|  |  |  |
| --- | --- | --- |
|  | 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 [*PRENUMELE ȘI NUMELE CANDIDATULUI*], student la specializarea [*DENUMIREA OFICIALĂ A SPECIALIZĂRII*] 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 [*TITLUL LUCRĂRII*],
* coordonată de [*TITLUL ȘTIINȚIFIC, PRENUMELE ȘI NUMELE COORDONATORULUI*],
* prezentată în sesiunea [*LUNA ȘI ANUL SESIUNII DE LICENȚĂ*].

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,

|  |  |  |
| --- | --- | --- |
|  | 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: | 01.12.2011 |
| Termenul estimat de predare a proiectului: | 01.06.2012 |
| 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 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.

**PROLOG**

**CUPRINSUL**

[1 Introducere 1](#_Toc309895962)

[1.1 Scopul 1](#_Toc309895963)

[1.2 Motivația 1](#_Toc309895964)

1.3 Definitii si acronime

1.4 Descriere generala

1.5 Rezumat al lucrarii pe capitole (despre ce e fiecare capitol)

[2 Aspecte teoretice si considerente tehnice 2](#_Toc309895965)

[2.1 Sisteme existente 2](#_Toc309895966)

[2.2 Avantaje si Dezavantaje 2](#_Toc309895967)

[2.3 Cerinte pentru sistemul implementat 3](#_Cerinte_pentru_sistemul)

[2.4 Abordare tehnica 3](#_Toc309895969)

[2.5 -------- 4](#_Toc309895970)

[2.6 ----- 4](#_Toc309895971)

[2.6.1 ----- 4](#_Toc309895972)

[2.6.2 ---- 4](#_Toc309895973)

[3 Proiectarea si implementarea sistemului 7](#_Toc309895975)

[3.1 arhitectura solutiei 7](#_Toc309895976)

[3.2 Proiectarea componentelor din arhitectura 7](#_Toc309895977)

3.3 Implementarea sistemului (componente din cod mai complexe)

3.4 Prezentarea aplicatiei (ss uri)

[4 Concluzii 8](#_Toc309895978)

[5 Bibliografie 9](#_Toc309895979)

[6 Referințe web 10](#_Toc309895980)

[A. ------ 11](#_Toc309895981)

[B. ------ 12](#_Toc309895982)

[C. CD / DVD 13](#_Toc309895983)

[Index 14](#_Toc309895984)

1 Oct Cerinte finalizate

1 Noi Proiectarea + Ce si Cum

15 Dec Varianta initiala a aplicatiei

**LISTA FIGURILOR**

[Figura 1. Selectarea prin click dreapta a opțiunii „Update field” 5](#_Toc309893908)

[Figura 2. Actualizarea întregului tabel 5](#_Toc309893909)

**LISTA TABELELOR**

[Tabelul 1. Nume de utilizatori și valorile rezumat ale parolelor acestora 5](#_Toc309893145)

# 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 3.1.6. Pentru autentificare și autorizare a fost integrat și folosit serviciul Identity versiunea 2.2.0. Aplicația a fost dezvoltată folosind IDE-ul Visual Studio 2019 si 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, pe parcursul dezvoltării, a fost folosit sistemul de control al versiunilor numit GIT, iar pentru găzduirea codului sursă a fost folosit serviciul Github.

De asemenea, 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 2 (folosit pentru a structura tabelele într-o manieră mai ușor de înțeles) și utilizarea 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, ORM-ul (Object–Relational Mapper) folosit este Entity Framework Core. ORM-ul le permite programatorilor să scrie query-uri și să mapeze date din baza de date folosind paradigma programării orientate pe obiecte, dar folosește și tehnica numită data binding, care protejează aplicația de atacurile de tipul SQLi (SQL injection).

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

În ceea ce privește serviciul Identity, acesta este un serviciu de autentificare și autorizare, fiind configurat să folosească o bază de date SQL Server pentru a stoca datele utilizatorilor cum ar fi parolele, email-urile, dar și alte date personale. Funcționalitățile de login, register, și editare a contului de utilizator au fost inițial create de serviciul Identity, fiind apoi customizate. De asemenea, serviciul Identity oferă și funcționalitatea de autorizare astfel încât utilizatorii vor putea accesa numai paginile destinate rolului pe care îl au, deci studenții nu vor avea acces la funcționalitățile profesorilor, etc.

În plus, 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 oferă și posibilitatea de a adăuga validări pentru parolele inserate. Aceste opțiuni fiind inserate în clasa de Startup.cs. Condițiile folosite în cadrul sistemului SPA sunt: parola trebuie să aibă minim 8 caractere, să conțină minim o cifră, o majusculă și un caracter alfanumeric. În plus, prin definirea opțiunilor permise pentru Identity, a fost activată și opțiunea folosită pentru a verifica dacă email-ul folosit există deja în baza de date, deci email-ul trebuie să fie unic.

