, este usor sa se poata scala pentru a rula pe mai multe servere. O eventuala schema cum ar putea fi extinsa configuratia este prezentata in capitolul 6 unde este explicata o eventuala configuratie in AWS.

5.2 Private Docker Registry

Aplicatia este implementata folosind Docker. Cum a fost prezentat si in subcapitolul 3.7, docker foloseste imagini pentru a rula containerele. Aceste imagini se pot stoca local (de unde se pot pierde usor), in cloud, sau se pot crea din Dockerfile la nevoie. Insa, din punct de vedere al performantei, cel mai practic este ca acestea sa fie tinute intr-un „cloud privat”.

Configuratia docker registry-ului este cea de mai jos:

```
registry:
  image: registry:2
  deploy:
    replicas: 1
    resources:
      limits:
        cpus: "0.2"
        memory: 100M
    restart_policy:
      condition: on-failure
  ports:
    - '444:444'
  environment:
    REGISTRY_HTTP_ADDR: 0.0.0.0:444
    REGISTRY_STORAGE_MAINTENANCE: |-
        uploadpurging:
            enabled: false
        readonly:
            enabled: false
  volumes:
    - /var/lib/car-planner/registry:/var/lib/registry
```

Fig. 18 - Configuratie Docker Registry

Acest Docker Registry este de fapt tot un container Docker care ruleaza din imaginea oficiala “registry” versiunea 2. Politica de restart este on-failure, adica Docker Swarm are grija ca serviciul registry sa fie mereu pornit. De mentionat este faptul ca serviciul ruleaza pe portul 444, astfel pentru a se face un pull de imagine se va folosi comanda:

```
docker pull localhost:444/flask-image:latest
```

Fig. 19 - Comanda docker pull

Aplicatia prezenta prezentata ruleaza pe un singur server (host-ul). In eventualitatea extinderii aplicatiei, se poate seta din fisierul docker-compose ca acest serviciu sa ruleze numai pe un anume server si astfel in loc de “localhost” din comanda va fi IP-ul acelui server.

5.3 Flask

Serviciul WEB este implementat cu ajutorul framework-ului de Python numit Flask. In configuratia prezenta, aceasta componenta ruleaza replicata de trei ori, fiecare replica putand fii independenta fata de cealalta.

Cum s-a explicat si in capitolul precedent, aplicatia are la baza trei entitati: User, Vehicul, Scadent. Operatiile CRUD ale acestor entitati sunt facute cu ajutorul framework-ului Flask si al pachetului Flask-SQLAlchemy, care, cum am vazut si in capitolul precedent, este un wrapper peste baza de date. Astfel sunt baza de data este abstractizata si manipularea datelor din aceasta se rezuma la manipularea unor instante are unor clase. Exemplu de creare / editare / stergere al unui user:

```
# Creare user
user = User(email=form.email.data,
            numeUser=form.numeUser.data,
            prenumeUser=form.prenumeUser.data,
            numeCompanie=form.numeCompanie.data,
            parola=form.parola.data)
db.session.add(user)
db.session.commit()

# Editare user
user = User.query.filter_by(email=form.email.data).first()
user.prenumeUser = form.prenumeUser.data
db.session.commit()

# Stergere user
user = db.session.query(User).filter(User.email == email).first()
db.session.delete(user)
db.session.commit()
```

Fig. 20 - CRUD User

Un aspect foarte practic la Flask-SQLAlchemy este acela ca baza de date este abstractizata astfel incat daca se doreste utilizarea altei baze de date, aplicatia ramane neschimbata cu exceptia setarilor initiale:

```
from flask import Flask
from flask_sqlalchemy import SQLAlchemy

app = Flask(__name__)

app.config['SQLALCHEMY_DATABASE_URI'] = 'mysql+mysqlconnector://root:****@mysql/car-planner'

db = SQLAlchemy(app)
```

Fig. 21 - Configurare baza de date

Un utilizator interactioneaza cu o pagina web cu ajutorul formularelor. Fie ca acesta cauta ceva pe Google, fie ca se logheaza pe Mail, fie ca incarca un fisier, acesta interactioneaza cu fisiere. In aplicatia carplanner, am folosit pachetul WTForms care este un wrapper peste formularele din HTML, acesta fiind mai usor de manipulat din Python. De aceea fiecare entitate (user, vehicul, scadent) contine in folderul lor un fisier numir forms.py care contine descrierea formularelor aferente operatiilor de creare si actualizare.

```
class LoginForm(FlaskForm):
    email = StringField(validators=[DataRequired(message = "Introduceti mailul"),
            Email(message = "Mail invalid")], render_kw={"placeholder": "Email*"})
    parola = PasswordField(validators=[DataRequired(message = "Introduceti parola")],
            render_kw={"placeholder": "Parola*"})
    submit = SubmitField("Log In")
```

Fig. 22 – Exemplu formular Login

Se poate observa ca pentru fiecare camp (field) avem o lista de validatori. Acestia pot fi impliciti, cum ar fi DataRequired sau Email, dar pot fi si creati special. Cu ajutorul acestor validatori se limiteaza spre exemplu lungimea imput-ului si astfel utilizatorul este avertizat ca datele de intrare introduse nu satisfac spre exemplu dimensiunile din baza de date.

