**JT HR**

**Admin**

**Echipa *ULBS20*:**

* **Bugner Ioana**
* **Cozma Alexandru – Cerinte personale**
* **Ghip Sabrina**

**Informatizarea procesului de recrutare**

**Echipa: ULBS20**

**Membrii:**

- Bugner Ioana (Gr. 231/2)

- Cozma Alexandru (Gr. 231/3) – Cerinte personale

- Ghip Sabrina (Gr. 231/1)

1. **Introducere**
2. **Cerinte functionale**

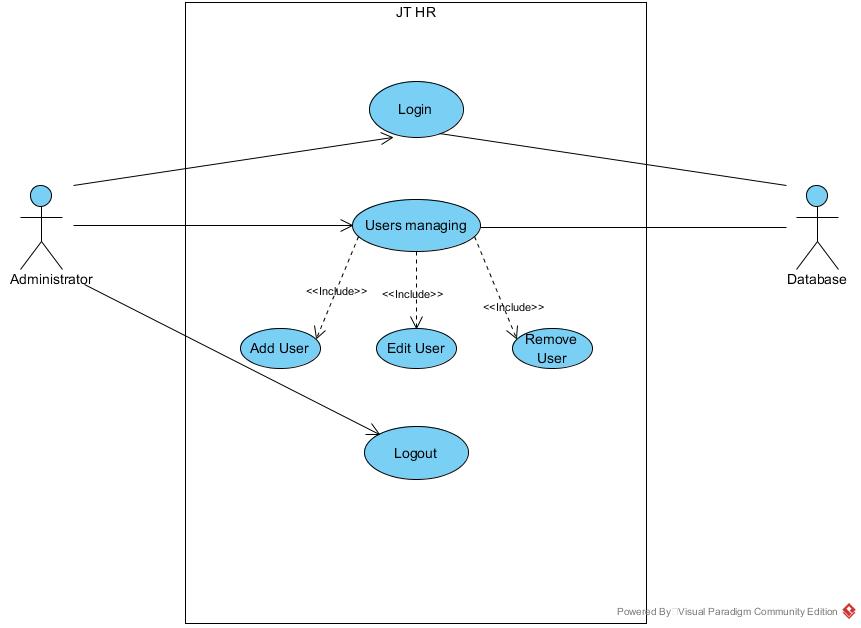
**Operatii CRUD implementate**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Rol**  **Operatie** | D. General | D. Dep. | D. HR | Recruiter | Admin | Viewer |
| Add user | x |  |  |  | x |  |
| Edit user | x |  |  |  | x |  |
| Delete user | x |  |  |  | x |  |
| Add pozitie | x | x | x | X |  |  |
| Edit pozitie | X | x | x | x |  |  |
| Close pozitie |  |  |  |  |  |  |

**Membru doi – Cozma Alexandru**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Rol**  **Operatie** | D. General | D. Dep. | D. HR | Recruiter | Admin | Viewer |
| Add user | X |  |  |  | x |  |
| Edit user | X |  |  |  | x |  |
| Delete user | X |  |  |  | x |  |
| Add pozitie | X | x | x | X |  |  |
| Edit pozitie | X | x | x | X |  |  |
| Close pozitie |  |  |  |  |  |  |

1. **Diagrame Use-Case**



**Use-Case List:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID Autor + nume** | **Rol** | **Use-Case ID** | **Use-Case Name** |
| **2** –Bugner Ioana | **Administrator** | **UC1** | **Login** |
| **1** - Cozma Alexandru | **UC2** | **Add User** |
| **3** - Ghip Sabrina | **UC3** | **Edit User** |
| **1** - Cozma Alexandru | **UC4** | **Remove User** |
| **2** –Bugner Ioana | **UC5** | **Logout** |

**User Management:**

**3.3 Use-Case: Add User**

|  |  |
| --- | --- |
| **Use Case ID:** | UC2 |
| **Use Case Name:** | Add User |
| **Created By:** | Cozma Alexandru |
| **Date Created:** | January 18, 2016 |
| **Actors:** | Administrator |
| **Description:** | To ensure the possibility to add a new user in the database |
| **Trigger:** | There is a need/demand for a new user |
| **Preconditions:** | NA |
| **Postconditions:** | NA |
| **Normal Flow:** | Administrator choose option to add new user  JT HR displays a form with the required user information  Admin completes the last and the first name, mobile phone no, email address (mandatory fields)  Admin completes the user position, description and the list of roles  JT HR autocompletes the username based on the last and first name  Admin. Selects to add the user with the entered information  JT HR validates the given information  JT HR creates the user and adds it in the database  JT HR restores the user management menu |
| **Alternative Flows:** | NA |
| **Exceptions:** | 7a. in the step 7 of the normal flow, if one of the fields from step 3 weren’t completed:  JT HR displays a warning message which tells that one or more mandatory fields weren’t complete  Use case resumes from step 3  7b. in the step 7 of the normal flow, if the email address and/or phone number is/are no valid:  JT HR displays a warning message which tells that email and/or phone number is/are not valid  Use case resumes from step 3  7c. in the step 7 of the normal flow, if a user with the same username(generated) already exists:  JT HR displays a warning message which tells that a user with the same username already exists |
| **Includes:** | NA |
| **Frequency of Use:** |  |
| **Special Requirements:** | NA |
| **Assumptions:** | NA |
| **Notes and Issues:** | What’s the best way to generate the username from last and first name? |

