PROIECT - INGINERIA SISTEMELOR SOFTWARE AN UNIVERSITAR 2020-2021 SEMESTRUL 2

Aplicatie

SISTEM

GESTIONARE BUGURI

**Nume si Prenume Student: Coordonator:**

Andrei Paul Prof. PETRASCU Vladiela

221/1

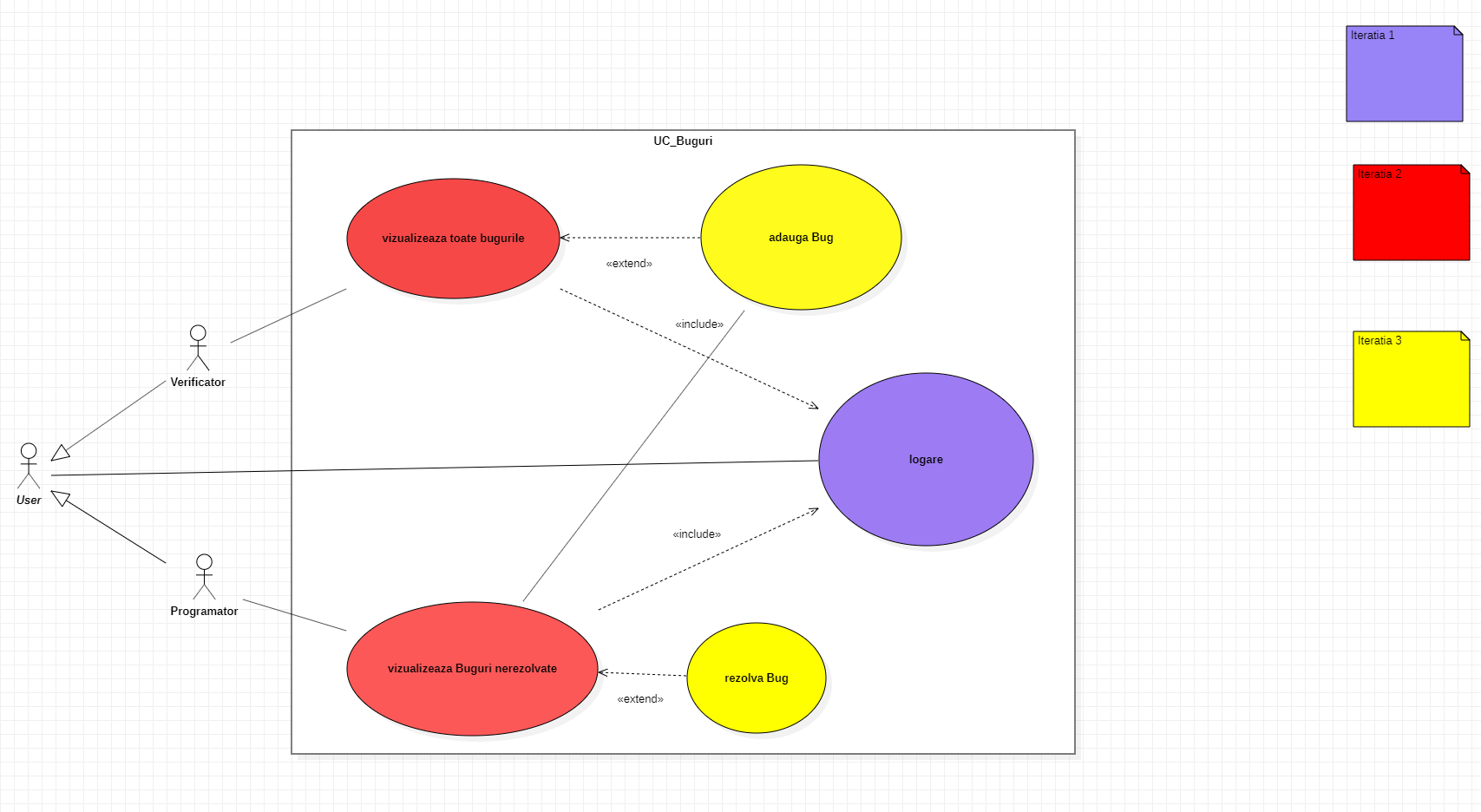
Informatica-romana

1.Prezentarea cerintei:

5. URMARIRE BUG-URI O firma producatoare de software pune la dispozitia programatorilor si verificatorilor sai un sistem prin care acestia pot sa comunice electronic. Astfel, fiecare dintre angajatii mentionati are la dispoziție un terminal prin care: •verificatorul poate înregistra un bug, dându-i o denumire si o descriere; imediat dupa înregistrarea bug-ului, toti programatorii vad lista bug-urilor actualizata cu obiectul nou introdus; •programatorul vizualizeaza lista bug-urilor; de asemenea, programatorul poate selecta un bug din lista si poate declansa un buton prin care declara ca bug-ul a fost eliminat, caz în care bug-ul este scos din lista tuturor programatorilor.

2.Modelul functional:

Diagrama cazuri de utilizare:[4]



Descriere cazuri de utilizare:

3.Modelul conceptual:

1)

|  |  |  |  |
| --- | --- | --- | --- |
| ID and name | Login | | |
| Primary actor | User | Secondary actors |  |
| Description | Functionalitatea de login care permite utilizatorului sa intre in aplicatie. | | |
| Trigger | Utilizatorul acceseaza fereastra de login. | | |
| Preconditions | Pre 1 – Db-ul este conectat la internet  Pre 2 – Utilizatorul introduce corect datele de logare. | | |
| Postconditions | Post 1 – Se deschide fereastra principala | | |
| Normal flow | 1. Utilizatorul intra pe fereastra de login.  2. Utilizatorul introduce email-ul si parola.  3. Sistemul va afisa un mesaj de eroare daca datele sunt eronate.  4. Utilizatorul este directionat catre pagina principala daca datele de logare sunt corecte | | |
| Alternative flows | None | | |
| Exceptions | 1. Nu exista conexiune la internet  - sistemul nu poate accesa baza de date.  2. Datale introduse nu sunt corecte sau nu exista in baza de date. | | |

