**Bug Tracking System**

Software Engineering - Semester Project

2022-2023

2nd Semester

|  |  |
| --- | --- |
| **Student**  Petec Răzvan-Gabriel  Group 225/2  Computer Science Specialization | **Coordinator**  Prof. Sima Ioan |
|  |  |

**[this page has been left intentionally blank]**

Contents

[**1. Requirements** 4](#_Toc136952941)

[**2. Functional Model** 4](#_Toc136952942)

[**3. Concept Model** 18](#_Toc136952943)

[**4. Database Diagram** 20](#_Toc136952944)

[**5. Dynamic Model** 21](#_Toc136952945)

[**6. Technical Documentation** 28](#_Toc136952946)

[**7. Help** 29](#_Toc136952947)

[**8.** **Bibliography** 32](#_Toc136952948)

# **Requirements**

A software company provides its programmers and testers with a system through which they can communicate online. Each employee has at his disposal a terminal through which:

* **the administrators** can manage the accounts and perform CRUD operations on them, such as creating, updating, and removing an account.
* **the** **testers** can report a bug by entering its name and its description, after the bug is registered, all programmers will see the updated list of bugs.
* **the programmers** can see the list of bugs; also, the programmer can select a bug from the list and trigger a button declaring that he assigned that bug, in which case the bug is marked as assigned for all the programmers, and the tester who registered the bug cannot longer modify it.

# **Functional Model**

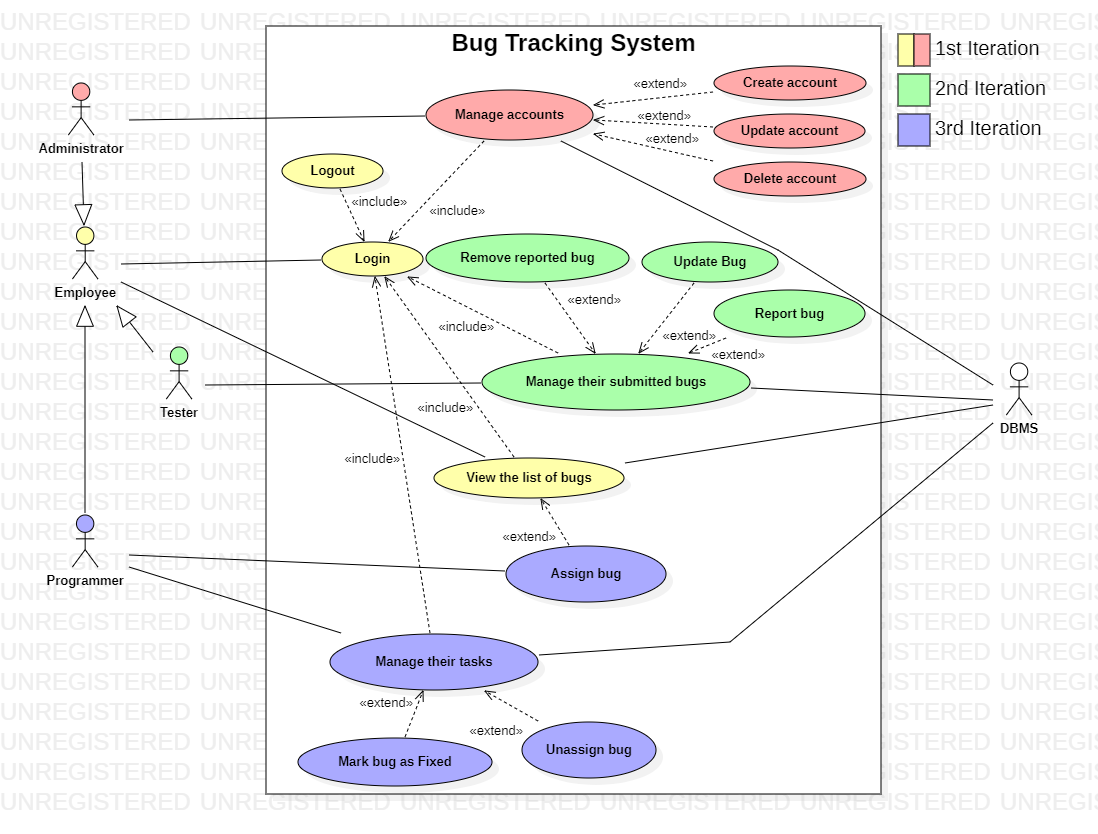


Figure 1. Use Cases Diagram

**Employee**:

* **login**: the employee opens the web application and, if their session key isn’t registered (they haven’t logged in before or they logged out at the last session), the system will show a form with two text fields and a button, where the employee needs to enter their username and password. If the credentials the employee will enter would be incorrect, the system will pop up an error message and the employee can try again, otherwise the system will show the main page of the application. If the employee logged in before with the same session key and they didn’t log out yet, the main page of the application will be shown directly.
* **log out**: the employee wants to log out, so they press the log out button in the main page. If the log out is successful, the system will log them out, otherwise an error message will pop up.
* **view list of bugs**: if the employee isn’t at the list of bugs page, they need to press the view bugs button, when the employee is at the list of bugs page the system will show them the list of bugs.

