Management-ul unei firme

Document de analiza si design

Numar echipa: 1

Studenti:

* Alisie Răzvan
* Martin Darius
* Murea Marius
* Pereanu Alexandru
* Pop Tudor

**Grupa: 30239**

Contents

[I Specificatia proiectului 3](#_Toc21463357)

[1.1 Diagrama Domain Model 3](#_Toc21463358)

[II Modelul Use-Case 3](#_Toc21463359)

[2.1 Identificarea Use-Case-uri 4](#_Toc21463360)

[2.2 Diagrama UML Use-Case 4](#_Toc21463361)

[III Design arhitectural 5](#_Toc21463362)

[3.1 Arhitectura conceptuala 5](#_Toc21463363)

[3.2 Diagrama de pachete 5](#_Toc21463364)

[3.3 Diagrama de clase 5](#_Toc21463365)

[3.4 Diagrama bazei de date 6](#_Toc21463366)

[3.5 Diagrame de secventa 6](#_Toc21463367)

[3.6 Diagrame de activitati 6](#_Toc21463368)

[IV Specificatii suplimentare 6](#_Toc21463369)

[4.1 Specificatii non-functionale 6](#_Toc21463370)

[4.2 Constrangeri de design 6](#_Toc21463371)

[V Testare 7](#_Toc21463372)

[5.1 Testarea functionalitatii aplicatiei 7](#_Toc21463373)

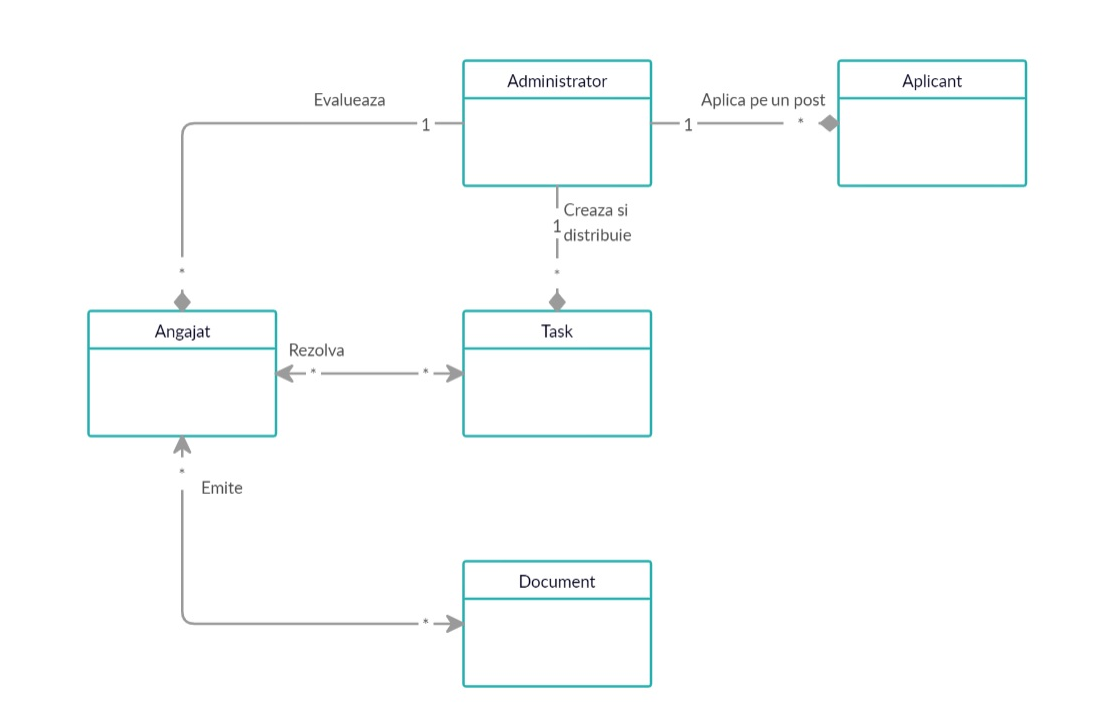
[5.2 Dezvolatari ulterioare 7](#_Toc21463374)

