Modelarea unor procese de resource management folosind oetpn

PROIECT DE DIPLOMĂ

Autor: **Ioana Petronela BOGDAN**

Conducător științific: **Titlu.ing. Prenume NUME**

|  |  |
| --- | --- |
| DECAN  **Prof.dr.ing. Liviu MICLEA** | Vizat,  DIRECTOR DEPARTAMENT AUTOMATICĂ  **Prof.dr.ing. Honoriu VĂLEAN** |

Autor: **Ioana Petronela BOGDAN**

Modelarea unor procese de resource management folosint OETPN

1. **Enunțul temei:** *O scurtă descriere a temei proiectului de diplomă*
2. **Conținutul proiectului:** *(enumerarea părților componente) Pagina de prezentare, Declarație privind autenticitatea proiectului, Sinteza proiectului, Cuprins, Titlul capitolului 1, Titlul capitolului 2,… Titlul capitolului n, Bibliografie, Anexe.*
3. **Locul documentării:** *Universitatea Tehnică din Cluj-Napoca, alte locuri dacă este cazul*
4. **Consultanți:** *ing. Prenume Nume (dacă este cazul)*
5. **Data emiterii temei:**
6. **Data predării:**

Semnătura autorului

Semnătura conducătorului științific

**Declaraţie pe proprie răspundere privind**

**autenticitatea proiectului de diplomă**

Subsemnatul(a) **Prenume NUME**  , legitimat(ă) cu CI/BI seria nr. , CNP ,

autorul lucrării:

elaborată în vederea susținerii examenului de finalizare a studiilor de licență la **Facultatea de Automatică și Calculatoare**, specializarea **Automatică și Informatică Aplicată,** din cadrul Universității Tehnice din Cluj-Napoca, sesiunea Iulie 2021 a anului universitar 2020-2021, declar pe proprie răspundere, că această lucrare este rezultatul propriei activități intelectuale, pe baza cercetărilor mele și pe baza informațiilor obținute din surse care au fost citate, în textul lucrării, și în bibliografie.

Declar, că această lucrare nu conține porțiuni plagiate, iar sursele bibliografice au fost folosite cu respectarea legislației române și a convențiilor internaționale privind drepturile de autor.

Declar, de asemenea, că această lucrare nu a mai fost prezentată în fața unei alte comisii de examen de licență.

În cazul constatării ulterioare a unor declarații false, voi suporta sancțiunile administrative, respectiv, *anularea examenului de licență*.

Data Prenume NUME

(semnătura)

**SINTEZA**

proiectului de diplomă cu titlul:

Titlul lucrării

Autor: **Prenume NUME**

Conducător științific: **Titlu.ing. Prenume NUME**

1. Cerințele temei:

2. Soluții alese:

3. Rezultate obținute:

4. Testări și verificări:

5. Contribuții personale:

6. Surse de documentare:

Semnătura autorului

Semnătura conducătorului științific

Cuprins

[1 Introducere 3](#_Toc104374792)

[1.1 Context general 3](#_Toc104374793)

[1.2 Prezentarea pe scurt a tehnologiilor folosite 4](#_Toc104374794)

[1.3 Motivația 4](#_Toc104374795)

[1.4 Conținutul lucrării 4](#_Toc104374796)

[2 Obiectivele proiectului 6](#_Toc104374797)

[2.1 Obiectivul principal 6](#_Toc104374798)

[2.2 Obiective funcționale 7](#_Toc104374799)

[2.2.1 Posibilitatea unui user de a se înregistra în aplicație 7](#_Toc104374800)

[2.2.2 Login si Logout 7](#_Toc104374801)

[2.2.3 Existența a doua roluri pentru utilizatori 7](#_Toc104374802)

[2.2.4 Vizualizare date cu privire la fiecare aplicant, impreuna cu realizarea operațiilor de baza (CRUD) pentru fiecare anunț in parte 8](#_Toc104374803)

[2.2.5 Trimitere e-mail cu detaliile cu privință la acceptarea sau respingerea pentru postul selectat 8](#_Toc104374804)

[Fiecare aplicant va primi un mail in care este informat cu privință la statusul aplicării. 8](#_Toc104374805)

[2.3 Obiective non- funcționale: 8](#_Toc104374806)

[2.3.1.1 **Cerințe non-funcționale** 8](#_Toc104374807)

[2.3.2 Accesabilitatea 8](#_Toc104374808)

[2.3.3 Performanța 8](#_Toc104374809)

[2.3.4 Mentenanța 9](#_Toc104374810)

[2.3.5 Independență 9](#_Toc104374811)

[2.3.6 Fiabilitate 9](#_Toc104374812)

[2.4 Cerințe funcționale 9](#_Toc104374813)

[**2.4.1.1** **Tabel cerințe funcționale ale sistemului realizat** 10](#_Toc104374814)

[2.5 Specificații 10](#_Toc104374815)

[3 Studiu bibliografic 11](#_Toc104374816)

[3.1 Concepte de bază 11](#_Toc104374817)

[3.1.1 Arhitectura REST 11](#_Toc104374818)

[3.1.2 OETPN (Object Enhanced Time Petri Nets) 11](#_Toc104374819)

[4 Analiză și Fundamentare Teoretica 12](#_Toc104374820)

[4.1 Cazuri de utilizare 12](#_Toc104374821)

[4.1.1 Actorii aplicației 12](#_Toc104374822)

[**4.1.1.1** **Caz de utilizare pentru Admin** 13](#_Toc104374823)

[**4.1.1.2** **Caz de utilizare pentru Aplicant** 13](#_Toc104374824)

[**4.1.1.3** **Caz de utilizare pentru HR Manager** 14](#_Toc104374825)

[4.1.2 Descrierea tuturor cazurilor de utilizare 14](#_Toc104374826)

[**4.1.2.1** **Caz de utilizare 1** 14](#_Toc104374827)

[**4.1.2.2** **Caz de utilizare 2** 15](#_Toc104374828)

[**4.1.2.3** **Caz de utilizare 3** 16](#_Toc104374829)

[**4.1.2.4** **Caz de utilizare 4** 17](#_Toc104374830)

[4.2 Tehnologii și concepte utilizate pentru dezvoltarea aplicației web 18](#_Toc104374831)

[4.2.1 Spring Boot 18](#_Toc104374832)

[4.2.2 React JS 20](#_Toc104374833)

[4.2.3 SQL Workbench 21](#_Toc104374834)

[4.2.4 Bootstrap 21](#_Toc104374835)

[5 Proiectare de Detaliu și Implementare 22](#_Toc104374836)

[5.1 Descrierea ficărei pagini prezente în aplicația web 22](#_Toc104374837)

[5.1.1 Pagina APLICANT 22](#_Toc104374838)

[5.1.2 Pagina HR Manager 22](#_Toc104374839)

[5.2 Proiectarea bazei de date 23](#_Toc104374840)