Exemple de mesaje pe care un utilizator le poate primi la introducerea unor valori care nu sunt acceptate de aplicatie:

Kilometrajul trebuie sa fie un numar
Capacitatea cilindrica trebuie sa fie un numar
Anul de fabricatie trebuie sa fie un numar
Codul motor introdus este prea lung
VIN-ul introdus este prea lung
Detaliile masinii introduse sunt prea lungi

Fig. 23 – Exemple de mesaje

Pentru a fi usor continuata dezvoltarea aplicatiei, au fost folosite „Blueprints”. Un blueprint de flask este un mod de a organiza aplicatia flask in aplicatii mai mici si refolosibile. La fel ca o aplicatie normala Flask, un blueprint defineste o colectie de view-uri, template-uri si obiecte statice. Astfel codul este mai usor de mentinut si de depanat.

5.4 Emailer

Aceasta componenta functioneaza ca un daemon. In fiecare zi scaneaza baza de date si trimite mail utilizatorului cu scadentele care sunt aproape de a expira. De asemenea, aceasta componenta trimite in fiecare saptamana o lista cu urmatoarele 10 scadente. La versiunea curenta a aplicatiei, utilizatorii nu isi pot modifica / anula notificarile prin mail. Se pot adauga spre exemplu feature-uri care sa permita utilizatorului sa selecteze pentru ce autovehicul sa primeasca mail-uri, sau pentru ce tipuri de scadente sa primeasca mailuri, dar in prezent utilizatorul va fi notificat in fiecare saptmana cu urmatoarele 10 scadente, si incepand cu 3 zile inainte de expirarea unui scadent, incluziv dupa ce acesta expira, pana la reimprospatare.

Exemplu de mail saptamnal cu urmatoarele 10 scadente:

Notificare saptamanala status scadente Mesaje primite ×

carplannerroot@gmail.com
către eu

Salut Ion!

Mai jos gasesti o lista cu urmatoarele 10 scadente pentru masinile tale:

Marca masina	Model masina	Numar Inmatriculare	Nume Scadent	Data expiraare	Km expirare
TOYOTA	RAV4	**AG99UNU**	Ulei + Filtre	2019-07-14	155433
VOLKSWAGEN	POLO	**AG16UNU**	Ulei + Filtre	2019-07-24	115435
VOLKSWAGEN	POLO	**AG16UNU**	Elemente franare	2019-08-20	135035
TOYOTA	RAV4	**AG99UNU**	Distributie	2019-08-22	195400
VOLKSWAGEN	POLO	**AG16UNU**	Rovigneta	2019-08-23	NA
LOTUS	ELISE	**AG77UNU**	ITP	2019-09-10	NA
VOLKSWAGEN	POLO	**AG16UNU**	ITP	2019-09-19	NA
LOTUS	ELISE	**AG77UNU**	Ulei + Filtre	2019-10-24	30430
MERCEDES	S	**AG01UNU**	ITP	2019-11-01	NA
MERCEDES	S	**AG01UNU**	Asigurare	2019-11-10	NA

Pentru mai multe detalii poti intra in contul tau

Fig. 24 – Exemplu de notificare

Mail-urile sunt trimise cu ajutorul pachetului “smtplib”, iar continutul lor este HTML:

```
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText

def sendMail(email, subject, body):

  msg = MIMEMultipart('alternative')
  msg['Subject'] = subject
  msg['From'] = gmail_user
  msg['To'] = email

  HTMLpart = MIMEText(body, 'html')
  msg.attach(HTMLpart)

  try:
    server = smtplib.SMTP_SSL('smtp.gmail.com', 465)
    server.ehlo()
    server.login(gmail_user, gmail_password)
    server.sendmail(gmail_user, email, msg.as_string())
    server.close()

    print('Email sent to ' + email)
  except:
    print('Something went wrong when sending email to ' + email)
```

Fig. 25 – Trimitere mail

5.5 MySQL

Pentru a stoca persistent cele trei entitati (user, vehicul, scadent) avem nevoie de o baza de date, accesibila atat din componenta Flask (pentru operatiile CRUD), cat si din componenta Emailer (pentru raportare scadente).