[VI Bibliografie 7](#_Toc21463375)

# I Specificatia proiectului

Proiectul acesta se referă la management-ul unei firme, unde ca și entități avem următorii: administrator-ul firmei (sau administratorii), angajații și cei care aplică la posturile disponibile din cadrul firmei. În mare, aplicația dispune de mai multe ferestre dintre care putem enumera următoarele: o fereastră de Log-in și trei ferestre principale pentru aplicanții la posturi, administratori și angajați din care vor rezulta alte ferestre secundare pentru diferitele task-uri pe care le are fiecare entitate mare în parte amintită mai sus. Aplicația are în spate o bază de date care se va actualiza de fiecare dată când cineva încearcă să lucreze pe ea (să rezolve task-uri, să aplice la un job, să angajeze oameni etc.).

## Diagrama Domain Model



# II Modelul Use-Case

În urma rulării aplicației se va deschide o fereastră de login. În funcție de natura contului se va deschide fie fereastra Adminului, fie fereastra Angajatului, fie fereastra Aplicantului.

Dacă s-a înregistrat în aplicație un admin, el va putea: să distribuie task-uri angajaților, să angajeze aplicanți la diferite posturi, să concedieze/evalueze angajați, să creeze/elimine task-uri.

Dacă s-a înregistrat în aplicație un angajat, el va putea: să rezolve un task și să creeze un document aferent acelui task rezolvat de el.

Dacă s-a înregistrat un aplicant la un post, el va putea: să aplice la un post disponibil.

## 2.1 Identificarea Use-Case-uri

1)

**Nume Use-case**: Admin Use-Case

**Nivel**: Administrare Task-uri

**Actor principal:** Administratorul

**Scenariul principal de success:** Administratorul poate adăuga/elimina task-uri sau distribui task-uri la angajați.

2)

**Nume Use-case**: Admin Use-Case

**Nivel**: Administrare Angajați

**Actor principal:** Administratorul

**Scenariul principal de success:** Administratorul poate concedia sau evalua angajați.

3)

**Nume Use-case**: Admin Use-case

**Nivel**: Administrare Aplicanți

**Actor principal:** Administratorul

**Scenariul principal de success:** Administratorul poate angaja diferiți aplicanți la posturi.

4)

**Nume Use-case**: Angajat Use-case

**Nivel**: Task-uri

**Actor principal:** Angajat

**Scenariul principal de success:** Angajatul poate rezolva task-urile pe care admin-ul i le-a distribuit lui.

5)

**Nume Use-case**: Angajat Use-case

**Nivel**: Document

**Actor principal:** Angajat

**Scenariul principal de success:** Angajatul poate crea un document aferent task-ului rezolvat de el.

6)

**Nume Use-case**: Aplicant Use-case

**Nivel**: Post

**Actor principal:** Aplicant

**Scenariul principal de success:** Aplicantul poate aplica la un post dacă este disponibil sau nu.

## 2.2 Diagrama UML Use-Case

# usecase

# III Design arhitectural

Cum s-a specificat și la capitolul cu “Specificația proiectului” acest proiect se ocupă cu management-ul unei firme când avem ca și entități administratori, angajați și aplicanți la diferite posturi. Pentru început dintr-o interfață numită “Meniu Principal” se intră într-un Log-In din care utilizatorul se poate înregistra ca Admin, Angajat sau Aplicant. În funcție de alegerea utilizatorului se va intra într-una din interfețele entităților administratorului, angajatului sau aplicantului.

Dacă s-a intrat pe interfața Admin, administratorul va putea să administreze task-urile din baza de date sau angajații firmei. Acesta va putea să adauge, să elimine task-uri sau dacă vrea, sa distribuie un task la un angajat. Dacă admin-ul va alege să administreze angajații el va putea să-i concedieze sau să-i evalueze în funcție de performanțele lor. În caz că s-a ales interfața de Aplicant din cadrul Administratorului, el va putea să angajeze diferiți oameni ce au aplicat la posturile disponibile.