[5.2.1 Tabela User- în interiorul acestei tabele se colectează toate informațiile cu privire la fiecare utilizator, iar mai apoi sunt salvate în baza de date. 23](#_Toc104374841)

[5.2.2 Tabela Email- această tabelă este responsabilă de trimiterea email-urilor pentru cazurile de admis/respins către aplicanți 23](#_Toc104374842)

[5.2.3 Tabel UserJob 24](#_Toc104374843)

[5.2.4 Tabela Form- scopul acestei tabele este acela de a stoca întreaga aplicare a tuturor userilor 24](#_Toc104374844)

[5.2.5 Tabela FileInfo- folosind aceasta tabelă, vom memora fiecare CV încărcat de către aplicanți 25](#_Toc104374845)

[5.2.6 Tabela Job 25](#_Toc104374846)

[6 Testare și validare 26](#_Toc104374847)

[6.1 Rezultate obținute 26](#_Toc104374848)

[6.2 Direcții de dezvoltare 26](#_Toc104374849)

[7 Manual de Instalare și Utilizare 27](#_Toc104374850)

[8 Concluzii 28](#_Toc104374851)

[9 Reguli de formatare 29](#_Toc104374852)

[9.1 Formatarea paginii 29](#_Toc104374853)

[9.2 Titluri și stiluri 29](#_Toc104374854)

[9.3 Figuri, tabele și ecuații 30](#_Toc104374855)

[9.3.1 Figuri 30](#_Toc104374856)

[9.4 Tabele 30](#_Toc104374857)

[9.5 Ecuații 30](#_Toc104374858)

[9.6 Referințe bibliografice 31](#_Toc104374859)

[10 Bibliografie 32](#_Toc104374860)

# Introducere

## Context general

În introducere familiarizați cititorul cu motivația lucrării, plasați lucrarea într-un context care să permită cititorului să înțeleagă obiectivele.

Descrieți importanța lucrării, de ce merita să o faceți, plasați ideile într-un context larg.

Susțineți studiul: de ce exact aceasta aplicație/implementare. Comentați asupra aspectelor teoretice sau practice care v-au făcut să o alegeți.

În funcție de natura lucrării, ar putea fi necesar să prezentați informații de fond asupra domeniului în care se încadrează aplicația, mai ales dacă aveți o lucrare într-un domeniu multidisciplinar. În acest caz, puteți introduce terminologia pe care o utilizați în continuare.

Descrieți pe scurt lucrarea: ce conține fiecare capitol.

Noțiunea de sistem software reprezintă un concept foarte des întâlnit în viețile noastre de zi cu zi. De-a lungul timpului, această tehnologie a evoluat într-un ritm alert. Acest fapt, a dus la necesitatea creării unor sisteme tot mai performante, care reușesc cu ușurință să țină pasul cu cerințele fiecăruia dintre noi.

Ritmul continuu în care s-a dezvoltat resursa software are ca si merit un acces mai facil si mai rapid la orice categorie de interes.

Cu ajutorul tehnologiei, lucrările noastre sunt realizate mai rapid, economic si convenabil. Fie ca vorbim de acasă sau la birou, tehnologia ne face mai eficienți și productivi.

Proiectul propus spre realizare, este constituit dintr-o aplicație web care are la bază management pentru întregul proces de recrutare pentru un job. Acesta vine atât in sprijinul aplicanților, cât și în sprijinul angajaților de la departamentul de Resurse Umane. Întreaga aplicație este realizată astfel încât să ofere fiecărui utilizator o interfață prietenoasă, ușor de navigat și totodată una optimă .

## Prezentarea pe scurt a tehnologiilor folosite

Întreaga aplicație realizată se bazează doar pe componente software, iar acestea sunt dupa cum urmează:

* MySQL Workbench- constituie baza de date, acolo unde toate datele utlizatorilor care folosesc aplicația software sunt stocate. Comunicarea între FrontEnd si BackEnd se realizareză prin servicii diferite.
* Partea de backend este realizată in limbajul de programare Java, cu ajutorul framework-ului Spring Boot.
* Frontend-ul aplicației este construit cu ajutorul framework-ului React JS. Acesta este constituit din acea interfață grafică, ce este vizibilă clientului.

## Motivația

Un motiv al alegerii acestei teme, a venit din dorința de a aprofunda mai mult acest domeniu al aplicațiilor web, un domeniu in continuă dezvoltare si cu un mare impact asupra fiecăruia dintre noi. Analizând mai multe platforme software, am constatat că majoritatea acestora se ocupa cu managementul intern al unei companii, ci nu pe întregul proces de recrutare.

## Conținutul lucrării

În acest capitol, am realizat prezentarea per ansamblu atât a naturii teoretice, cât și a celei funcționale a acestei lucrări. In capitolele care urmează, vor fi expuse o serie de concepte care au dus la conturarea întregului proiect. Acestea sunt precum urmează:

* **Capitolul 1- Introducere-** Primul capitol reprezintă în linii mari atât natura funcțională cât și cea funcțională a lucrării.
* **Capitolul 2- Obiectivul Proiectului –** Acest capitol va descrie principalele obiective pe care dorește să le atingă proiectul, impreună cu detaliile semnificative care vor servi ca bază pentru construcția proiectului înca de la primele faze
* **Capitolul 3- Studiu Bibliografic-** Acest capitol constă in reexaminarea materialului bibliografic existent cu privire la subiectul care urmează a fi studiat și vine în sprijinul realizării proiectului. Seria de documentări din revistele de specialitate a fost necesară a fi realizată, asfel conturându-se întregul ansamblu.
* **Capitolul 4- Analiză și Fundamentare Teoretică-** În acest capitol se va analiza atât proiectarea, cât și implementarea proiectului. În același capitol se vor descrie pe scurt și detaliile care țin de tehnologiile utilizate pe parcursul realizării proiectului.
* **Capitolul 5- Proiectare de Detaliu si Implementare-** Pe parcursul acestui capitol, se va detalia arhitectura conceptuală a proiectului. Acest capitol va conține detaliile care au stat la baza elaborării codului aplicatiei, schemele descriptive, dar si diagrama bazei de date.
* **Capitolul 6- Testare, Validare și Evaluare-** Acest capitol va conține o serie de moduri prin care s-au realizat testăriile pentru fiecare componentă prezentă în proiect. Pentru a fi pusă în funcțiune, o aplicație are nevoie de mai multe validări pentru a fi considerată potrivită pentru a fi pus în producție
* **Capitolul 7- Manual de Instalare si Utilizare-** În cadrul acestui capitol se vor evidenția pașii necesari pentru instalarea si utilizarea aplicației efectuate, precum si configurările aferente acesteia.
* **Capitolul 8- Concluzii-** Acest capitol este predestinat concluziilor deduse dupa efectuarea aplicației, dar și posibilele îmbunătățiri care ar putea fi aduse aplicației

# Obiectivele proiectului

În cadrul acestui capitol, se vor prezenta toate obiectivele propuse pentru realizarea intregului sistem software. Scopul acestei aplicații fiind acela de a facilita întregul proces de recrutare pentru un job dorit. Aceasta aplicație este compusă din 2 părți: o parte care conține serverul (Java Spring) și cea de a doua parte, partea de client(React JS). Toate acestea vor atrage spre sine o îmbunătățire a întregului proces, atât intern cât și extern.