Pe lângă aceste tehnologii, au fost folosite și pachete NuGet, acestea fiind instalate prin intermediul managerului de pachete NuGet, acesta permițându-le utilizatorilor să folosească și să creeze librării .Net sub formă 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.

De asemenea, au fost folosite și plugin-uri pentru jQuery, dar și toolkit-ul Bootstrap. jQuery este o librărie de JavaScript care este caracterizată printr-o dimensiune mică, prin rapiditate și este folosită pentru procesarea evenimentelor, parsarea și manipularea elementelor dintr-un document HTML (HyperText Markup Language), dar și pentru cereri de tip AJAX. În plus, jQuery a fost dezvoltat cu scopul de a fi compatibil tuturor tipurilor de browsere. De asemenea, JavaScript este un limbaj de programare folosit mai ales pentru introducerea unor funcționalități în paginile web, iar codul JavaScript din aceste pagini este rulat de către browser.

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.

Un prim plugin folosit este DataTables, care a fost conceput pentru a modifica tabelele HTML cu scopul de a fi interactive. De asemenea, prin folosirea 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 rândurile în ordine crescătoare sau descrescătoare în funcție de o anumită coloană pe care o poate alege utilizatorul, împărțirea rezultatelor pe mai multe pagini, posibilitatea de a alege câte rezultate sunt afișate în tabel, filtrarea / căutarea anumitor rezultate pe baza unui input dat de utilizator și afișarea numărului total de rezultate.

De asemenea, pe pagina de acasă 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.

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” (Skwiers-Koballa, Sharkey & Leavitt, 2021).

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)

## 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.

* un 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.
* resource group: include toate resursele folosite si descrise mai jos, un grup de resurse este asignat unui abonament (subscription).
* app service plan: defineste un set de resurse care sunt asignate catre aplicatia respectiva, care se platesc, pentru ca aceasta sa ruleze pe cloud.
* 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.
* sql database: baza de date folosita.
* Dashboard:

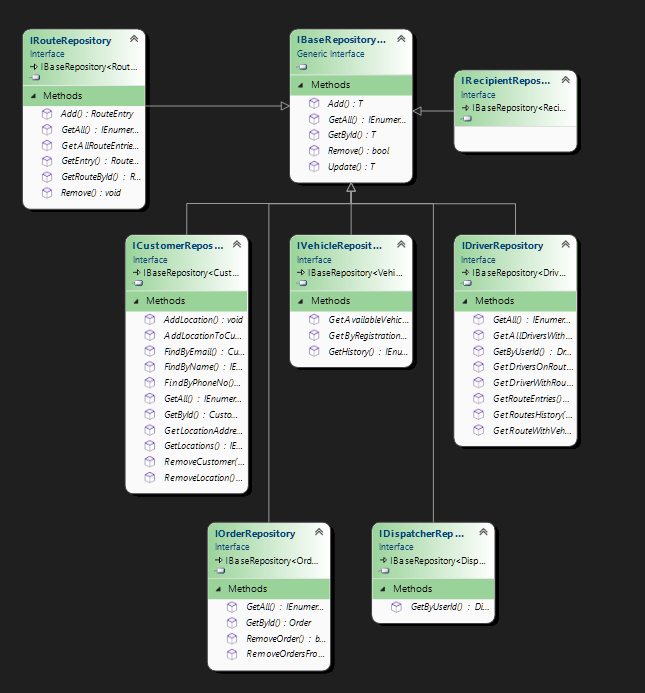
## Structura bazei de date

Baza de date este una relațională și a fost creată folosind strategia CodeFirst. Mai exact, pentru crearea bazei de date, au fost create modelele, iar fiecare model reprezintă o mapare 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.

