|  |  |  |
| --- | --- | --- |
|  | UNIVERSITATEA DIN CRAIOVA  FACULTATEA DE AUTOMATICĂ, CALCULATOARE ȘI ELECTRONICĂ  DEPARTAMENTUL DE CALCULATOARE ȘI TEHNOLOGIA INFORMAȚIEI |  |

PROIECT DE DIPLOMĂ

*Marcu Andrei Cristian*

COORDONATOR ȘTIINȚIFIC

*Asist. drd. ing. Petcușin Felix Alin*

*Iulie 2021*

CRAIOVA

|  |  |  |
| --- | --- | --- |
|  | UNIVERSITATEA DIN CRAIOVA  FACULTATEA DE AUTOMATICĂ, CALCULATOARE ȘI ELECTRONICĂ  DEPARTAMENTUL DE CALCULATOARE ȘI TEHNOLOGIA INFORMAȚIEI |  |

*Application for Human Resources Management in a Company*

*Marcu Andrei Cristian*

COORDONATOR ȘTIINȚIFIC

*Asist. drd. ing. Petcușin Felix Alin*

*Iulie 2021*

CRAIOVA

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

Proverb popular

**DECLARAȚIE DE ORIGINALITATE**

Subsemnatul Marcu Andrei Cristian, student la specializarea Calculatoare cu predare in limba engleză 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 Application for Human Resources Management in a Company**,**
* coordonată de Asist. drd. ing. Inginer Petcușin Felix Alin,
* prezentată în sesiunea Iulie 2021.

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,

20.06.2021

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

**PROIECTUL DE DIPLOMĂ**

|  |  |
| --- | --- |
| Numele și prenumele studentului/-ei: | Marcu Andrei Cristian |
| Enunțul temei: | Application for Human Resources Management in a Company  Description: Aplicatia ofera o solutie pentru gestionarea angajatiilor dintr-o companie. Oferta statistici si o vedere mai detaliata asupra companiei. |
| Datele de pornire: | ASP.NET Core, React, React Material UI, EF Core, Devexpress React, SQL Server |
| Conținutul proiectului: | [*Descrierea succintă a conținutului fiecărui capitol al lucrării*] |
| Material grafic obligatoriu: | Schema aplicatiei, diagrame UML, screenshot-uri din aplicatie |
| Consultații: | Periodice |
| Conducătorul științific  (titlul, nume și prenume, semnătura): | Asist. drd. ing. Petcușin Felix Alin |
| Data eliberării temei: | 27.11.2020 |
| Termenul estimat de predare a proiectului: | 24.06.2021 |
| Data predării proiectului de către student și semnătura acestuia: |  |

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

**REFERATUL CONDUCĂTORULUI ȘTIINȚIFIC**

|  |  |
| --- | --- |
| Numele și prenumele candidatului/-ei: |  |
| Specializarea: | Calculatoare cu predare în limba engleză |
| Titlul proiectului: | Application for Human Resources Management in a Company |
| 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,

**PROJECT SUMMARY**

The Application for Human Resources Management in a Company is an application designed to simplify the work of the organization on an IT company. Shortly called “*HRDesk*”, the application handles the holiday requests of the employees, offers an overview of all the holidays, has the possibility to book meeting rooms and future plan the organization of the internal calls. The application also generates reports for different types of dates like employees, holiday requests, leave requests, upcoming meetings.

The reason I’ve chose this topic it’s because the application is useful in real world situations and offers multiple ways of expansion. From the point where the application is right now it can be extended to also support polls, salary reports, virtual office management, food menus, and multiple internal company necessities.

The following technologies were used in the development of the website:

Frontend:

* [React](https://reactjs.org/)
* [React Redux](https://react-redux.js.org/)
* [Material UI](https://material-ui.com/)
* [DevExtreme Reactive](https://devexpress.github.io/devextreme-reactive/docs/)

Backend:

* [ASP.NET Core](https://docs.microsoft.com/en-us/aspnet/core/?view=aspnetcore-5.0)
* [Entity Framework Core](https://docs.microsoft.com/en-us/ef/core/)

Database:

* [Sql Server](https://www.microsoft.com/en-us/sql-server/sql-server-downloads)

Version control:

* [Github](https://github.com/)
* [Sourcetree](https://www.sourcetreeapp.com/)

The application is divided on multiple levels, and each user needs permission in order to access them.

1. Dashboard – Contains general data about the company
2. Leave requests – Possibility to add a leave request(a leave request is a unexpected leave from the work, like going to a doctor, which can be maximum a few hours)
3. Dayoff requests – Possibility to add a holiday request
4. Reports – Charts about the company
5. Meetings – Upcoming meetings of the logged-in user
6. Team – Company employees overview
7. Holiday Calendar – Holiday tracker for the user and it’s company(can see his holidays, teams holidays and free national days)
8. Book room – Possibility to create a Meeting which will be assigned to a Team\
9. Manage employees – Add, Edit and Delete employees
10. Manage holidays – Approve or Decline Leave requests, Holidays and create National Days
11. Manage organization – Management data like offices, meeting rooms, teams and functions.

***Termenii cheie***: website, React, Redux, SQL, ASP.NET Core, Entity Framework, Human resource management, HRDesk, Devextreme Reactive, Material UI

**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 introduction 1](#_Toc75555941)

[1.1 The objective 1](#_Toc75555942)

[1.2 The motivation 2](#_Toc75555943)

[1.3 Document structure 2](#_Toc75555944)

[2 TECHNOLOGIES 3](#_Toc75555945)

[2.1 ASP.NET Core 3](#_Toc75555946)

[2.1.1 What is ASP.NET Core? 3](#_Toc75555947)

[2.1.2 NuGet packages 3](#_Toc75555948)

[2.1.3 Why should you use ASP.NET Core? 3](#_Toc75555949)

[2.1.4 ASP.NET Core vs ASP.NET 4](#_Toc75555950)

[2.2 Entity Framework Core 4](#_Toc75555951)

[2.2.1 What is Entity Framework Core? 4](#_Toc75555952)