**Administrator**:

* **manage accounts**: if the administrator isn’t at the accounts management page, they need to press the manage accounts button; once there, the system will show them the list of accounts and the administrator can do three operations:
  + **create an account**: the administrator selects the type of the account, enters a username and a password, then they press the create account button, if the data they entered is incorrect, the system will pop up an error message, otherwise a confirmation dialog will pop up, if the administrator will choose to not continue, the system will return to its initial state, otherwise, if the creation fails, the system will pop up an error message, otherwise the system will show a success message and the accounts list will refresh.
  + **update an account**: the administrator selects an employee, selects the new type of the accounts, enters the new username and the new password, then they press the update account button, if the data they entered is incorrect, the system will pop up an error message, otherwise a confirmation dialog will pop up, if the administrator will choose to not continue, the system will return to its initial state, otherwise, if the update fails, the system will pop up an error message, otherwise the system will show a success message and the accounts list will refresh
  + **delete an account**: the administrator selects an employee, then they press the delete account button, if no account has been selected, an error message will pop up, otherwise a confirmation dialog will pop up, if the administrator will choose to not continue, the system will return to its initial state, otherwise, if the deletion fails, the system will pop up an error message, otherwise the system will show a success message and the accounts list will refresh.

**Tester**:

* **manage submitted bugs**: if the tester isn’t at the submitted bugs management page, they need to press the manage submitted bugs button; once there, the system will show them the list of their submitted bugs and the tester can do three operations:
  + **report a bug**: the tester enters the title of the bug and its description, then they press the report bug button, if the data they entered is incorrect, the system will pop up an error message, otherwise a confirmation dialog will pop up, if the tester will choose to not continue, the system will return to its initial state, otherwise, if the creation fails, the system will pop up an error message, otherwise the system will show a success message and the bugs list will refresh
  + **update a bug**: the tester selects a bug, enters the new title and the new description, then they press the update bug button, if the data they entered is incorrect, the system will pop up an error message, otherwise a confirmation dialog will pop up, if the tester will choose to not continue, the system will return to its initial state, otherwise, if the update fails, the system will pop up an error message, otherwise the system will show a success message and the bugs list will refresh
  + **remove a bug**: the tester selects a bug, then they press the remove bug button, if no bug has been selected, an error message will pop up, otherwise a confirmation dialog will pop up, if the tester will choose to not continue, the system will return to its initial state, otherwise, if the deletion fails, the system will pop up an error message, otherwise the system will show a success message and the bugs list will refresh.

**Programme**r:

* **assign a bug**: if the programmer isn’t at the list of bugs page, they need to press the view bugs button; once there the system will show them the list of all bugs, if the programmer selects an unassigned bug, the system will show them an assign bug button, if the programmer presses it and the assignation is failed, the system will show them an error message, otherwise the bug will be assigned and the bugs list will refresh.
* **manage assigned bugs**: if the programmer isn’t at the assigned bugs management page, they need to press the manage assigned bugs button; once there, the system will show them the list of their assigned bugs and the programmer can do two operations:
  + **mark bug as unassigned**: the programmer selects an bug, then they press the mark as unassigned button, if no bug has been selected, an error message will pop up, otherwise a confirmation dialog will pop up, if the programmer will choose to not continue, the system will return to its initial state, otherwise, if the status change fails, the system will pop up an error message, otherwise the system will show a success message and the bugs list will refresh.
  + **mark bug as fixed**: the programmer selects an bug, then they press the mark as fixed button, if no bug has been selected, an error message will pop up, otherwise a confirmation dialog will pop up, if the programmer will choose to not continue, the system will return to its initial state, otherwise, if the status change fails, the system will pop up an error message, otherwise the system will show a success message and the bugs list will refresh.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and name** | **UC-0: Login into an Account** | | |
| **Primary actor** | Employee | **Secondary actors** | SGBD |
| **Description** | An Employee wants to log in into the application. | | |
| **Trigger** | The Employee wants to use the application. | | |
| **Preconditions** | PRE-0. Employee’s data should be in the SGBD. | | |
| **Postconditions** | POST-0. Employee will see the application based on his permissions. | | |
| **Normal flow** | * 1. **Login into an Account**  1. Employee accesses the web page of the application. 2. BTS displays the login page. 3. Employee enters the credentials and asks to log in. 4. BTS redirects to the main application. | | |
| **Alternative flows** |  | | |
| **Exceptions** | **0.4.E1 Wrong credentials**   1. BTS informs the Employee that the data he typed is incorrect. 2. Return to step 3 to normal flow. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and name** | **UC-1: Create an Account** | | |
| **Primary actor** | Administrator | **Secondary actors** | Employee, SGBD |
| **Description** | An Employee without an account needs one. He reaches an Administrator to create him one. The Administrator accesses the Bug Tracking System, accesses the ‘Manage accounts’ tab, enters an username, a password and the type of the account for the employee, then he gives the account info to him. | | |
| **Trigger** | The Administrator wants to create an account for the Employee. | | |
| **Preconditions** | PRE-1. Administrator is logged into BTS. (See UC-0) | | |
| **Postconditions** | POST-1. The account is created and saved into the SGBD with the given type. | | |
| **Normal flow** | 1. **Create an Account** 2. Administrator asks to view the Manage Accounts tab. 3. BTS displays the Manage Accounts tab. 4. Administrator enters the username, the password and selects the type of the account, then asks to Create Account. 5. BTS displays if he is sure that he wants to create the account with the given information. 6. Administrator selects the confirmation option. 7. BTS displays that the account was successfully created and reloads the list of accounts. | | |
| **Alternative flows** | **1.5.a. Administrator selects the cancel option.**  6.a. BTS displays that the action was canceled. | | |
| **Exceptions** | **1.6.E1 There is already an Employee with the same username**   1. BTS informs the administrator that there is already an user in SGBD who have the same username 2. Return to step 5 to normal flow.   **1.6.E2 Invalid username / password**   1. BTS informs the administrator that the information he provided is invalid. 2. Return to step 5 to normal flow. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and name** | **UC-2: Update an Account** | | |
| **Primary actor** | Administrator | **Secondary actors** | Employee, SGBD |
| **Description** | An Employee forgot his password. He reaches an Administrator to update his password. The Administrator accesses the Bug Tracking System, accesses the ‘Manage accounts’ tab, enters the username and the new password, then he gives him back the new password to him. | | |
| **Trigger** | The Employee came to the Administrator to update his password. | | |
| **Preconditions** | PRE-2. Administrator is logged into BTS. (See UC-0) | | |
| **Postconditions** | POST-2. The account is updated in the SGBD with the new password. | | |
| **Normal flow** | 1. **Create an Account** 2. Administrator asks to view the Manage Accounts tab. 3. BTS displays the Manage Accounts tab. 4. Administrator selects the user from the table and enters a new password, then asks to Update Account. 5. BTS displays if the administrator is sure that he wants to update the account with the given information. 6. Administrator selects the confirmation option. 7. BTS displays that the account was successfully updated and reloads the list of accounts. | | |
| **Alternative flows** | **2.5.a. Administrator selects the cancel option.**  6.a. BTS displays that the action was canceled. | | |
| **Exceptions** | **2.6.E1 There is already no Employee with the given username**   1. BTS informs the administrator that there is no employee with the username given as input. 2. Return to step 5 to normal flow.   **2.6.E2 Invalid user**  **1.** Informs the administrator he did not select the user 2. Return to step 5 to normal flow. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and name** | **UC-3: Delete an Account** | | |
| **Primary actor** | Administrator | **Secondary actors** | Employee, SGBD |
| **Description** | An Employee leaves the company. Because memory is money, the Administrator removes him from the SGBD. | | |
| **Trigger** | The Employee leaved the company. | | |
| **Preconditions** | PRE-3. Administrator is logged into BTS. (See UC-0) | | |
| **Postconditions** | POST-3. The account is deleted from the SGBD. | | |
| **Normal flow** | 1. **Delete an Account** 2. Administrator asks to view the Manage Accounts tab. 3. BTS displays the Manage Accounts tab. 4. Administrator presses the Create Account button. 5. BTS displays an window with the data to complete for the account creation to be done. 6. Administrator enters the username, the password and selects the type of the account, then asks to Delete Account. 7. BTS displays that the account was deleted successfully and reloads the list of accounts. | | |
| **Alternative flows** | **3.5.a. Administrator selects the cancel option.**  6.a. BTS displays that the action was canceled. | | |
| **Exceptions** | **3.6.E1 There is already an Employee with the same username**   1. BTS informs the administrator that there is already an user in SGBD who have the same username 2. Return to step 5 to normal flow.   **3.6.E2 Invalid username / password**   1. BTS informs the administrator that the information he provided is invalid. 2. Return to step 5 to normal flow. | | |

