PROIECT - INGINERIA SISTEMELOR SOFTWARE

AN UNIVERSITAR 2019-2020

SEMSTRUL 2

Bug Tracker

Student:

Tanislav Cosmin-Gabriel

227/1

Informatica

Coordonator:

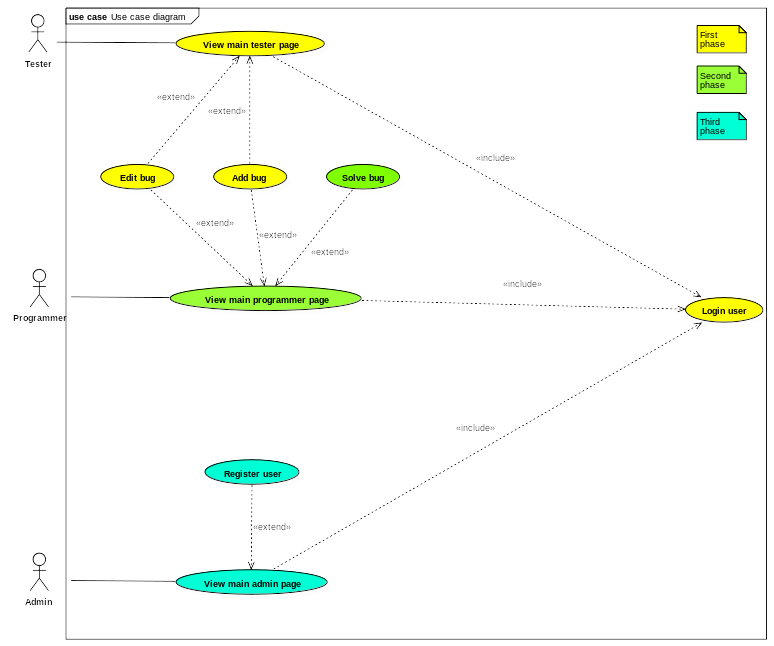
Prof. Sima Ioan

# Prezentarea cerintei

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.

# Diagrama cazuri de utilizare



# Descriere cazuri de utilizare

|  |  |
| --- | --- |
| **Nume caz de utilizare** | Autentificare |
| **Actor** | Verificator (IT1)  Programator (IT2)  Administrator (IT3) |
| **Scenariu normal** | 1. Actorul activeaza functia de autentificare din aplicatie 2. Aplicatia afiseaza un formular actorului ce conține campuri pentru numele de utilizator si parola 3. Actorul completeaza formularul și confirma autentificarea 4. Aplicația afiseaza pagina principala specifica actorului |
| **Scenariu alternativ** | Numele de utilizator sau parola sunt greșite, caz în care aplicația afiseaza un mesaj de eroare și permite utilizatorului sa modifice valorile introduse în campurile formularului și sa încerce din nou, sau sa iasă din aplicatie |
| **Conditie de intrare** | Actorul se afla în fata terminalului sau |
| **Conditie de iesire** | Actorul este autentificat |
| **Cerinta de calitate** |  |
| **Iteratia** | 1-3 |

|  |  |
| --- | --- |
| **Nume caz de utilizare** | Vizualizare pagina principala |
| **Actor** | Verificator (IT1)  Programator (IT2)  Administrator (IT3) |
| **Scenariu normal** | Verificator:   * Aplicația afiseaza lista bug-urilor create de către verificator, cu acțiunile specifice fiecaruia și acțiuni globale   Programator:   * Aplicația afiseaza lista bug-urilor create de către programator și lista bug-urilor create de către toți verificatorii și programatorii, cu acțiunile specifice fiecaruia și acțiuni globale   Administrator:   * Aplicația afiseaza lista angajatilor, cu acțiunile specifice fiecaruia și acțiuni globale |
| **Scenariu alternativ** |  |
| **Conditie de intrare** | Actorul este autentificat |
| **Conditie de iesire** |  |
| **Cerinta de calitate** |  |
| **Iteratia** | 1-3 |