De pe interfața Angajat, un angajat va putea să rezolve un task ce îl are la dispoziție sau să creeze un document aferent unui task rezolvat de el.

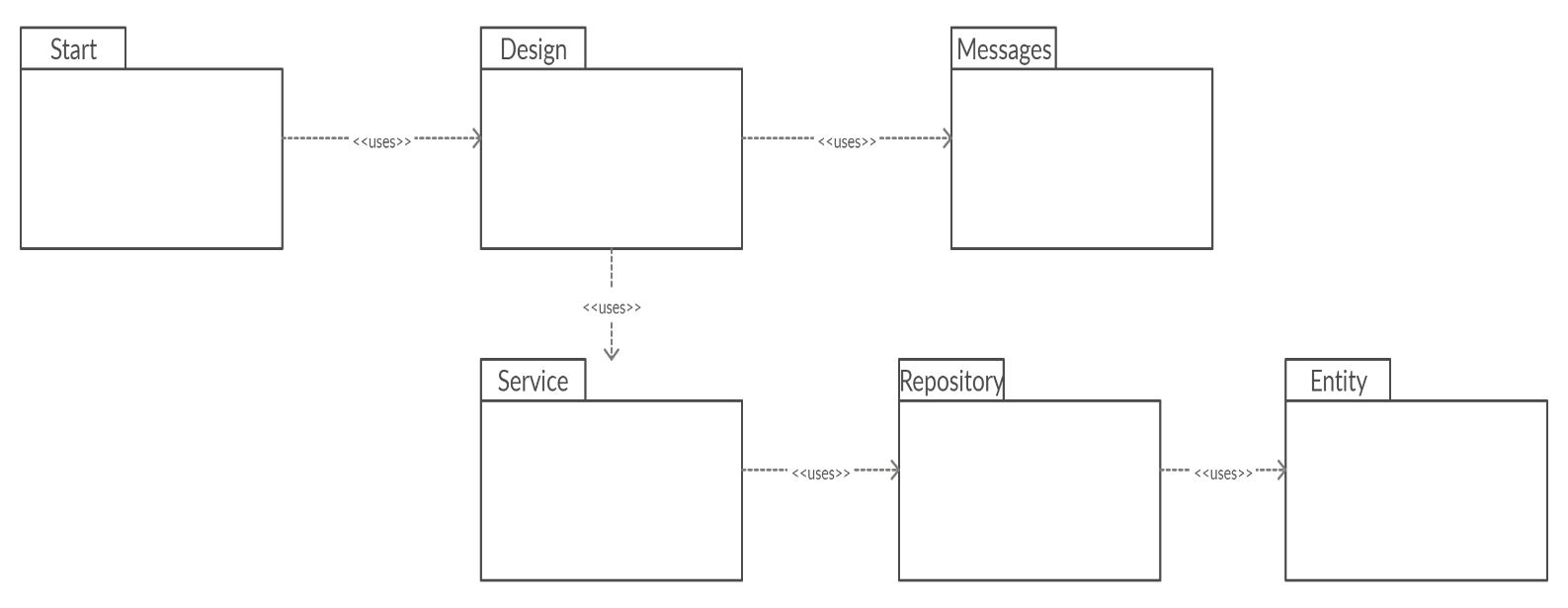
De pe interfața Aplicant, un utilizator va putea să aplice la un post, dacă este disponibil, iar el va aștepta pe urmă să fie acceptat sau nu de administratorul/administratorii firmei.