[2.2.2 LINQ 4](#_Toc75555953)

[2.2.3 Code first approach 5](#_Toc75555954)

[2.3 React 7](#_Toc75555955)

[2.3.1 What is React? 7](#_Toc75555956)

[2.3.2 JSX 7](#_Toc75555957)

[2.3.3 React Redux 8](#_Toc75555958)

[2.3.4 Axios 9](#_Toc75555959)

[2.3.5 React Material UI 10](#_Toc75555960)

[2.3.6 Devexpress Scheduler 10](#_Toc75555961)

[2.4 SQL Server 10](#_Toc75555962)

[2.5 Github 10](#_Toc75555963)

[2.6 Sourcetree 11](#_Toc75555964)

[3 PROJECT STRUCTURE 12](#_Toc75555965)

[3.1 HRDesk backend solution 12](#_Toc75555966)

[3.2 HRDesk frontend solution 12](#_Toc75555967)

[3.3 Repository pattern 13](#_Toc75555968)

[3.4 Unit of Work 14](#_Toc75555969)

[3.5 Exception middleware 14](#_Toc75555970)

[3.6 Dependency Injection 15](#_Toc75555971)

[4 Database structure 16](#_Toc75555972)

[4.1 Database schema 16](#_Toc75555973)

[4.2 Tables 16](#_Toc75555974)

[4.2.1 Users 16](#_Toc75555975)

[4.2.2 Teams 16](#_Toc75555976)

[4.2.3 Functions 17](#_Toc75555977)

[4.2.4 Offices 17](#_Toc75555978)

[4.2.5 Leave requests 17](#_Toc75555979)

[4.2.6 Daysoff 17](#_Toc75555980)

[4.2.7 Company Details 17](#_Toc75555981)

[4.2.8 Personal Details 17](#_Toc75555982)

[4.2.9 Meeting Rooms 18](#_Toc75555983)

[4.2.10 Meetings 18](#_Toc75555984)

[4.2.11 National days 18](#_Toc75555985)

[4.2.12 Permissions 18](#_Toc75555986)

[4.2.13 Polls, PollAnswers, UserPollAnswers 18](#_Toc75555987)

[4.2.14 UserPermissions 18](#_Toc75555988)

[5 HRDesk modules 19](#_Toc75555989)

[5.1 Dashboard module 19](#_Toc75555990)

[5.2 Leave request module 19](#_Toc75555991)

[5.3 Dayoff request module 20](#_Toc75555992)

[5.4 Reports module 21](#_Toc75555993)

[5.5 Meetings module 22](#_Toc75555994)

[5.6 Team module 23](#_Toc75555995)

[5.7 Holiday Calendar module 24](#_Toc75555996)

[5.8 Booking module 25](#_Toc75555997)

[5.9 Manage employees module 27](#_Toc75555998)

[5.10 Manage holidays module 29](#_Toc75555999)

[5.11 Manage organization 31](#_Toc75556000)

[5.12 Unauthorized access 32](#_Toc75556001)

[5.13 Login page 33](#_Toc75556002)

[6 uml diagrams 34](#_Toc75556003)

[6.1 Class diagram 34](#_Toc75556004)

[6.2 Use case diagram – Basic user 35](#_Toc75556005)

[6.3 Use case diagram – Administrator 36](#_Toc75556006)

[6.4 Activity diagram – day off 37](#_Toc75556007)

[7 hOw to use 38](#_Toc75556008)

[7.1 Fetch or download the code 38](#_Toc75556009)

[7.2 Link and create the database 38](#_Toc75556010)

[7.3 Frontend solution setup 39](#_Toc75556011)

[8 Termeni de utilizare 41](#_Toc75556012)

[8.1 Autorii 41](#_Toc75556013)

[8.2 Licența de utilizare 41](#_Toc75556014)

[9 Concluzii 42](#_Toc75556015)

[10 Bibliografie 43](#_Toc75556016)

[11 Referințe web 44](#_Toc75556017)

[A. Codul sursă 45](#_Toc75556018)

[B. Site-ul web al proiectului 46](#_Toc75556019)

[C. CD / DVD 47](#_Toc75556020)

[Index 48](#_Toc75556021)

**LISTA FIGURILOR**

[Figure 1. ASP.NET Core logo [1] 3](#_Toc75563071)

[Figure 2. Linq query syntax 5](#_Toc75563072)

[Figure 3. Linq extension methods 5](#_Toc75563073)

[Figure 4. Code first approach[6] 5](#_Toc75563074)

[Figure 5. HRDesk User entity example 6](#_Toc75563075)

[Figure 6 - React logo[8] 7](#_Toc75563076)

[Figure 7. JSX Code[9] 7](#_Toc75563077)

[Figure 8. How JSX really looks behind after being parsed by Babel[9] 8](#_Toc75563078)

[Figure 9. React Redux logo[10] 8](#_Toc75563079)

[Figure 10. Axios wrapper from HRDesk 9](#_Toc75563080)

[Figure 11. GitHub logo[13] 10](#_Toc75563081)

[Figure 12. HRDesk sourcetree history 11](#_Toc75563082)

[Figure 13. HRDesk solution 12](#_Toc75563083)

[Figure 14. HRDesk frontend solution 13](#_Toc75563084)

[Figure 15. Unit of work pattern flow 14](#_Toc75563085)

[Figure 16. HRDesk database diagram 16](#_Toc75563086)

[Figure 17. Dashboard module 19](#_Toc75563087)

[Figure 18. Leave request module 20](file:///E:\Licenta-Human-resources-management\Documentation\Licenta%20-%20MarcuAndreiCristian.docx#_Toc75563088)

[Figure 19. Dayoff module 21](file:///E:\Licenta-Human-resources-management\Documentation\Licenta%20-%20MarcuAndreiCristian.docx#_Toc75563089)

[Figure 20. Reports module 22](file:///E:\Licenta-Human-resources-management\Documentation\Licenta%20-%20MarcuAndreiCristian.docx#_Toc75563090)

