|  |  |  |
| --- | --- | --- |
|  | UNIVERSITATEA DIN CRAIOVA  FACULTATEA DE AUTOMATICĂ, CALCULATOARE ȘI ELECTRONICĂ  DEPARTAMENTUL DE [CALCULATOARE ȘI TEHNOLOGIA INFORMAȚIEI / AUTOMATICĂ, ELECTRONICĂ ȘI MECATRONICĂ] |  |

PROIECT DE DIPLOMĂ

Popescu Alexandru-Iulian

COORDONATOR ȘTIINȚIFIC

[*Titlul științific, prenumele și numele coordonatorului*]

[*Luna (în litere) și anul susținerii proiectului*]

CRAIOVA

|  |  |  |
| --- | --- | --- |
|  | UNIVERSITATEA DIN CRAIOVA  FACULTATEA DE AUTOMATICĂ, CALCULATOARE ȘI ELECTRONICĂ  DEPARTAMENTUL DE [CALCULATOARE ȘI TEHNOLOGIA INFORMAȚIEI / AUTOMATICĂ, ELECTRONICĂ ȘI MECATRONICĂ] |  |

Platforma pentru intermedierea activităților de curierat folosind tehnologiile Cloud

Popescu Alexandru-Iulian

COORDONATOR ȘTIINȚIFIC

[*Titlul științific, prenumele și numele coordonatorului*]

[*Luna (în litere) și anul susținerii proiectului*]

CRAIOVA*„Învățătura este o comoară care își urmează stăpânul pretutindeni.”*

Proverb popular

**DECLARAȚIE DE ORIGINALITATE**

Subsemnatul Popescu Alexandru-Iulian, student la specializarea Calculatoare cu predare în limba română din cadrul Facultății de Automatică, Calculatoare și Electronică a Universității din Craiova, certific prin prezenta că am luat la cunoştinţă de cele prezentate mai jos şi că îmi asum, în acest context, originalitatea proiectului meu de licenţă:

* cu titlul Platforma pentru intermedierea activităților de curierat folosind tehnologiile Cloud,
* coordonată de [*TITLUL ȘTIINȚIFIC, PRENUMELE ȘI NUMELE COORDONATORULUI*],
* prezentată în sesiunea Iulie 2022

La elaborarea proiectului de licenţă, se consideră plagiat una dintre următoarele acţiuni:

* reproducerea exactă a cuvintelor unui alt autor, dintr-o altă lucrare, în limba română sau prin traducere dintr-o altă limbă, dacă se omit ghilimele şi referinţa precisă,
* redarea cu alte cuvinte, reformularea prin cuvinte proprii sau rezumarea ideilor din alte lucrări, dacă nu se indică sursa bibliografică,
* prezentarea unor date experimentale obţinute sau a unor aplicaţii realizate de alţi autori fără menţionarea corectă a acestor surse,
* însuşirea totală sau parţială a unei lucrări în care regulile de mai sus sunt respectate, dar care are alt autor.

Pentru evitarea acestor situaţii neplăcute se recomandă:

* plasarea între ghilimele a citatelor directe şi indicarea referinţei într-o listă corespunzătoare la sfărşitul lucrării,
* indicarea în text a reformulării unei idei, opinii sau teorii şi corespunzător în lista de referinţe a sursei originale de la care s-a făcut preluarea,
* precizarea sursei de la care s-au preluat date experimentale, descrieri tehnice, figuri, imagini, statistici, tabele et caetera,
* precizarea referinţelor poate fi omisă dacă se folosesc informaţii sau teorii arhicunoscute, a căror paternitate este unanim cunoscută și acceptată.

Data, Semnătura candidatului,

05.09.2022

|  |  |  |
| --- | --- | --- |
|  | UNIVERSITATEA DIN CRAIOVA  Facultatea de Automatică, Calculatoare şi Electronică  Departamentul de [Calculatoare și Tehnologia Informației / Automatică, Electronică și Mecatronică] | Aprobat la data de  …………………  Şef de departament,  Prof. dr. ing.  Marius BREZOVAN/  Emil PETRE |

**PROIECTUL DE DIPLOMĂ**

|  |  |
| --- | --- |
| Numele și prenumele studentului/-ei: |  |
| Enunțul temei: | [*Titlul lucrării / descrierea pe scurt a temei*] |
| Datele de pornire: | [*Descrierea datelor inițiale de la care s-a început activitatea de cercetare/dezvoltare a tezei*] |
| Conținutul proiectului: | [*Descrierea succintă a conținutului fiecărui capitol al lucrării*] |
| Material grafic obligatoriu: |  |
| Consultații: | [*Periodice/zilnice/săptămânale/lunare*] |
| Conducătorul științific  (titlul, nume și prenume, semnătura): | Șef lucrări dr. ing. Marius MARIAN |
| Data eliberării temei: | 01.12.2022 |
| Termenul estimat de predare a proiectului: | 01.06.2022 |
| Data predării proiectului de către student și semnătura acestuia: |  |

|  |  |  |
| --- | --- | --- |
|  | UNIVERSITATEA DIN CRAIOVA  Facultatea de Automatică, Calculatoare şi Electronică  Departamentul de [Calculatoare și Tehnologia Informației / Automatică, Electronică și Mecatronică] |  |

**REFERATUL CONDUCĂTORULUI ȘTIINȚIFIC**

|  |  |
| --- | --- |
| Numele și prenumele candidatului/-ei: |  |
| Specializarea: | [*Denumirea oficială a specializării absolvite de candidat*] |
| Titlul proiectului: | [*Titlul lucrării*] |
| Locația în care s-a realizat practica de documentare (se bifează una sau mai multe din opțiunile din dreapta): | În facultate □ |
| În producție □ |
| În cercetare □ |
| Altă locație: [*se detaliază*] |