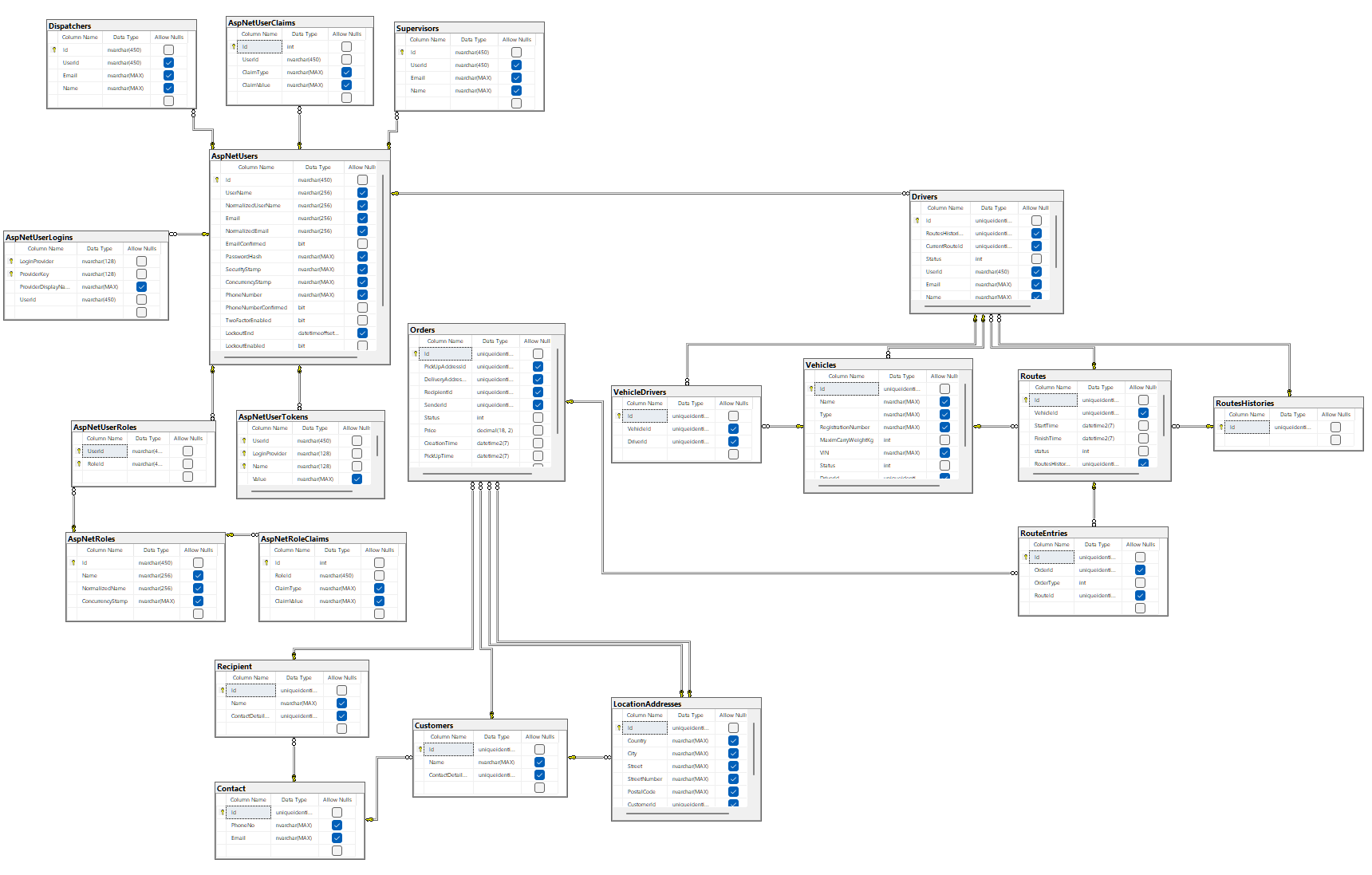
Mai jos au fost adăugate diagramele de clase care sunt împărțite în funcție de principalele funcționalități. Deoarece fiecare model avea nevoie de un Id, sub forma de Guid, pentru a putea fi găsit in baza de date, toate modele moștenesc o clasa comuna si anume DataEntity:







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



# 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 nici-o 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

## Funcționalitățile clientului final

## Elemente de tehnoredactare

Pe scurt:

* dimensiunea paginilor va fi A4, 21 x 29,7 cm;
* marginile recomandate sunt: sus/jos/stânga/dreapta – 2,54 cm;
* fontul recomandat este Times New Roman;
* corpul literelor va avea dimensiunea de 11 puncte;
* spațiul dintre rânduri va avea dimensiunea de 1 rând și jumătate (1,5);
* indentarea unui paragraf se va face cu 1,27 cm;
* textul paragrafelor trebuie să fie aliniat în mod echilibrat stânga-dreapta (în eng. - *justify*);
* paginile trebuie să fie numerotate conform acestui șablon.

## Formulele matematice

Pentru redactarea formulelor matematice recomandăm utilizarea instrumentului Microsoft Equation Editor și importul lor (o data terminate) în Microsoft Word.

## Ilustrațiile

### Figurile

Partea grafică a lucrării are o pondere semnificativă în nota finală acordată lucrării candidatului. Se recomandă prin urmare acordarea unei atenții sporite tehnoredactării figurilor.