[Figure 21. Meetings module 22](file:///E:\Licenta-Human-resources-management\Documentation\Licenta%20-%20MarcuAndreiCristian.docx#_Toc75563091)

[Figure 22. Team module 23](file:///E:\Licenta-Human-resources-management\Documentation\Licenta%20-%20MarcuAndreiCristian.docx#_Toc75563092)

[Figure 23. Team module - Employee details 24](file:///E:\Licenta-Human-resources-management\Documentation\Licenta%20-%20MarcuAndreiCristian.docx#_Toc75563093)

[Figure 24. Holiday calendar module 25](file:///E:\Licenta-Human-resources-management\Documentation\Licenta%20-%20MarcuAndreiCristian.docx#_Toc75563094)

[Figure 25. Booking module 26](#_Toc75563095)

[Figure 26. Booking module menu 26](file:///E:\Licenta-Human-resources-management\Documentation\Licenta%20-%20MarcuAndreiCristian.docx#_Toc75563096)

[Figure 27. Manage employees module 27](file:///E:\Licenta-Human-resources-management\Documentation\Licenta%20-%20MarcuAndreiCristian.docx#_Toc75563097)

[Figure 28. Manage employees module step 1 27](file:///E:\Licenta-Human-resources-management\Documentation\Licenta%20-%20MarcuAndreiCristian.docx#_Toc75563098)

[Figure 29. Manage employees module step 2 28](file:///E:\Licenta-Human-resources-management\Documentation\Licenta%20-%20MarcuAndreiCristian.docx#_Toc75563099)

[Figure 30. Manage employees module step 3 28](file:///E:\Licenta-Human-resources-management\Documentation\Licenta%20-%20MarcuAndreiCristian.docx#_Toc75563100)

[Figure 31. Manage employees module step 4 29](file:///E:\Licenta-Human-resources-management\Documentation\Licenta%20-%20MarcuAndreiCristian.docx#_Toc75563101)

[Figure 32. Manage holidays module leave requests 30](file:///E:\Licenta-Human-resources-management\Documentation\Licenta%20-%20MarcuAndreiCristian.docx#_Toc75563102)

[Figure 33. Manage holidays module national days 31](file:///E:\Licenta-Human-resources-management\Documentation\Licenta%20-%20MarcuAndreiCristian.docx#_Toc75563103)

[Figure 34. Manage organization module offices and meeting rooms 32](file:///E:\Licenta-Human-resources-management\Documentation\Licenta%20-%20MarcuAndreiCristian.docx#_Toc75563104)

[Figure 35. Access denied 32](file:///E:\Licenta-Human-resources-management\Documentation\Licenta%20-%20MarcuAndreiCristian.docx#_Toc75563105)

[Figure 36. Login page 33](file:///E:\Licenta-Human-resources-management\Documentation\Licenta%20-%20MarcuAndreiCristian.docx#_Toc75563106)

[Figure 37. Employees excel report from HRDesk 33](file:///E:\Licenta-Human-resources-management\Documentation\Licenta%20-%20MarcuAndreiCristian.docx#_Toc75563107)

[Figure 38. Leave request excel report from HRDesk 34](#_Toc75563108)

[Figure 39. National days excel report from HRDesk 34](#_Toc75563109)

[Figure 40. UML Class diagram 35](#_Toc75563110)

[Figure 41. UML Use case diagram – User 36](#_Toc75563111)

[Figure 42. UML Use case diagram – Administrator 37](#_Toc75563112)

[Figure 43. UML Activity diagram - day off 38](#_Toc75563113)

[Figure 44. Connection string 39](#_Toc75563114)

[Figure 45. Apply migration 40](#_Toc75563115)

[Figure 46. Default user 40](#_Toc75563116)

**LISTA TABELELOR**

[Table 1 - ASP.NET Core vs ASP.NET 4](#_Toc75196245)

# introduction

## The objective

The objective of the Human resource management application in a company called „HRDesk” from now on is to come in help of the companies, granting an alternative of employees management. The application offers different statistics, possibility to export data in Excels and it is designed to be used however the company wants. This was achieved by avoiding creating predefined roles(which would’ve got access to a certain module) and instead when an user is created it will be assigned to multiple modules.(For each menu item in the left drawer there is a module). This way the users can be splitted on responsabilities(for example there can be a user which accepts/declines the holiday requests of the employees). This user can also be a developer.

Available modules at the time of the presentation:

1. Dashboard – Contains general data about the company and the upcoming meetings of the user
2. Leave requests – Employee/Administrative level – the user can add or delete a leave request. A leave request is an urgent leave, for example going to a doctor
3. Dayoff requests – Employee level – the user can add or delete a day off. Also in here he can keep the track of his number of free days. When we calculate the number of free days we check the overlap with weekends, free national days and another holidays.
4. Reports – Employee/Administrative level – Different charts about the company(Age average chart, Country of provenience chart, Functions chart)
5. Meetings – Employee level – Upcoming meetings between two dates
6. Team – Employee level – Company employees ordered by employment date
7. Holiday Calendar – Employee level – Can see his own holidays, team members holidays and also national days
8. Book room – Administrative level – Reserve a meeting room for a team. This meeting will be displayed on the Meetings module.
9. Manage employees – Administrative level – A more detailed overview of the employees. Can add, delete, update them. In here we can assign the user to a module and also we can generate a report(Excel) of all the employees
10. Manage holidays – Administrative level – This module handles the leave and day off requests created before by the employee. The administration can either accept and decline them. Also on this level we have the national days, which are considered as day offs for all employees.
11. Manage organization – Administrative level – Handles the location management. For example offices, meeting rooms(used for booking), teams and functions.

## The motivation

The main reason for choosing this specific topic is because this is a real world usable application that can be extended with many functionalities. Also I am working in a company of this kind and I’ve seen the benefits an internal application can provide. Keeping track of employees data, like number of free days, salary, and multiple details can simplify the work of the management team of a company.