2)

|  |  |  |  |
| --- | --- | --- | --- |
| ID and name | AfiseazaBuguri | | |
| Primary actors | Verificator | Secondary actors |  |
| Description | Utilizatorul poate vedea lista cu toate bugurile (rezolvate sau nerezolvate). | | |
| Trigger | Utilizatorul acceseaza fereastra cu lista de buguri. | | |
| Preconditions | Pre 1 – Db-ul este conectat la internet  Pre 2 – Utilizatorul este logat | | |
| Postconditions | Post 1 – Sistemul afiseaza lista cu toate bugurile. | | |
| Normal flow | 1.Utilizatorul cu rolul de verificator adauga un bug.  2.Utilizatorul cu rolul de verificator acceseaza pagina cu lista de buguri.  3.Sistemul afiseaza lista cu buguri. | | |
| Alternative flows | None | | |
| Exceptions | 1. Nu este conectat la internet  - sistemul nu poate accesa baza de date  - sistemul va afisa un mesaj de eroare  2. Nu exista niciun element in lista  - sistemul va afisa un mesaj sugestiv | | |

3)

|  |  |  |  |
| --- | --- | --- | --- |
| ID and name | AfiseazaBuguriNerezolvate | | |
| Primary actors | Programator | Secondary actors |  |
| Description | Utilizatorul poate vedea lista cu toate bugurile nerezolvate. | | |
| Trigger | Utilizatorul acceseaza fereastra cu lista de buguri nerezolvate. | | |
| Preconditions | Pre 1 – Db-ul este conectat la internet  Pre 2 – Utilizatorul este logat | | |
| Postconditions | Post 1 – Sistemul afiseaza lista cu bugurile nerezolvate. | | |
| Normal flow | 1.Utilizatorul cu rolul de verificator adauga un bug care initial e marcat “nerezolvat”.  2.Utilizatorul cu rolul de programator acceseaza pagina cu lista de buguri nerezolvate.  3.Sistemul afiseaza lista cu buguri nerezolvate. | | |
| Alternative flows | None | | |
| Exceptions | 1. Nu este conectat la internet  - sistemul nu poate accesa baza de date  - sistemul va afisa un mesaj de eroare  2. Nu exista niciun element in lista  - sistemul va afisa un mesaj sugestiv | | |

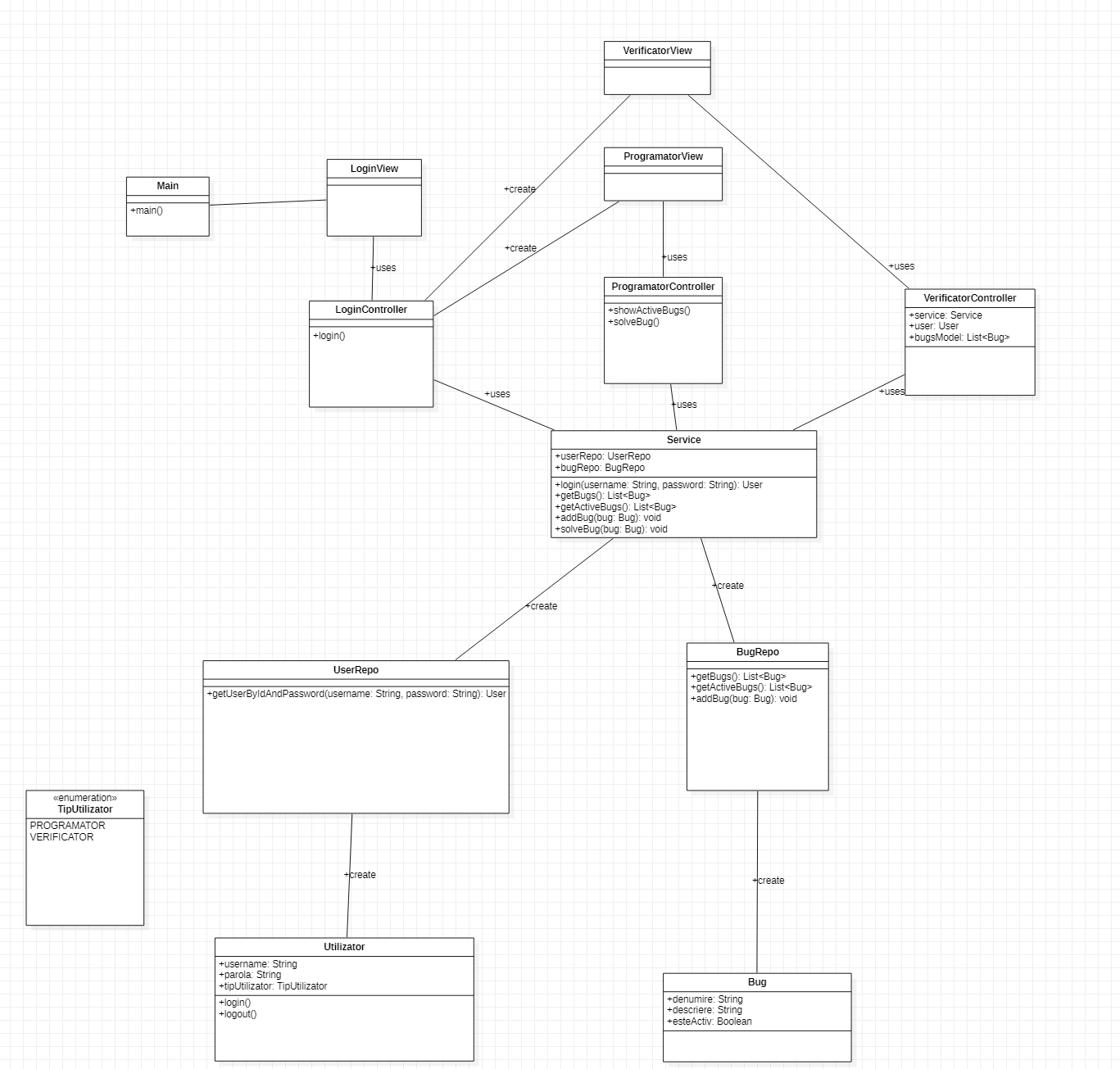
4)