‘

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and name** | **UC-4: Report bug** | | |
| **Primary actor** | Tester | **Secondary actors** | SGBD |
| **Description** | A Tester found a bug in the application and needs to report it. He accesses the Bug Tracking System, accesses the ‘Register Bug’ window from the ‘Manage Submitted Bugs’ tab, enters the bug’s name and a description, then submits it. | | |
| **Trigger** | The Tester found a bug in the application and wants to report it. | | |
| **Preconditions** | PRE-4. Tester is logged into BTS. (See UC-0) | | |
| **Postconditions** | POST-4. The bug is updated in the SGBD and in the list of bugs of all the users logged in the application at that moment. | | |
| **Normal flow** | 1. **Report a bug** 2. Tester asks to view the Manage Submitted Bugs tab. 3. BTS displays the Manage Submitted Bugs tab. 4. Tester enters the name and the description of the bug, then asks to Register Bug. 5. BTS displays if the Tester is sure that he wants to Report the bug. 6. Tester selects the confirmation option. 7. BTS displays that the bug was successfully reported and reloads its list of submitted bugs. | | |
| **Alternative flows** | **4.5.a. Tester selects the cancel option.**  6.a. BTS displays that the action was canceled. | | |
| **Exceptions** | **4.4.E1 Invalid bug data**   1. BTS informs the tester that the name or the description he introduced is invalid. 2. Return to step 3 to normal flow. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and name** | **UC-5: Update bug** | | |
| **Primary actor** | Tester | **Secondary actors** | SGBD |
| **Description** | A Tester needs to change some details at a bug. He accesses the Bug Tracking System, accesses the ‘Register Bug’ window from the ‘Manage Submitted Bugs’ tab, selects the bug, enters the bug’s name and a description, then submits it. | | |
| **Trigger** | The Tester found an error in the reported bug and wants to report it. | | |
| **Preconditions** | PRE-5. Tester is logged into BTS. (See UC-0) | | |
| **Postconditions** | POST-5. The bug is updated in the SGBD and in the list of bugs of all the users logged in the application at that moment. | | |
| **Normal flow** | 1. **Update a bug** 2. Tester asks to view the Manage Submitted Bugs tab. 3. BTS displays the Manage Submitted Bugs tab. 4. Tester selects the bug, enters the name and the description of the bug, then asks to Update Bug. 5. BTS displays if the Tester is sure that he wants to Update the bug. 6. Tester selects the confirmation option. 7. BTS displays that the bug was successfully updated and reloads its list of submitted bugs. | | |
| **Alternative flows** | **5.5.a. Tester selects the cancel option.**  6.a. BTS displays that the action was canceled. | | |
| **Exceptions** | **5.4.E1 No bug selected**   1. BTS informs the tester that the name or the description he introduced is invalid. 2. Return to step 3 to normal flow.   **5.4.E2 Invalid bug data**   1. BTS informs the tester that the name or the description he introduced is invalid. 2. Return to step 3 to normal flow. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and name** | **UC-6: Remove reported bug** | | |
| **Primary actor** | Tester | **Secondary actors** | SGBD |
| **Description** | A Tester submitted a bug, but something went wrong, like a wrong name or a wrong description. He can remove it and then report it again, using the valid information. | | |
| **Trigger** | The Tester did a mistake when reporting a bug. | | |
| **Preconditions** | PRE-6. Tester is logged into BTS. (See UC-0) | | |
| **Postconditions** | POST-6. The bug is updated in the SGBD and in the list of bugs of all the users logged in the application at that moment. | | |
| **Normal flow** | 1. **Remove reported bug** 2. Tester asks to view the Manage Submitted Bugs tab. 3. BTS displays the Manage Submitted Bugs tab. 4. Tester selects the bug, then asks to Remove Bug. 5. BTS displays if the tester is sure that he wants to remove the reported bug. 6. Tester selects the confirm option. 7. BTS displays that the reported bug was successfully removed and reloads its list of submitted bugs. | | |
| **Alternative flows** | **6.5.a. Tester selects the cancel option.**  6.a. BTS displays that the action was canceled. | | |
| **Exceptions** | **6.4.E1 No bug selected**   1. BTS informs the tester that he did not select a bug. 2. Return to step 3 to normal flow. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and name** | **UC-7: Assign bug** | | |
| **Primary actor** | Programmer | **Secondary actors** | SGBD |
| **Description** | A Programmer sees a bug that he knows how to solve, so he marks it as his task. | | |
| **Trigger** | The Programmer decided that he wants to solve a bug. | | |
| **Preconditions** | PRE-7. Programmer is logged into BTS. (See UC-0) | | |