În urma analizei lucrării candidatului au fost constatate următoarele:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nivelul documentării | | Insuficient  □ | Satisfăcător □ | Bine  □ | Foarte bine  □ |
| Tipul proiectului | | Cercetare  □ | Proiectare  □ | Realizare practică □ | Altul  [*se detaliază*] |
| Aparatul matematic utilizat | | Simplu  □ | Mediu  □ | Complex □ | Absent  □ |
| Utilitate | | Contract de cercetare □ | Cercetare internă □ | Utilare  □ | Altul  [*se detaliază*] |
| Redactarea lucrării | | Insuficient  □ | Satisfăcător □ | Bine  □ | Foarte bine  □ |
| Partea grafică, desene | | Insuficientă  □ | Satisfăcătoare □ | Bună  □ | Foarte bună  □ |
| Realizarea practică | Contribuția autorului | Insuficientă  □ | Satisfăcătoare □ | Mare  □ | Foarte mare  □ |
| Complexitatea  temei | Simplă  □ | Medie  □ | Mare  □ | Complexă  □ |
| Analiza cerințelor | Insuficient  □ | Satisfăcător □ | Bine  □ | Foarte bine  □ |
| Arhitectura | Simplă  □ | Medie  □ | Mare  □ | Complexă  □ |
| Întocmirea specificațiilor funcționale | Insuficientă  □ | Satisfăcătoare □ | Bună  □ | Foarte bună  □ |
| Implementarea | Insuficientă  □ | Satisfăcătoare □ | Bună  □ | Foarte bună  □ |
| Testarea | Insuficientă  □ | Satisfăcătoare □ | Bună  □ | Foarte bună  □ |
| Funcționarea | Da  □ | Parțială  □ | Nu  □ | |
| Rezultate experimentale | | Experiment propriu  □ | | Preluare din bibliografie  □ | |
| Bibliografie | | Cărți | Reviste | Articole | Referințe web |
| Comentarii  și  observații | |  | | | |

În concluzie, se propune:

|  |  |
| --- | --- |
| ADMITEREA PROIECTULUI  □ | RESPINGEREA PROIECTULUI  □ |

Data, Semnătura conducătorului științific,

**REZUMATUL PROIECTULUI**

Platforma web poate fi folosita de către orice firma de curierat, atât de către partea interna (gestionarea pachetelor, a șoferilor) cat si de către clienții acestei firme pentru crea cereri de ridicare a pachetelor. Aplicația poate fi folosita pentru orice tip de comenzi care folosesc ansamblul general care este folosit de către aplicațiile ca Fancurier, Glovo sau Takeaway (șoferi, comenzi, Google maps, rute, dispecerat).

**Etapele întregului proces de crearea si livrare de pachete**

Prima etapa a procesului este reprezentata de folosirea unui cont de Administrator pentru a crea conturi pentru șoferul angajat si pentru dispeceratul care va avea grija de logistica din spate. După ce administratorul a creat conturile, dispeceratul poate adaugă mașinile care vor fi folosite pentru livrări in baza de date si poate vedea toate pachetele care au fost create de client. Acesta poate intra apoi pe pagina de rute pentru a crea o ruta care este făcută din mai multe comenzi si o poate asigna unui șofer împreuna cu o mașina care este libera. Șoferul poate începe ruta când este pregătit prin apăsarea butonului de start si o poate actualiza si finaliza când este gata.

***Termenii cheie***: curierat, administrare, CRUD, dispecerat, Cloud, maps

**MULȚUMIRI**

În această secțiune opțională (în eng., *Acknowledgements*), autorul are ocazia de a face o declarație de recunoștință față de oricine (conducătorul științific/alte persoane apropiate autorului/instituții/organizații/et caetera) a susținut sau a contribuit la realizarea lucrării sale.

**CUPRINSUL**

[1 Introducere 15](#_Toc105882320)

[1.1 Scopul 15](#_Toc105882321)

[1.2 Motivația 15](#_Toc105882322)

[1.3 Descriere generală 15](#_Toc105882323)

[1.3.1 Tehnologii și aplicații folosite 15](#_Toc105882324)

[1.3.2 Tehnologii și framework-uri folosite 16](#_Toc105882325)

[1.4 Arhitectura sistemului 19](#_Toc105882326)

[1.5 Azure cloud services 20](#_Toc105882327)

[1.6 Structura bazei de date 23](#_Toc105882328)

[1.7 Proiectarea aplicației 27](#_Toc105882329)

[2 Cerinte pentru sistemul implementat 29](#_Toc105882330)

[2.1 Funcționalitățile administratorului 29](#_Toc105882331)

[2.1.1 Autentificare 29](#_Toc105882332)

[2.1.2 Dashboard conturi 29](#_Toc105882333)

[2.1.3 Adăugare cont 29](#_Toc105882334)

[2.1.4 Editare cont 29](#_Toc105882335)

[2.1.5 Logout 29](#_Toc105882336)

[2.2 Funcționalitățile dispeceratului 29](#_Toc105882337)

[2.2.1 Autentificare 29](#_Toc105882338)

[2.2.2 Dashboard vehicule 30](#_Toc105882339)

[2.2.3 Dashboard comenzi 30](#_Toc105882340)

[2.2.4 Dashboard șoferi 30](#_Toc105882341)

[2.2.5 Dashboard rute 30](#_Toc105882342)

[2.3 Funcționalitățile șoferului 30](#_Toc105882343)

[2.3.1 Operatii pe ruta curenta 30](#_Toc105882344)

[2.4 Funcționalitățile clientului final 31](#_Toc105882345)

[2.4.1 Estimare cost transport 31](#_Toc105882346)

[2.4.2 Editare profil si locatii 31](#_Toc105882347)

[2.4.3 Creeare comanda 31](#_Toc105882348)

[2.4.4 Verificare stadiu comanda 32](#_Toc105882349)

[3 Manualul utilizatorului 32](#_Toc105882350)

[3.1 Pagina de home (utilizator neinregistrat) 32](#_Toc105882351)

[3.2 Crearea unui cont de utilizator 34](#_Toc105882352)

[3.3 Pagina de home (utilizator inregistrat) 35](#_Toc105882353)

[3.4 Pagina de profil 35](#_Toc105882354)

[3.5 Pagina de comenzi 36](#_Toc105882355)

[3.6 AWB Tracking 39](#_Toc105882356)

[4 Concluzii 40](#_Toc105882357)

[4.1 Îmbunătățiri si dezvoltare 40](#_Toc105882358)

[5 Bibliografie 41](#_Toc105882359)

[Site-ul web al proiectului 42](#_Toc105882360)

# Introducere

## Scopul

Acest document are rolul de a prezenta aplicatia web de curierat care poate fi folosita atat de utilizatorul final cat si de catre firma de curierat pentru gestionarea pachetelor.