|  |  |  |  |
| --- | --- | --- | --- |
| ID and name | Adauga Bug | | |
| Primary actors | Verificator | Secondary actors | Programator |
| Description | Utilizatorul cu rolul de verificator completeaza field-uri, se face o validare, si se adauga sau nu in lista de buguri un bug nou. | | |
| Trigger | Utilizatorul apasa pe butonul de adauga bug. | | |
| Preconditions | Pre 1 – Db-ul este conectat la internet  Pre 2 – Utilizatorul are rolul de verificator  Pre 3 – Utilizatorul completeaza datele corect. | | |
| Postconditions | Post 1 – Sistemul salveaza in baza de date noul bug.  Utilizatorul se logheaza ca Verificator | | |
| Normal flow | 1. Utilizatorul introduce denumire si descriere bug.  2.Utilizatorul apasa pe butonul de “adauga bug”  3. Sistemul adauga inregistrarea in lista de buguri.  4. Bugul apare la programatori | | |
| Alternative flows | None | | |
| Exceptions | 1. Nu este conectat la internet  - sistemul nu poate accesa baza de date  - sistemul va afisa un mesaj de eroare  2. Datele nu au fost introduse corect  - sistemul afisa un mesaj sugestiv  - utilizatorul introduce din nou datele | | |

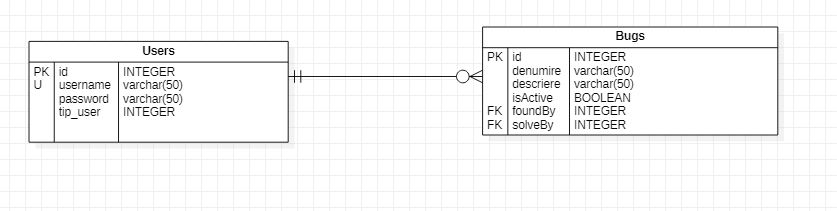
5)

|  |  |  |  |
| --- | --- | --- | --- |
| ID and name | Rezolva Bug | | |
| Primary actors | Programator | Secondary actors | Verificator |
| Description | Utilizatorul cu rolul de programator marcheaza un bug ca si rezolvat. | | |
| Trigger | Utilizatorul selecteaza un bug si apasa pe butonul de rezolva bug. | | |
| Preconditions | Pre 1 – Db-ul este conectat la internet  Pre 2 – Utilizatorul are rolul de programator | | |
| Postconditions | Post 1 – Sistemul updateaza starea bugului in baza de date.  Post 2 – Utilizatorul se logheaza ca Programator. | | |
| Normal flow | 1.Utilizatorul apasa pe butonul de “rezolva bug”  2. Sistemul sterge inregistrarea din lista de buguri pentru | | |
| Alternative flows | None | | |
| Exceptions | 1. Nu este conectat la internet  - sistemul nu poate accesa baza de date  - sistemul va afisa un mesaj de eroare  2. Datele nu au fost introduse corect  - sistemul afisa un mesaj sugestiv  - utilizatorul introduce din nou datele | | |

Diagrama de clase:

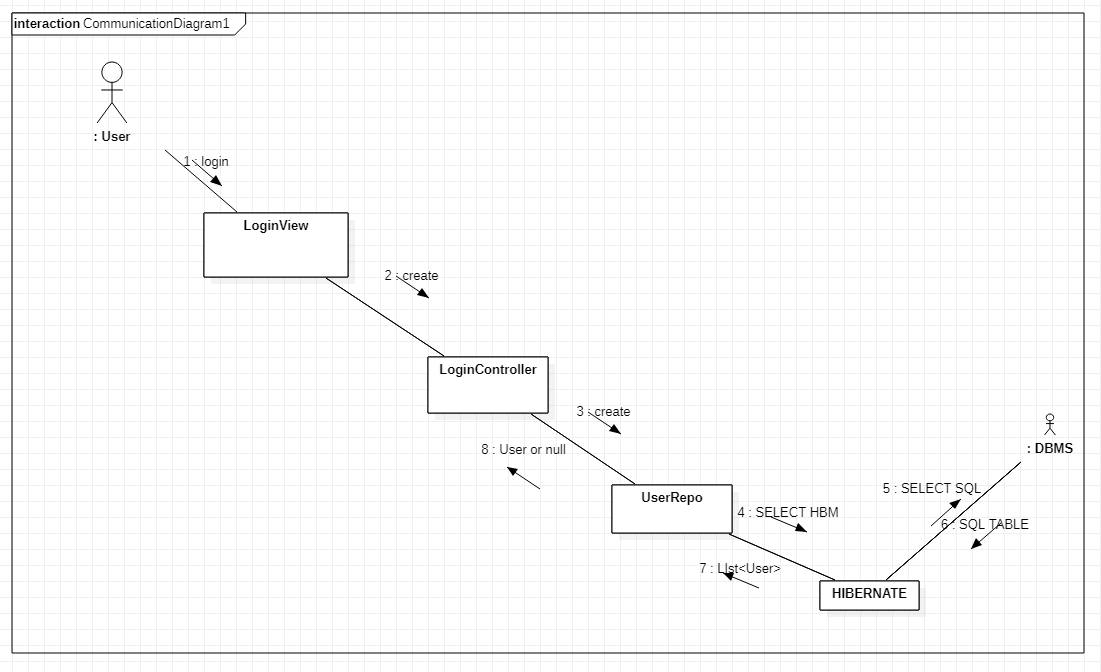


4.Diagrama bazei de date:

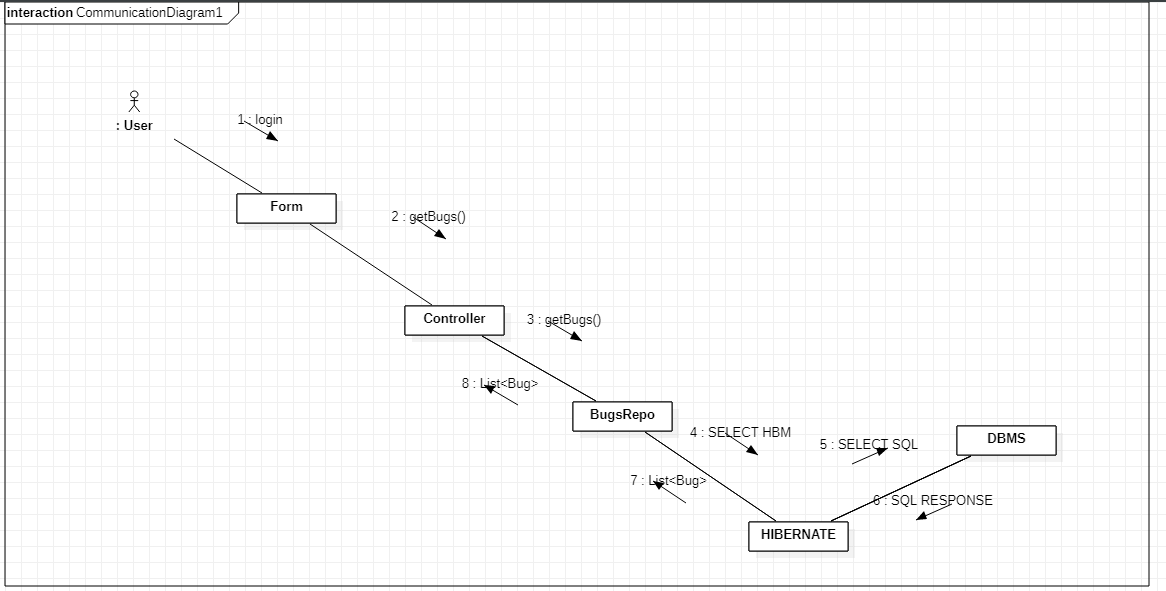


5.Model dinamic:

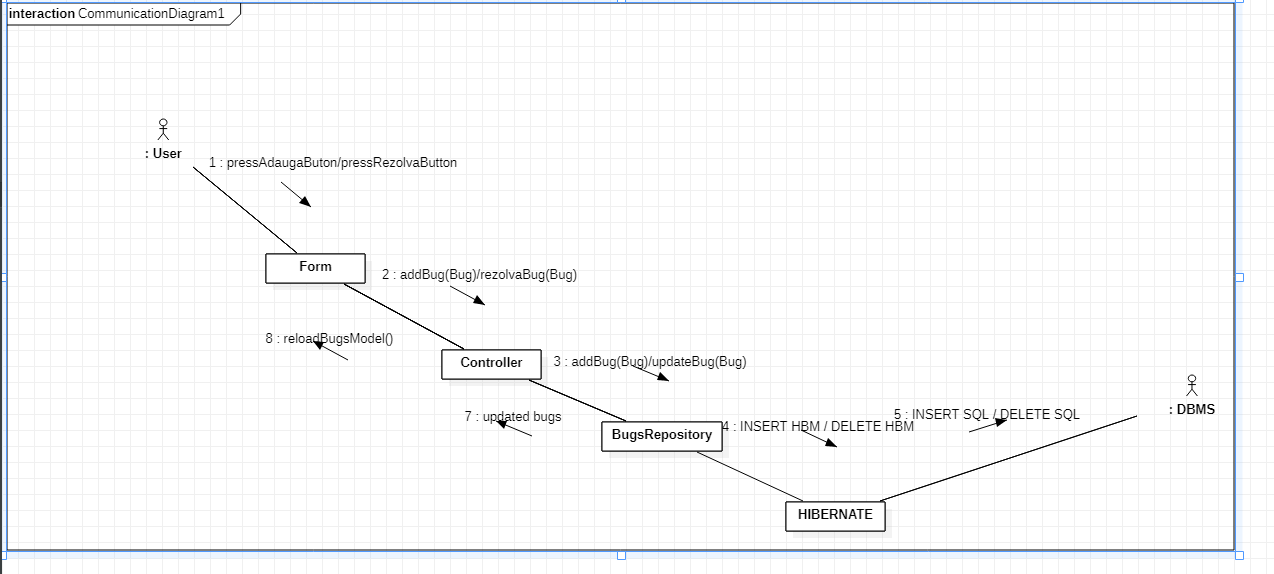
**Iteratia 1:**



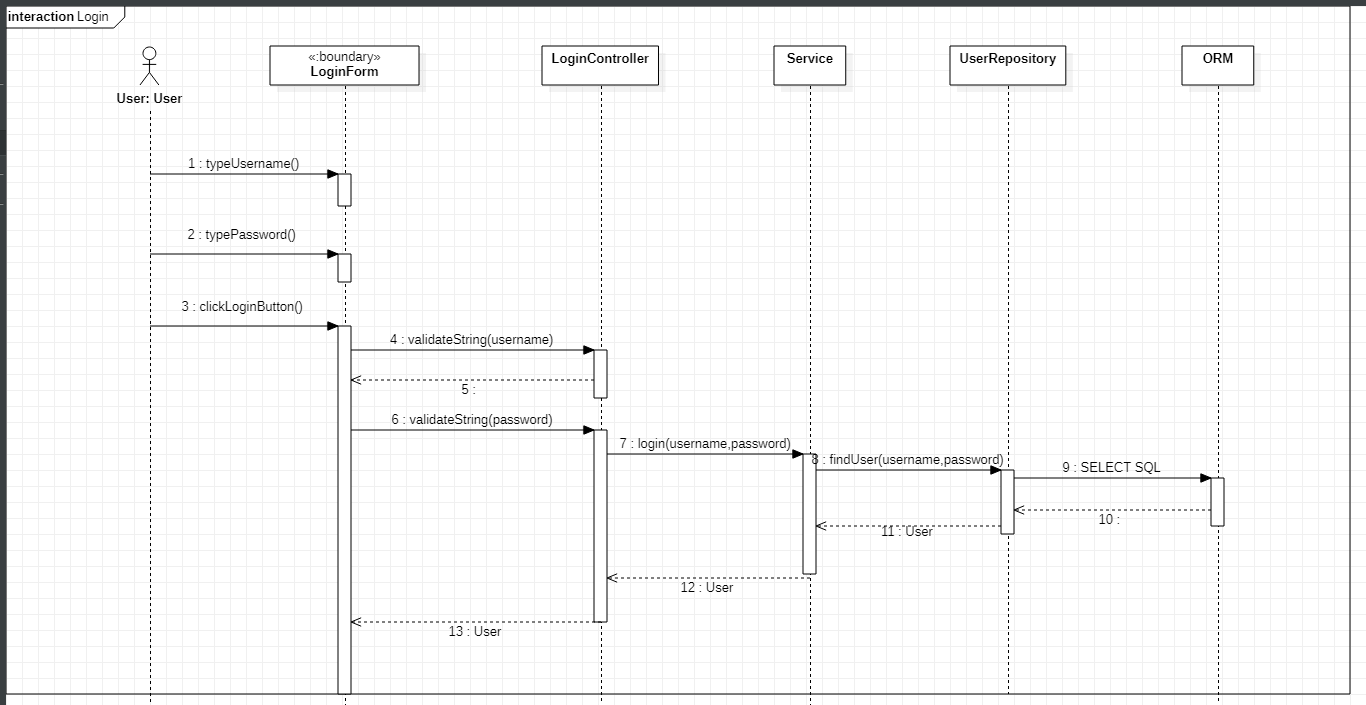
Iteratia 2:

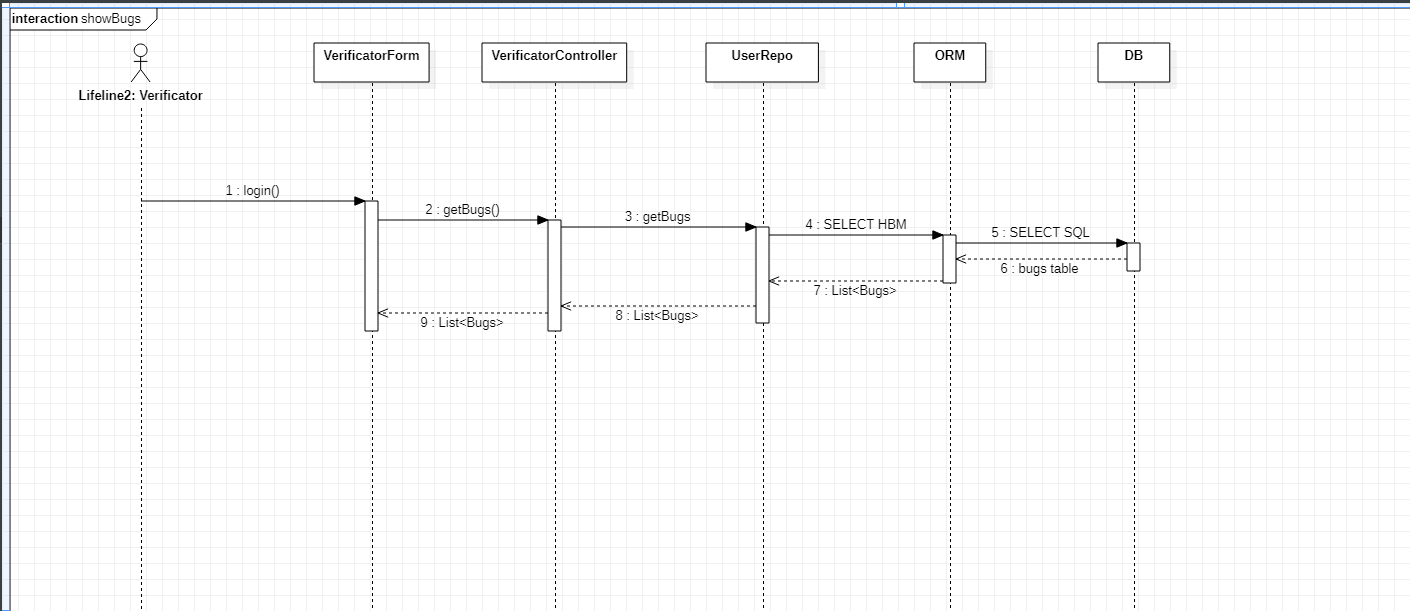


Iteratia 3

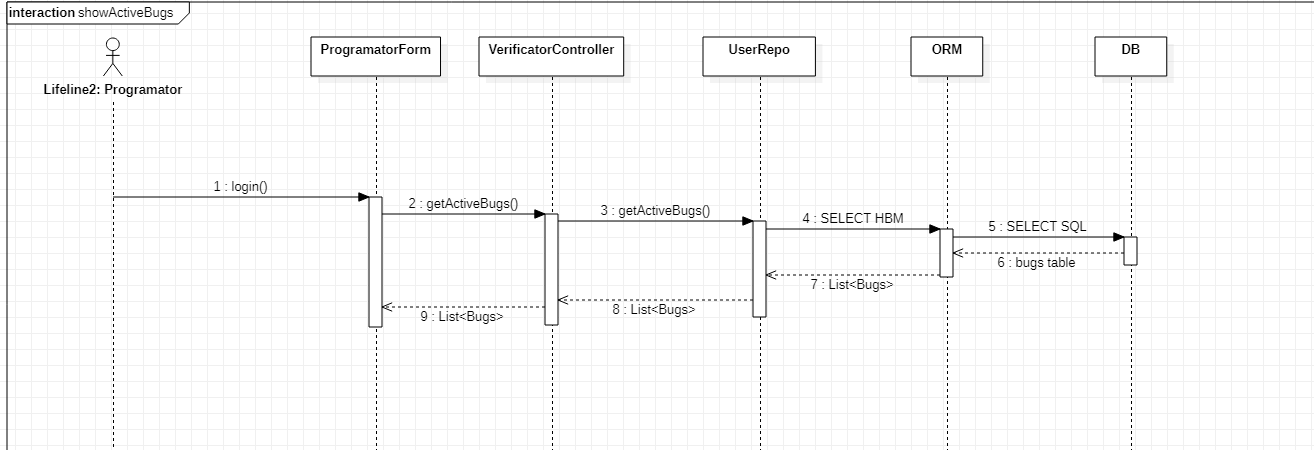


Login – diagrama de secventa

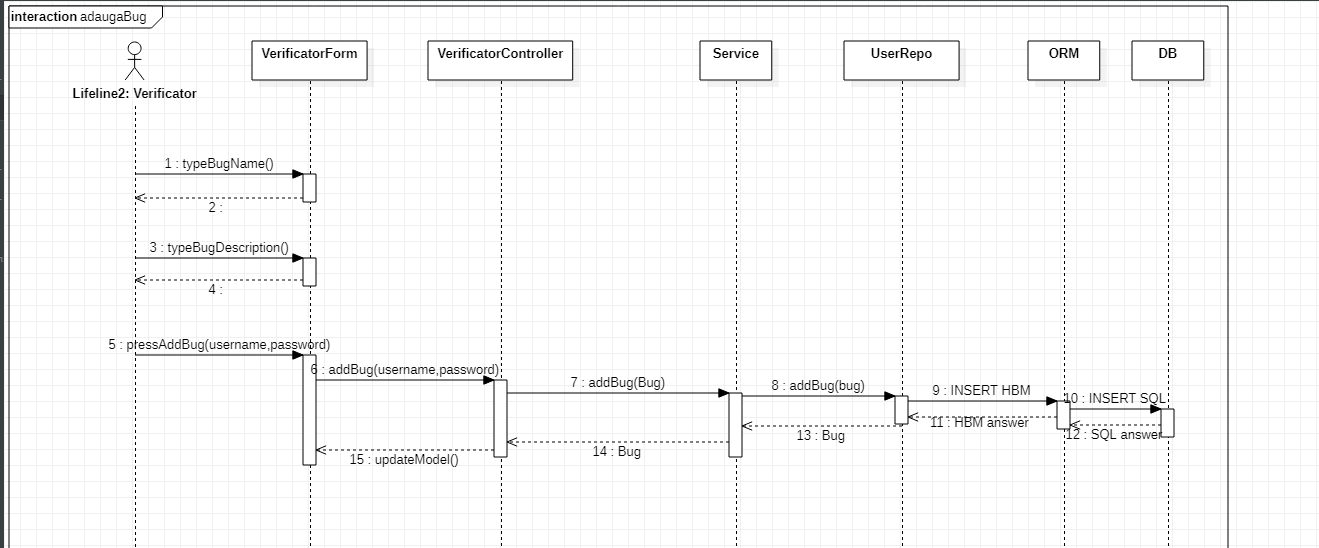


showBugs() - diagrama de secventa 

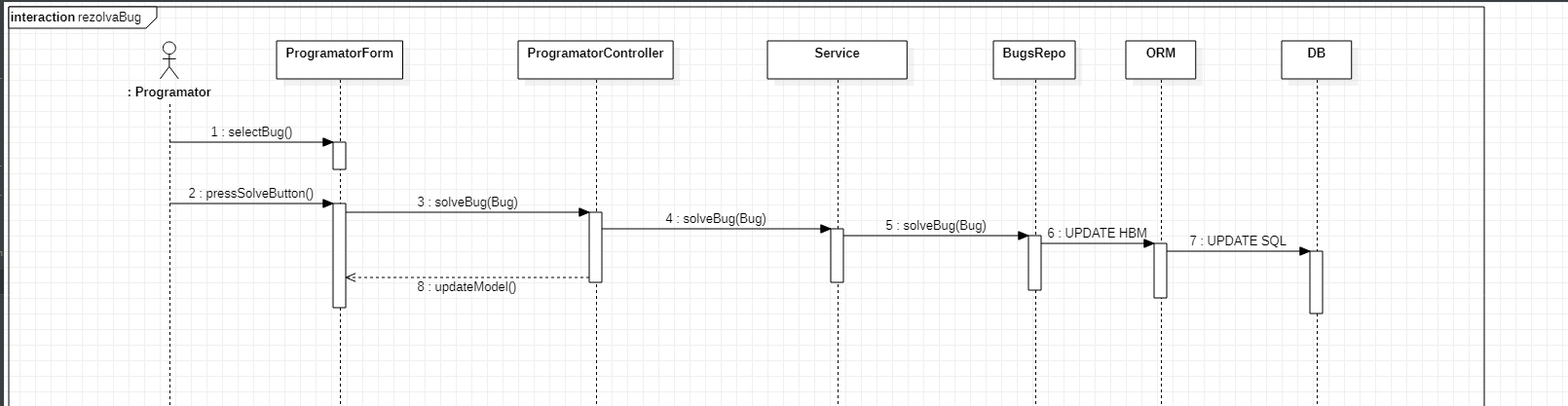
showActiveBugs() – diagrama de secventa



Adauga bug – diagrama de secventa



rezolva bug – diagrama de secventa



6.Documentația tehnică:

Funcționalitățile:

**Programator:**

* poate sa vizualizeze bugurile active
* poate sa marcheze bugurile ca rezolvate

**Verificator:**

* poate sa vizualizeze toata bugurile
* poate sa creeze buguri noi

Etapele proiectării:

**Analiza cerintelor:**

• Am indentificat cerintele impuse, functionalitatiile(adaugarea de buguri, rezolvare de buguri, afisarea de buguri).