Baza de date MySQL, o baza de date OpenSource, de tip relational. Aceasta componenta este, la fel ca celelalte, un serviciu al aplicatiei. Configurarea din docker-compose este urmatoarea:

```
mysql:
  image: mysql:5.7
  environment:
    MYSQL_DATABASE: 'car-planner'
    MYSQL_ROOT_PASSWORD: 'mypassword'
  deploy:
    replicas: 1
    restart_policy:
      condition: on-failure
  volumes:
    - /var/lib/car-planner/mysql:/var/lib/mysql
    - /var/lib/car-planner/mysqlconf/mycustom.cnf:/etc/mysql/conf.d/mycustom.cnf
  networks:
    - webnet
```

Fig. 26 – Configurare MySQL docker-compose

Pentru a porni o baza de date intr-un container si nu direct pe host, trebuiesc setate cel putin doua variabile de mediu, acelea fiind: „MYSQL_DATABASE” si „MYSQL_ROOT_PASSWORD”. Se poate observa, de asemenea, maparea fisierului „mycystom.cnf” in container, fisier care contine cativa parametri de fine-tuning.

Structura bazei de date este urmatoarea:

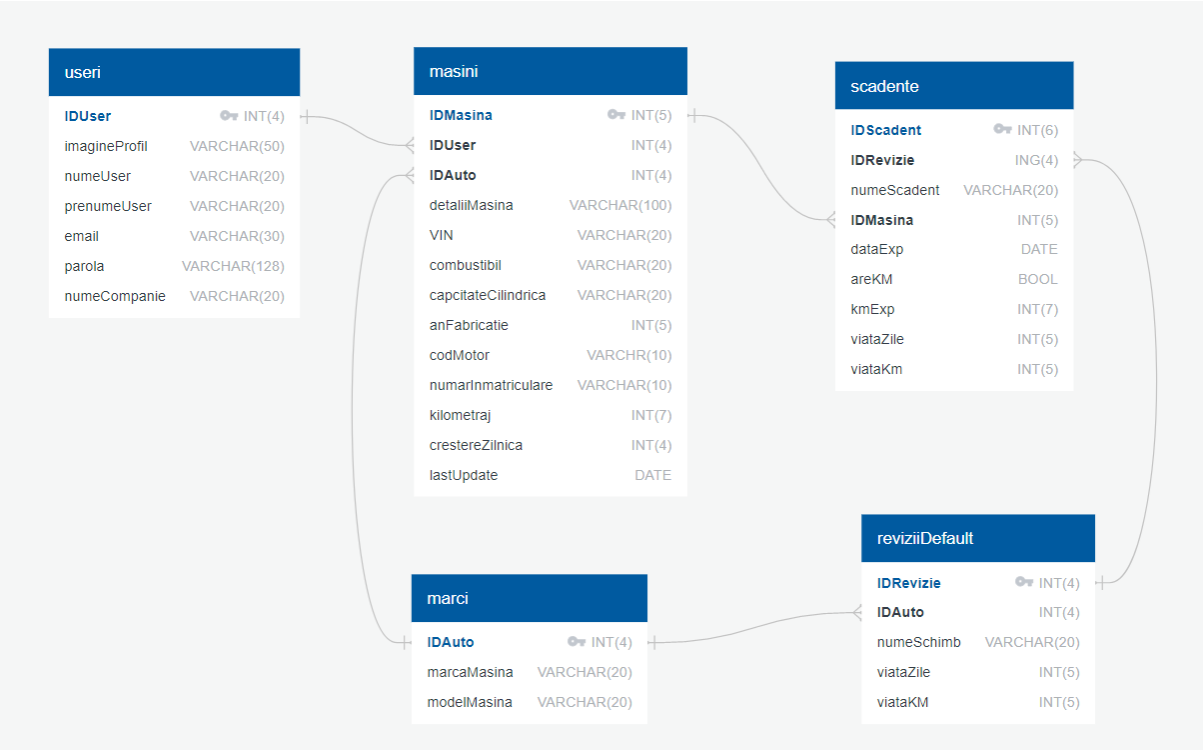

Fig. 27 – Structura bazei de date

Mai jos voi incerca sa descriu tabelele prezente in baza de date si legaturile dintre acestea:

1) Marci – tabela este populata cu cele mai comune marci si modele de masini populare in Europa. Cheia primara este IDAuto
2) ReviziiDefault – tabela care contine, pentru fiecare model de masina, o serie de revizii default pe care utilizatorul le poate asocia cu o masina proaspat adaugata. Cheia primara este IDRevizie. Are si o cheie straina IDAuto catre tabela Marci
3) Useri – tabela care contine toti userii inregistrati in aplicatie. Aici sunt stocate numele, prenumele, email-ul (care este unic), parola (stocata in hash) si numele copaniei (daca este cazul). Cheie primara este IDUser
4) Masini – tabela care contine toate masinile adaugate de utilizatori. Aici sunt stocate pe langa detaliile masinii, ultimele actualizari ale kilometrajului pentru fiecare masina.