## Motivația

Platforma de curierat a fost facuta cu scopul de a putea fi folosita si de catre alte firme de start-up care vor sa inceapa o firma de curierat. Evolutia omenirii si a tehnologiei a facut ca livrarea de orice tip sa fie foarte cautata si folosita in orice domeniu datorita usurintei de a obtine orice este necesar din comoditatea locuintei, chiar si la distante mari.

## Descriere generală

### Tehnologii și aplicații folosite

Aplicatia este o platformă web dezvoltată folosind framework-ul ASP .Net Core 3.1 cu modelul arhitectural MVC (Model-View-Controller) și cu Entity Framework Core 3.1.4 iar pentru autentificare și autorizare a fost integrat și folosit serviciul Identity versiunea 2.2.0. Aplicația a fost dezvoltată folosind IDE-ul Visual Studio 2022 și Microsoft SQL Server Management Studio 18 ca și mediu integrat care se ocupă cu managementul bazei de date. Partea de Cloud a fost facută folosind Azure, portalul celor de Microsoft pentru management si Visual studio pentru publicare si modificare.

În plus, pentru a avea un istoric al dezvoltarii si pentru siguranta, a fost folosit sistemul de control al versiunilor numit GIT, iar pentru găzduirea codului sursă a fost folosit serviciul Github.

Pe langa acestea, jQuery a fost folosit pentru comunicarea cu server-ul folosind apeluri AJAX.

Pentru frontend, au fost folosite mai multe plugin-uri JavaScript care au scopul de a face aplicația mai intuitivă, deci de a crește gradul de utilizabilitate cat si cateva animații, fie reutilizate din diferite pachete gasite pe internet, fie personificate pentru a avea o imagina de ansamblu a aplicației cât mai placută. Aceste plugin-uri sunt:

* DataTables, folosit pentru a structura tabelele într-o manieră mai ușor de înțeles
* Google Maps prin scripturi pentru a afișa in mod dinamic distanta si ruta pe harta.

### Tehnologii și framework-uri folosite

Aplicația este o platformă web care a fost implementată folosind framework-ul ASP .Net Core 3.1 cu serviciul de autentificare și autorizare Identity. De asemenea, Entity Framework Core este folosit ca si ORM (Object–Relational Mapper). ORM-ul ne permitem să mapam date din baza de date folosind paradigma programării orientate pe obiecte.

ASP .Net Core este un framework folosit pentru dezvoltarea de site-uri web independente de platformă (cross-platform) acesta folosind șablonul determinat de injectarea dependențelor (en. Dependency Injection sau DI). DI este o tehnică folosită pentru îndeplinirea principiului al 5-lea SOLID numit Dependency Inversion Principle între clase și dependențele lor. Așadar, repository-urile, dar și serviciile vor fi înregistrate în metoda ConfigureServices din clasa Startup.

În cazul serviciul Identity, acesta este un serviciu de autorizare și autentificare, fiind legat la o bază de date SQL Server pentru a stoca datele utilizatorilor cum ar fi parolele, email-urile, dar și alte date personale. Funcționalitățile de register, login, și editare a contului de utilizator au fost inițial create de serviciul Identity, acestea fiind apoi customizate pentru a se mula mai bine pe scopul proiectului. De asemenea, serviciul Identity oferă și funcționalitatea de autorizare astfel încât utilizatorii vor putea accesa numai paginile destinate rolului pe care îl au, deci clientii nu vor avea acces la funcționalitățile soferilor, etc.

Dacă un utilizator care nu este autentificat încearcă să acceseze o pagină pentru care este nevoie să se autentifice, acesta va fi redirectat către pagina de autentificare, iar dacă utilizatorul este autentificat, dar încearcă să acceseze o pagină destinată altui tip de utilizator, va fi redirectat către o pagină de acces interzis („Access denied”).

De asemenea, Identity detine si optiunea validării parolelor inserate. Aceste opțiuni fiind inserate în clasa de Startup.cs. Condițiile folosite sunt: parola trebuie să aibă minim 8 caractere, să conțină minim o cifră, o majusculă și un caracter alfanumeric.

Pe lângă tehnologiile descrise, au fost folosite și pachete NuGet, acestea fiind instalate prin intermediul managerului de pachete. Pachetele folosite sunt: GoogleMapsApi (folosit pentru implementarea de funcționalității de harți) si Razor runtime compilation pentru a putea face debug si hot reload fara a închide aplicația.

jQuery, care este o librarie de javascript ce poate procesa evenimente, manipula elemente html si a utiliza cereri de tip AJAX, a fost de asemenea folosit impreuna cu anumite plugin-uri.

În plus, Bootstrap a fost folosit pentru view-uri pentru a face ca site-ul să fie responsive. Bootstrap este un toolkit open-source, care include un sistem grid responsive, diferite componente, dar și plugin-uri JavaScript.

Unul din plugin-urile folosite este DataTables, care a fost conceput pentru a modifica tabelele HTML cu scopul de a fi interactive. Prin utilizarea acestei librării, se vor stiliza tabelele folosind stilurile cuprinse de librărie. Acest plugin depinde doar de jQuery și necesită ca jQuery să fie integrat pentru a putea să fie folosit.

Plugin-ul DataTables oferă multiple funcționalități care fac aplicația mult mai ușor de utilizat și de înțeles cum ar fi:

* posibilitatea de a sorta orice coloana aleasa de utilizator fie în ordine crescătoare sau descrescătoare
* împărțirea rezultatelor pe mai multe pagini
* posibilitatea de a alege câte rezultate sunt afișate în tabel
* filtrarea anumitor rezultate pe baza unui input dat de utilizator
* afișarea numărului total de rezultate

De asemenea, pe pagina de dashboard au fost create animații folosind fișiere de tip SVG care au fost customizate și adaptate. Fișierele SVG au fost descărcate de pe un website care oferă ilustrații gratuite.

În ceea ce privește proiectarea aplicației și reducerea gradului de cuplare, aplicația a fost împărțită în trei straturi: Stratul de prezentare, care include controllerele, view-urile, fișierele de stilizare (CSS), fișierele care conțin scripturile JavaScript, dar și componentele introduse de Identity în aplicație, stratul cu logica aplicației (Application Logic layer) și stratul folosit pentru comunicarea cu baza de date (Data Access layer). Pentru a crea straturile Application Logic și Data Access, au fost folosite cate două librării de clase (class libraries).

