**JT HR**

**Admin**

**Echipa *ULBS20*:**

* **Bugner Ioana**
* **Cozma Alexandru**
* **Ghip Sabrina**

**Informatizarea procesului de recrutare**

**Echipa: ULBS20**

**Membrii:**

- Bugner Ioana (Gr. 231/2)

- Cozma Alexandru (Gr. 231/3)

- 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 unu – Bugner Ioana**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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 |  |  |  |  |  |  |

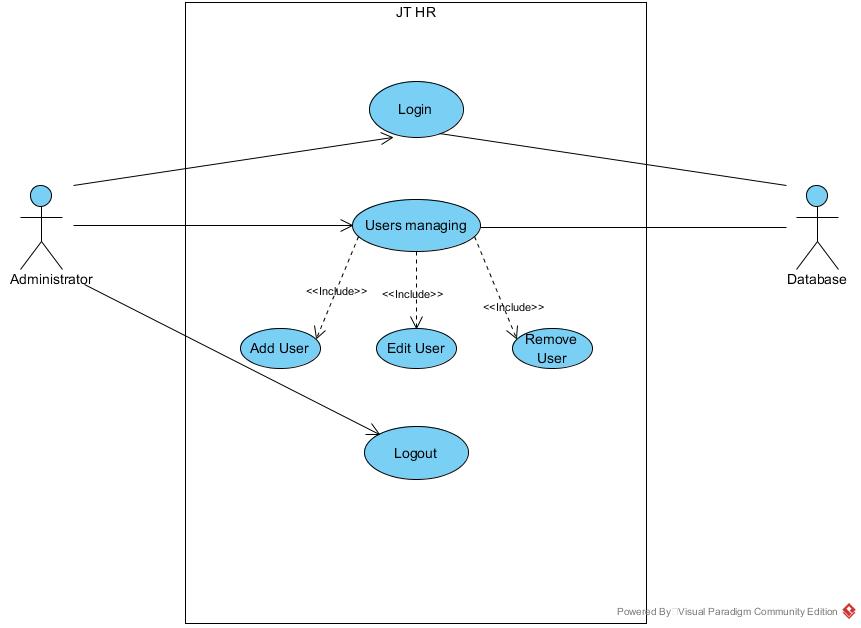
**Membru doi – Cozma Alexandru**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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 |  |  |  |  |  |  |

**Membru trei – Ghip Sabrina**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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** |

**Authentication: Login and Logout**

* 1. **Use-Case: Login**

|  |  |
| --- | --- |
| **Use Case ID:** | UC1 |
| **Use Case Name:** | Login |
| **Created By:** | Bugner Ioana |
| **Date Created:** | January 18, 2016 |
| **Actors:** | Admin |
| **Description:** |  |
| **Trigger:** |  |
| **Preconditions:** | Administrator User account exists  Administration rights are configured |
| **Postconditions:** | NA |
| **Normal Flow:** | Administrator enters username and password  JT HR validates if username and password are correct  JT HR displays user management menu  Administrator does his job |
| **Alternative Flows:** | NA |
| **Exceptions:** | 2a. In step 2 of the normal flow, if the username is not in the database  JT RH displays a message which tells that there is no user with that username  2b. In step 2 of the normal flow, if the password is not valid  JT RH displays a message which tells that the inserted password is incorrect  Administrator retry login or quits |
| **Includes:** | NA |
| **Frequency of Use:** |  |
| **Special Requirements:** |  |
| **Assumptions:** | The Administrator has a default user and password that exist in the database since the implementation of the functionality |
| **Notes and Issues:** | Should admin use the same login interface as other users or a separate one |

* 1. **Use-Case: Logout**

|  |  |
| --- | --- |
| **Use Case ID:** | UC5 |
| **Use Case UC5:** | Logout |
| **Created By:** | Bugner Ioana |
| **Date Created:** | January 18, 2016 |
| **Actors:** | Admin |
| **Description:** |  |
| **Trigger:** |  |
| **Preconditions:** | The administrator must be logged in |
| **Postconditions:** | Can’t return to user management menu unless the admin. login again |
| **Normal Flow:** | 1. Administrator clicks on ‘’Logout’’ button  2. JT RH closes the user management menu  3. JT RH displays normal viewer’s interface |
| **Alternative Flows:** | NA |
| **Exceptions:** | NA |
| **Includes:** | NA |
| **Frequency of Use:** |  |
| **Special Requirements:** |  |
| **Assumptions:** | NA |
| **Notes and Issues:** |  |