| **Postconditions** | POST-7. The bug is updated in the list of bugs of all the users logged in the application at that moment. | | |
| **Normal flow** | 1. **Assign bug** 2. Programmer asks to view the View Bugs List tab. 3. BTS displays the Manage Submitted Bugs tab. 4. Programmer selects the bug, then asks to Add bug to their tasks. 5. BTS displays if the programmer is sure that he wants to mark the bug as their task. 6. Programmer selects the confirmation option. 7. BTS displays that the bug was successfully marked as their task and reloads its list of tasks. | | |
| **Alternative flows** | **7.5.a Programmer selects the cancel option.**  6.a BTS displays that the action is canceled. | | |
| **Exceptions** | **7.4.E1 No bug selected**   1. BTS informs the programmer that he did not select a bug. 2. Return to step 3 to normal flow. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and name** | **UC-8: Mark bug as Fixed** | | |
| **Primary actor** | Programmer | **Secondary actors** | SGBD |
| **Description** | A Programmer finally solved a bug, so he enters the application and mark it as solved. | | |
| **Trigger** | The Programmer decided that he wants to mark the bug as solved. | | |
| **Preconditions** | PRE-8. Programmer is logged into BTS. (See UC-0) | | |
| **Postconditions** | POST-8. The bug is removed from the list of bugs of all the users logged in the application at that moment. | | |
| **Normal flow** | 1. **Mark bug as Fixed** 2. Programmer asks to view the Manage Tasks tab. 3. BTS displays the Manage Tasks tab. 4. Programmer selects the bug, then asks to Mark it as Solved. 5. BTS displays if the programmer is sure that he wants to mark the bug as solved. 6. Programmer selects the confirmation option. 7. BTS displays that the bug was successfully marked as solved and reloads its list of bugs. | | |
| **Alternative flows** | **8.5.a Programmer selects the cancel option.**  6.a BTS displays that the action is canceled. | | |
| **Exceptions** | **8.4.E1 No bug selected**   1. BTS informs the programmer that he did not select a bug. 2. Return to step 3 to normal flow. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and name** | **UC-9: Unassign Bug** | | |
| **Primary actor** | Programmer | **Secondary actors** | SGBD |
| **Description** | A Programmer gave up on his bug and decides to remove it from its list, the other programmer can take it. | | |
| **Trigger** | The Programmer decided that he wants to mark the bug as solved. | | |
| **Preconditions** | PRE-9. Programmer is logged into BTS. (See UC-0) | | |
| **Postconditions** | POST-9. The bug is marked as unsolved in the list of bugs of all the users logged in the application at that moment. | | |
| **Normal flow** | 1. **Unassign bug** 2. Programmer asks to view the Manage Tasks tab. 3. BTS displays the Manage Tasks tab. 4. Programmer selects the bug, then asks to remove it from his tasks. 5. BTS displays if the programmer is sure that he wants to remove it from his tasks. 6. Programmer selects the confirmation option. 7. BTS displays that the bug was successfully removed from its tasks and reloads its list of bugs. | | |
| **Alternative flows** | **9.5.a Programmer selects the cancel option.**  6.a BTS displays that the action is canceled. | | |
| **Exceptions** | **9.4.E1 No bug selected**   1. BTS informs the programmer that he did not select a bug. 2. Return to step 3 to normal flow. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and name** | **UC-10: View the list of bugs** | | |
| **Primary actor** | Employee | **Secondary actors** | SGBD |
| **Description** | An Employee wants to see how the projects are doing, so how many bugs are there. | | |
| **Trigger** | The Employee wanted to see the list of bugs. (See UC-0) | | |
| **Preconditions** | PRE-10. Employee is logged into BTS. | | |
| **Postconditions** |  | | |
| **Normal flow** | 1. **View the list of bugs** 2. Employee asks to view the View Bugs List tab. 3. BTS displays the View Bugs List tab. | | |
| **Alternative flows** |  | | |
| **Exceptions** |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID and name** | **UC-11: Log out** | | |
| **Primary actor** | Employee | **Secondary actors** | SGBD |
| **Description** | An Employee may want to leave the computer for some moments, and, to avoid another to do some malicious stuff, he logs out of the application. | | |
| **Trigger** | The Employee wanted to log out from the application. | | |
| **Preconditions** | PRE-11. Employee is logged into BTS. (See UC-0) | | |
| **Postconditions** | POST-11. Employee is no longer logged into BTS. | | |
| **Normal flow** | 1. **Logout** 2. Employee asks to log out of the application. 3. BTS logs out the Employee and shows the login page. | | |
| **Alternative flows** |  | | |
| **Exceptions** |  | | |