Cheie primara este IDUser, o cheie straina este IDUser catre tabela Useri si alta cheie straina IDAuto catre tabela Marci

5) Scadente – tabela care contine toate scadentele ale tuturor masinilor ale tuturor utilizatorilor. Cheia primara este IDScadent, o cheie straina este IDMasina catre tabela Masini si alta cheie straina IDRevizie catre tabela ReviziiDefault

5.6 Securitate

In fiecare zi, suntem expusi, atat acasa cat si la locul de munca, la amenintari care isi au originea in spatiul virtual. In majoritatea cazurilor nici macar nu suntem constienti de acest lucru sau daca il realizam, nu reactionam la aceste amentintari intr-o maniera adecvata. De aceea, securitatea unei aplicatii ar trebui sa fie pentru orice persoana un criteriu definitoriu in alegerea folosirii unei solutii software, cu atat mai mult cand acea solutie presupune stocarea de informatii personale. Niciodata nu sunt destule metode de siguranta pentru a preveni compromiterea unei aplicatii, insa in implementarea acestei aplicatii am incercat sa tin cont de cat mai multe aspecte de securitate cum ar fi:

1) Autentificare. Fiecare utilizator, pentru a putea folosi aplicatia trebuie sa isi faca un cont personal cu care va trebui sa se autentifice ulterior pentru a putea adauga, edita sau sterge vehicule si scadente. Totodata, un utilizator nu poate accesa nici o pagina a altui utilizator, dupa cum se poate vedea in urmatoarea secventa de cod:

```
if email != current_user.email:
  # Forbidden, No Access
  abort(403)
```

Fig. 28 – Restrictionarea accesului

Daca totusi un utilizator va incerca sa acceseze o pagina a altui utilizator (pagina care necesita autentificae bineinteles), va fi redirectat catre pagina custom „403”:

Fig. 29 – Pagina 403 – Acces interzis

2) Verificare email. Atunci cand un utilizator doreste sa isi creeze un cont, contul nu va fi implicit creat, ci va fi intr-o stare de asteptare, pana cand utilizatorul va accesa linkul primit in mailul de verificare. Astfel, o persoana de rea-credinta nu isi poate face un cont in aplicatie cu o adresa de mail ca're nu ii apartine. Acelasi rationament este valabil si cu functionalitatea “Am uitat parola”. Pentru a se resta parola, utilizatorul va trebui sa acceseze un link primit pe email la adresa specificata in formular.

3) Parolele nu sunt stocate in clar in baza de date. In baza de date se tine un hash al parolei, hash calculat cu ajutorul functiei “generate_password_hash” din pachetul “werkzeug.security”, pachet Python cunoscut pentru nivelul de securitate pe care il ofera. In figura de mai jos se poate observa cum sunt tinute parolele in baza de date:

```
mysql> select email, parola from useri;
+---------------------------+-----------------------------------------------------------------------------------------------------+
| email                     | parola                                                                                              |
+---------------------------+-----------------------------------------------------------------------------------------------------+
| carplannertest1@gmail.com | pbkdf2:sha256:50000$tHe34VXH$bee840acfab392f2502029f8faa638f10ac7765a55949af2e8e5d61a674482a4 |
| carplannertest2@gmail.com | pbkdf2:sha256:50000$1v6XHET7$294f7752663ac27a31061667b97ac49b83342e7dee945c2441b495ef6224bb0a |
+---------------------------+-----------------------------------------------------------------------------------------------------+
2 rows in set (0.00 sec)
```

Fig. 30 – Stocare parola baza de date

4) SQL Injection. Datorita faptului ca se foloseste wrapper-ul SQLAlchemy, si nu direct comenzi de inserare / updatare in baza de date, posibilitatea de SQL injection este eliminata deoarece acest pachet Python are inclusa sanitizarea fiecarei operatii cu baza de date in vederea SQL injection