## Obiectivul principal

Principalul obiectiv al acestei aplicații este de a implementa un sistem care să fie util și totodată să mențină mereu in legătura aplicatul, cel care dorește să aplice la un job, cu partea de resurse umane a unei anumite firme. Pentru îndeplinirea acestora, sistemul creat va pune la dispoziția utilizatorului o aplicație software. În momentul în care un utilizator va deschide aplicația și se conectează cu credențialele specifice, acesta va fi introdus automat ca si utilizator în aplicație. Acest proces de selectare a candidaților necesita mai multe etape si totodată mai multe date care ar trebui să fie accesate. În această privință, realizarea unei aplicații web, este cea mai benefică opțiune.

Principalul avantaj pe care îl deține această aplicatie software, este acela că faciliteaza tot procesul de recrutare pentru un post dorit. Astfel va exista o buna comunicare între aplicant și departamentul de resurse umane, care se ocupă de întregul proces de recrutare.

Un alt avantaj este reprezentat de faptul că la sfârșitul aplicării pentru un job, partea de HR-ul, va putea genera o ierarhie a aplicanților în funcție de CV-ul incarcat de catre aceștia.

Prin această implementare, se urmărește implementarea unei aplicații care sa fie in același timp una concretă, dar și prietenoasă cu utilizatorul.

Sistemele software se caracterizează atât prin funcționalitate(ceea ce realizează sistemul), cat si prin non-funcționalitate(comportamentul sistemului cu privire la unele atribute observabile). Ambele aspecte sunt importante in dezvoltarea unui software.

## Obiective funcționale

Obiectivele funcționale au rolul de a descrie interacțiunile prezente intre sistem si mediu acestuia.

Principalele funcționalități implementate in cadrul aplicație sunt următoarele:

* Posibilitatea unui user de a își crea un cont în aplicație
* Posibilitatea ca în momentul in care un utilizator și-a creat un cont, si s-a logat, să poată intra in cont si să poată aplica pentru job-ul dorit
* Adăugarea joburilor disponibile împreună cu data până la care se poate aplica de catre admin
* Ștergerea unui job de catre admin
* Actualizarea unui job de catre “admin”
* Vizualizarea tuturor joburilor
* Trimitere email catre aplicant pentru cazul de accept/respins
* Realizarea unui rating al aplicanților
* Existența a doua roluri pentru utilizatori
* Vizualizare date cu privire la fiecare aplicant
* Cu ajutorul interfeței web prezentate, utilizatorul poate să încarce datele cu privire la job-ul dorit.
* Utilizatorul poate naviga facil prin toate paginile aplicației

### Posibilitatea unui user de a se înregistra în aplicație

Cu ajutorul acestei funcționalități, utilizatorul are posibilitatea de a-și crea cont, introducând datele personale în câmpurile prezente.

### Login si Logout

Pentru a putea efectua operații in aplicația existenta, toți utilizatorii trebuie in primul rând să fie autentificati. Întregul proces se realizeaza prin introducerea de către fiecare utilizator a unor credentiale (email, parolă). Aceste credențiale provin de la pasul anterior al înregistrării. O alta funcție prezentă aici, este aceea de logout a utilizatorului.

### Existența a trei roluri pentru utilizatori

In cadrul acestei funcționalități, vom avea prezente doua tipuri de utilizator: **HR Manger** (membrii echipei de HR al firmei prezentate), **user** (aplicantul care doreste să aplice pentru job-ul dorit), **admin** (cel care acces la întreaga gestiune a informațiilor aplicației)

### Vizualizare date cu privire la fiecare aplicant, impreuna cu realizarea operațiilor de baza (CRUD) pentru fiecare anunț in parte

Utilizatorii care detin rolul de “admin” au posibilitatea de a vizualiza intregul panou cu aplicanti, la ce job au aplicat, pot adauga o nota specifica CV-ului incarcat si pot face o ierarhie a tuturor acestora.

### Trimitere e-mail cu detaliile cu privință la acceptarea sau respingerea pentru postul selectat

### Fiecare aplicant va primi un mail în care este informat cu privință la statusul aplicării.

## Obiective non- funcționale:

Cerințele non-funcționale sunt acele comportamente ale sistemului cu privire la unele atribute observabile. Acestea descriu toate capacitățile sistemelor, dar si a serviciilor prezente intr-o aplicatie.

Diagram

Description automatically generated

#### **Cerințe non-funcționale**

### Accesabilitatea

* Un sistem trebuie să fie unul accesibil, adică un sistem care garantează faptul ca toate persoanele cu o gamă cât de larga de capacități pot accesa facil informațiile prezente in aplicație, iar această aplicație este disponibilă pe Internet. Scopul accesabilității software este acela de a asigura disponibilitatea si capacitatea de utilizare a aplicație software pentru o gamă cât mai largă de utilizatori.

### Performanța

* această cerință are ca și scop măsurarea vitezei și eficienței sistemului realizat.

### Mentenanța

* constă în inspectarea frecventă a aplicației. Scopul acestei inspectări este acela de a preveni situațiile de nefuncționare a anumitor componente și totodată remedierea defectelor constante.

### Independență

* în cadrul acestei cerințe se asigură faptul că aplicația va avea doar dependințe proprii

### Fiabilitate

* Este reprezentată de acea abilitate a sistemului de a reuși sa îndeplinească toate funcțiile cerute în condițiile stabilite.

## Cerințe funcționale

|  |  |  |
| --- | --- | --- |
| Identificator | Descriere a cerinței funcționale | Tipul utilizatorului |
| CF1 | Creare cont/Conectare către un cont existent | Toți utilizatorii |
| CF2 | Gestionare conturilor pentru **utilizatori** | Admin |
| CF2.1 | Gestionare conturi pentru **Aplicant** | Admin |
| CF2.2 | Gestionare conturi pentru **personalul HR** | Admin |
| CF2.3 | Gesionare conturi pentru **Admin** | Admin |
| CF3 | Gestionarea procesului de aplicare către un job dorit | HR manager |
| CF3.1 | Adăugare job | HR manager |
| CF3.2 | Ștergere job | HR manager |
| CF3.3 | Actualizare detalii privind un job | HR manager |
| CF3.4 | Vizualizare joburi | HR manager, Aplicant |
| CF3.5 | Alegerea unui job dorit | Aplicant |
| CF3.6 | Completarea unui form cu date personale si incarcarea CV-ului aferent aplicantului | Aplicant |
| CF4 | Administrarea tututor aplicanților | HR manager |
| CF4.1 | Vizualizarea CV-urilor aferente fiecărui aplicant | HR manager |
| CF4.2 | Acordarea unei note de la 1-10, in funcție de CV, pentru fiecare aplicant si ordonarea acestora într-un tabel | HR manager |
| CF4.3 | Vizualizarea unui grafic care indica numărul de aplicanți către un anumit job | HR manager |
| CF4.4 | Vizualizarea numărului total de aplicanti respinsi | HR manger |
| CF4.5 | Vizualizarea numărului total de aplicanti acceptați | HR manger |
| CF4.6 | Trimitere mail pentru acceptare/respingere pentru job | HR manger |
| CF4.7 | Vizualizare tabel cu datele personale ale aplicaților | HR manger |
| CF4.8 | Vizualizarea pe baza unei retele OETPN a progresului in procesul de aplicare a fiecărui user | HR manger |