Also another reason is that the requirement of the topic was to be a Web application, the domain I’m interested in evolving the most. This way I was able to use all the knowledge acquired during the university and create an usable and opened for extension application.

## Document structure

# TECHNOLOGIES

## ASP.NET Core

### What is ASP.NET Core?



Figure 1. ASP.NET Core logo [1]

ASP.NET Core is the successor of the .NET Framework developed by Microsoft. It is a cross-platform framework targeted to run on the .NET Core platform. It’s free, open-source and supports the development of multiple types of applications(Web, Cloud, Mobile etc.).[2]

ASP.NET Core features can be extended by the use of NuGet packages.

### NuGet packages

A NuGet package is an essential tool for any modern development platform. It’s a mechanism through which developers can create, share and consume useful code. Put simply, a NuGet package is a single ZIP file that contains compiled code. This packages can be private(used by a restrained number of users) or can be public. One example of NuGet package is Entity Framework which will be detailed later on.[3]

### Why should you use ASP.NET Core?

* Cross-platform support
* Good performance
* Expandable using NuGet packages
* Cloud support
* Open source
* Can be used for many types of applications(Web, Mobile, Cloud etc)
* Created by Microsoft
* Updated constantly

### ASP.NET Core vs ASP.NET

|  |  |
| --- | --- |
| ASP.NET Core | ASP.NET |
| Supports Windows, Linux and macOS | Supports Windows |
| Bigger CPU usage and performance | Lower CPU usage and performance |
| Can be hosted on multiple platforms | Can be hosted on IIS(Internet Information Server) |
| Supports Docker | Doesn’t support Docker |
| It is focused on Web applications development | It is focused on both Desktop and Web applications |

Table 1. ASP.NET Core vs ASP.NET [4]

## Entity Framework Core

### What is Entity Framework Core?

Entity Framework Core is an open source and closs platform object-relational mapper. It was initially part of .NET Framework and starting with Entity Framework 6 was delivered separately. This can be acquired from the NuGet packages. [5]

In my application Entity Framework Core was used in order to simplify the relation with the database. Writing LINQ code directly in the main project, without the need of SQL stored procedures makes the code more readable. In my case the database was created using a code first approach with migrations

### LINQ

Language-Integrated Query(LINQ) is the technology that allow the user to do queries directly from C# code.

Linq syntax is of two types:

* Query syntax. Similar two classic SQL syntax.

Graphical user interface, text

Description automatically generated

Figure 2. Linq query syntax

* Extension method syntax

Text

Description automatically generated

Figure 3. Linq extension methods

### Code first approach

In entity framework code first approach the focus is on creating entities from C# code. Based on the Context of the application Entity Framework will create the corresponding tables via migrations. Using Data Annotations on Entity fields you can add limitations like in SQL(for example maximum length of string).

Diagram

Description automatically generated

Figure 4. Code first approach[6]

In HRDesk I’ve used the code first approach. The following picture displays an Entity example from the application. In this case the Entity also inherits a “Base Entity” which is DeleteEntity. This entity automatically adds some extra fields, which are: IsDeleted(used for soft delete), Id(unique identifier), Creation date, Updated date(if it is updated by anyone).

Text

Description automatically generated

Figure 5. HRDesk User entity example

## React

### What is React?

Icon

Description automatically generated

Figure 6 - React logo[8]

React is an open-source front-end JavaScript library created by Facebook. The code is structured on UI Components. The main reason of the component is that the code is modular and one component can be used by multiple other components. This way the user is forced to create reusable code. React is mainly focused on state management and displaying the data in the DOM. There are additional libraries that extend it’s functionalities. For example react-router-dom for routing, axios for api calls, material-ui for CSS components.[7]

### JSX

“JSX is an XML/HTML-like syntax used by React so that XML/HTML-like text can co-exist with JavaScript/React code. The syntax is intended to be used by preprocessors (i.e., Babel) to transform HTML-like text found in JavaScript files into standard JavaScript objects that a JavaScript engine will parse.” [9]

Basically, you write HTML-like code which is then parsed by Babel or another processor into Javascript code. The differences between normal HTML and JSX are not noticeable.

Text

Description automatically generated

Figure 7. JSX Code[9]

Text

Description automatically generated

Figure 8. How JSX really looks behind after being parsed by Babel[9]

### React Redux

Icon

Description automatically generated

Figure 9. React Redux logo[10]

In React the state it’s placed inside a component and it’s available only there. This state can be passed to the Parent Component or the Child component and this causes a limitation. React Redux provides an alternative way to create the state. React Redux creates a global state named store.

Redux is composed of the following components:

* Actions: It’s a JavaScript object that contains a type field. The type field describes the action. For example: “getUser”.
* Reducer: It’s a function that updates the state(if needed) based on the action
* Dispatch: Receives an action as parameter and calls the reducer function of that specific function.
* Store: The actual global state. This can be accessed all over the application using Redux connect.
* Selector: Extracts a field from the state

### Axios

Axios is a HTTP client from Node.js. I’ve created a wrapper using this package for handling api requests. The wrapper automatically creates the request body, attaches the API base url and the authorization token if it’s available

Text

Description automatically generated

Figure 10. Axios wrapper from HRDesk

From this point the wrapper can handle any type of requests, by simply receiving the controller url and the payload.

### React Material UI

Material UI is an open-source library that provides a multitude of UI components for React application. Material UI follows the guidelines provided by Google on Material Design. It is similar to Bootstrap, it’s also based on a 12 column grid structure. Multiple components from Material UI were used in the development of the HRDesk application. The most encountered ones in my application are: Container, Grid, Data Grid, Card, Button, Transfer List, Drawer, Paper, App bar, Avatar and Material Icons. Using the grid system from Material UI caused the application to be as mobile responsive as possible. There are indeed some limitations on data grid with many columns, which can not be resized on a mobile device and another component should be used for this case.

### Devexpress Scheduler

The Devexpress scheduler was used for the holiday and booking calendar. It offers a similar type of calendar to the one in Outlook. This calendar can be customized in any means by the user since it accepts any component to be overwritten. The scheduler supports recurrent events and for a booking system this was ideal. It’s user friendly and it’s interface is similar to the one from the Material Design.