În plus, a fost folosit șablonul arhitectural MVC (Model-View-Controller), care separă logica de business de logica de prezentare. În timp ce modelele sunt folosite pentru a mapa tabelele din baza de date relațională, controllerele sunt folosite pentru a randa view-urile, pentru a face redirectările și pentru a procesa cererile emise de utilizator. A treia componentă vizată de acest șablon este reprezentată de view-uri, acestea fiind reprezentate de componentele care sunt afișate în browser.



Figură Diagrama MVC

În ceea ce privește baza de date, aceasta este una relațională. Tipul de bază de date relațională este cel mai răspândit tip de baze de date în care datele sunt memorate în tabele. „Pe lângă tabele, o bază de date relațională mai poate conține: indecși, proceduri stocate, declanșatori, utilizatori și grupuri de utilizatori, tipuri de date, mecanisme de securitate și de gestiune a tranzacțiilor etc.” (Wikipedia, 2022). În plus, pentru baza de date a fost folosit utilitarul Microsoft SQL Server Management Studio 18, care reprezintă un „mediu integrat folosit pentru managementul oricărei infrastructuri SQL [...] SSMS oferă funcții folosite pentru configurarea, monitorizarea și administrarea instanțelor de SQL Server și a bazelor de date” (Microsoft Docs).

Toata aceasta aplicatie a fost publicata online folosind tehnologia Cloud, mai precis Microsoft Azure. Pentru acest lucru a fost nevoie de mai multe resurse, descrise in detaliu in [capitolul 1.5](#_Azure_cloud_services).

## Arhitectura sistemului

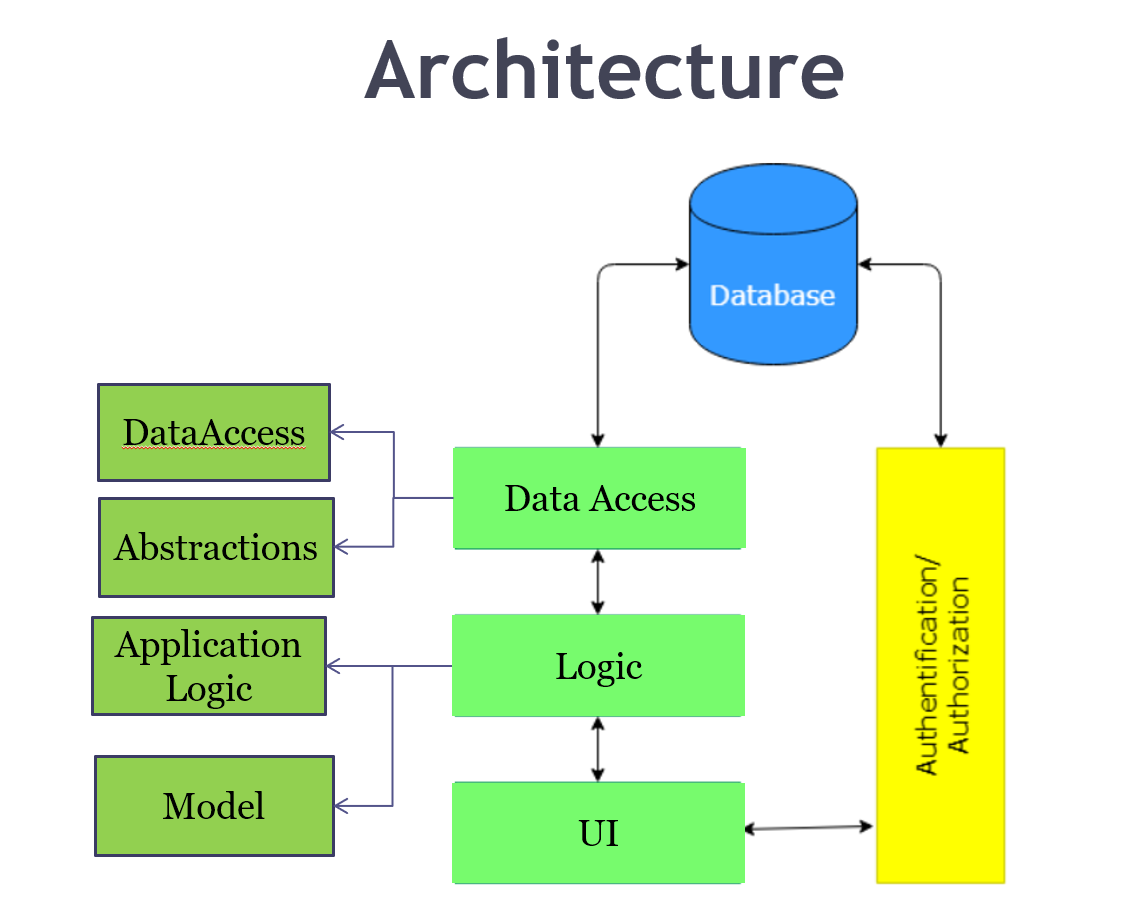
Sistemul nu este unul monolitic, ci este unul format din mai multe straturi care comunică între ele cu scopul de a reduce gradul de cuplare al aplicației, fiecare strat putând fi reutilizat. Cele trei straturi principale folosite sunt: *stratul de prezentare (Presentation layer)*, stratul care cuprinde logica aplicației *(Application logic layer)* și stratul *Data access (Data Access layer)*.

Stratul de prezentare este stratul folosit direct de utilizator și cuprinde: controllerele, view-urile, script-urile JavaScript, fișierele de stilizare CSS, imagini, documentele încărcate de administrator, fișierele de configurare și funcționalitățile generate folosind serviciul de autentificare și autorizare Identity.