#### **Tabel cerințe funcționale ale sistemului realizat**

## Specificații

În specificațiile lucrării detaliați cerințele. Descrieți ce intenționați să obțineți. Vă puteți referi la funcțiile aplicației, interfață, nivele de performanță, structuri de date, elemente, securitate, fiabilitate, calitate, limitări, etc.

# Studiu bibliografic

Conține o analiză a ceea ce s-a realizat/studiat anterior. Arătați că ați studiat materiale bibliografice și că ați înțeles ceea ce ați citit.

Puteți include diferite puncte de vedere asupra problemei pe care o rezolvați în lucrare.

Nu uitați să citați corespunzător autorii oricărei idei extrase dintr-o sursă bibliografică.

1. Putting Non-Functional Requirements into Software Architecture- Xavier Franch, Pere Botella
2. Software Accessibility: Recommendation and Guidelines- Alenka Kavcic

Diagram

Description automatically generated

## Concepte de bază

### Arhitectura REST

Arhitectura REST (Representational State Transfer) reprezintă un stil arhitectural software predestinat sistemelor hipermedia distribuite. XMLUn exemplu de astfel de sistem ar fi cel word wide web. Cu ajutorul acesteia, se dezvoltă cele mai multe aplicații web. Se vor realiza mesaje între client-server, însă starea acestora nu va putea fi reținută. Datele între cei care comunică se transmit folosind protocolul HTTP și sunt reprezentate în format XML sau în orice alt format.

Așa cum ne este specificat în [11], principalele metode ale serviciilor REST sunt:

* **PUT:** va concepe o nouă resursă
* **POST:** realizează o actualizare pentru o nouă resursă. Este folosită pentru a afla dacă serverul acceptă o entitate înglobată în cerere.
* **GET:** citește o resursă, dar nu o modifică, realizeaza o cerere de accesare cu privire la accesarea unor informații
* **DELETE:** are ca effect stergerea unei resurse

Printre aspectele esențiale amintite în cadrul [11], se enumără:

* Față de un model SOAP, acest stil are la baza standardele Web si HTTP
* Clienții au la dispoziție mai multe tehnici legate de caching care au ca scop creșterea de scalabilitate, dar și de performanță
* Sunt reprezentări diferite pentru resurse, cum ar fi: XML, JSON, dar pot fi definite chiar si de un user

Exemple de adnotări prezente în această arhitectură:

### OETPN (Object Enhanced Time Petri Nets)

Implementarea acesteia a fost realizata cu ajutorul limbajului de programare Java.

Chiar dacă noțiunea de OEPTN nu este des folosită, acasta este un real ajutor in detalierea dezvoltării de orice aplicație. Așa cum ne este explicat în [16] modelele OETPN, pot fi realizate să fie cât mai aproape de gândirea umană, dar totodată și mai aproape de programele în care urmează să fie implementate. OETPN poate fi folosit pe parcursul tuturor etapelor de dezvoltare a unei aplicații.

Exemplu de rețea OETPN împreună cu principiul după care funcționează o astfel de rețea:

Diagram, schematic

Description automatically generated

#### [16] Rețea OETPN

Se consideră:

* u,v – locații de intrare
* y ieșire
* x1,x2,x3 locații
* t\_1, t\_2, t\_3 tranziții

Grd reprezintă acel set de condiții

**Tranziția t\_1:**

* 

Tranziția t\_1 va putea transmite jetonul mai departe doar dacă în x1 vom avea jeton. Odată ce locația x1 va avea jeton, tranzitia t\_1 se va valida, iar locația x2 va lua valoarea locației x1+ intrarea u.

**Tranziția t\_2:**



Comportamentul prezent la tranziția 1 este același ca și în acest caz.

**Tranziția t\_3:**



Validarea acestei tranziții se realizează în momentul în care jetonul a ajuns in locația x3. După această validare, unul dintre jetoane va merge in locația x1, iar celălalt in ieșirea y .

# Analiză și Fundamentare Teoretica

Aceasta parte a lucrării este flexibilă și depinde foarte mult de natura lucrării, poate fi organizată în mai multe capitole și conține contribuțiile personale ale autorului.

Includeți:

* + Detalii referitoare la analiză și proiectare:
    - descrierea metodelor pe care le-ați aplicat pentru rezolvarea problemei,
    - descrierea materialelor, procedurilor
    - calcule, tehnici, descrierea echipamentelor
    - metodologia de proiectare
    - informațiile necesare pentru ca cineva să poată reface lucrarea
  + Implementare :
    - Descrieți detaliile tehnice ale implementării aplicației: mediul de implementare, modul de prezentare, modul de utilizare al aplicației, etc.
  + Testare si validare :
    - Descrieți metodologia de testare a aplicației și rezultatele
    - Includeți experimentele pe care le-ați realizat, analiza rezultatelor pe care le-ați obținut.

Cu ajutorul acestui capitol, vom parcurge toate tehnologiile și principiile legate de funcționare utilizate pe tot parcursul lucrării. Toate acestea se vor examina din punctul de vedere al cerințelor non-funcționale și se vor urmări modul de funcționare a fiecărei componente. Din toate acestea, vom extrage cerințele funcționale și vom realiza și o detaliere a tehnologiilor care au stat la baza creării întregii aplicații.

## Cazuri de utilizare

### Actorii aplicației

Întregul sistem conține trei tipuri de utilizatori:

* **Admin**- este acel tip de utilizator care se ocupă de intreaga gestionare a aplicației.
* **HR Manager**- utilizatorul care administrează toate etapele recrutării pentru un job dorit.
* **Aplicant**- utilizatorul cu cele mai putine acțiuni, cel care poate vizualiza întreg panoul de posturi disponibile și cel care poate aplica la unul din job-uri.