# **Concept Model**



Figure 2. Model Classes Diagram

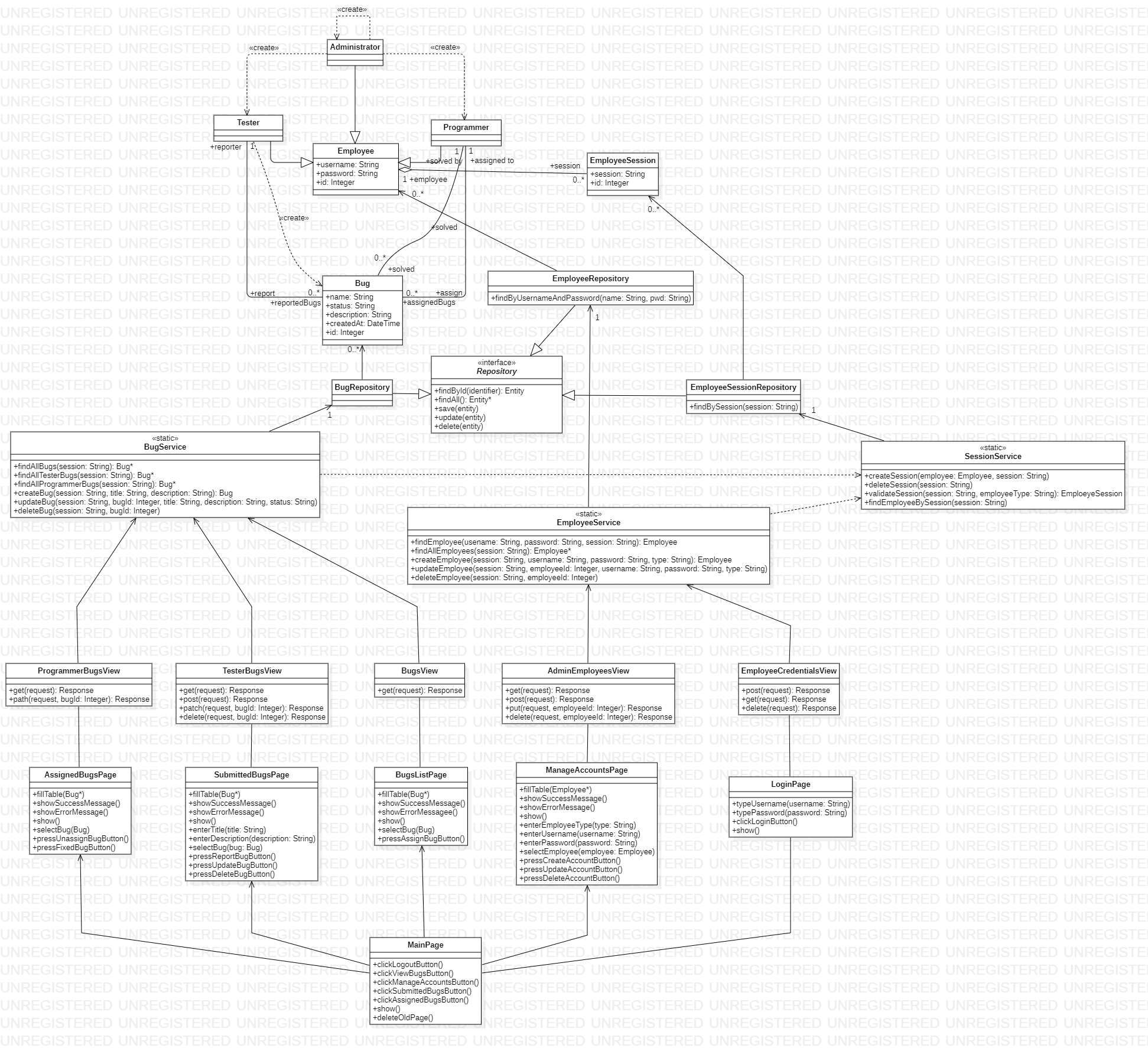


Figure 3. Finished Class Diagram

# **Database Diagram**

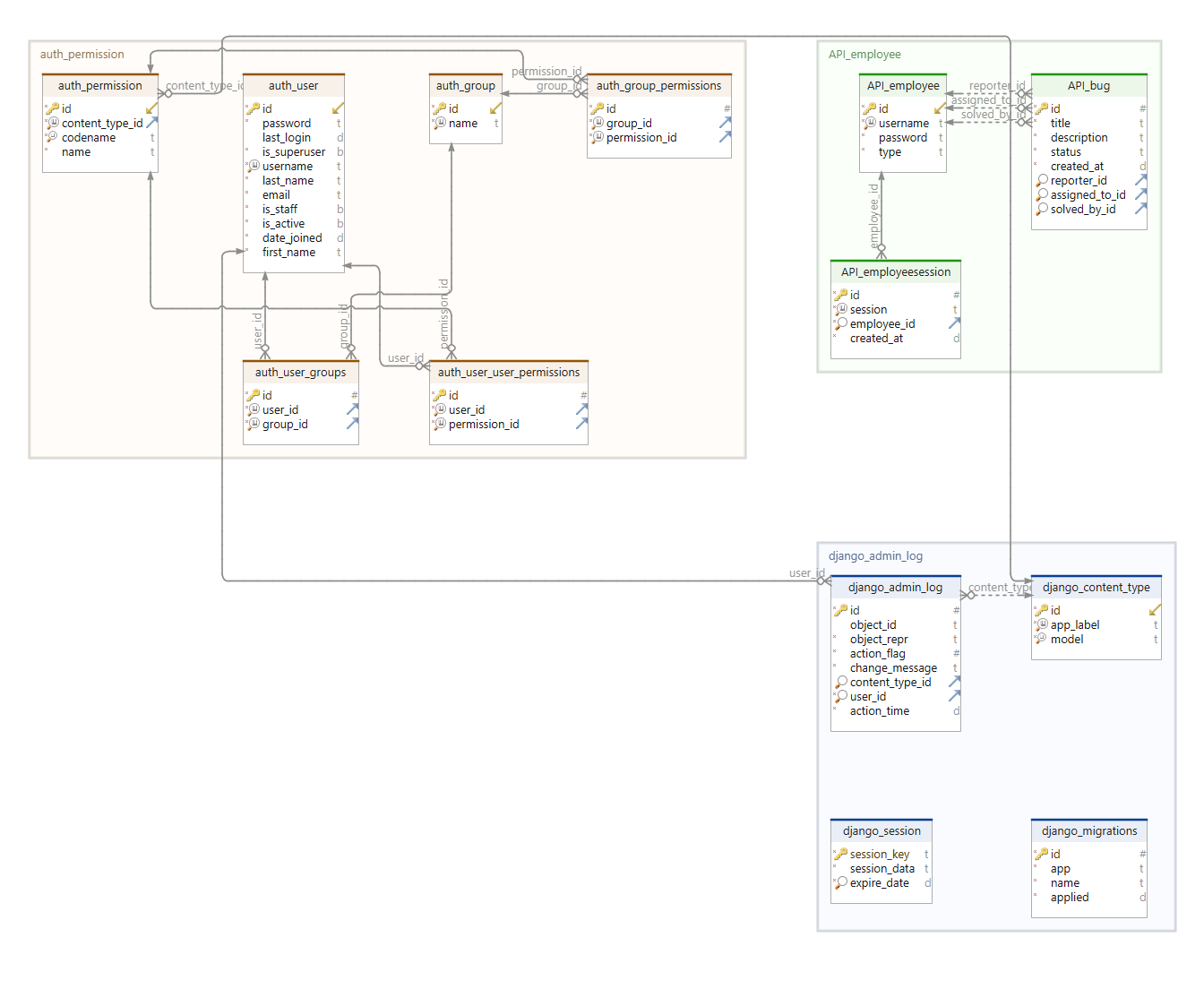


Figure 4. Database Diagram

# **Dynamic Model**



Figure 5. Login Sequence Diagram - Normal Flow



Figure 6. Logout Sequence Diagram - Normal Flow



Figure 7. View Bugs List Sequence Diagram - Normal Flow



Figure 8. Manage Accounts Sequence Diagram - Normal Flow



Figure 9. Create Account Sequence Diagram - Normal Flow



Figure 10. Update Account Communication Diagram - Normal Flow

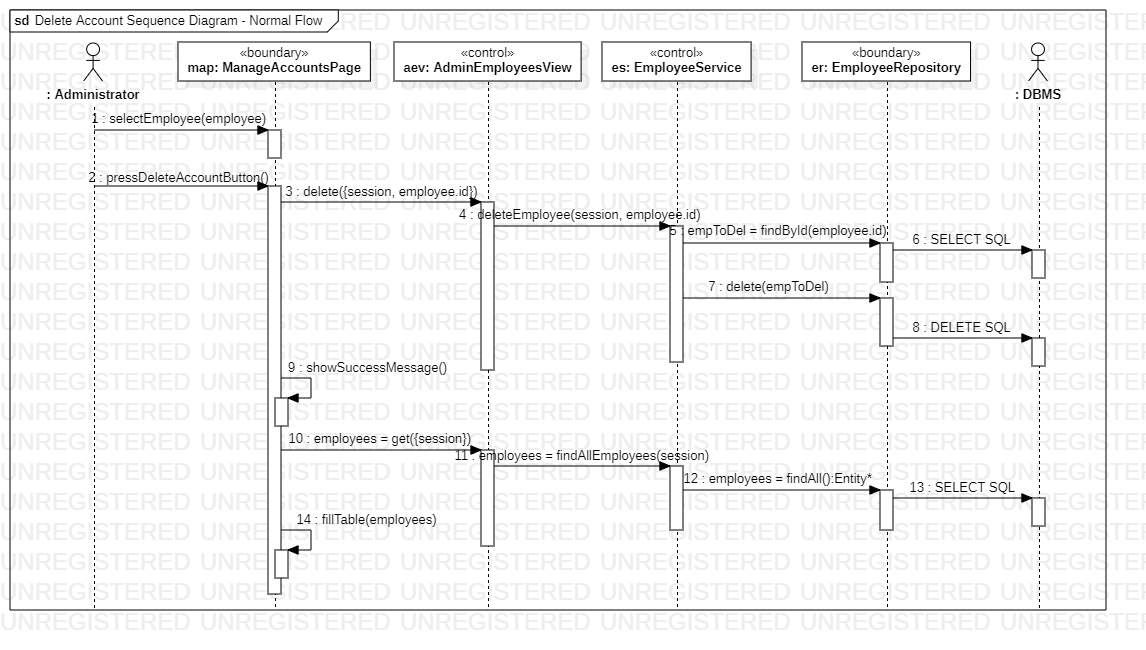


Figure 11. Delete Account Sequence Diagram - Normal Flow



Figure 12. Manage Submitted Bugs Sequence Diagram - Normal Flow



Figure 13. Report Bug Sequence Diagram - Normal Flow



Figure 14. Update Bug Communication Diagram - Normal Flow



Figure 15. Remove Bug Sequence Diagram - Normal Flow



Figure 16. Manage Assigned Bugs Sequence Diagram - Normal Flow



Figure 17. Assign Bug Sequence Diagram - Normal Flow



Figure 18. Mark Bug as Unassigned Communication Diagram - Normal Flow