Stratul care cuprinde logica aplicației este cel care include serviciile folosite în cadrul aplicației. Pentru fiecare serviciu a fost folosită o interfață pentru a putea reduce gradul de cuplare al aplicației deoarece ultimul principiu SOLID (Dependency Inversion Principle) care se referă la faptul că abstractizările ar trebui folosite în loc de clase concrete este astfel respectat. Acest strat cuprinde: interfețele care sunt implementate de repository-uri, interfețele implementate de servicii, serviciile (clasele concrete) care conțin logica aplicației, modelele care sunt folosite la maparea tabelelor din baza de date, viewmodelele care sunt clase asemănătoare modelelor, dar care nu au un corespondent în baza de date. Viewmodelele sunt modele care conțin multiple proprietăți care nu sunt mapate în baza de date.

Stratul Data Access este stratul folosit pentru interacțiunea cu baza de date și reprezintă o abstractizare a sa. Acest strat cuprinde clasele concrete care reprezintă repository-urile și directorul care include migrările folosite pentru generarea bazei de date (folosind strategia CodeFirst).

De asemenea, a fost folosit Repository Pattern, care este un șablon de proiectare care constă în abstractizarea accesului datelor stocate în baza de date prin intermediul unor metode. Folosind acest șablon, va scădea gradul de cuplare al aplicației. Acest șablon consistă în crearea unei clase de bază generice, în cadrul aplicației fiind numită EFBaseRepository, care va conține metode generice folosite pentru operații de tip CRUD (Create, Read, Update, Delete). În EFBaseRepository sunt definite metodele următoare: GetAll (tipul de return este IEnumerable), GetById (cu tip de return DataEntity), Add (folosită pentru inserarea de noi date/records în baza de date), Update (pentru a actualiza datele existente în baza de date), Remove (pentru a șterge date). De asemenea, pentru fiecare tabel din baza de date a fost creată o clasă de repository care extinde EFBaseRepository. În plus, pentru a reduce gradul de cuplare al aplicației, prin respectarea pricipiilor 2 și 5 SOLID, fiecare repository va implementa o interfață, iar repository-urile, dar și serviciile vor fi înregistrate ca și servicii Scoped în clasa Startup pentru a putea fi folosite prin Dependency Injection. (Wikipedia 2022)



Figură Arhitectura sistemului

## Azure cloud services

Platforma Azure cloud este un serviciu de cloud computing operat de Microsoft pentru gestionarea aplicațiilor prin centre de date gestionate de Microsoft. Oferă software ca serviciu (SaaS), platformă ca serviciu (PaaS) și infrastructură ca serviciu (IaaS) și acceptă multe limbaje de programare, instrumente și cadre diferite, inclusiv software și sisteme specifice Microsoft și terțe părți. (Wikipedia)

* Cont, a fost folosit cel de „@robotics.ucv” pentru a avea beneficiile unui „Azure for students”.
* Subscription: Un abonament Azure este un container de bază care cuprinde un grup de resurse comerciale sau tehnice conexe. Grupul de resurse este utilizat și facturat împreună.



Figură Azure subscription

* Resource group: include toate resursele folosite si descrise mai jos, un grup de resurse este asignat unui abonament (subscription).



Figură Azure resource group

* App service plan: defineste un set de resurse care sunt asignate catre aplicatia respectiva, care se platesc, pentru ca aceasta sa ruleze pe cloud.

Graphical user interface, text, application

Description automatically generated

Figură Azure app service plan

* Sql server: server-ul care sustine bazele de date folosite de aplicatie, acestea pot exista fie separat, fie folosind un elastic pool pentru a impartii un anumit set de resurse.



Figură Azure sql server

* Sql database: baza de date folosita. Aceasta are deja un query editor implementat in Azure, fara a fi nevoie de SQL Management Studio

A picture containing text

Description automatically generated

Figură Azure database/query editor

* Dashboard: poate fi folosit de catre developeri/product owneri pentru a vedea diferite informatii legate de utilizarea resurselor azure, de exemplu, numarul de requesturi catre aplicatie, numarul de mesaje transmite pe un service bus, numarul de timeouts pentru unele requesturi etc.

Graphical user interface, application

Description automatically generated

Figură A Azure dashboard

* Application Insights care este o caracteristică de monitorizare Azure ce oferă un management extensibil al performanței și monitorizarii aplicației web live. Insights poate de exemplu sa detecteze automat anomalii de performanta intampinate, ajuta la diagnosticarea problemelor gasite si poate vedea ce apeluri sunt facute de catre utilizatori

A screenshot of a computer

Description automatically generated with medium confidence

Figură Azure application insights

## Structura bazei de date

Baza de date este una relațională și a fost creată folosind strategia CodeFirst. Mai exact, inainte sa fie creata baza de date, au fost create modelele, iar fiecare model reprezintă o entitate a tabelelor din baza de date în aplicație. După ce modelele au fost create, au fost adăugate în clasa de context, iar cu ajutorul Entity Framework Core, a fost creată o migrare. Migrările sunt localizate în stratul de Data Access în directorul Migrations. Migrările reprezintă clase care conțin două metode: Up și Down. Metoda Up este folosită pentru a aplica schimbările pe baza de date, iar metoda Down este folosită pentru a reseta starea bazei de date, astfel încât după resetare, baza de date se va afla într-o stare stabilă, mai exact starea stabilă inițială. Migrările oferă avantajul că se poate face revert la starea precedentă dacă se constată că migrarea nu ar fi trebuit rulată. Un alt avantaj este că în cazul în care migrarea nu se poate executa cu succes, se aplică mecanismul de roll back, deci baza de date nu va fi alterată, ci se va afla în starea inițială. Migrările sunt construite pe baza migrărilor precedente și prin compararea modelelor folosite în aplicație cu starea curentă a bazei de date. Astfel se poate menține un istoric al schimbărilor și se poate reconstrui baza de date, această acțiune presupunând rularea migrărilor într-o manieră secvențială și cronologică.