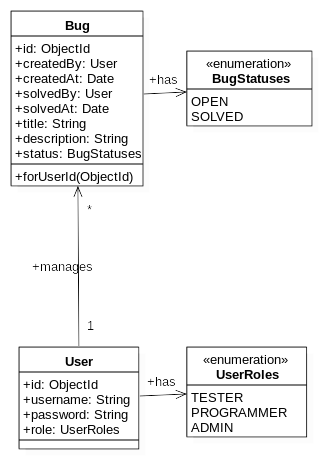
|  |  |
| --- | --- |
| **Nume caz de utilizare** | Rezolvare bug |
| **Actor** | Programator |
| **Scenariu normal** | 1. Programatorul activeaza functia de *Rezolvare bug* din dreptul bug-ului pe care dorește sa îl rezolve 2. Aplicația afiseaza programatorului un formular ce conține un câmp pentru motivul rezolvarii 3. Programatorul completeaza formularul și confirma rezolvarea bug-ului 4. Bug-ul dispare atât din lista de bug-uri proprii ale angajatului cât și din lista de bug-uri a tuturor programatorilor |
| **Scenariu alternativ** |  |
| **Conditie de intrare** | Programatorul este autentificat |
| **Conditie de iesire** | Bug-ul este eliminat din lista de bug-uri a tuturor programatorilor |
| **Cerinta de calitate** |  |
| **Iteratia** | 2 |

|  |  |
| --- | --- |
| **Nume caz de utilizare** | Adaugare bug |
| **Actor** | Verificator (IT1)  Programator (IT2) |
| **Scenariu normal** | 1. Angajatul activeaza functia *Adaugare bug* din aplicatie 2. Aplicatia afiseaza un formular ce contie campuri pentru denumirea și descrierea bug-ului 3. Angajatul completeaza formularul și confirma adaugarea noului bug 4. Formularul se inchide 5. Bug-ul apare atât în lista de bug-uri proprii ale angajatului cât și în lista de bug-uri a tuturor programatorilor |
| **Scenariu alternativ** |  |
| **Conditie de intrare** | Angajatul este autentificat |
| **Conditie de iesire** | Bug-ul este adaugat în lista de bug-uri a tuturor programatorilor |
| **Cerinta de calitate** |  |
| **Iteratia** | 1-2 |

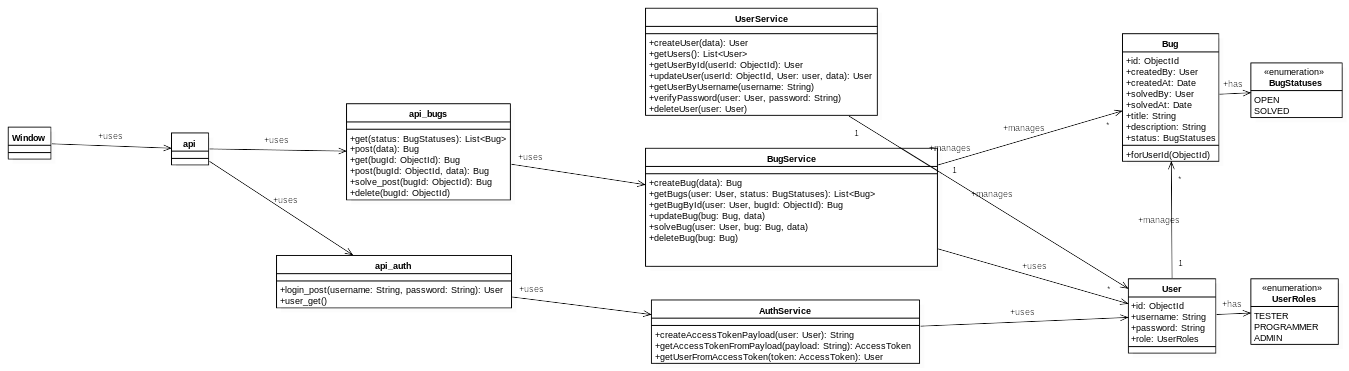
|  |  |
| --- | --- |
| **Nume caz de utilizare** | Modificare bug |
| **Actor** | Verificator (IT1)  Programator (IT2) |
| **Scenariu normal** | 1. Angajatul activeaza functia de *Modificare bug* din dreptul bug-ului pe care dorește sa îl modifice 2. Aplicația afiseaza angajatului un formular ce conține campuri pentru denumirea și descrierea bug-ului, completate cu valorile actuale 3. Angajatul modifica valorile campurilor și confirma modificarea bug-ului 4. Bug-ul este actualizat atât în lista de bug-uri proprii ale angajatului cât și din lista de bug-uri a tuturor programatorilor |
| **Scenariu alternativ** |  |
| **Conditie de intrare** | Angajatul este autentificat |
| **Conditie de iesire** | Bug-ul este actualizat în lista de bug-uri a tuturor programatorilor |
| **Cerinta de calitate** |  |
| **Iteratia** | 1-2 |