**Analiza:**

* Cerințe funcționale: Aplicația trebuie să îi permit verificatorul să adauge buguri si programatorului sa le rezolve.
* Cerințe non-funcționale: Aplicația trebuie să fie scrisă în java, să fie ușor de utilizat și să folosească un ORM.

**Proiectarea sistemului:**

* Am folosit StarUML pentru a realiza diagrame pentru a planifica dezvoltarea aplicației.Tipurile de diagrame pe care le-am folosit pentru a-mi facilita munca sunt: diagramă de utilizare, diagramă de secvență, diagramă de comunicare, diagram de clase și diagrama bazei de date.

**Implementarea:**

* Am utilizat o abordare top-down, începând cu interfața grafică. Am continuat cu controlerul și baza de date.Am repetat această abordare pentru toate funcționalitățile

**Testare:**

* Am testat codul în 2 modalități:am rugat o persoană să folosească aplicația și am testat toate cazurile aparute după implementarea codului (white-box testing)

Tehnologii folosite:

**Proiectare:**

* Pentru proiectare și dezvoltarea diagramelor am folosit StarUML, pentru posibilitatea de a grupa diagramele în același proiect și deoarece a fost prezentat atât la această materie cât și la altele, ceea ce mi-a permis să ma familiarizez cu acesta.

**Mediu de dezvoltare:**

* Mediul de dezvoltare pe care l-am folosit este IntelliJ Idea, deoarece este un mediu stabil, în care se pot integra cu ușurință framework-uri.De asemenea acesta oferă posibilitatea de a gestiona baza de date pe care am folosit-o.

**Baza de date:**

* Am utilizat SQLite fiindcă este ușor de folosit și am considerat ca biblioteca nu va avea nevoie de o capacite mai mare de un milion de înregistrări (capacitate până la care SQLite fucționeaza foarte bine)

**Limbaj de programare:**

* Limbajul de programre ales este Java.

**GUI (Interfață grafică):**

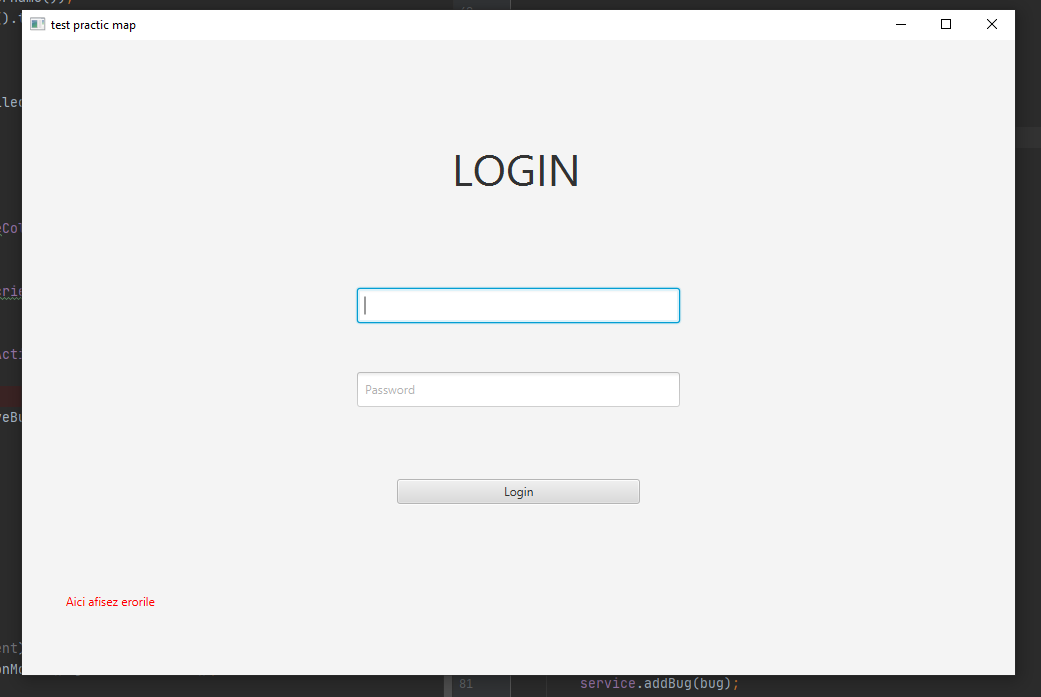
* Am folosit JavaFX și Scene Builder pentru ușurința de a crea interfețe cu un design plăcut prin drag and drop.