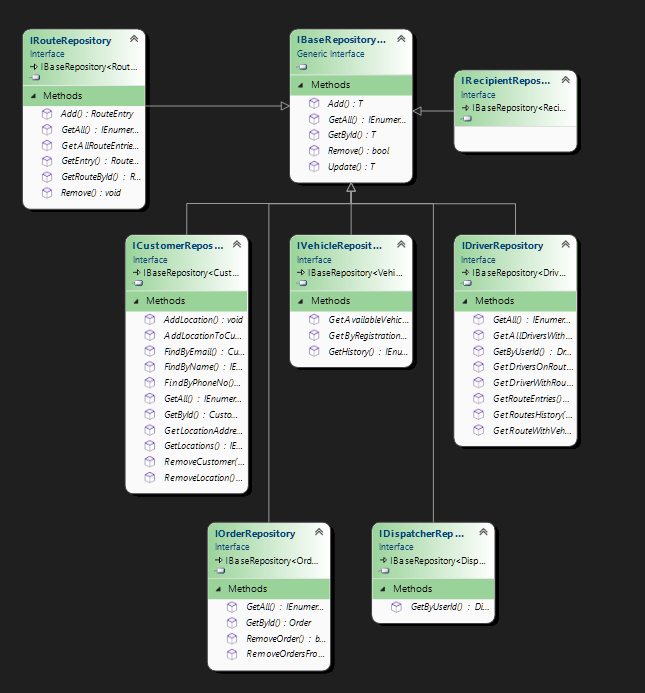
Mai jos au fost adăugate diagramele de clase care sunt împărțite în funcție de principalele funcționalități. Deoarece fiecare model avea nevoie de un Id, sub forma de Guid, pentru a putea fi găsit in baza de date, toate modele moștenesc o clasa comuna si anume DataEntity:



Figură Diagrama de clase (1)

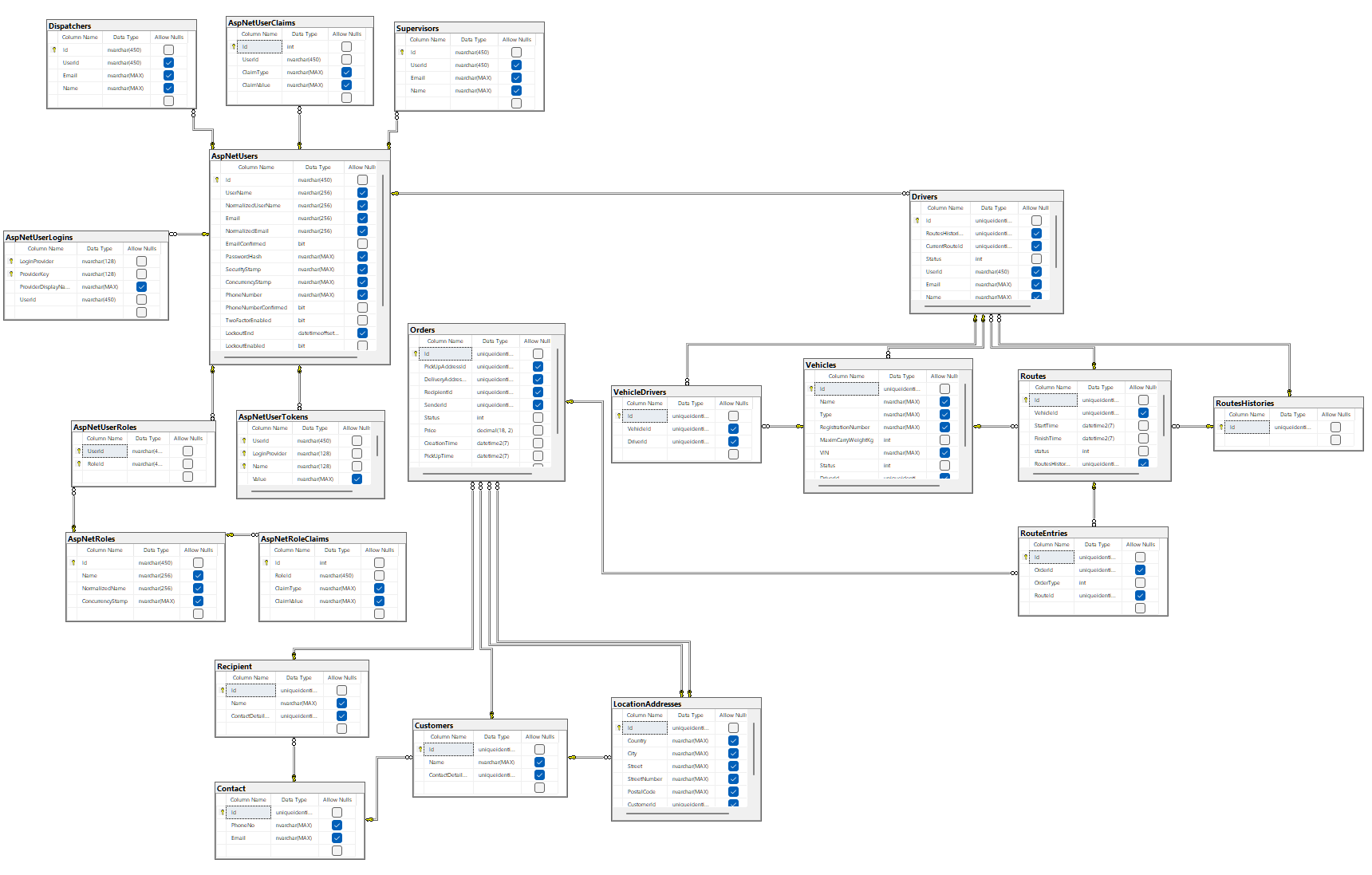


Figură Diagrama de clase (2)



Figură Diagrama de clase (3)

În figurelele următoare se pot observa legăturile dintre tabelele bazei de date:



Figură Modelul relational al bazei de date

## Proiectarea aplicației

Aplicatia de curierat a fost conceputa cu scopul de a fi folosita de toate persoanele care ar putea exista in cercul acestei aplicatiei. Cu acest scop, aplicatia este creata cu mai multe roluri in minte. Mai jos sunt prezentate use-case-urile acestor utilizatori, acestea fiind explicate in detaliu in cadrul cerintelor:

* Client

Diagram

Description automatically generated

Figură Diagrama cazuri de utilizare - Client

* Admin

Diagram

Description automatically generated

Figură Diagrama cazuri de utilizare - Administrator

* Dispecerat

Diagram

Description automatically generated

Figură Diagrama cazuri de utilizare - Dispecerat

* Șofer

Diagram

Description automatically generated

Figură Diagrama cazuri de utilizare - Sofer

# Cerinte pentru sistemul implementat

## Funcționalitățile administratorului

### Autentificare

Contul de administrator este destinat managerilor din firma. Acesta nu se poate crea ca un cont normal de client, ci este creat automat la start-up. Administratorul se poate autentifica folosind ca si user: admin@gmail.com si parola Admin1!. După ce s-a autentificat, administratorul va fi redirectat către pagina de dashboard.