|  |  |
| --- | --- |
| **Nume caz de utilizare** | Inregistrare angajat nou |
| **Actor** | Administrator |
| **Scenariu normal** | 1. Administratorul activeaza functia *Inregistrare angajat nou* din aplicatie 2. Aplicatia afiseaza un formular ce conține campuri pentru numele de utilizator și tipul noului angajat 3. Administratorul completeaza formularul și confirma inregistrarea noului angajat 4. Aplicatia afiseaza detaliile noului angajat, inclusiv o parola temporara pe care angajatul o poate folosi la logare |
| **Scenariu alternativ** | Exista deja un angajat cu numele de utilizator respectiv, caz în care aplicația afiseaza o eroare și cere recompletarea formularului |
| **Conditie de intrare** | Administratorul este autentificat |
| **Conditie de iesire** | Contul angajatului devine valid |
| **Cerinta de calitate** |  |
| **Iteratia** | 3 |

# Diagrama de clase a modelului

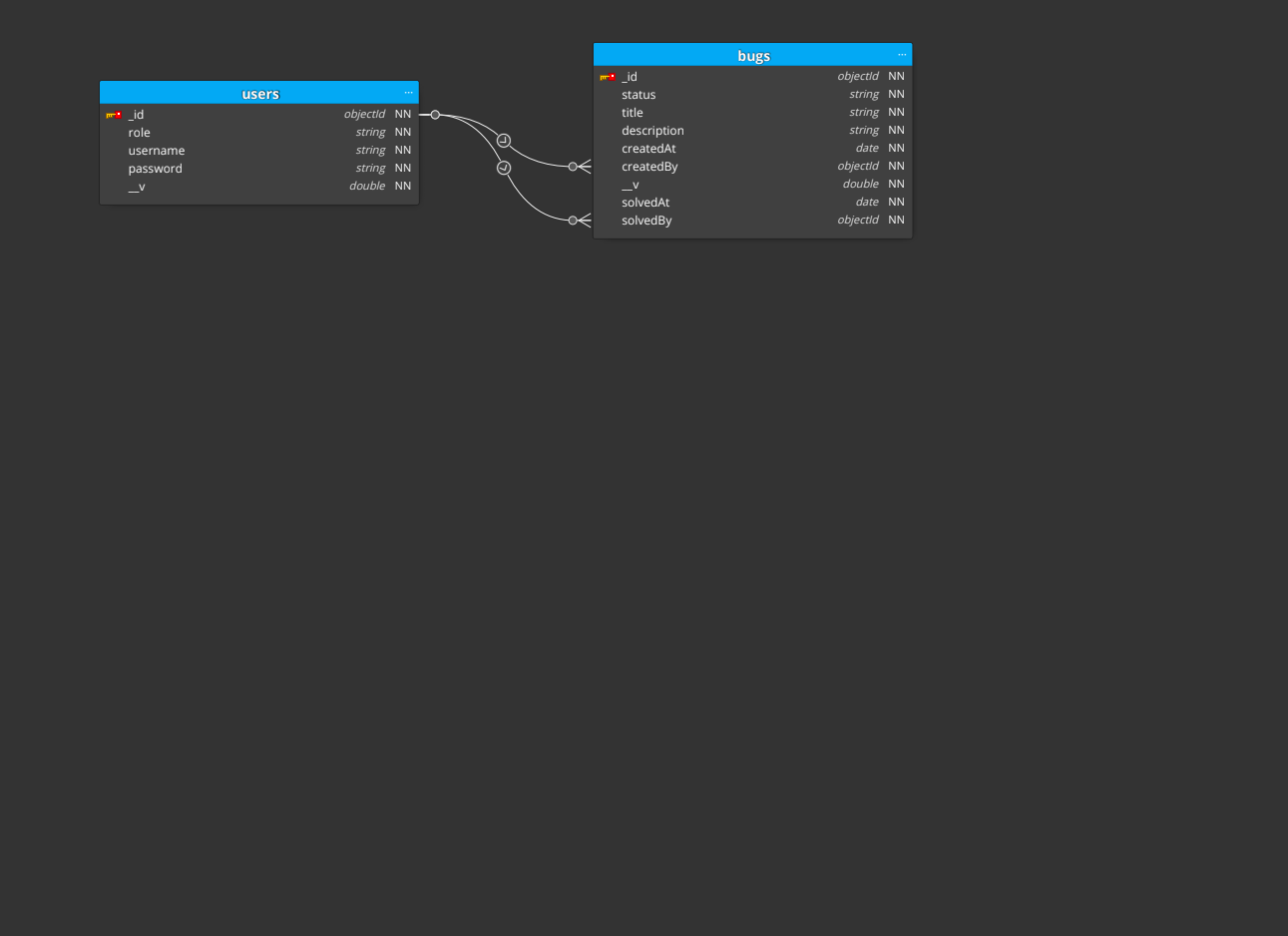


# Diagrama de clase pentru a doua iteratie



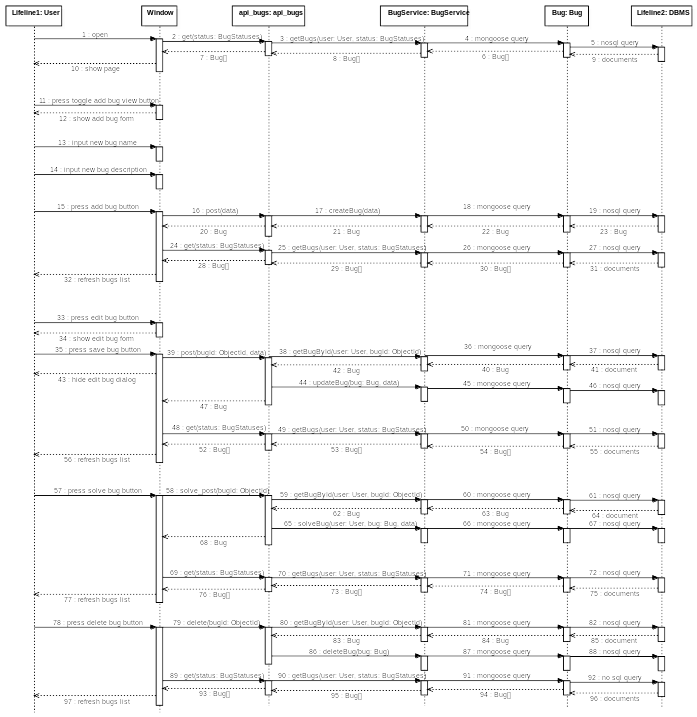
# Diagrama de clase finala

# Diagrama bazei de date



# Diagrama de secventa pentru login

# Diagrama de secventa pentru **manageriere bug-uri**



# **Diagrama de secventa pentru inregistrare user**

# Tehnologii folosite

Front-end:

* vue.js – framework MVVM
* vue-router – plugin vue.js pentru front-end routing
* vue-sse – plugin vue.js pentru server side events
* vuex – plugin vue.js pentru storage stateful
* moment – librarie pentru formatarea timpului
* axios – librarie pentru request-uri AJAX

Back-end:

* node.js
* express – librarie node.js pentru routing
* express-session – librarie node.js pentru session storage
* webpack – bundler de module
* mongoose – MongoDB ORM
* bcrypt – pentru encriptia parolelor
* MongoDB – baza de date

Altele:

* StarUML – pentru design-ul diagramelor
* git, GitHub – controlul surselor

## vue.js

Am ales vue.js ca framework pentru realizarea interfetei grafice deoarece este unul dintre cele mai populare framework-uri din aceasta categorie.

Acesta face reactivitatea interfetei mult mai usor de realizat, oferind abilitatea crearii unui website cat mai interactiv, fara reload-uri. De asemenea, vue.js faciliteaza crearea unor componente reutilizabile, utile pentru minimizarea codului duplicat.

## node.js

Am ales node.js ca limbaj pentru realizarea partii de server a aplicatiei deoarece este foarte flexibil, are o comunitate foarte mare, fiind unul dintre cele mai folosite limbaje pentru crearea unei aplicatii web in 2020. Acesta ofera avantajul folosirii JavaScript atat pe client cat si pe server, de asemenea facilitand comunicarea bi-directionala, necesara notificarii utilizatorilor.

## MongoDB

Am ales MongoDB ca baza de date deoarece acesta este cel mai des utilizat in combinatie cu node.js, comunitatea oferind o multitudine de librarii ajutatoare.

## Structura proiectului

Proiectul este structurat intr-o serie de directoare in functie de tipul fisierelor sursa si scopul acestora.