## SQL Server

SQL Server is a relational database management developed by Microsoft. It is based on the SQL query language. SQL Server can handle database with a huge amount of data. SQL Server has it’s own SQL derivation, called T-SQL. T-SQL extends the classic SQL queries by adding additional syntax for stored procedures and transactions. It support multiple types of data, from numbers and strings to binary data.[11][12]

SQL Server is also available as a cloud storage on Microsoft Azure. The Azure SQL Database is based on the last version of SQL Server

## Github

Text

Description automatically generated with low confidence

Figure 11. GitHub logo[13]

GitHub is a version control website for software development. It offers multiple services like version control, code management, pull requests, task management, bug tracking. It’s free to use, the user can create both public and private repositories. There are some limitations for the free plan, like the number of contributors but that is not an issue for a final exam project. In 2018 GitHub was bought by Microsoft. It the largest host of source code in the world. [14]

While developing HRDesk I’ve committed after each major change on GitHub, this way whenever an issue occurred I could check the old code, find the cause of the issue easier and or course keep the code safe on the GitHub storage. For my relation with GitHub I’ve used a Git client available for windows and iOS, called Sourcetree. HRDesk git repository will be found in the how to use chapter.

## Sourcetree

Sourcetree is a free to use Git client developed by Atlassian. It is a middleware between the user and a Git. It’s intuitive and there are only a few buttons, which makes it very easy to use. Writing the same commands from git bash would be way harder. Also, in case if something is not supported by it’s buttons available in the UI, for example hard resetting a merge, or rollbacking a commit it also offers a git bash directly from the UI.

In my opinion the best thing about the Sourcetree is the history view, where you can see all the commits and branches, each branch with it’s own color and line

Graphical user interface, text, application, email

Description automatically generated

Figure 12. HRDesk sourcetree history

In my case I’ve used only one branch, so there is only one line. In this view I can see all the changes I’ve made in that commit and rollback to an exact commit if needed.

# PROJECT STRUCTURE

## HRDesk backend solution

HRDesk backend solution is found in the WebApi folder

HRDesk solution is splitted on 3 projects:

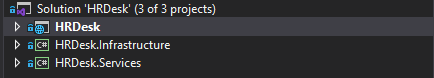


Figure 13. HRDesk solution

* HRDesk contains the controllers and the Startup. Also there is a wrapper for dependency injection, which is simply called on Startup.
* HRDesk.Infrastructure contains everything needed by the database. In this project we can find the entities, repositories, migrations, DbContext, UnitOfWork.
* HRDesk.Services contains the services of the application. All the logic needed all over the application, mappers, error handlers, exception middlewares.

## HRDesk frontend solution

HRDesk frontend solution is found in the WebApp folder

The project is composed of the following structure:

* Api : Contains the Axios wrapper for api calls
* Common: Contains the actual components. A component is composed of a container(which contains the dispatch methods from Redux), general file which contains the actual React code and styles which contains css and material ui styles
* Media: Images, or any file needed in application
* Redux: The actions, reducers and selectors needed by Redux
* Services: Different services that were needed between multiple components, and in order to not duplicate the code this was exported from an external file
* Utils: Constant values used in the application

Text

Description automatically generated

Figure 14. HRDesk frontend solution

## Repository pattern

Repositories are classes that contain the logic to access the database. Most of the code is made out of Linq code using extensions methods. I’ve also created a Base Repository which contains CRUD operations. For each entity(table in database) there is one repository. The normal repository extends the CRUD operations provided by the BaseRepository(for example GetUserByEmail, GetUserByName etc.).

The repositories contain one extra level of abstraction, created by the Unit of Work pattern.Using repository pattern will minimize duplicate logic and will decouple the services from the actual database framework. The services are independent of the framework, as long as the logic is kept in the repositories. If the database relation framework (Entity Framework in my case) needs to be changed at any time, the code will be modified only in the repositories.

## Unit of Work

An Unit of Work is a class that controls all the repositories, providing an extra layer of abstractisation between the services and the repositories. This way, the user doesn’t depend on a framework. For example if entity framework will be replaced in the future the code needs to be changed only in the repositories, leaving the services untouched. This way the application is flexible, maintainable and opened for extension.

Diagram

Description automatically generated

Figure 15. Unit of work pattern flow

## Exception middleware

The application has an automatic exception handler, whenever an exception is thrown the message is captured by the middleware and it’s sent to the frontend. In frontend the code is catched by axios and the error message is automatically displayed in a red toaster. This way the user will know whenever a request fails, and a message will indicate the cause of the issue. One example of exception is thrown when the administrator tries to create an user who’s email address is already found in the database. The actual user will receive the error message in toaster “Email already in use “.

## Dependency Injection

ASP.NET Core supports dependency injection pattern. The classes and their interfaces are registered in the startup of the application. There are three types of lifetimes:

Singleton: Wherever the dependency injection is made the same instance of class will be used for all the calls

Scoped: The instance is the same within the scope of a request, but it’s changed when another request is made.

Transient: The instance is always different

Dependency Injection decouples dependencies between services by using abstractions.

# Database structure

## Database schema

A picture containing text, indoor, screenshot

Description automatically generated

Figure 16. HRDesk database diagram

## Tables

### Users

Users is the main table of the application, we can call it the core table. This table is related with most of the other tables. It contains all the data about the employee. The details are spllited in two, personal details and company details. An user needs to be linked to a function, table and office in order to be usable in the application. The account and the password of the user also encoded and stored in this database. The type of the user entity is DeleteEntity, meaning that when an user is marked as deleted but not actually removed from the table.

### Teams

Teams table is used to store the teams available in the companies. An user needs to be assigned to a team, and a team is needed when a booking is made.

### Functions

Functions are stored in this table. An user needs to be linked to a function. The functions are used all over the application. Also there is a chart related to the functions

### Offices

An user needs to be linked to an office. The offices are used for a better organization in the company.