* 1. **User-Case: Remove User**

|  |  |
| --- | --- |
| **Use Case ID:** | UC4 |
| **Use Case Name:** | Remove User |
| **Created By:** | Cozma Alexandru |
| **Date Created:** | January 18, 2016 |
| **Actors:** | Administrator |
| **Description:** | To ensure the possibility to remove a user from the data base |
| **Trigger:** | There is a need/demand to remove user |
| **Preconditions:** | There must be at least a user in the data base |
| **Postconditions:** | The removed user must not appear in the database any more |
| **Normal Flow:** | Administrator choose option to remove new user  JT HR displays the list of all users  Admin selects the user wanted to be removed and clicks the remove button  JT RH prompts the Admin for confirmation of the operation  Admin. accepts  JT HR removes the user from the database  JT HR restores the user management menu |
| **Alternative Flows:** | 5a. in step 5 of the normal flow, if Admin choose to cancel operation:  Use case resumes at step 2 |
| **Exceptions:** | NA |
| **Includes:** | NA |
| **Frequency of Use:** |  |
| **Special Requirements:** | NA |
| **Assumptions:** | NA |
| **Notes and Issues:** |  |

1. **Modelul de date si alte informatii**

**Fisierele JSP (Java Server Pages)**

**Fisierele JSP realizate personal:**

* **addUser.jsp**
  + Afisarea interfetei corespunzatoare pentru adaugarea unui nou utilizator pe pagina web
* **index.jsp**
  + Afisarea paginii principale “home”
* **login.jsp**
  + Asigura conectarea si pastrarea autentificarii prin pornirea unei sesiuni
* **menu.jsp**
  + Afisarea meniului pe pagina web

**Servleti**

**Fisiere Servlet realizate personal**

* Adaugare utilizator (AddUserServlet.java)
* Autentificarea utilizatorilor (LoginServlet.java)

**Validatori**

**Validator Numar de telefon**

“PhoneValidator.java” respectarea pattern-ului a unei expresii regulate, precum numărul telefonului: - stringul trebuie format doar din cifre si de lungima maxima 10.

**Sesiune – variabile de sesiune (Autentificare)**

Pentru ca dupa user-ul este autentificat (introducand username-ul si parola) si sa ramana autentificat cat timp sesiunea browserului folosit e deschisa (sau pana cand decide sa se delogheze), se seteaza o variabila booleana cu valoarea true si se salveaza ca atribut de sesiune.

Aceasta variabila este verificata in modului de menu.jsp prin intermediul careia sunt accesibile paginile dedicate user-ului logat.

Pentru a inchide sesiunea in care user-ul este "logged in" variabila de sesiune se seteaza pe false, astfel paginile ptr adaugare si editare/stergere devin inaccesibile.

Se mai folosec doua variabile de sesiune pentru semnalarea finalizarea operatiunii de adaugare sau de editare. Acestea sunt setate cu valoarea true imediat dupa ce operatiunea a fost validata, data spre executie si inainte de redirectare catre pagina principala.

In pagina principala se verifica, se anunta utilizatorul printr-un mesaj despre executia operatiunii si se reseteaza cu valoarea false.

**Conectarea la Baza de Date**

Pentru conectarea la baza de date MySQL s-a implementat clasa *“SQLdb”*

Membri:

final String db\_driver;

final String db\_url;

final String db\_user;

final String db\_password;

Connection conn;

Statement statem;

PreparedStatement ps;

ResultSet result;

Metode:

public void Connect() – Pentru conectare

* + Se face instantarea driver-ului de MySql
  + Se incearca realizarea conexiunii prin apelarea functiei ***DriverManager.getConnection(url, user, password);***
  + Se creaza statement-ul conexiunii

public void executeSQL(String sql) – Pentru trimiterea unei cereri catre baza de date

* + Se verifica daca este o cerere de “update”, “insert” sau “delete”
    - In acest caz se pregateste statement-ul apoi se executa
  + Daca este un simplu *“query”* se executa cererea care returneaza rezultatul in membru *“result”*

**DAO Pattern (Data Access Object)** (+ Ghip Sabrina)

“UserDao.java”

DAO este un obiect care oferă o interfață abstractă pentru un anumit tip de baze de date sau alt mecanism de persistență . Prin cererea de cartografiere cheamă la stratul de persistența , DAO oferă unele operațiuni specifice de date , fără a expune detalii ale bazei de date.

In aceasta aplicatie nu am respectat in totalitate pattern-ul de DAO intrucat nu am implementat interfata abstracta.