Figure 19. Mark Bug as Fixed Sequence Diagram - Normal Flow

# **Technical Documentation**

**Non-Functional requirements**:

* the application must be a web application, it must offer a good user experience for any employee, and it must be easy to use for anybody.
* the application must use some colors such that no colorblind person will face problems.

**Used technologies**:

* **database**: the used dialect was SQLite because is easy to use, doesn’t take up much space and the required problem doesn’t need many types of entities which other databases come with.
* **backend**: the used language was Python 3 because is easy to maintain if the code is written correctly and the code needed to solve this problem isn’t that big, which makes it easier to find problems with the application.
* **ORM**: the object-relation mapping used was Django because it offers many features that makes the development of the application cleaner and easier to understand, it also offers tools to make the testing of the application easier.
* **frontend**: the used language was React 18 because it’s easy to use and it offers strong features, such as hooks, that make the user experience better.
* **diagrams**: the design of the diagrams was made using StarUML because it offers the required tools to project any diagram and it’s easy to use, and DbSchema for the databases diagrams.
* **environment**: the code was edited in PyCharm because it offers many features and shortcuts to make the coding experience better.

# **Help**

Once the user opens the application, they must introduce their username and password, then press the login button [Fig. 20]. If the login will not be successful, an error message will pop up, otherwise the user will be redirected to the main page.

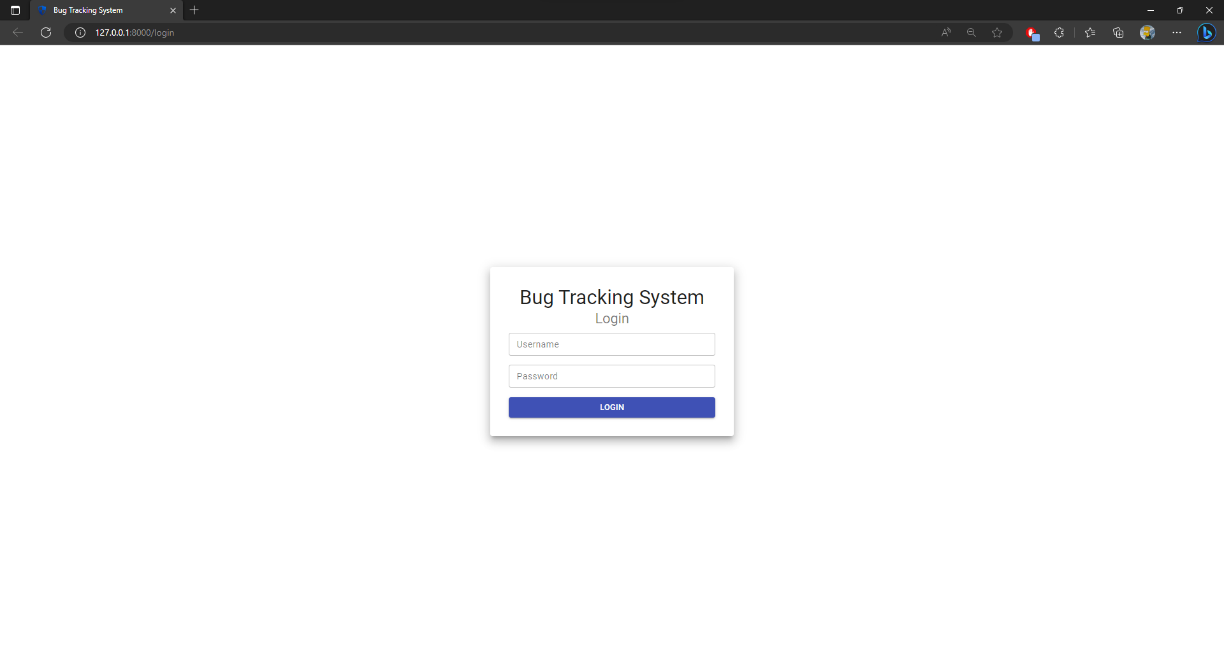


Figure 20. Login Page

Any user will have the option to view the list of bugs (they need to select the option on the left navigation bar if they are in another page), if any bug is selected, some details about it will be shown below the table, if the user is also a programmer and the bug is not assigned yet, an option to assign the bug will be available [Fig. 21]. The other option each user has is the log out button, and when that’s pressed, the user will be logged out of the application and redirected to the login page [Fig. 20].