### Leave requests

The leave requests contain two FKs to the user table. One points to the user that made the leave request and the other one points the administrator that has to accept or decline it. The status of a leave request is defined by the following enum:

* Status = 0 -> Waiting
* Status = 1 -> Refused
* Status = 2 -> Approved

The type of this entity is DeleteEntity, meaning that a soft delete is performed instead of an actual remove.

### Daysoff

The daysoff are similar to the leave request, the main difference between these entities is that a dayoff can be made on multiple days, while a leave request is limited to hours in a day. The status enum is the same one from the leave request.

* Status = 0 -> Waiting
* Status = 1 -> Refused
* Status = 2 -> Approved

### Company Details

It’s an extension of the user table. Contains user’s company related data, like salary, employment date, number of free days.

### Personal Details

It’s an extension of the user table. Contains user’s personal data like birth date, name, country of provenience, email, telephone number.

### Meeting Rooms

Meeting rooms are similar to offices. The main use of them is by the booking calendar. For each meeting room there will be a column displayed in the calendar.

### Meetings

A meeting is related to a Team and a Meeting room. The recurrence rule is a string defined by the devexpress scheduler. This string will tell the scheduler if the meeting is recurrent, or if it’s an all day meeting. The meetings are created from the booking calendar and can be seen in both the Booking calendar, home page and meetings module.

### National days

National days are added by an administrator. A national day is similar to a dayoff but it’s not assigned to an user and doesn’t need to be approved. It’s a standalone entity that it’s reflected to all the employees in the company.

### Permissions

In the permissions table we can find the modules. There is no way to add or remove them. The data is inserted by the seed method at the start of the aplication. The permissions are linked to an user using a many to many relationship, making the user to see the actual module.

### Polls, PollAnswers, UserPollAnswers

Those tables are related.

* Polls contains the actual question of the poll and if the poll accepts multiple asnwers from an user.
* PollAnswers contains the possible answers for that poll, we can have one or many answers for a poll.
* UserPollAnswers is the many to many relationship between the answers and the user. If the poll accepts multiple answers each user will have multiple entries in this table. Also this table support the case when the poll accepts only one entry

### UserPermissions

This table is the many to many table between users and permissions. Based on this data the user gets access to modules

# HRDesk modules

## Dashboard module

Graphical user interface, website

Description automatically generatedThe dashboard is a module available to all the user in the application. It’s main purpose it’s of a home page. Contain details about the number of employees in the company and also an overview about the upcoming meetings.

Figure 17. Dashboard module

## Leave request module

As mentioned earlier in the presentation a leave request is a temporary necessity to leave the workplace, the most encountered example is going to a doctor. After the user talks with a person which can approve it’s leave request he can come to this module and create a leave request. When creating a leave request he needs to specify the interval he will be gone, and also the person which he spoke with. The leave request will be sent to the person he spoke with, and from here the administrator can accept or decline the leave request. This request doesn’t impact the number of free days. Let’s call them unexpected holidays.

Graphical user interface, application

Description automatically generated

Figure 18. Leave request module

## Dayoff request module

Similar to leave requests, but those are the actual holiday requests of the user. Compared to a leave request a day off can contain multiple days, while a leave request is limited only to hour intervals. Also, when creating a holiday request the user needs to specify the administrator he spoke with, and the administrator will have to accept or decline the dayoff.

If the dayoff is accepted then the number of free days of the user will be automatically adjusted. This number can be noticed on the chart from the top of the page. With blue is the number of free days and with red the number of used days. The number of free days used by a holiday request is calculated without the days that are in weekend, or overlap a national day, or another holiday.

Graphical user interface

Description automatically generated with medium confidence

Figure 19. Dayoff module

## Reports module

Is composed of three pie charts at the moment of the presentation. On this module there are already multiple tabs available for future use charts.

Right now there are three charts:

* One which displays the number of users between different ranges of age(ex 20-30, 30-40 etc).
* The second one displays the number of users that come from a certain country.
* And the last chart displays the number of users of each function.

Both the functions and the countries of provenience are dynamically retrieve in function of the data inserted, meaning that whenever a function or another country is added this will be automatically available in the charts.

Graphical user interface, chart, application

Description automatically generated

Figure 20. Reports module

## Meetings module

Graphical user interface, application

Description automatically generated It’s more of an informative module, the user can see all the meetings between two dates (start and end) which are assigned to it’s team. This module is only for display, seeing the meeting room where the call will take place and the description of the call. The meetings will be added from the Booking module which is on the administrative level

Figure 21. Meetings module

## Team module

Graphical user interface, application

Description automatically generatedDisplays all the employees of the company ordered by the employment date. The entries are displayed in a mobile friendly way using Cards from Material UI. Each card is customized by the data of the user (picture, function, name, etc. ).

Figure 22. Team module

Clicking a card opens a more detailed view of the employee, where you can see non-sensitive data like Name, Office, Function, Email, Age, Date of Employment and Avatar.

Both the card and the employee dialog contain the picture of the user. The picture of the user is created by the administrator at the same time with the user, and stored in the database as a byte64 array. Then the picture is cropped by the Material-UI Avatar component.

Graphical user interface, application

Description automatically generated

Figure 23. Team module - Employee details

## Holiday Calendar module

The module displays a read-only calendar similar to the one available in Microsoft Outlook. This calendar displays three types of entries.

* The first ones are the days off of the current logged in user.
* This user is assigned to a team, so in the calendar the user will also see all the holidays of it’s team members.
* In addition to this the national days defined from the administrative level are also displayed. This way the team can make a better development organization, by knowing everytime when a colleague is going to be missing, or when all the team will be gone and the sprint needs to be shorted.

Calendar

Description automatically generated with medium confidence

Figure 24. Holiday calendar module

## Booking module

It’s the administrative level of the Module 5(Meetings). It’s also a calendar similar to the one in Microsoft Outlook but this one is not read-only. In the calendar we can see all the meeting rooms defined in the application ( the calendar will automatically change when a new meeting room is inserted or deleted).