### Dashboard conturi

Apăsând pe butonul de „Accounts”, administratorul va fi întâmpinat de un tabel populat cu toți utilizatorii, pe care acesta poate face operații CRUD.

### Adăugare cont

Administratorul este singurul rol care poate adaugă alte conturi folosite in interiorul firmei (alt admin, dispecerat, șofer).

### Editare cont

Administratorul poate sa modifice conturile deja existente, in caz ca este nevoie de un schimb de username, parola sau rol.

### Logout

Utilizatorii autentificați au posibilitatea de a se deconecta prin folosirea butonului de logout prezent în bara de navigație.

## Funcționalitățile dispeceratului

### Autentificare

După crearea contului de utilizator, dispecerul se poate autentifica folosind email-ul și parola setate la pasul de înscriere. Dacă autentificarea nu a avut succes (date invalide), utilizatorul va primi mesaje de eroare. În cazul în care autentificarea a avut loc cu succes, dispecerul va fi redirectat către pagina de dashboard.

### Dashboard vehicule

Dispecerul are acces la un tabel care in include toate vehiculele adăugate in baza de date. Pe lângă detaliile generale despre un vehicul ( nume, tip, VIN), dispecerul poate vedea si statusul vehiculului ( Free daca este liber, Busy daca este deja atașat unei rute).

### Dashboard comenzi

In tabelul de comenzi, dispecerul poate vedea adresa de ridicare, adresa de livrare, recipientul, user-ul care a făcut comanda (in caz ca este nevoie sa îl sune pentru informații adiționale), statusul comenzii (Created, Assigned, PickedUp, Delivering, Delivered, Canceled), prețul si un buton pentru deschiderea unui pop-up pentru Google maps unde dispecerul poate vedea cea mai rapida ruta creata de Google maps.

### Dashboard șoferi

In tabelul de șofer, dispecerul poate vedea toți șoferii înregistrați (id-ul lor, email-ul si numele), ruta curenta cu cate comenzi sunt asignate pe acea rute, statusul comenzii si daca rute a fost asignata sau nu unui șofer.

### Dashboard rute

Aceasta pagina este folosita pentru deferite acțiuni ce se pot efectua asupra unei rute. Dispecerul poate crea o ruta noua, fiind obligat sa ii adauge un vehicul care are statusul de „Free”. Ruta este creata fără nicio comanda, acestea trebuind sa fie adăugate de pe butonul de „Add Order”. Se poate vedea de asemenea ruta pe Google maps, aceasta unind rutele de la fiecare comanda si încercând sa facă cel mai scurt drum. Daca un pachet este adăugat din greșeala, user-ul poate elimina doar un pachet sau toata ruta. De asemenea, se poate schimba vehiculul cu un altul liber daca este necesar.

## Funcționalitățile șoferului

### Operatii pe ruta curenta

In aceasta pagina, soferul poate vedea ce ruta a fost asignata acestuia de catre unul dintre dispeceri. Dupa ce soferul are o ruta, acesta poate vedea pachetele care sunt include in acea ruta si de asemenea poate vedea, folosind Google Maps, cel mai scurt drum pentru a putea ridica si lasa toate comenzile incluse in ruta respectiva. In acest moment, pachetele au tag-ul de „Assigned”. Cand soferul este gata de plecare, acesta poate apasa pe butonul de „Start Route”, tag-ul pachetelor devening „Delivering”. O data ce ruta este pornita, fiecare pachet are doua optiuni: Delivered (pachetul a ajuns cu succes la destinatie) si Cancel ( au fost diferite probleme cu pachetul). O data ce toate pachetele din ruta respectiva ajung la una din cele doua stari de mai sus, ruta se poate incheia apasand pe butonul „End Route”, soferul devenind iar liber si pregatit de o noua ruta.

## Funcționalitățile clientului final

### Estimare cost transport

Clientul are acces la un calculator de costuri estimativ pentru transport. Acesta trebuia sa introduca valoarea comenzii, ce tip de transport este nevoie sa fie chemat pentru comanda respectiva in functie de dimensiuni si kilometraj (bicicleta, masina, camion) si un numar de kilometrii.

### Editare profil si locatii

Clientul poate folosi pagina de profil pentru a edita detalii despre acesta cum ar fi numele, numarul de telefon, email-ul si, in special, locatiile de ridicare a coletelor. O locatie noua are nevoie obligatoriu de urmatoarele date (tara, oras, strada, nr.strada, cod postal) si optional de un „Tag”, acesta fiind folosit in loc de toata adresa pentru a f mai usor de gasit (de ex. Home).

### Creeare comanda

Pe pagina de dashboard a comenzilor, clientul poate apasa pe butonul de „New Order” pentru a deschide modalul cu informatiile necesare pentru a crea o comanda noua:

* PickUp Address: este un dropdown cu toate locatiile care le are clientul respectiv (adresa sau, daca exista, tag-ul)
* Country
* City
* Street
* Street Number
* Postal Code
* Recipient Name
* Recipient Email
* Recipient Phone Number
* Price

### Verificare stadiu comanda

Stadiul unei comenzi poate fi vazut din doua locuri: Daca clientul este deja logat pe site, acesta poate intra pe dashboard-ul de comenzi si poate analiza statusul comenzii in tabel. In schimb, daca acesta nu este logat si doar vrea sa verifice statusul undei comenzi, acesta o poate face din dashboard-ul site-ului principal, fara a se loga, prin folosirea awb-ului generat dupa crearea comenzii. Clientul poate introduce AWB-ul in inputul specificat pentru a afla statusul, in timp real, al comenzii.

# Manualul utilizatorului

## Pagina de home (utilizator neinregistrat)

Pagina de home poate fi accesata atat de un utilizator autentificat cat si de unul care nu s-a autentificat. Pe aceasta pagina utilizatorul poate vedea cateva detalii despre aplicatie, o parte de „About Us”, o parte de contact si un estimator de costuri de transport.