**ORM(Object Relational Mapping)**

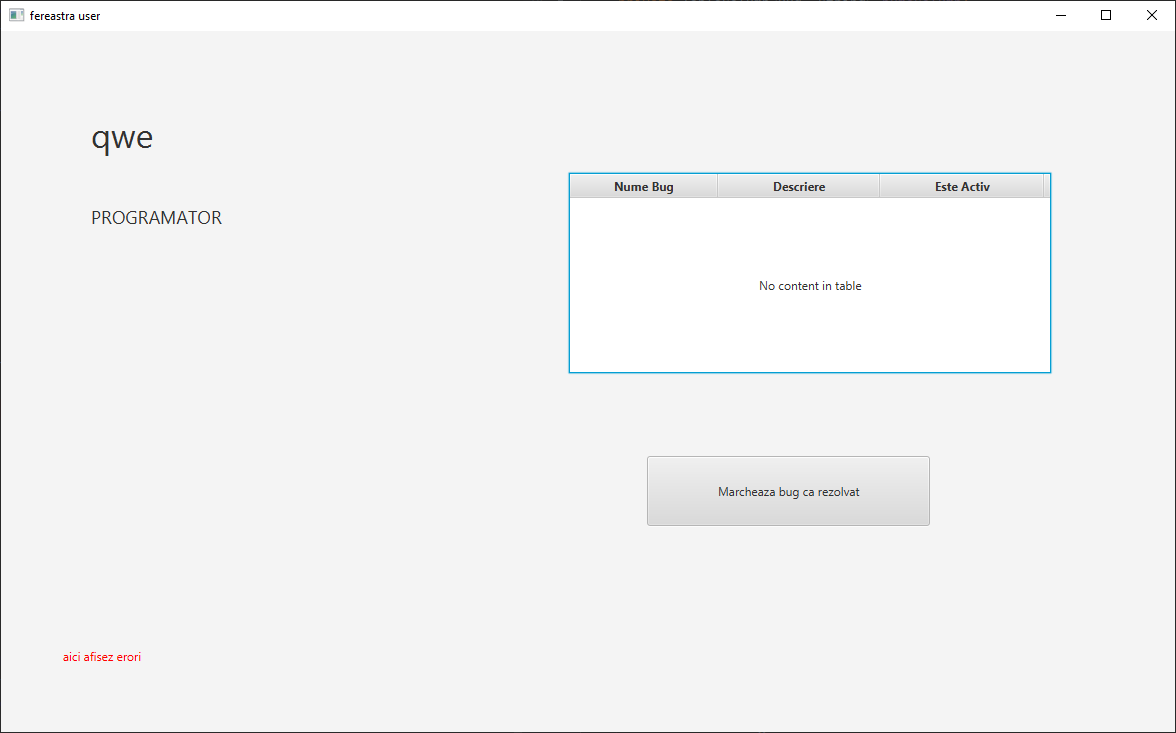
* ORM-ul pe care l-am folosit este Hibernate, deoarece am considerat ca este framework-ul despre care pot gasi cea mai multă informație care să mă ajute la implementarea aplicației

7.Tutorial

Cand utilizatorul deschide aplicația prima fereastră care se deschide este:



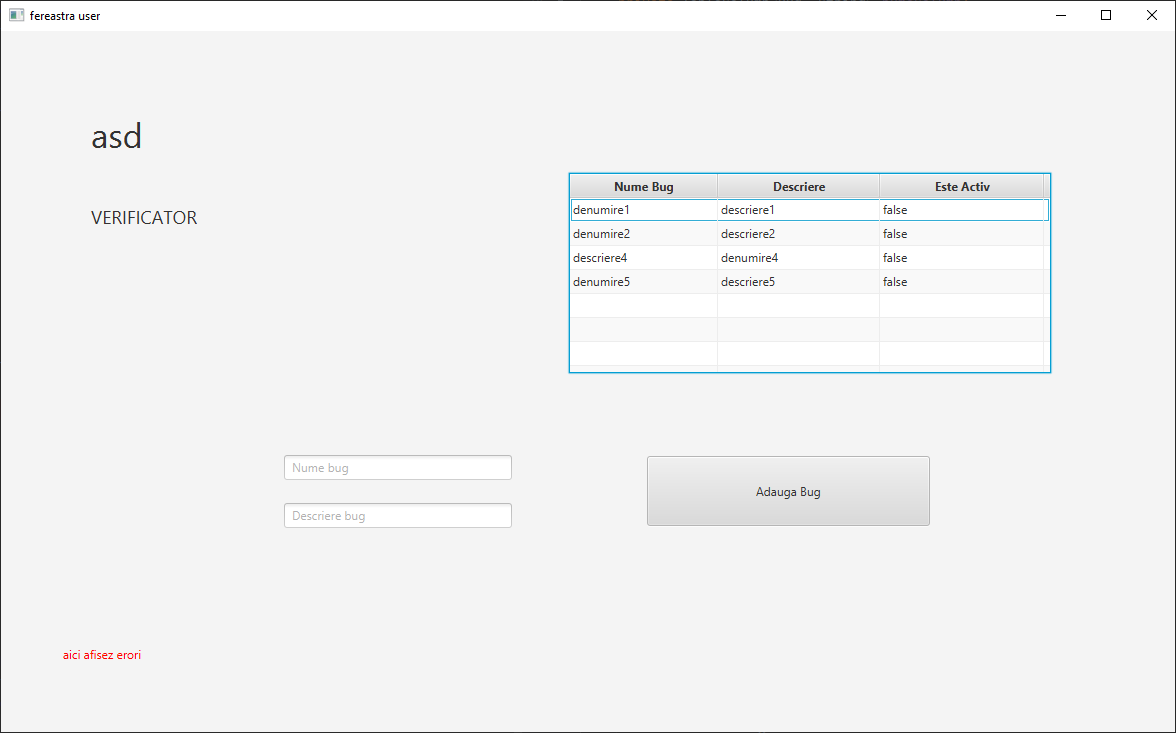
După apasearea pe butonul log in (dacă datele sunt valide se deschide fereastra corespunzatoare tipului de utilizator).Pentru Programator:



se selecteaza un bug si se apasa butonul marcheaza bug ca rezolvat pentru a rezolva un bug

in coltul din stanga jos vor aparea erorile

Dacă utilizatorul care se loghează este verificator se dechide următoarea fereastră:



se introduce numele bugului in nume bug si descriere sa in descriere bug apoi se apasa butonul adauga bug pentru a adauga un bug

in coltul din stanga jos vor aparea erorile

8.Bibliografie

1) https://ubbcluj.sharepoint.com/sites/ISS2021/Class%20Materials/Forms/AllItems.aspx?RootFolder=%2Fsites%2FISS2021%2FClass%20Materials%2FCurs&FolderCTID=0x012000A4CC77AA1514674AAC5B8B8289E7BC18

2) <https://stackoverflow.com/>

3) http://www.cs.ubbcluj.ro/~grigo/mpp/Curs/

4) “Use Case Diagrams”: http://csis.pace.edu/~marchese/CS389/L9/Use%20Case%20Diagrams.pdf

5) “Sequence diagram tutorial”: https://creately.com/blog/diagrams/sequence-diagram-tutorial/

6) “What is communication Diagram?” : https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-communication-diagram/