From here an user which has the permission to access the calendar can book a meeting room for a certain team. This way the planning inside the company will be way easier, there will be no mistaken overlaps in meeting rooms. The application also support future planning and recurrent planning. When a meeting is created for a team, this will be displayed in Module 5 to all team members.

Chart

Description automatically generated

Figure 25. Booking module

Graphical user interface, application

Description automatically generatedIf we double click inside a cell from the calendar a menu opens from where we can create a meeting. If we double click a meeting the same menu opens and we can edit or delete the meeting. Also dragging a meeting across the calendar is possible, changing it’s hour/meeting room.

Figure 26. Booking module menu

## Manage employees module

Graphical user interface, application, table

Description automatically generatedIt’s the main administrative level, in here all the employees are created, deleted or updated. The creation of an employee is made out of 4 steps.

Figure 27. Manage employees module

Graphical user interface, text, application

Description automatically generatedThe first step is adding personal data like Name, Email, Phone Number, Date of Birth, Country of Provenience.

Figure 28. Manage employees module step 1

Graphical user interface, text, application, email

Description automatically generated The second step contains company specific data like Office, Function, Team, Salary, Number of free days, Work email, Account password.

Figure 29. Manage employees module step 2

The third step is adding a picture of the user, after a picture is uploaded the user can see a preview of it.

Graphical user interface, application

Description automatically generated

Figure 30. Manage employees module step 3

And the last step, the most important one is the module assigning step. In here the administrator can select which modules of the application the user will be able to see.

If the user is missing a module and he tries to access that page in any way(ex: from url) he will be automatically redirected to the Access Denied page. Also if an used tries to make a request to a module controller that is missing permission he will receive an Unauthorized Http Code.

Between all the steps the data is changed at any time, meaning that if we go back to a step, when we come back the data will always be saved. This way the user can go back and make changes without losing the work.

The modules of an user can be modified at any time by the administrator.

Graphical user interface, application

Description automatically generated

Figure 31. Manage employees module step 4

## Manage holidays module

It’s the administrative module of the requests. It strongly related to the Module 2 and Module 3. In here the person marked as “Administrator” when a leave request or day off is created will see all it’s pending request.

There are two button for each request: “Accept” and “Decline”. If a leave request is accepted or declined this will be automatically reflected in the status of the request from Module 2 and Module 3. If a request was accepted and the user changed his mind he can always change it’s option. In addition when a day off is accepted the number of free days of the user is automatically calculated.

There are 3 statuses for leave requests and days off:

* Status = 0 -> Waiting
* Status = 1 -> Refused
* Graphical user interface, application

  Description automatically generatedStatus = 2 -> Approved

Figure 32. Manage holidays module leave requests

There is one extra tab on this module, the National Days tab, which defines the free days across all the company employees. This national days are displayed in each user’s holiday calendar. All the grids from this levels offer the user the possibility to generate a report of all the data. The report is generated as excel and contains the data visible in the grid.

Graphical user interface, table

Description automatically generated

Figure 33. Manage holidays module national days

## Manage organization

It’s also an important administrative module. In here the administrator can create, update or delete it’s own company details. There are three tabs:

* Teams where all the teams are defined(the teams are used all over the application, for example when creating a meeting or an user).
* Functions tab is also needed when creating an user. Whenever we add a Function the Functions chart from Module 4 is automatically updated
* Rooms tab contains the information about offices and meeting rooms

For all these tabs there are some predefined entries created by the DbContext onModelCreating seed method, which is called the first time when a migration is ran on the database. There is also an user with all the modules enabled created on startup. The credentials for this user will be provided later in the presentation.

## Table Description automatically generatedUnauthorized access

Figure 34. Manage organization module offices and meeting rooms

A picture containing graphical user interface

Description automatically generatedThe routing is strongly protected by the permissions, meaning that if the user tries to access a module which it’s missing from it’s own attributions he will always end on the Access Denied page. The only way to end on a missing permission page is to try to access the page from the URL. Otherwise the menu items won’t be available in the drawer.

Figure 35. Access denied

## Login page

Graphical user interface, website

Description automatically generatedThe login page doesn’t have a register possibility, the only person that can create an user is the administrator or a person with Module 9. When the user logs in he has the possibility to check the remember me option which will store it’s token to the localStorage instead of sessionStorage. For the time being the token is valid a month. If the token expires the user is logged out of the application.

Figure 36. Login page

## Graphical user interface, application, table, Excel Description automatically generatedExcel reports

Figure 37. Employees excel report from HRDesk

Graphical user interface, application, table, Excel

Description automatically generated

Figure 38. Leave request excel report from HRDesk

Graphical user interface, table

Description automatically generated

Figure 39. National days excel report from HRDesk

# uml diagrams

## Class diagram

Diagram, schematic

Description automatically generated

Figure 40. UML Class diagram

## Use case diagram – Basic user

Diagram

Description automatically generated

Figure 41. UML Use case diagram – User

## Use case diagram – Administrator

Diagram

Description automatically generated

Figure 42. UML Use case diagram – Administrator

## Activity diagram – day off

Diagram

Description automatically generated

Figure 43. UML Activity diagram - day off

# hOw to use

## Fetch or download the code

The code can be fetched from Github: <https://github.com/mandrei28/Human-resources-management> from the branch feature/initial\_configuration or main branch / from the CDs. The repository is now public for everyone. In order to fetch it I recommend using Sourcetree, all you need is a Github account, then from sourcetree clone the repository to your local computer. Make sure you’re on the feature/initial\_configuration or main branch. Now that we have the code locally we can move on to the next step.

## Link and create the database

First of all we need to have SQL Server installed. After we install SQL Server we need to modify the connection string from the “appsettings.json”.

Text

Description automatically generated

Figure 44. Connection string

Server = SQL Server hostname(if it’s local then you can use localhost instead of the computer name)

Database = Database name, this can be anything.