Actualizarea Listei Figurilor este obligatorie (procedura este similară cu cea exemplificată în secțiunea 2.5.2).

### Tabelele

Exemplificăm aici utilizarea tabelor. Pentru fiecare tabelă adăugată lucrării, autorul trebuie să prevadă și adăugarea unei legende (în eng., *Caption*). La final, este recomandată actualizarea listei figurilor. Pașii necesari pentru actualizarea listei tabelelor sunt:

1. Click dreapta pe Lista Tabelelor. Va apărea un meniu similar cu cel din Figura 1.

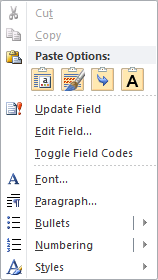


Figura 1. Selectarea prin click dreapta a opțiunii „Update field”

1. Selectarea opțiunii „Update entire table” conform cu Figura 2.

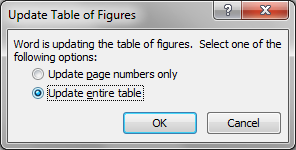


Figura 2. Actualizarea întregului tabel

1. Verificarea fontului folosit pentru conținutul propriu-zis al Listei Tabelelor și alegerea fontului Times New Roman în caz de incongruență.

|  |  |  |
| --- | --- | --- |
| **Index** | **Nume utilizator** | **Valoarea rezumat a parolei (folosind SHA1)** |
| 1 | dpopescu | 8fb9e6763269ae7cba85f02668c3c32041bf00ed |
| 2 | eganea | 3b42ba8a586dd1589e949c28c9cf2810f7d65bb4 |
| 3 | mmarian | bfc01d16d1944f3b6caba515556713f4aeeb2d0b |

Tabelul 1. Nume de utilizatori și valorile rezumat ale parolelor acestora

În interiorul lucrării, tabelul poate fi citat folosind eticheta și numărul de ordine ale sale precum în exemplul acesta (vezi Tabelul 1).

### Legenda (unei figuri/tabele)

În cele doua secțiuni de mai sus s-au demonstrat două modele de legende atașate fiecărei tabele sau figuri. Microsoft Word permite modificarea, respectiv adăugarea de etichete noi corespunzătoare unui anumit tip de legendă.

# Termeni de utilizare

## Autorii

Acest șablon de document a fost creat de colectivul Facultății de Automatică, Calculatoare și Electronică pentru uniformizarea structurii proiectelor de diplomă ale studenților săi.

## Licența de utilizare

Nu există restricții de utilizare. Documentul nu este constrâns de nicio licență.

# Concluzii

Autorul prezintă concluziile sale…

# Bibliografie

Bibliografia va fi ordonată alfabetic dupa eticheta fiecărei element (de ex. DOOM05 în lista de mai jos este o etichetă). Etichetele materialelor consultate vor fi formatate folosind:

* primele litere ale primului autor urmate de cele două cifre semnificative ale anului apariției materialului, sau
* dintr-un acronim popular al lucrării respective, urmat din nou de cele două cifre semnificative ale anului apariției.

[DOOM05] – *Dicţionarul ortografic, ortoepic şi morfologic al limbii române*, Editura Univers Enciclopedic, Bucureşti, 2005

# Referințe web

Recomandăm și aici respectarea regulilor enunțate pentru secțiunea 5.

[Alm08] – Pedro de Almeida, Patrik Fuhrer, Documentation Guidelines for Diploma and Master Thesis, Universitatea din Fribourg, Elveția, 2008, disponibil on-line la adresa http://diuf.unifr.ch/drupal/softeng/teaching/guidelines

[Olt07] – Th. Olteanu, C. Albu, *Ghid pentru redactarea lucrării de diplomă sau a disertaţiei de masterat*, Universitatea Română de Arte și Științe „Gheorghe Cristea”, 2007, disponibil via web la adresa http://www.ugc.ro/tpl/GHID REDACTARE DIPLOMA LICENTA.pdf

# Codul sursă

În această anexă se adaugă codul sursă al aplicației…

# Site-ul web al proiectului

Autorul prezintă în această anexă (opțională) site-ul web asociat proiectului său.

# CD / DVD

Autorul atașează în această anexă obligatorie, versiunea electronică a aplicației, a acestei lucrări, precum și prezentarea finală a tezei.



# Index

B

Bibliografie 9

C

CUPRINSUL xi

D

Dimensiuni 3

F

Figuri 4

Formulele matematice 4

I

Ilustrațiile 4

L

Legenda 6

LISTA FIGURILOR xii

LISTA TABELELOR xiii

R

Referințe web 10

S

Structura documentului 2

T

Tabele 5