**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. **Use-Case: Edit User**

|  |  |
| --- | --- |
| **Use Case ID:** | UC3 |
| **Use Case Name:** | Edit User |
| **Created By:** | Ghip Sabrina |
| **Date Created:** | January 18, 2016 |
| **Actors:** | Administrator |
| **Description:** | To ensure the possibility to modify(update) user information in the database |
| **Trigger:** | There is a need/demand for an update to a user information |
| **Preconditions:** | There must be at least a user in the database |
| **Postconditions:** |  |
| **Normal Flow:** | Administrator choose option to edit user  JT HR displays the list of all users  Admin selects the user wanted to be edit and clicks the edit button  JT HR displays a editable form with the user information  Admin edits the last and the first name,(or) mobile phone no,(or) email address (mandatory fields)  Admin completes the user position, (or) description, (or) the list of roles  JT HR autocompletes the username based on the last and first name (if first and/or last name were modify)  Admin. Selects to update the user with the information from the form  JT HR validates the given information  JT HR updates the user in the database  JT HR restores the user management menu |
| **Alternative Flows:** | NA |
| **Exceptions:** | 7a. 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 5  7b. 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 email and/or phone number is/are not valid  Use case resumes from step 5 |
| **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**

**4.1 Descrierea tabelei din baza de date**

**Nume tabela: „jthr\_admin“**

**Atributele tabelei:**

* **„userid“**  - reprezinta un id unic, *“autoincrement”* care ramane ascuns pe parcursul rularii aplicatiei, avand ca scop accesarea din spatele codului a tuplurilor din baza de date
* **„username“** – este de asemenea un atribut unic la un moment de timp, format din primele 5 litere ale “firstname-ului” (sau intregul camp atunci cand acesta are o lungime mai mica de 5) si prima litera din *“lastname”.*  In cazul in care *“username-ul”* exista pentru o altă inregistrare el va continua sa adauge litere din campul *“lastname”* pana va devein unic.
* **„firstname“** – numele de familie al utilizatorului
* **„lastname“** – prenumele utilizatorului
* **„phone“**
* **„cellphone“**
* **„userfunction“** – functia utilizatorului
* **„rolelist“** – lista de roluri pe care trebuie sa o indeplineasca, dupa caz
* **„description“** – descrierea in amanunt a functiei
* **„password“** – parola utilizatorului care la inceput va fi formata din concatenearea *“username-ului”* cu anul prezent “2016”

**4.2 Fisierele JSP (Java Server Pages)**

**Java Server Pages**este o simpla dar puternica tehnologie folosita pe partea de server pentru a genera continut HTML dinamic. JSP este o extensie directa aJava Servletssi furnizeaza o modalitate de a separa partea de procesare de cea de prezentare. Motorul JSP este doar un altServlet, mapat la extensia„\*.jsp”.

O pagina JSP poate crea si/sau accesa anumite obiecte Java pentru a procesa o anumita cerere.

Asemenea aplicatiei prezente, JSP-urile sunt folosite in aproape toate aplicatiile “Java Enterprise”.

**Directive principale utilizate in fisierele JSP ale aplicatiei prezente:**

* + “import” - importa anumite clase precum “User.java” sau diferite pachete
  + “contentType” – specifica setul de caractere de raspuns

In aproape toate fisierele JSP (exceptie login.jsp si logout.jsp) sunt incluse fisierele \*.css corestunzatoare pentru stilul de afisare pe paginile web si fisierul “menu.jsp” care include partea de afisare a meniului.

**Fisierele JSP prezente in aceasta aplicatie sunt:**

* **AllUsers.jsp** 
  + Afisarea listei de utilizatori din baza de date intr-un tabel pe pagina web
* **addUser.jsp**
  + Afisarea interfetei corespunzatoare pentru adaugarea unui nou utilizator pe pagina web
* **editUser.jsp**
  + Afisarea interfetei corespunzatoare pentru editarea unui anumit utilizator pe pagina web
* **index.jsp**
  + Afisarea paginii principale “home”
* **login.jsp**
  + Asigura conectarea si pastrarea autentificarii prin pornirea unei sesiuni
* **logout.jsp**
  + Distruge sesiune de autentificare
* **menu.jsp**
  + Afisarea meniului pe pagina web

**4.3 Servleti**

Un servlet este o clasã Java care primeste si interpreteazã cererile HTTP primite de la browser, dupã care transmite un rãspuns sub forma unei pagini HTML cãtre browser.

**Metodelor folosite din clasele de servlet pentru o cerere HTTP in aplicatia noastra:**

- protected void **processRequest** (HttpServletRequest, HttpServletResponse);

- protected void **doGet**(HttpServletRequest, HttpServletResponse);

- protected void **doPost**(HttpServletRequest, HttpServletResponse);

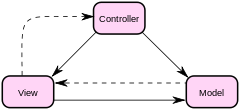
**Fiecare dintre aceste clase au fost adaptate dupa caz:**

* Adaugare utilizator (AddUserServlet.java)
* Afisarea tuturor utilizatorilor (AllUsersServlet.java)
* Editarea unui utilizator (EditUserServlet.java)
* Stergerea unui utilizator (RemoveUserServlet.java)
* Autentificarea utilizatorilor (LoginServlet.java)

**4.4 Alte fisiere si specificatii**

**4.4.1 MVC Pattern (Model-View-Controller) – business layer**

S-a respectat **pattern-ul MVC** pentru izolarea **logicii de business** fata de considerentele interfetei cu utilizatorul, rezultându-se astfel o aplicație unde aspectul vizual sau/și nivelele inferioare ale regulilor de business sunt mai ușor de modificat, fără a afecta alte nivele.