În figura următoare am reprezentat actorul **ADMIN** cu fiecare acțiune care îi aparține:

Diagram

Description automatically generated

#### **Caz de utilizare pentru Admin**

În figura care urmează este ilustrat profilul de **Aplicant:**

Diagram

Description automatically generated

#### **Caz de utilizare pentru Aplicant**

În următoarea figura am prezentat actorul principal, **HR Manager:**

Diagram, schematic

Description automatically generated

#### **Caz de utilizare pentru HR Manager**

### Descrierea tuturor cazurilor de utilizare

În componența acestui subcapitol, se va evidenția o scurta descriere a cazurilor de utilizare a aplicației

#### **Caz de utilizare 1**

**Numele cazului de utilizare**: Logarea unui utilizator în aplicație

**Actorul**: Aplicant, HR Manager, Admin

Această parte de adreseaza utilizatorului care s-a înregistrat cu succes. Prin inregistrare, fiecare actor va putea fi direcționat către pagina asociată acestuia. Doar parcurgând acest pas, se va putea merge mai departe.

**Precondiții**: accesarea de către utilizator a paginii de login

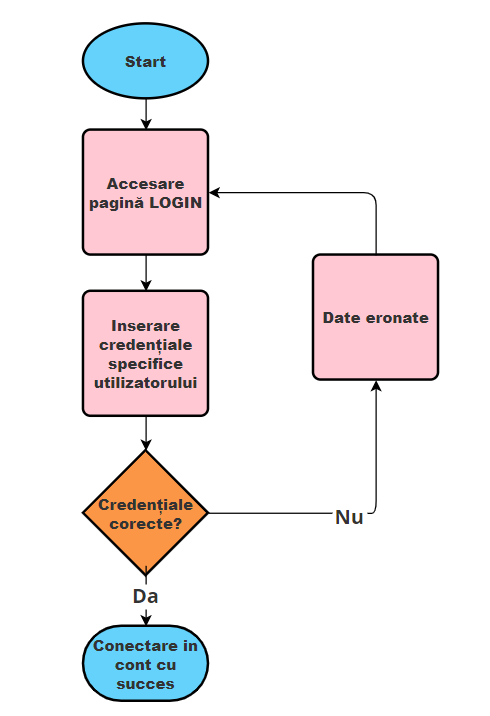
**Postconditii**: inregistrarea utlizatorului respectiv în sistem

**Scenariu favorabil**:

* Accesare pagină login
* Introducere date personale(credențiale proprii)
* Dupa intregistrare, se vor trimite datele catre baza de date
* Autentificare utilizator

**Scenariu nefavorabil**:

Datele menționate de către utilizator conțin unele greșeli și drept urmare va apărea pe ecran un mesaj specific care nu permite trecerea mai departe



##### **Diagrama flow-chart pentru logarea utilizatorilor**

#### **Caz de utilizare 2**

**Numele cazului de utilizare**: Adăugare/Ștergere/Actualizare anunțuri privind job-urile disponibile

**Actorul**: HR Manager

Această parte de adresează personalului HR Manager care va realiza operații de tipul CRUD asupra anunțurilor.

**Precondiții**:

* Logare în sistemul prezent, cu credențialele specifice
* Utilizatorul trebuie să fie în rolul de HR Manager

**Postconditii**: Adăugare/Ștergere/Actualizare/Vizualizare job-uri în sistem

**Scenariu favorabil**:

* Înregistrare utilizator
* Realizare operații CRUD pentru job-uri
* Trimitere date disponibile către sistem

**Scenariu nefavorabil**:

* HR Manager-ul nu va putea transmite datele cu privire la job-uri

Diagram

Description automatically generated

##### **Diagrama flow-chart pentru efectuarea operațiilor CRUD asupra anunțurilor privind job-urile**

#### **Caz de utilizare 3**

**Numele cazului de utilizare**: Accesare tablou cu datele privind aplicanții și realizarea unor serii de operații pentru o evidență clară a fiecărui aplicant

**Actorul**: HR Manager

Această parte de adresează tot personalului HR Manager și are scopul de a obține toate informțiile cu privire la aplicant, dar si realizarea unei statistici

**Precondiții**:

* Logare în sistemul prezent, cu credențialele specifice
* Utilizatorul trebuie să fie în rolul de HR Manager
* Accesare pagină specifică HR Manager-ului

**Postconditii**: Logare pentru rolul de HR Manager

**Scenariu favorabil**:

* Înregistrare utilizator
* Realizare statistici aplicant(acordare notă aplicant în funcție de CV-ul încărcat, realizare diagrama chart in raport cu numărul aplicanților la un anumit job, vizualizare număr total de aplicanți, respinși, acceptați, trimitere email pentru acceptare/respingere )
* Vizualizare date importante aplicanți

**Scenariu nefavorabil**:

* În cazul în care user-ul nu se logheaza cu credențialele de HR Manager, acesta nu va putea avea access la cele menționate mai sus

#### **Caz de utilizare 4**

**Numele cazului de utilizare**: Adăugare candidatură pentru un job dorit

**Actorul**: Aplicant

Această parte de adreseaza aplicantului care va avea la dispoziție un form pentru înregistrarea pentru un post dorit.

**Precondiții**:

* Logare utilizator
* Acest utilizator trebuie să dețina rolul de aplicant
* Accesare pagină pentru alegerea și completarea form-ului de aplicare

**Postconditii**: Crearea unei candidaturi pentru un job

**Scenariu favorabil**:

* Logare utilizator cu credențialele specifice
* Alegere job dorit din lista prezentă
* Trimitere datelor din form-ul asociat, prin apăsarea unui buton care va stoca toate aceste informații în baza de date

**Scenariu nefavorabil**:

Daca utilizatorul nu completează toate câmpurile obligatorii, sistemul ii va afișa un mesaj si drept urmare nu se va putea trece la pasul următor.

Diagram

Description automatically generated

##### **Diagrama flow-chart pentru adăugarea aplicantului a candidaturii pentru un job**

## Tehnologii și concepte utilizate pentru dezvoltarea aplicației web

### Spring Boot

Framework-ul Spring se numără printre cele mai populare framework-uri de tipul open source și se adresează aplicațiilot de tipul Java enterprise.

Acest framework, acceptă operații CRUD (Create, Read, Update, Delete), operații care sunt considerate necesare pentru implementarea oricărei aplicații. Toate aceste operații ajută aplicația pentru a avea o stocare persistentă (orice dispozitiv de stocare a datelor care chiar si dupa ce dispozitivul este oprit, se păstreaza alimentarea).

Seria de pași pentru folosirea acestui framework ar fi:

* Crearea de entități(@Entity)- reprezintă un obiect persistent care ne informează cu privire la faptul că o clasă adnotată este un depozit.
* Crearea de depozite(@Repository)- au rolul de depozita specificația metodelor care vor fi implementate ulterior. Acestă adnotare poate fi de două feluri: CRUD sau JPA
* Crearea de controllere(@Controller)- rol în a acumula implementările operațiilor dorite.
* @Autowired-

Pentru transferul datelor din partea de BackEnd spre partea de FrontEnd se folosesc următoarele adnotații:

* @PostMapping- rol în inserarea anumitor date in baza de date
* @GetMapping- obținem toate datele dintr-un anumit câmp din baza de date
* @PutMapping- realizăm update informațiilor din baza de date
* @DeleteMapping- ștergem date din baza de date

Realizarea legăturilor între tabelele bazei de date:

* Crearea de celor patru tipuri de legături între entități (One-to-One(@OneToOne), One-to-Many(@OneToMany), Many-to-One(@ManyToOne), Many-to-Many(@ManyToMany). Direcția acestor legături poate fi atat unidirectională, cât si bidirectional. Împreună cu aceste tipuri de legături, vom avea nevoie si de atributul mappedBy cu rol de a indică acea proprietate de acolo de unde este referită.