5) Folosirea protocolului HTTPS, care este un protocol pentru o cumunicare securizata in cadrul nagivarii web. Este folosit in cadrul siteurilor care contin date sensibile cum ar fi institutiile banzare, site-urile de cumparaturi, retele sociale, si astfel a devenit un standard in ceea ce priveste securitatea web. Acest protocol se bazeaza pe certificate acordate site-ului de catre o companie emitenta de certificate. Majoritatea companiilor care emit certificate si sunt recunoscute implicit de un broswer web, efectueaza o verificare a companiei care cere certificat pentru e elimina posibilitatea fraudei si a firmelor fantoma. Bineinteles aceste certificate costa si de multe ori pot costa destul de

mult. Pentru aceasta aplicatie, am folosit certificate self-signed, adica am creat o autoritate care semneaza certificate, pe care am adaugat-o in browser-ul web ca fiind recunoscuta. Apoi am semnat un certificat in numele aceste autoritati si astfel comunicare se face prin HTTPS. Daca aceasta aplicatie va ajunge in piata, este nevoie a se cumpara un certificat de la o autoritate recunoscuta de majoritatea browserelor web pentru a se evita clasicul mesaj din broswere “Back to safety”.

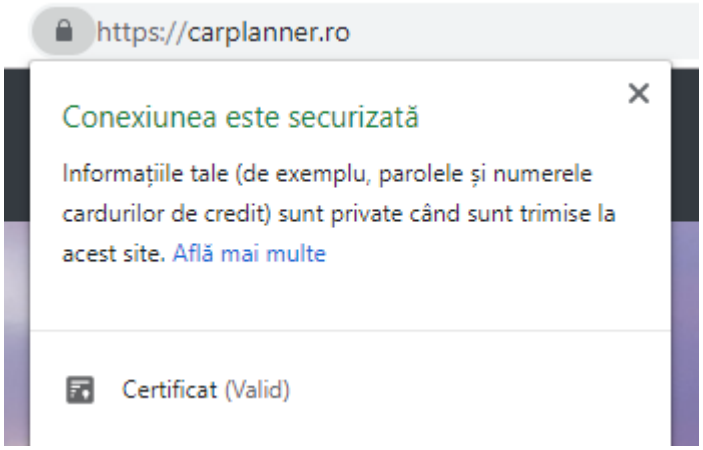

Fig. 31 - HTTPS

Aplicatia ruleaza folosint certificatele generate de autoritatea locala de semnare:

```
app.run(host='0.0.0.0', ssl_context=('servercert.pem', 'serverkey.pem'))
```

Fig. 32 – Rulare aplicatie HTTPS

6) WTForms au in plus fata de formularele clasice din HTML o securitate preimplementata, cum ar fi verificarea contra “cross-site request forgery” sau CRSF.

7) Folosirea SECRET_KEY din cagrup framework-ului Flask. Aceasta cheie este folosite pentru semnarea securizata a cookie-urilor sesiunii si poate fi de asemenea folosita pentur orice alta extensie a aplicatiei care necesita securitate.

6 CONCLUZII SI DEZVOLTARI ULTERIOARE

6.1 Concluzii

Aplicatia Carplanner este un instrument care vine in ajutorul posesorilor de masini dar si firmelor care detin numeroase vehicule, si le ofera un suport pentru gestionarea mai eficienta a documentelor si a reviziilor vehiculelor.

Ca orice solutei nou aparuta, trebuie incercata in piata pentru a ne da seamna daca are succes sau nu. In momentul de fata, aplicatia poate fi pusa pe un server in cloud (spre exemplu Amazon AWS), achizitionat un domeniu si un certificat si pusa in piata fara probleme.

In realizarea acestei solutii am folosit mai multe tehnologii, unele deja cunoscute de mine MySQL, Python, Docker iar aletele in totalitate noi, cum ar fi Flask, WTForms, HTML+CSS.

6.2 Dezvoltari ulterioare

Pentru a creste sansele de a avea succes, aplicatia necesita multe imbunatatiri viitoare:

- Deploy in cloud, pe mai multe servere, pentru a creste fiabilitatea in caz de esuare. Un exemplu ar fi urmatorul: 3 manageri si 3 workeri. Pe manageri nu ruleaza containere (astfel pot avea resurse mai putine), acestia doar coordonand alocarea cintainerelor pe nodurile worker:

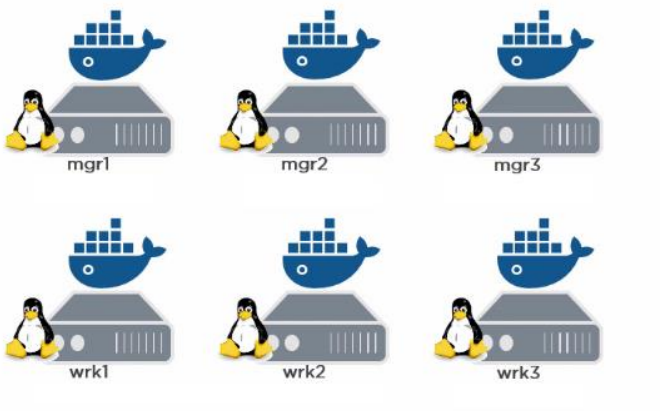

Fig. 33 – Eventuala configuratie cloud

```
ubuntu@ip-172-31-42-177:~$ sudo docker node ls
ID                            HOSTNAME            STATUS              AVAILABILITY        MANAGER STATUS      ENGINE VERSION
i95jajoo39o70dkg2ujbbxjfi     ip-172-31-24-232    Ready               Active                                  18.09.2
l4iy5bvxjf2suoc773657pm48     ip-172-31-26-25     Ready               Active              Reachable           18.09.2
thnkaphwpxvg806m66369ffmb     ip-172-31-26-96     Ready               Active                                  18.09.2
u5wjjln6yifpcaaj33s1chqyt     ip-172-31-34-38     Ready               Active              Reachable           18.09.2
zi7oebfhutrkn27fvakww6ylf     ip-172-31-35-220    Ready               Active                                  18.09.2
ygobfg02ki3otd2fqpmb2jdhp *   ip-172-31-42-177    Ready               Active              Leader              18.09.2
ubuntu@ip-172-31-42-177:~$
```

Fig. 34 – Eventuala configuratie cloud

- Componenta MySQL sa poata fi replicata, pentru a creste eficienta in caz ca numarul de utilizatori va creste. Se poate utiliza modul default de replicare al MySQL, master-slave sau se poate utiliza spre exemplu GlusterFS pentur a rula in modul master-master (ambele noduri pot scrie simultan fata de modul default master-slave unde doar nodul master poate scrie, iar nodurile slave pot doar citi)

- In cazul in care se opteaza pe un deployment pe mai multe servere, se poate configura si un load-balancer, pentru o mai buna manipulare a cererilor clientilor

- Pentru a putea sugera corect lista de revizii default pe care un utilizator este intrebat daca doreste sa le adauge unui nou vehicul adaugat, acestea trebuiesc sa fie corespunzatoare cu ce recomanda producatorul. Astfel trebuie populata corect si cat mai complet tabela „reviziiDefault” pentru fiecare model de masina in parte

- O alta imbunatatire ar fi permiterea utilizatorului in a customiza vehiculele si scadentele pentru care doreste sa primeasca notificari, dar si cu cate zile sa fie notificat inainte de expirarea fiecarui scadent

- Se poate, de asemnea, opta si pentru notificarea prin SMS a utilizatorului. Trebuie modificata baza de date astfel incat sa se stocheze si numarul de telefon al utilizatorului, si de asemnea achizitionat o solutie care sa trimita efectiv notificarile SMS

- Putem adauga un newsletter cu articole referitoare la noile legislatii si amenzi, noutati in lumea automobilelor, detalii tehnice si orice altceva ce are legatura cu domeniul auto.

7 BIBLIOGRAFIE

[1]https://adevarul.ro/locale/suceava/povestea-inginerului-automobilul Accesat: 17.06.2019

[2]https://www.forbes.com/sites/kateashford/americans-in-collections/ Accesat: 12.05.2019

[3]http://www.ziare.com/auto/inmatriculare-auto/masini-inmatriculate Accesat: 20.06.2019

[4]http://www.ziare.com/auto/inmatriculare-auto/sunt-aproape-8-milioane-de-masini-inmatriculate-in-romania-1521407 Accesat: 17.06.2019

[5]https://www.my-car.co/ Accesat: 17.05.2019

[6]https://www.autominder.ro/ Accesat: 06.06.2019

[7]https://www.alertemasina.ro/ Accesat: 06.06.2019

[8]https://home.cern/science/computing/birth-web/short-history-web Accesat: 10.06.2019

[9]https://webdesign.tutsplus.com/articles/world-webdesign-8710 Accesat: 17.03.2019

[10]https://webfoundation.org/about/vision/history-of-the-web/ Accesat: 17.06.2019

[11]https://www.w3schools.com/html/html_basic.asp Accesat: 17.03.2019

[12]https://www.w3schools.com/whatis/whatis_htmldom.asp Accesat: 10.06.2019

[13]https://www.w3schools.com/html/html_elements.asp Accesat: 12.06.2019

[14] https://www.tutorialspoint.com/html/html_overview.htm Accesat: 13.06.2019

[15]https://www.w3schools.com/html/html_classes.asp Accesat: 14.06.2019

[16]https://www.w3schools.com/html/html_attributes.asp Accesat: 06.06.2019

[17]https://www.w3schools.com/html/html5_intro.asp Accesat: 17.06.2019

[18]https://www.w3schools.com/bootstrap4/bootstrap_get_started.asp Accesat: 17.05.2019