**4.4.1.1 Modelul -** manipulează operațiunile logice și de utilizare de informație

intr-o forma usor de inteles.

Fisierul **“User.java”** realizeaza astfel aceasta manipulare prin clasa **“User”** cu:

**Membrii:**

* private int **id**;
* private String **username**;
* private String **password**;
* private String **firstName**;
* private String **lastName**;
* private String **phone**;
* private String **cellPhone**;
* private String **email**;
* private String **userFunction**;
* private String **roleList**;
* private String **description**;

**Constructorii si metodele:**

* public **User**(int id,String username,String password,String firstName,

String lastName,String phone,String cellPhone,String email,

String userFunction,String roleList,String description);

* public **User**(String firstName, String lastName, String phone, String cellPhone, String email,

String userFunction, String roleList, String description);

* public int **getId**();
* public void **setId**(int id) ;
* public String **getUsername**() ;
* public void **setUsername**(String username) ;
* public String **getPassword**() ;
* public void **setPassword**(String password) ;
* public String **getFirstName**() ;
* public void **setFirstName**(String firstName);
* public String **getLastName**();
* public void **setLastName**(String lastName)
* public String **getPhone**()
* public void **setPhone**(String phone)
* public String **getCellPhone**()
* public void **setCellPhone**(String cellPhone)
* public String **getEmail**()
* public void **setEmail**(String email)
* public String **getUserFunction**()
* public void **setUserFunction**(String userFunction)
* public String **getRoleList**()
* public void **setRoleList**(String roleList)
* public String **getDescription**()
* public void **setDescription**(String description)

**4.4.1.2 View** – reprezinta interfața grafică ce interacționează cu utilizatorul final

In aceasta aplicatie este realizat prin fisierele ***\*. JSP*** descrise si prezentate mai sus.

**4.4.1.3 Controller** –ne permite a controla accesul la aplicația noastră

Fisierele din cadrul aplicatiei nostre responsabile cu acest lucru sunt fisierele ***\*Servlet.java***, care au fost prezentate mai sus.

**Conectarea la baza de data MySQL** “SQLdb.java”

* realizeaza conectarea la baza de date
* inchiderea conexiunii spre baza de date
* trimiterea si primirea cereriilor spre/dinspre baza de date

Metode:

* public void Connect();
* public void closeConnection();
* public void executeSQL(String);
  + string-ul reprezinta cererea trimisa bazei de date pentru executare
* public ResultSet getResult();
  + salveaza rezultatul trimit de catre baza de date intr-un membru de tip *ResultSet*

**Validatori**

**Validator Email**

“EmailValidator.java” – respectarea pattern-ului a unei expresii regulate, precum emailui care in cazul nostru urmareste structura:

“nume@k.m’

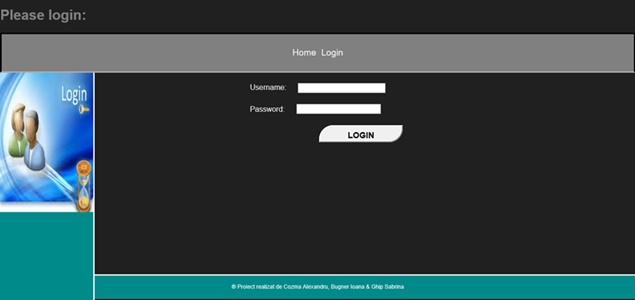
* nume – trebuie sa inceapa cu o litera, cifra sau caracterul “\_”, contine mai multe cifre, litere, caractere
* trebuie sa contina neaparat caracterul “@”
* k – un camp de litere sau cifre
* m – camp format din cel putin 3 caractere

**Validator Numar de telefon**

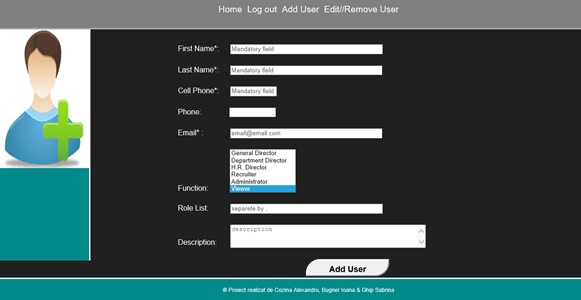
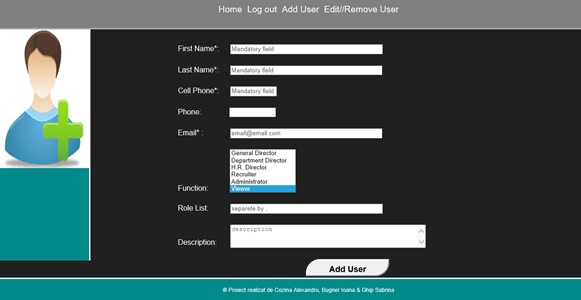
“PhoneValidator.java” respectarea pattern-ului a unei expresii regulate, precum numărul telefonului: - stringul trebuie format doar din cifre si de lungima maxima 10.

**4.5 Fotografii pagini web**

**Login:**



**Add User:**



**All Users:**



**Edit User:**