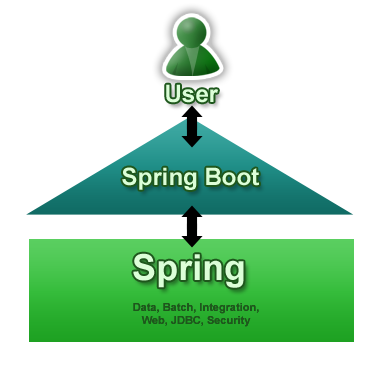
Pentru a asigura o persistență între legăturile dintre date și obiecte, vom folosi și adnotările @Id (realizează cheia primară), @Column (mapează un câmp din tabelă, cu o anumită proprietate a unei clase), @Query (scop în asigurarea implementării unei interogări).

@Entity – are rolul de anunța framework-ul Spring că prezenta clasă este asociată unei anumite tabele din baza de date.

@GeneratedValue- rol in stabilirea ordinii generării de id-uri în cadrul cheii primare

@Column- asocierea unui câmp al clasei către o coloană din tabelă

Integrarea cu ajutorul JPA va crea funcționalități mai puternice oricărei aplicații.



### React JS

După cum se specifică si în [7], React JS este o bibliotecă din categoria JavaScript de tipul open source. Cu ajutorul acestei biblioteci, se facilitează întregul proces de construție de interfețe utilizator, interactive. În cadrul acestei biblioteci, se regaseste și componenta MVC(Model-View-Controller). Acest framework este printre cele mai folosite până în present și împletește cu ușurință performanța, complexitatea, dar și accesibilitatea.

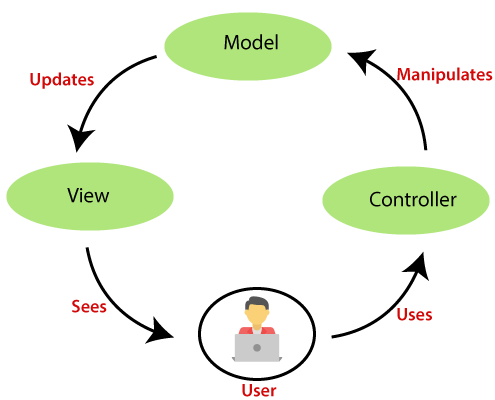
Printre caracteristicile care sunt de remarcat se numără:

* Componente- întregul cod este format din entități numite componente. Toate aceste componente pot fi redate pe un anumit element din DOM, cu ajutorul bibliotecii React DOM
* DOM virtual- model de obiect de document virtual (DOM). Cu ajutorul acestei componente, React va creea pentru structura de date, in memorie, o memorie cache.

Graphical user interface, text

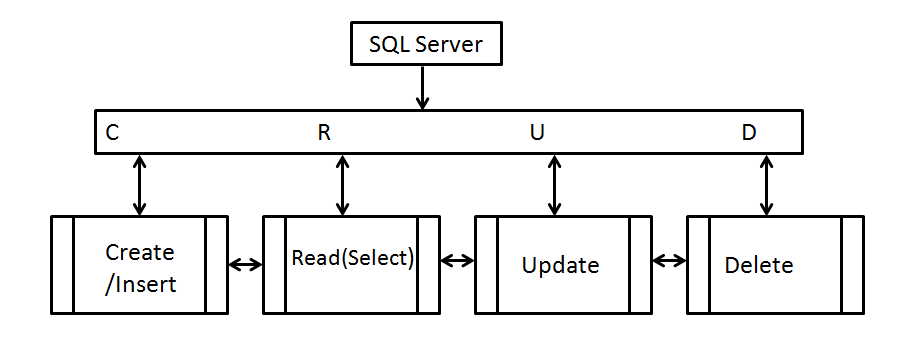
Description automatically generated

Arhitectura React, bazată pe servicii și componente?



### SQL Workbench

Baza de date este una importantă, deoarece cu ajutorul acesteia vom stoca toate informațiile și datele specifice aplicației realizate. Datele prezente in bazele de date sunt organizate in rânduri, coloane, tabele, fiind indexate pentru facilitarea găsirii informațiilor. SQL Workbench este o bază de date și un instrument pentru toți ahitecții de baze de date, dezvoltatori și DBA. Acest instrument ajută la configurarea serverului, administrarea utilizatorilor, backup, fiind disponibil pe toate sistemele de operare.



#### Operațiile CRUD efectuate de către SQL Server

### Bootstrap

Bootstrap reprezintă un framework de tipul open source și constituie o componentă importantă in crearea atât a site-urilor web, cât si a aplicațiilor web. Acest framework este construit pe HTML, CSS si JavaScript(JS) și are ca scop facilitarea dezvoltării de site-uri și aplicații. Design-ul responsive face posibilă ca o pagină web sau o aplicație să detecteze dimensiunea si orientarea ecranului și totodată să adapteze automat afișarea.

Principalele avantaje care le constituie acest framework ar fi:

* Utilizare ușoară și personalizată
* Folosirea mai multor plugin-uri JavaScript care folosesc JQuery
* Un framework care acceptă majoritatea browser-elor, dar și soluțiile de compatibilitate CSS

### JSON- JavaScript Objectiv Notation

JSON [17] este un format care este apropiat de codul care se folosește pentru crearea obiectelor JavaScript. Fiindcă formatul acestuia este unul exclusiv text, datele pot fi transmise între pc și totodată pot si utilizate de orixce fel de limbaj de programare.

Text

Description automatically generated

##### Exemplu de interogare JSON

# Proiectare de Detaliu și Implementare

Întreaga aplicație a fost realizată din cele două părți mari: atât front-end, cât și back-end. Fiecare dintre aceste doua părți, la rândul lor au fost realizate cu ajutorul mai multor tehnologii și module. În cele ce urmează, voi descrie, pe rând părțile importante din cod, dar și unele componente.

## Descrierea ficărei pagini prezente în aplicația web

### Pagina APLICANT

După ce fiecare aplicant se va înregistra cu propriile credențiale, va avea la dispoziție o pagină care va conține o listă cu toate posturile disponibile din firmă, impreuna cu data scadenta depunerii cererilor. După selectarea postului dorit, utilizatorul va fi direcționat catre o pagină în care va avea posibilitatea sa completeze un formular cu privire la datele personale, impreuna cu CV-ul.