## 3.1 Arhitectura conceptuala

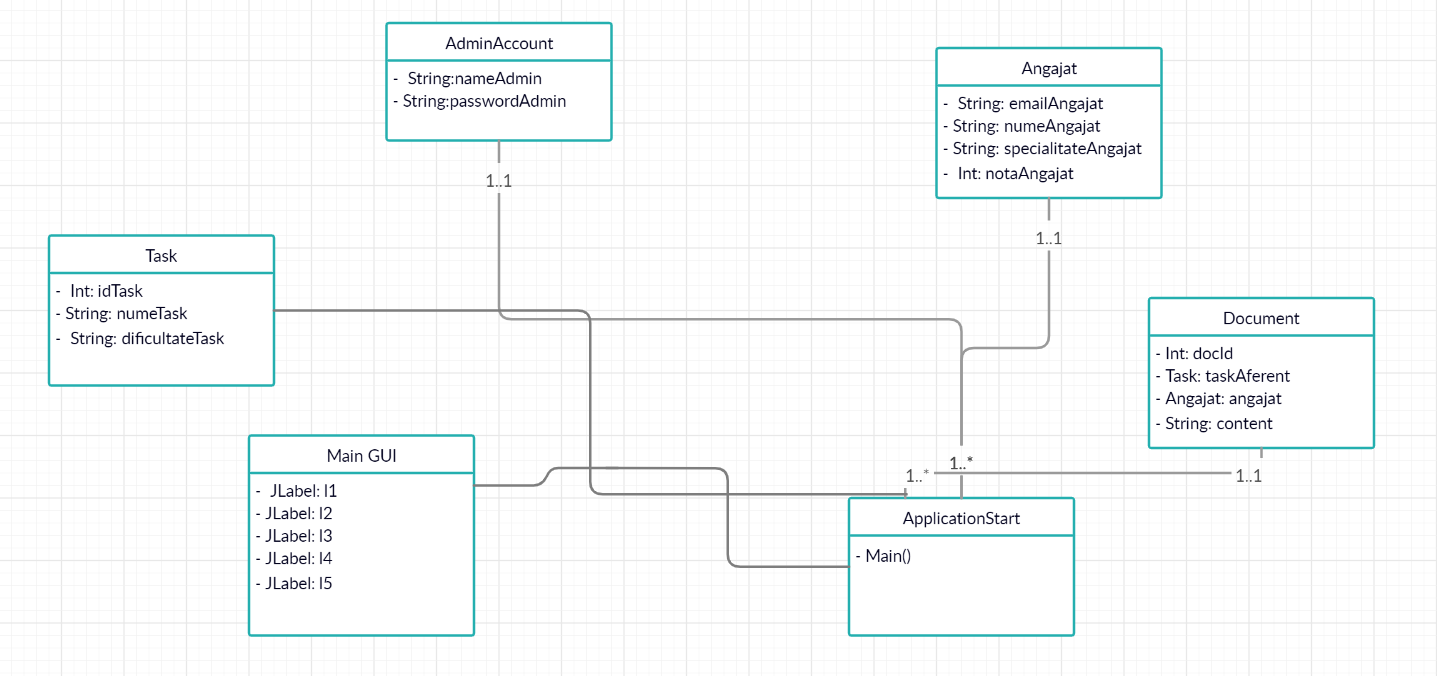
Aplicația este de tip desktop. Este implementată în principal în limbajul Java și are elemente de Spring deoarece se execută în spate, în partea de back-end și instrucțiuni SQL, ajutându-ne de Java.

Aplicația este legată la o bază de date ce conține tabele pentru administratori, angajați, aplicanți la diferite posturi și eventual și task-uri. În interiorul ei există 7 pachete principale, dintre care, cele mai voluminoase sunt cele de design, repository și entity. La design se pot găsi toate interfețele GUI, unde se află butoane, txt field-uri, imagini etc. Pachetul de repository reprezintă un pachet în care se găsesc toate clasele aferente entităților ce se pot regăsi în aplicație și cu care se fac legăturile la tabelele bazei de date ce se află în spatele aplicației. La entity sunt organizate toate clasele ce sunt entități în cadrul aplicației (Admin, User, Task, etc.).