[19]https://startbootstrap.com/ Accesat: 17.03.2019

[20]https://getbootstrap.com/docs/4.0/getting-started/introduction/ Accesat: 17.06.2019

[21]https://getbootstrap.com/docs/4.0/getting-started/accessibility/ Accesat: 17.06.2019

[22] https://www.pythonforbeginners.com/learn-python/what-is-python/ Accesat: 17.06.2019

[23] https://www.learnpython.org/en/Variables_and_Types Accesat: 17.06.2019

[24] https://www.w3schools.com/python/python_variables.asp Accesat: 01.06.2019

[25] https://github.com/ Accesat: 04.05.2019

[26] http://flask.pocoo.org/docs/1.0/foreword/ Accesat: 17.05.2019

[27] https://pypi.org/project/WTForms/ Accesat: 17.05.2019

[28] https://wtforms.readthedocs.io/en/stable/forms.html Accesat: 01.06.2019

[29] https://searchitoperations.techtarget.com/definition/Docker-Swarm Accesat: 17.05.2019

[30] https://www.siteground.com/tutorials/php-mysql/mysql/ Accesat: 01.05.2019

[31] https://www.hostinger.com/tutorials/what-is-mysql Accesat: 01.06.2019

[32] https://www.sqlalchemy.org/ Accesat: 01.06.2019

8 ANEXE

8.1 Baza de date

```
mysql> show tables;
+-----------------------+
| Tables_in_car-planner |
+-----------------------+
| marci                 |
| masini                |
| reviziiDefault        |
| scadente              |
| useri                 |
+-----------------------+
5 rows in set (0.00 sec)

mysql> desc marci;
+-------------+-------------+------+-----+---------+----------------+
| Field       | Type        | Null | Key | Default | Extra          |
+-------------+-------------+------+-----+---------+----------------+
| IDAuto      | int(11)     | NO   | PRI | NULL    | auto_increment |
| marcaMasina | varchar(20) | YES  |     | NULL    |                |
| modelMasina | varchar(20) | YES  |     | NULL    |                |
+-------------+-------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)
```

Fig. 35 – Tabele 1

```
mysql> desc masini;
+----------------------+--------------+------+-----+---------+----------------+
| Field                | Type         | Null | Key | Default | Extra          |
+----------------------+--------------+------+-----+---------+----------------+
| IDMasina             | int(11)      | NO   | PRI | NULL    | auto_increment |
| IDUser               | int(11)      | NO   | MUL | NULL    |                |
| IDAuto               | int(11)      | NO   | MUL | NULL    |                |
| detaliiMasina        | varchar(100) | YES  |     | NULL    |                |
| VIN                  | varchar(20)  | YES  |     | NULL    |                |
| combustibil          | varchar(20)  | YES  |     | NULL    |                |
| capacitateCilindrica | varchar(20)  | YES  |     | NULL    |                |
| anFabricatie         | int(11)      | YES  |     | NULL    |                |
| codMotor             | varchar(20)  | YES  |     | NULL    |                |
| numarInmatriculare   | varchar(20)  | NO   |     | NULL    |                |
| kilometraj           | int(11)      | NO   |     | NULL    |                |
| crestereZilnica      | int(11)      | YES  |     | NULL    |                |
| lastUpdate           | date         | YES  |     | NULL    |                |
+----------------------+--------------+------+-----+---------+----------------+
13 rows in set (0.00 sec)

mysql> desc reviziiDefault;
+------------+-------------+------+-----+---------+----------------+
| Field      | Type        | Null | Key | Default | Extra          |
+------------+-------------+------+-----+---------+----------------+
| IDRevizie  | int(11)     | NO   | PRI | NULL    | auto_increment |
| IDAuto     | int(11)     | NO   | MUL | NULL    |                |
| numeSchimb | varchar(20) | NO   |     | NULL    |                |
| viataZile  | int(11)     | YES  |     | NULL    |                |
| viataKm    | int(11)     | YES  |     | NULL    |                |
+------------+-------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)
```