În cazul in care utilizatorul va fi acceptat, va primi un mail de confirmare cu data si ora interviului, link catre interviu.

### Pagina HR Manager

Pagina principală destinată HR-ului va conține diferite opțiuni pe care le are cu redirecționare către pagina specifică opțiunii alese.

Primul tab, “Dashboard” a acestei pagini este împărțit după cum urmează:

* Componente in care ne sunt contorizate detaliile privind numărul aplicanților acceptați, respinși, dar si numărul total al candidaților.
* Grafic care ne prezintă numărul total de candidați pentru un anumit job
* Tabel cu fiecare aplicant, butoane de admis/respins care vor si trimite email candidatului, dar și o parte în care HR Manager-ul acordă o notă fiecărui candidat, în funcție de CV-ul încărcat de fiecare
* Tabel cu aplicanții și notele obținute de către aceștia

Al doilea tab este alcătuit dintr-un tabel care conține următoarele detalii:

* Numele aplicantului
* Email-ul la care poate fi contactat
* Număr de telefon
* Job-ul la care a aplicat
* Calificativul de admis/respins

Al treilea tab, va contine toate posturile disponibile, împreuna cu data până la care se poate aplica. HR Manager-ul va putea face operațiile CRUD (Create, Read, Update, Delete) pentru fiecare anunț.

Al patrulea tab, va conține o retea OETPN, în care pentru fiecare user ni se va ilustra câți pași s-au realizat din tot procesul de recrutare. Aici vom vedea unde anume a ajuns jetonul, în funcție de pașii efectuați.

## Proiectarea bazei de date

### Tabela User

În interiorul acestei tabele se colectează toate informațiile cu privire la fiecare utilizator, iar mai apoi sunt salvate în baza de date.

Graphical user interface, text, application

Description automatically generated

### Tabela Email

Această tabelă este responsabilă de trimiterea email-urilor pentru cazurile de admis/respins către aplicanți

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

Description automatically generated

### Tabel UserJob

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

Description automatically generated

### Tabela Form

Scopul acestei tabele este acela de a stoca întreaga aplicare a tuturor userilor

Text

Description automatically generated

### Tabela FileInfo

Folosind aceasta tabelă, vom memora fiecare CV încărcat de către aplicanți

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

Description automatically generated

### Tabela Job

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

Description automatically generated

În realizarea acestei baze de date, s-a dorit abordarea datelor cât se poate de corect pentru a evita cazul în care informațiile ajung eronate sau se pierd.

Pentru realizarea operațiilor CRUD asupra bazei de date, am folosit metodele definite de interfața JpaRepository. Pentru inceput, am creat șase clase DTO(Data Transfer Object) după cum urmează: Email( trimitere email), FileInfo( incărcare fișiere), Form( formular de înscriere pentru un job), Job(memorarea numelui și id-ului fiecărui job), User(tabel pentru datele de înregistrare cont), userJob(legătura între formular si id-ul jobului ales).

În primul rând, am creat două clase – model, aferente entităților din baza de date (Position și User), fiecare având atributele necesare. Aceste atribute au, la rândul lor, adnotări specifice pachetelor Swagger, Jackson și JavaX, identificând entitatea (*@Entity*), id-ul (*@Id*), câmpuri care trebuie ignorate de JSON (*@JsonIgnore*) sau nenule (*@NotNull*).

Mai apoi, am creat cinci interfețe (EmailRepositor, FormRepository, JobRepository, UserJobRepository, UserRepository), având adnotarea *@Repository*, care extind JpaRepository, cu primul argument aferent entității, al doilea fiind tipul de date al ID-ului. Aici am apelat anumite metode pentru a realiza operațiile de interes asupra bazei de date, fără a scrie manual interogările. Aceste metode sunt create automat, depinzând de atributele modelului folosit.

Au fost create și cinci servicii (PositionService și UserService) cu adnotarea *@Service*, care folosește o instanță a interfețelor definite anterior. Se folosesc aceleași nume de metode, care returnează rezultatul metodelor aferente din Repository. De asemenea, am făcut anumite verificări, precum dacă data inițială sau finală este nulă să apeleze diferite metode (de exemplu dacă data finală este nulă, să se apeleze metoda care returnează lista de poziții a unui utilizator, de la data inițială, fără o limită finală).

Pentru ca un client să știe unde să trimită datele și să poată primi informații, au fost implementate două controllere (PositionController și UserController), adnotarea *@Controller*, folosind de asemenea și *@RequestMapping*, pentru a știi pe ce rută trebuie să ajungă datele. Fiecare metodă are o anumită mapare individuală, folosind parametrii primiți, și returnând rezultatele interogărilor. Am folosit și adnotările *@ResponseBody*, pentru a identifica răspunsul, sau *@PathVariable*, pentru a știi care argumente ale metodei se folosesc în mapare.

În fișierul „application.properties” am scris setările necesare conexiunii server-ului la baza de date, precum port-ul folosit, numele bazei de date, driver-ul folosit.

# Testare și validare

## Rezultate obținute

Evidențiați toate rezultatele pe care le-ați obținut și trageți concluzii din ele. Puteți prezenta o analiză critică a ceea ce ați realizat comparativ cu alte lucrări/studii anterioare.

Includeți o listă a contribuțiilor pe care le-ați avut în domeniul temei abordate.

## Direcții de dezvoltare

Descrieți direcțiile posibile de dezvoltare.

# Manual de Instalare și Utilizare

Acest capitol este predestinat detaliilor ce privesc instalarea și cum anume se va folosi aplicația dezvoltată. Tot aici sunt prezenți și pașii care trebuie parcurși pentru instalarea tuturor componentelor sistemului împreună cu manualul de utilizare a aplicației.

## Instalarea tuturor resurselor necesare aplicației

Aplicația descrisă în capitolele precedente, este dezvoltată pe partea de back-end în IntelliJ IDEA, o aplicație pentru limbajul Java. Pentru partea de front-end, am folosit limbajul de programare REACT JS, iar pentru stocarea integrală a datelor am folosit SQL WORKBENCH( s-a creat automat cu ajutorul codului din IntelliJ).