web/ ← fisiere statice servite browser-ului

css/ ← fisiere css

js/ ← fisiere js

vue/ ← componente vue

dist/ ← rezultatul module bundling-ului

index.html ← singurul fisier HTML al aplicatiei

config/ ← fisiere de configurare ale server-ului

data/ ← stocarea bazei de date

docs/ ← documentatie, diagrame

api/ ← rute expuse de server

services/ ← servicii necesare rutelor

models/ ← modelele bazei de date

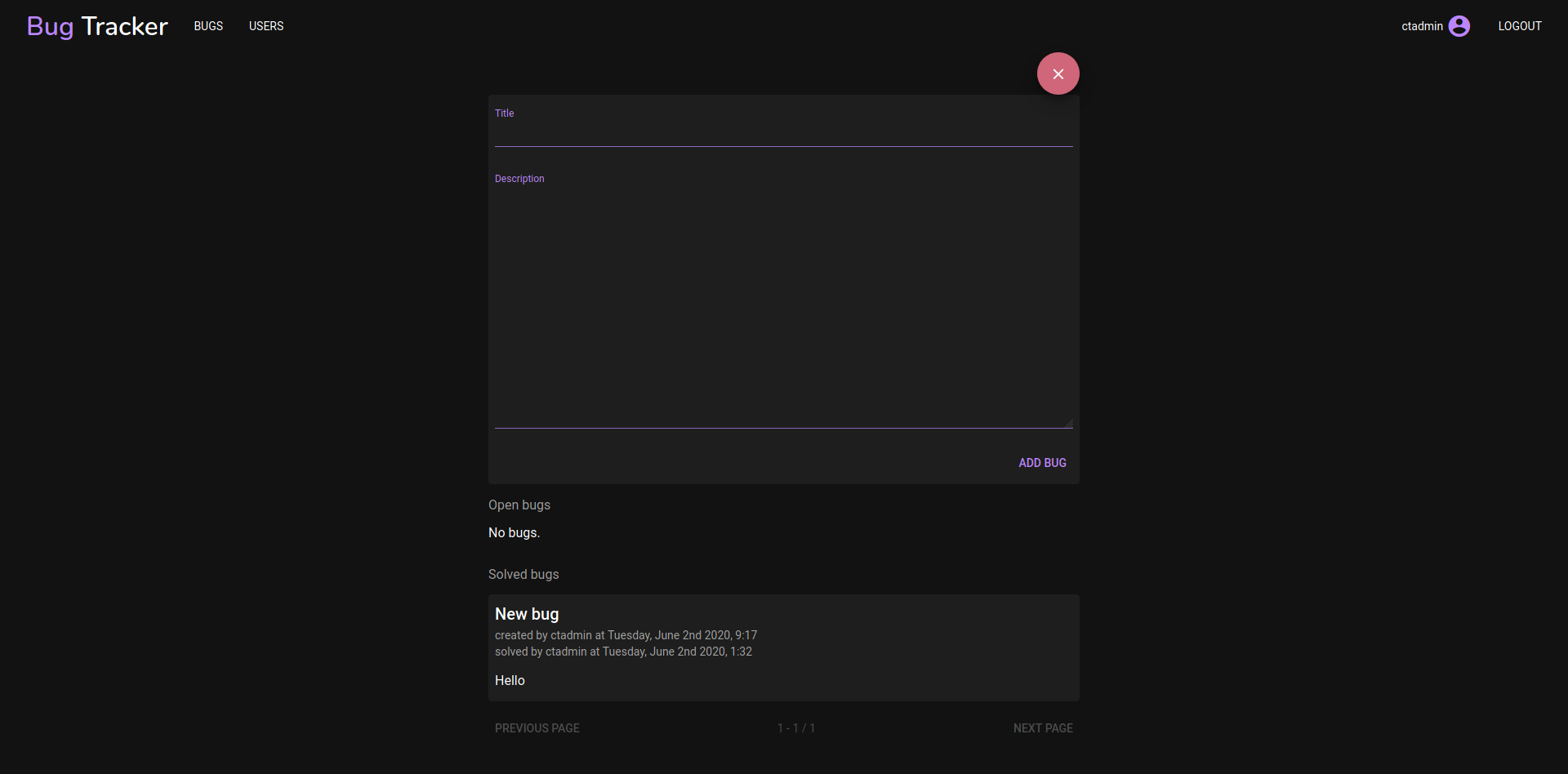
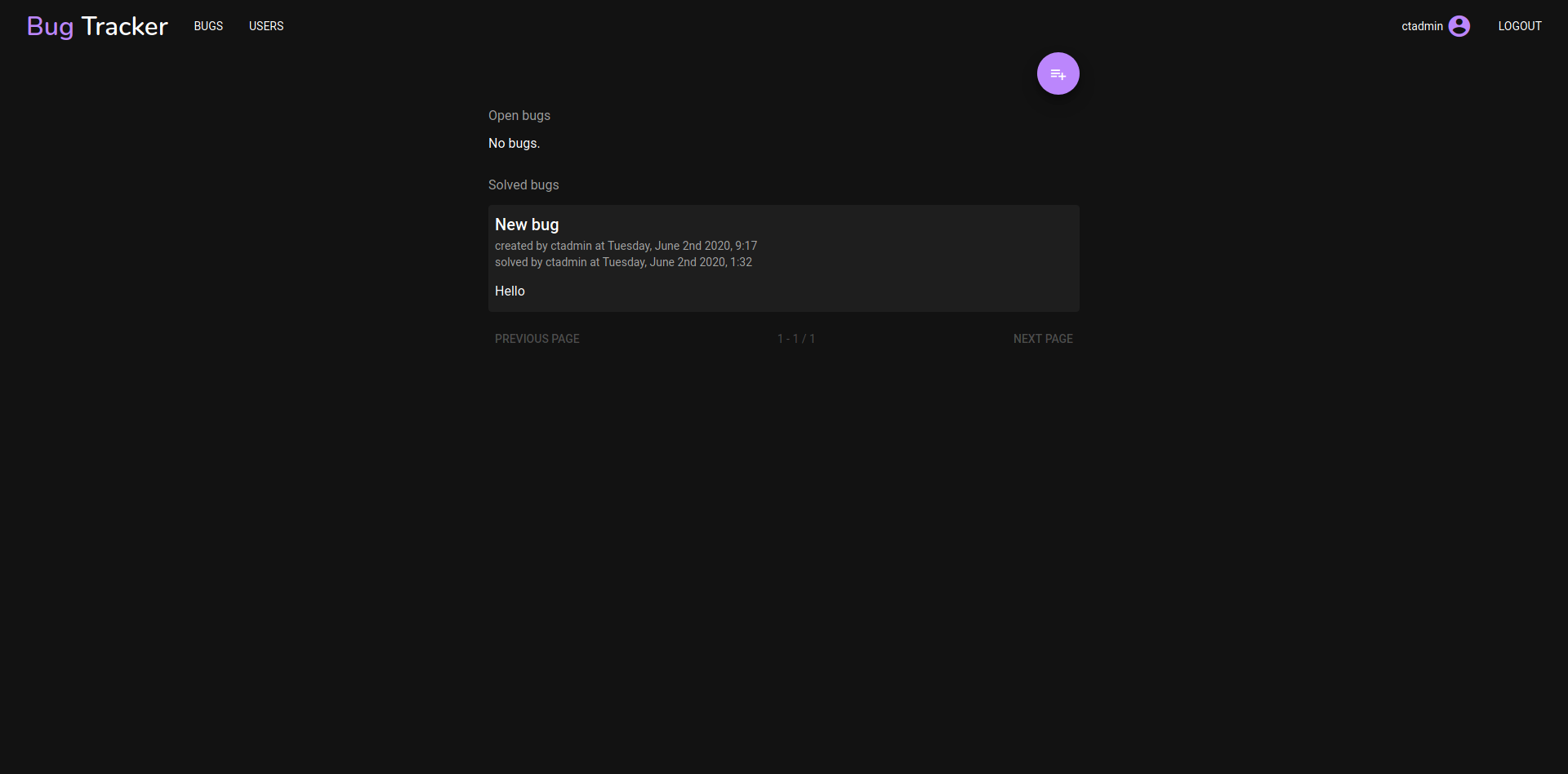
lib/ ← librarii utilitare