Now that the connection is made, the next step is to actually create the database. In order to create the database we need to open the package manager console from Tools -> NuGet package manager -> Package manager console. We select the target project as HRDesk.Infrastructure(because the database context is placed there) and then we run the command Update-Database. This will automatically create the tables. Also there are some entries inserted by the seed method, like the first user.

A screenshot of a computer

Description automatically generated

Figure 45. Apply migration

After this the backend solution should be up and running.

Text

Description automatically generated

Figure 46. Default user

The default user has all the modules enabled. Credentials:

Username: [administrator@admin.com](mailto:administrator@admin.com)

Password: administrator(encrypted in the database).

## Frontend solution setup

All the packages needed by the frontend solution are stored in the package.json file. Due to this running the “npm install” command will automatically get the dependencies.

While running the commands the path needs to look like this: Licenta-Human-Resources-Management\WebApp\hrdesk

So the steps are: First run the “*npm install*” command

Then *npm build* to check that everything is ok

And in the end *npm start,* which will cause the website to open at[*http://localhost:44332/*](http://localhost:44332/)

Using the credentials provided above the user can log in the application and begin the manual setup phase. He needs to go to the administrative levels and add offices, teams, functions, meeting rooms and employees. From this point the application can be ran in anyway the user desires. The setup is complete.

# Termeni de utilizare

## Autorii

This bachelor thesis document is done by Marcu Andrei Cristian, student at Faculty of Automatics, Computers and Electronics in Craiova, Romania.

## Licența de utilizare

The application has no licenses constraints, everything that was used inside the application was free to use. Anyone is allowed to use the code, the application was developed for educational purpose.

# Concluzii

Both ASP.NET Core API and React are modern technologies that are used by many companies in the development of web applications. These technologies are continously growing by receiving constant updates. The apparition of React and the other similar framework(Angular, Vue.js) simplified the web programming process by a lot. Since the development became easier a new function appeared „Full stack developer” which handles both backend and frontend of the application. In the past those two were splitted on Frontend Developer and Backend developer.

Given the topic of the application I consider that choosing React and ASP.NET Core was the perfect choice for a web application. While developing the applicaiton I’ve improved my skills with these two technlogies and also applied some new knowledge like Redux and axios.

Since the application isn’t working with big sets of data, Entity Framework can easily handle the database query part. If at any time the application will begin to move slow Entity Framework LINQ queries will have to be replaced by classic Stored Procedure, where we can control the code that is actually executed behind. Entity Framework creates it’s own query, which cannot be modified by the actual user.

At the moment of this document there are some limitations regarding the performance, the biggest one being the pagination on the grids. The pagination is implemented on the frontend side but on the backend is not yet added in the queries. An improvement would be, when working with a lot of data, to use skip and take while retrieving data.

The topic of the bachelor thesis, human resources management, can be extended way more in the future, we can add for example possibility to add food menus, to create polls, internal competitions, chat, maybe it’s own project media storage and many more.

I’ve tried, as much as possible, to avoid writing duplicate code. On the backend side especially I’ve applied some patterns(Repository Pattern + Unit Of Work) in order to avoid duplicate queries. Also these gave me an extra layer of abstractization between the database frameworks and the actual code. This way if at any time we will have to change the Entity Framework for example this will be done only by changing the code in the repositories without affecting the actual services and controllers.

The application should also be usable on smaller devices, like tablets and laptops, since the layout was made as responsive as possible. There might be issues on the grids with multiple columns but yet it’s still usable with a scroll option.

From security standpoint I’ve tried to keep the application as secure as possible since it contains sensitive data, meaning that accessing a module from frontend which is not available for that specific used would redirect the user to the access denied page, also if the user tries to request some data from a module that is not available he will receive a HTTP 401 ERROR Unauthorized, and even more sensitive data like password is encrypted in the database.

In conclusion, this bachelor thesis gave me the opportunity to use the knowledge acquired across the years of university one more time in a single project. This it the most complex application I’ve developed on my own, and I hope it will mark all the requirements of the Human Resources Management topic.

# Bibliografie

[ANDR19] – *ASP.NET Core in Action – First edition*, Andrew Lock, Manning Publications, 2019

[JUL10] – *Programming Entity Framework, 2nd Edition*, Julia Lerman, O’Reilly Media, Inc., 2010

# Referințe web

1. ASP.NET Core logo - <https://commons.wikimedia.org/wiki/File:.NET_Core_Logo.svg>
2. ASP.NET Core Wikipedia - <https://en.wikipedia.org/wiki/ASP.NET_Core>
3. Microsoft – Introduction to NuGet - <https://docs.microsoft.com/en-us/nuget/what-is-nuget>
4. Netsolutions – ASP.NET Core vs ASP.NET - <https://www.netsolutions.com/insights/net-core-vs-net-framework/>
5. Entity Framework Core Wikipedia - <https://en.wikipedia.org/wiki/Entity_Framework>
6. Entity Framework Code First Approach – EntityFrameworkTutorial - <https://www.entityframeworktutorial.net/code-first/what-is-code-first.aspx>
7. React ( JavaScript library ) - <https://en.wikipedia.org/wiki/React_(JavaScript_library)>
8. React logo - <https://en.wikipedia.org/wiki/React_(JavaScript_library)#/media/File:React-icon.svg>
9. What is JSX? - <https://www.reactenlightenment.com/react-jsx/5.1.html>
10. React Redux logo - <https://commons.wikimedia.org/wiki/File:Redux.png>
11. SQL Server Wikipedia RO - <https://ro.wikipedia.org/wiki/Microsoft_SQL_Server>
12. SQL Server Wikipedia EN - <https://en.wikipedia.org/wiki/Microsoft_SQL_Server>
13. GitHub logo - <https://en.wikipedia.org/wiki/GitHub#/media/File:GitHub_logo_2013.svg>
14. GitHub wikipedia - https://en.wikipedia.org/wiki/GitHub

# Codul sursă

Source code can be found on Github: https://github.com/mandrei28/Human-resources-management

# Site-ul web al proiectului

The website is not hosted, but here is the GitHub repository : https://github.com/mandrei28/Human-resources-management

# 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