* Instalare IntelliJ IDEA [18]
* Se va descărca fișierul de instalare al Java IDE ca Java IDE, de la adresa https://www.jetbrains.com/idea/download/#section=windows
* Se va parcurge întregul fișier de instalare după ce s-a ales opțiunea Run as Administrator
* Activare licență IntelliJ pe baza unui JetBrains account
* După finalizarea tuturor pașilor, se va alege opțiunea de Lunch
* Instalare Visual Studio Code (pentru ReactJS)
* Se descarcă fișierul sursă
* Se parcurg toți pașii spre instalare
* După instalare, ni se va deschide IDE-ul
* Instalare SQL Workbench
* Decărcare sursă de pe site-ul mysql.com/products/workbench
* Dupa descărcare și executare, va fi necesar sa configurăm portul pe care dorim sa lucram, un username și o parolă. Acestea ne vor fi necesare mereu atunci când vom folosi bază de date
* După toate configurarile, vom putea accesa cu ușurință baza de date

## Manual de utilizare

### Înregistrare și logare în aplicație

Graphical user interface

Description automatically generated with medium confidence

#### Înregistrare user nou în sistem

Graphical user interface, application

Description automatically generated

#### Pagină login

### Detalierea funcționalităților

Graphical user interface

Description automatically generated

#### Vizualizare status al aplicantului

În figura de mai sus este reprezentată pagina în care HR Manager-ul are posibilitatea de a căuta un aplicant, iar mai apoi pe rețeaua OETPN să vizualizeze câți pași a reușit să realizeze aplicantul. Tot acest progres ne va fi afișat cu ajutorul jetoanelor în locațiile specifice.

# Concluzii

# Reguli de formatare

## Formatarea paginii

* + Dimensiunea paginii: A4
  + Margini: 2.5 cm (sus, jos, stânga, dreapta)
  + Antet și subsol: 1.27 cm de la marginea paginii
  + În antetul paginii (header): titlul capitolului, centrat, stil: Header\_style
  + În subsolul paginii: numărul paginii, centrat

## Titluri și stiluri

Titlurile capitolelor și subcapitolelor se marchează cu stilurile Heading 1 – 4, conform documentului model anexat în format Word. Descrierea stilurilor utilizate în document este prezentată în Tabelul 5.1.

Tabelul 5.1. Stiluri utilizate în acest document

| Nr. | Stil | Utilizat pentru | Format |
| --- | --- | --- | --- |
| 1 | Normal | Text normal | Font: (Default) Cambria, 12 pt, Justified, Line spacing: Multiple 1.1 li, Space After: 6 pt |
| 2 | Titlu | Titlul proiectului, prima pagină | Font: 24 pt, Small caps, Centered Line spacing: single, Space Before: 126pt, After: 0 pt, |
| 3 | Titlu2 | Titlul proiectului, pagina de prezentare | Font:14pt, Bold, Centered |
| 4 | Heading 1 | Titlurile capitolelor (nivel 1) | Font: 24 pt, Indent: Left: 0 cm Hanging: 0.76 cm, Space Before: 24pt, After: 12pt |
| 5 | Heading 2 | Titlurile subcapitolelor (nivel 2) | Font: 14 pt, Bold, Indent: Left: 0 cm  Hanging: 1.02 cm, Space Before: 18pt, After: 12pt |
| 6 | Heading 3 | Titlurile secțiunilor (nivel 3) | Font: Bold, Indent: Left: 0 cm Hanging: 1.27 cm, Space Before: 6 pt, After: 6pt |
| 7 | Heading 4 | Titlurile secțiunilor (nivel 4) | Font: Italic, Indent: Left: 0 cm Hanging: 1.52 cm, Space Before: 2 pt, After: 0 pt |
| 8 | Caption | Legenda figurilor și tabelelor | Font: Italic, Font color: Text 1, Line spacing: single, Space After: 10 pt, |
| 9 | Header\_style | Antetul paginii | Font: 10 pt, Italic, Centered, Border: Bottom: (Single solid line, Background 1, 0.5 pt Line width) |

## Figuri, tabele și ecuații

### Figuri

Figurile se inserează în text centrate, cu etichetă de numerotare și legendă (Caption) în partea de jos a figurii. Numărul figurii include și numărul capitolului, după exemplul prezentat în Figura 5.1.



Figura 5.1. Figură exemplu, stil: Caption

## Tabele

Tabelele se inserează în text centrate, cu etichetă și legendă (Caption) în partea de sus a tabelului, aliniată la stânga. Numărul tabelului include și numărul capitolului, după cum este prezentat, de exemplu, în Tabelul 5.1.

## Ecuații

Ecuațiile se inserează în text centrate, cu numerotare în partea dreaptă. Numărul ecuației include și numărul capitolului, conform exemplului din relația (5.1).

|  |  |
| --- | --- |
|  | (5.1) |

## Referințe bibliografice

Se recomandă ca citarea referințelor bibliografice să fie făcută în formatul IEEE.

În secțiunea Bibliografie sunt prezentate exemple pentru: o citare a unui capitol dintr-o carte [1], un articol publicat într-o revistă [2] și un articol publicat la o conferință [3].

Detalii cu privire la formatul citării diverselor tipuri de referințe pot fi găsite în [4] sau [5].

Referințele bibliografice se pot insera în text utilizând facilitățile Word de a adăuga surse și bibliografie unui document (References -> Citations & Bibliography). Dacă formatul IEEE pentru bibliografie nu este instalat implicit în Word, se poate descărca gratuit de la:

<https://bibword.codeplex.com/wikipage?title=Styles&referringTitle=Home>

Instrucțiunile de instalare pentru diferite versiuni de Word se pot obține de la aceeași adresă.

# Bibliografie

|  |  |
| --- | --- |
| [1] | P. Nume, "Titlul capitolului," în *Titlul cartii*, Oras, Editura, 2016, pp. 1-24. |
| [2] | P. Nume, "Titlul articolului," *Titlul revistei,* vol. 1, no. 2, pp. 22-30, 2016. |
| [3] | P. Nume, "Titlul articolului," în *Numele conferintei*, Oras, 2015. |
| [4] | "IEEE Citation Reference," 2009. [Online]. Available: https://www.ieee.org/documents/ieeecitationref.pdf. |
| [5] | "IEEE Editorial Style Manual," 2016. [Online]. Available: https://www.ieee.org/documents/style\_manual.pdf.  [6] https://www.sumologic.com/glossary/crud/  https://ro.education-wiki.com/9050114-what-is-react  [7] https://www.tutorialspoint.com/reactjs/  [8]https://koaha.org/wiki/React\_(web\_framework)  [9] https://www.mysql.com/products/workbench/  [10] Tiberiu S. Leția, Dahlia Al- Janabi- The Development of Complex Data Structures Using Object Enhanced Time Petri Nets, IEEE International Conference on System Theory, Control and Computing 2018  [11] T.Leția- Programarea concurentă în limbajul Java standard  [12] • Leonard Richardson, RESTful Web Services, O'Reilly, 2007  [12] Leonard Richardson, RESTful Web Services, O'Reilly, 2007  [13] https://www.todaysoftmag.ro/article/81/restful-web-services-folosind-jersey  [14] B. S. Alboaie Lenuța, Servicii Web: concepte de bază și implementări, Iași: Editura POLIROM, 2006.  [15] Borcan Dumitru - Marius - Integrarea Spring cu Java Persistence API.pdf  [16] Programarea concurentă în limbajul Java standard- Tiberiu Leția  [17] https://www.w3schools.com/js/js\_json\_intro.asp  [18] http://www.cs.ubbcluj.ro/~sergiu/MAP/ |