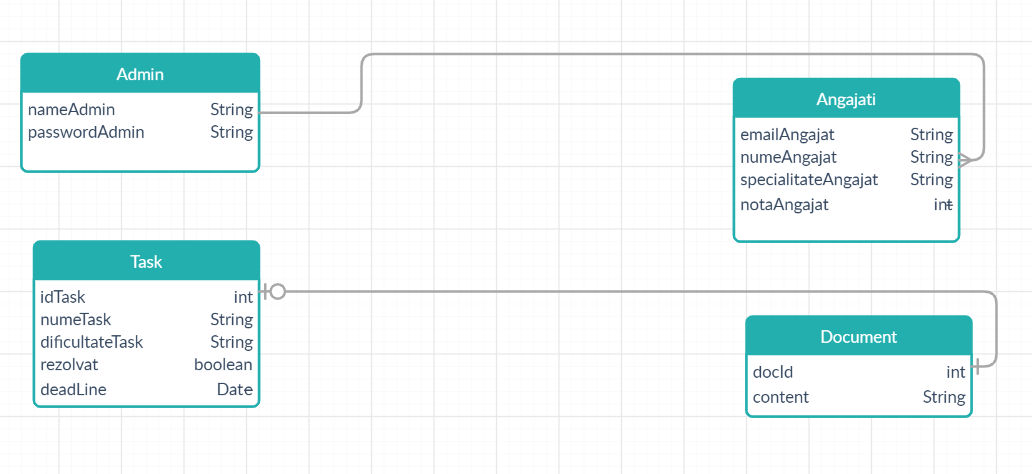
## 3.2 Diagrama de pachete

**

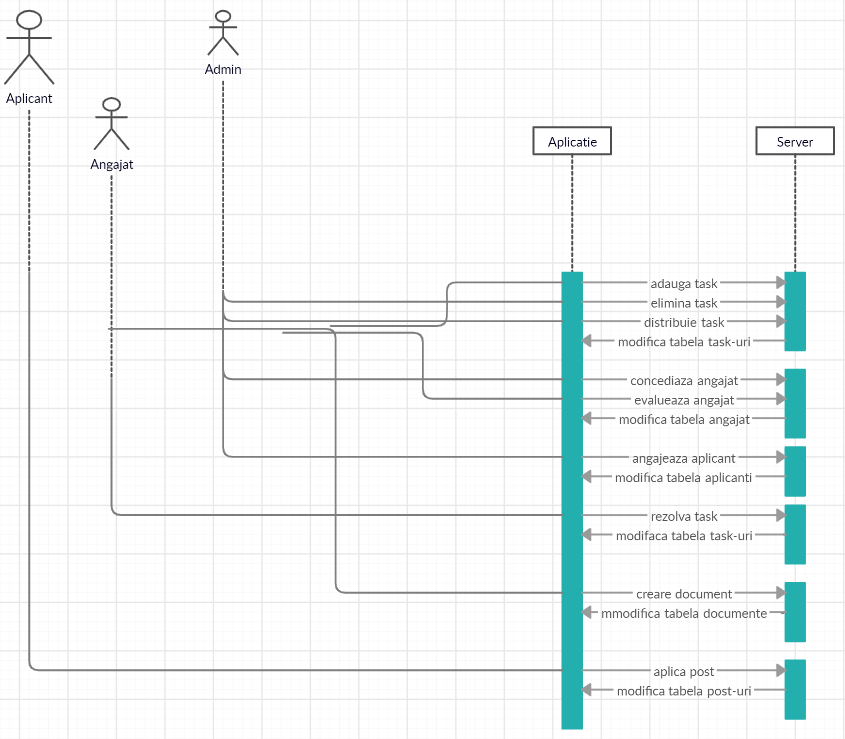
## 3.3 Diagrama de clase



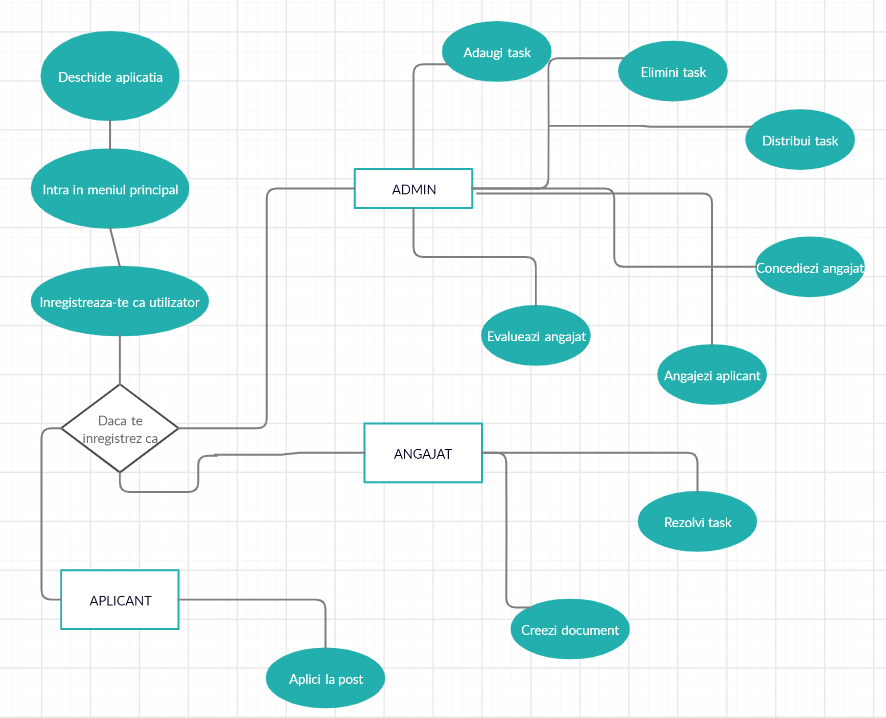
## 3.4 Diagrama bazei de date



## 3.5 Diagrame de secventa



## 3.6 Diagrame de activitati



## IV Specificatii suplimentare

Aplicația a fost dezvoltată în mare parte folosind limbajul Java. Există unele părți unde s-a folosit și Spring pentru a putea face conexiunile cu baza de date. Versiunea de Java folosită este JDK 8, iar poate pe viitor se va migra la o versiune mai nouă, de exemplu JDK 10. Baza de date a proiectului este una relatională și anume MySQL.

## 4.1 Specificatii non-functionale

La această aplicație se pot observa următoarele specificații non-funcționale:

1. **Usability**: aplicația în sine este una ușor de folosit de toată lumea, interfețele sunt prietenoase, destul de simple și ușor de înțeles. Această versiune conține în spate o bază de date minimală, cu doar câteva tabele. Se poate folosi în principal în firme ce au nevoie de baze de date pentru a stoca angajații, administratorii (dacă sunt mai mulți), task-uri, etc.
2. **Portability**: pentru viitor este în plan a se face și o versiune de android sau poate chiar web dar deocamdată aplicația este disponibilă doar pe desktop.