Fig. 36 – Tabele 2

```
mysql> desc scadente;
+-------------+-------------+------+-----+---------+----------------+
| Field       | Type        | Null | Key | Default | Extra          |
+-------------+-------------+------+-----+---------+----------------+
| IDScadent   | int(11)     | NO   | PRI | NULL    | auto_increment |
| IDRevizie   | int(11)     | NO   | MUL | NULL    |                |
| numeScadent | varchar(20) | NO   |     | NULL    |                |
| IDMasina    | int(11)     | NO   | MUL | NULL    |                |
| dataExp     | date        | YES  |     | NULL    |                |
| areKM       | tinyint(1)  | YES  |     | NULL    |                |
| kmExp       | int(11)     | YES  |     | NULL    |                |
| viataZile   | int(11)     | YES  |     | NULL    |                |
| viataKm     | int(11)     | YES  |     | NULL    |                |
+-------------+-------------+------+-----+---------+----------------+
9 rows in set (0.01 sec)

mysql> desc useri;
+---------------+--------------+------+-----+---------+----------------+
| Field         | Type         | Null | Key | Default | Extra          |
+---------------+--------------+------+-----+---------+----------------+
| IDUser        | int(11)      | NO   | PRI | NULL    | auto_increment |
| imagineProfil | varchar(50)  | NO   |     | NULL    |                |
| numeUser      | varchar(30)  | NO   |     | NULL    |                |
| prenumeUser   | varchar(30)  | NO   |     | NULL    |                |
| email         | varchar(30)  | NO   |     | NULL    |                |
| parola        | varchar(128) | YES  |     | NULL    |                |
| numeCompanie  | varchar(30)  | YES  |     | NULL    |                |
+---------------+--------------+------+-----+---------+----------------+
7 rows in set (0.00 sec)
```

Fig. 37 – Tabele 3

8.2 Script build si rulare

```
#! /bin/bash

if [ "$1" == "help" ]; then
  echo "./build.sh help      <-  help"
  echo "./build.sh first     <-  first run on machine"
  echo "./build.sh flask      <-  build flask-image and start
docker-compose"
  echo "./build.sh emailer   <-  build emailer-image and start
docker-compose"
  echo "./build.sh all         <-  build all images and start
docker-compose"
  echo "./build.sh start     <-  start docker compose"
  echo "./build.sh stop      <-  stop docker compose"
  echo "./build.sh populate  <-  populate database tables"
  exit
fi

if [ "$1" != "populate" ]; then
  echo Exit Docker Swarm
```

```
  sudo docker swarm leave --force
  sleep 3
  if [ "$1" != "stop" ]; then
    echo Init Docker Swarm
    sudo docker swarm init
  fi
  sleep 3
fi

if [ "$1" == "first" ]; then
  # delete all containers
  sudo docker rm -f $(docker ps -a -q)

  sleep 1

  # delete all images
  sudo docker image rm -f $(docker images -q -a)

  # clean tree
  sudo rm -rf /var/lib/car-planner

  # create tree
  sudo mkdir /var/lib/car-planner
  sudo mkdir /var/lib/car-planner/registry
  sudo mkdir /var/lib/car-planner/mysql
  sudo mkdir /var/lib/car-planner/flask
  sudo mkdir /var/lib/car-planner/emailer

  # copy application files
  sudo cp -R ./Flask/* /var/lib/car-planner/flask
  sudo cp -R ./Emailer/* /var/lib/car-planner/emailer

  sleep 1
fi

if [ "$1" != "stop" ] && [ "$1" != "first" ]; then
  echo "Removing /var/lib/car-planner/flask/*"
  sudo rm -rf /var/lib/car-planner/flask/*
  echo "Removing /var/lib/car-planner/emailer/*"
  sudo rm -rf /var/lib/car-planner/emailer/*
  echo "Copying to /var/lib/car-planner/flask/*"
  sudo cp -R ./Flask/* /var/lib/car-planner/flask/
  echo "Copying to /var/lib/car-planner/emailer/*"
  sudo cp -R ./Emailer/* /var/lib/car-planner/emailer/
fi

if [ "$1" == "flask" ] || [ "$1" == "first" ]; then
  echo "Start building flask-image"
  sudo    docker    build    --no-cache    ./Dockerfiles    -f
./Dockerfiles/FlaskDockerfile -t flask-image
```

```
fi

if [ "$1" == "emailer" ] || [ "$1" == "first" ]; then
  echo "Start building emailer-image"
  sudo    docker    build    --no-cache    ./Dockerfiles    -f
./Dockerfiles/EmailerDockerfile -t emailer-image
fi

if [ "$1" == "all" ]; then
  echo "Start building flask-image"
  sudo    docker    build    --no-cache    ./Dockerfiles    -f
./Dockerfiles/FlaskDockerfile -t flask-image
  echo "Start building emailer-image"
  sudo    docker    build    --no-cache    ./Dockerfiles    -f
./Dockerfiles/EmailerDockerfile -t emailer-image
fi


if [ "$1" != "stop" ]; then
  sleep 1
  sudo docker stack deploy -c docker-compose.yml car-planner
fi

if [ "$1" == "stop" ]; then
  # delete all containers
  echo "Delete all containers"
  sudo docker rm -f $(docker ps -a -q)
fi

if [ "$1" == "populate" ]; then
  docker exec $(docker ps | grep flask | cut -d " " -f1 | head
-1) python3 populateTables.py
fi
```