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, chat or text message

Description automatically generated

Pentru calculatorul de cost, orice client isi poate calcula cu aproximatie pretul pe care trebuie sa il plateasca curierului, in functie de 3 clasificari: valoarea declarata a pachetului, dimensiunile aproximative si numarul de kilometrii.

Text

Description automatically generated with medium confidence

De asemenea, calculator are si un buton de „Read More”, unde detinatorul final al aplicatiei poate scrie detalii despre cum este calculata rata de transport, etc.

Text, timeline

Description automatically generated

## Crearea unui cont de utilizator

Pentru a crea un cont de utilizator, se va accesa pagina de inregistrare (apasant pe butonul de „Register” din card-ul de autentificare. Dupa apasare, utilizatorul va fi redirectionat catre pagina de register

Graphical user interface, text, application, chat or text message

Description automatically generated

## Pagina de home (utilizator inregistrat)

Pagina de start este „imbunatatita” cand un utilizator este inregistrat. Pe langa card-urile de autentificare si cost estimate se mai adauga un card pentru comenzi si unul pentru editarea profilului.

Graphical user interface

Description automatically generated

## Pagina de profil

Apasand pe butonul de profil, utilizatorul este redirectional la pagina de editare a profilului, unde poate modifica numele, adresa de email si, in special, locatiile.

A screenshot of a computer

Description automatically generated with medium confidence

Graphical user interface, text, application, chat or text message

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

Pentru a adauga o locatie, este nevoie de urmatoarele informatii obligatorii: Tara, Oras, Strada, Numarul strazii, codul posta. Optional, utilizatorul isi poate alege si un „Tag” pentru locatia adaugata, fie la creare, fie la editare. Acest tag poate fi ceva familiar pentru a fi mai usor de gasit locatia in liste (de ex. Home). Daca un tag este introdus, acesta v-a aparea in locul adresei cand este aleasa pentru o comanda.

## Pagina de comenzi

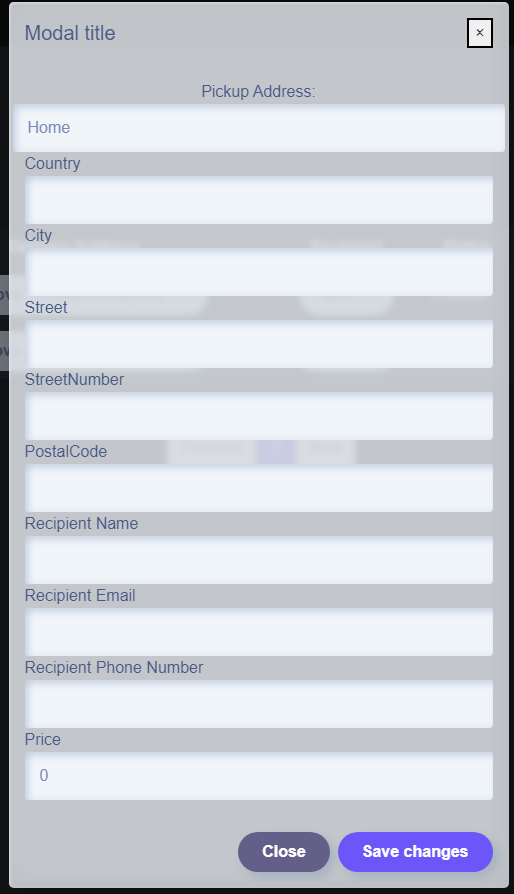
Pagina de comenzi include un tabel, facut cu ajutotul plugin-ului „DataTables”, care include detalii despre comenzi cum ar fi: adresa de ridicare, adresa de destinatie, recipientul si informatiile despre el (intr-un dropdown), statusul comenzii, awb, pretul si optiunea de a deschide google maps pentru a vedea ruta pe care v-a merge soferul pentru a efectua comanda.

A screenshot of a computer screen

Description automatically generated with medium confidence

Dupa apasarea butonului de „New Order” userul trebuie sa introduca anumite date pentru o noua comandă:

* Tara
* Oras
* Strada
* Numarul strazii
* Cod postal
* Informatii despre recipient
* Pret`



Adresele de ridicare pot fi alese dintr-un meniu de dropdown, unde meniul arata fie cateva detalii despre adresa (cod postal, oras) sau tag-ul (de ex. Home). Dupa ce se apasa pe butonul de „Save Changes”, o comanda este salvata. Aceasta se poate vedea si pe google maps apasand SVG-ul din coloana de „Action”:

Map

Description automatically generated

## AWB Tracking

Optiunea de AWB Tracking este valabila atat pentru un user inregistrat cat si pentru un user fara un cont. In cardul corespunzator, utilizatorul poate introduce awb-ul comenzii, care l-a vazut cand a creat comanda pentru a afla statusul comenzii.

Graphical user interface, application

Description automatically generated

# Concluzii

In concluzie, aceasta aplicatie web a fost dezvoltata cu scopul de a adauga in repertoriul aplicatiilor de curierat, o aplicatie dinamica, care poate fi utilizata de mai multe tipuri de firme de transport si/sau curierat. Folosind aceasta aplicatie, angajatilor le va fi mult mai usor sa gestioneze toate flow-urile de care sunt responsabili, de la atribuirea de rute, pana la livrarea coletelor.

## Îmbunătățiri si dezvoltare

Pentru directiile de dezvoltare, se poate introduce o aplicatie mobile, care poate fi folosita de soferi pentru a avea un tracking la comenzi mult mai usor. Aceasta ar putea lega rutele direct la aplicatia Google Maps sau la Waze, pentru a vedea in timp real locatia soferului si locatia de ridicare/livrare (la fel ca un sofer Uber).

# Bibliografie

Azure documentation (2022), accesibil la: <https://docs.microsoft.com/en-us/azure/?product=popular>

C. Sbora (2021), *platforme de laborator PAW*

Google Maps documentation (2022), accesibil la: <https://developers.google.com/maps/documentation>

S. McConnell (2004), Code Complete, Second Edition

R. C. Martin, M. Martin (2006), Agile Principles, Patterns, and Practices in C#

jQuery (2022), accesibil la: <https://jquery.com/>

Wikipedia (2022), *accesibil la:* <https://ro.wikipedia.org/>

# Site-ul web al proiectului

<https://licenta20220601113900.azurewebsites.net/>