3. **Response time**: timpul de răspuns este imediat, aplicația rulează rapid (depinde mult totuși și de performanțele stației de pe care rulează aplicația).
4. **Scalability**: pe viitor poate să apară o versiune extinsă a acestei aplicații (o bază de date mai complexă, versiune și pe web/android).
5. **Security**: aplicația are un log-in securizat, foarte greu de spart parola, necesită ani de experiență în hack-ing și mult timp.
6. **Fault tolerance**: pot apărea în cadrul rulării aplicației diferite bug-uri și excepții dar aplicația în sine nu va da crash.
7. **Reusability**: în cadrul aplicației există bucăți de cod ce pot fi folosite în alte sisteme, cum ar fi partea de Spring-Java-SQL ce poate fi folosită în sisteme de management al restaurantelor, ca de exemplu.
8. **Platform compatibility**: da, aplicația este făcută în mare parte în Java, deci e platform independent.
9. **Open source**: deocamdată nu, aplicația este în stadiul Beta.
10. **Internationalization and localization**: pe viitor poate va apărea o versiune ce va fi disponibilă și în mai multe limbi.

## 4.2 Constrangeri de design

# V Testare

## 5.1 Testarea functionalitatii aplicatiei

## 5.2 Dezvolatari ulterioare

# VI Bibliografie