## Ghid pentru utilizator

### **Pagina** de login

Se introduct datele de logare in campurile aferente si se apasa pe butonul de login.

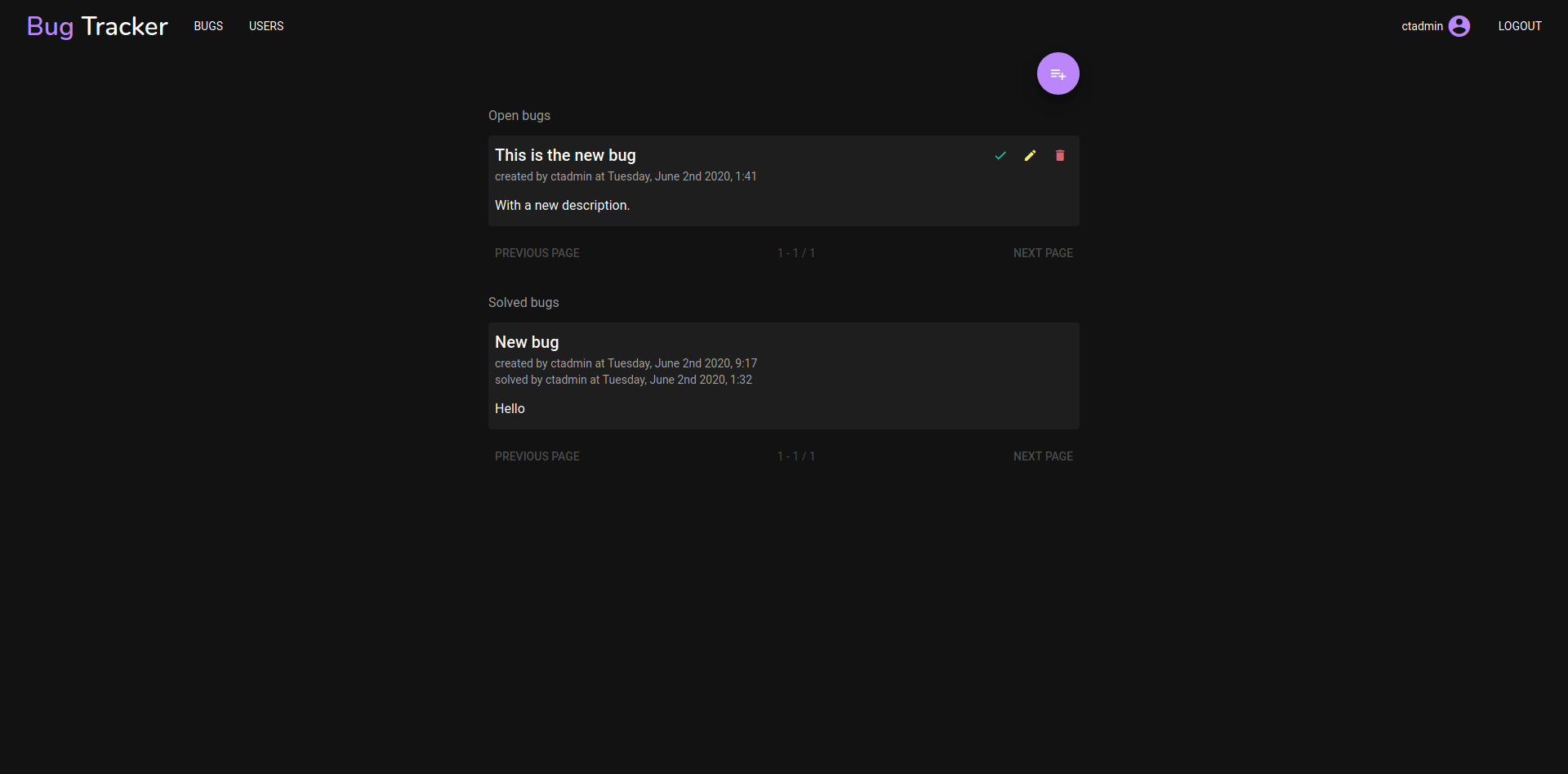
Va fi afisata o eroare in cazul unor date de logare incorecte. In cazul in care datele de logare sunt corecte, se afiseaza pagina cu bug-uri.



### Pagina cu bug-uri

Pentru adaugarea unui bug, se apasa butonul de add.

Un formular pentru adaugarea bug-ului va fi afisat.

Se introduc datele noului bug in campurile afisate, si se apasa pe butonul de add.

Formularul se va inchide si noul bug va fi vizibil in lista de bug-uri.

Acesta poate fi rezolvat, editat sau sters.

# Bibliografie

http://www.cs.ubbcluj.ro/~vladi/Teaching/Didactic/ISS%202019-2020/

<https://simaioan.wordpress.com/>

https://nodejs.org/en/docs/

https://expressjs.com/en/api.html

https://vuejs.org/