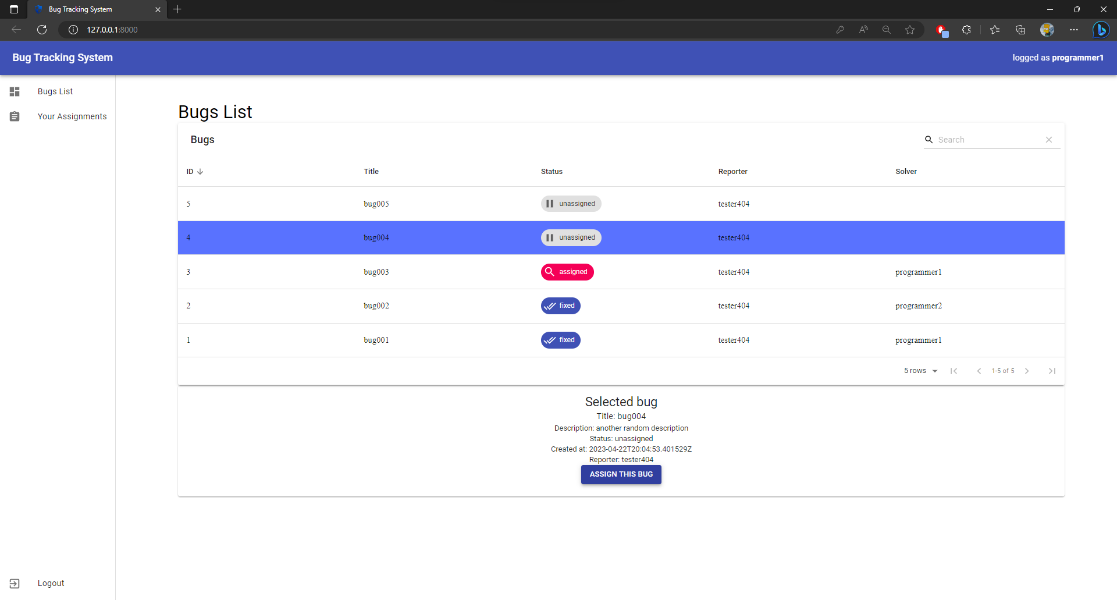


Figure 21. Bugs List and the Bug Assignation Option

The administrators will see the Manage Accounts button, when they press it, they will be redirected to a page with all the accounts registered in the system, they have the options to create a new account, to update and to delete an existing one [Fig. 22]. The update and the delete option will be available only when an account is selected, the create account will be available all times, and if one of them is pressed and the operation fails, an error message will pop up, otherwise a success message will be shown.

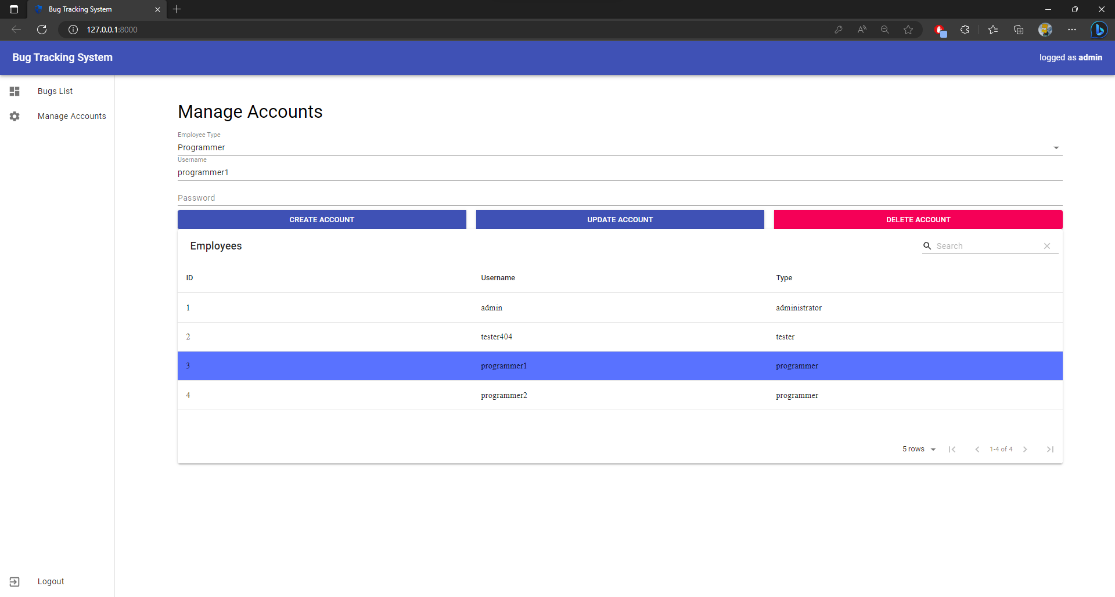


Figure 22. Manage Accounts Page

The tester will see the Submitted Bugs button, when they press it, they will be redirected to a page with all their submitted bugs. There, they can report a bug by entering its title and its description, to update a bug and to delete a bug [Fig. 23]. The update and the delete operations can be done only on unassigned bugs, if the bug is already assigned, those options will be disabled. When one of the buttons is pressed and the operation fails, an error message will pop up, otherwise a success message will be shown.

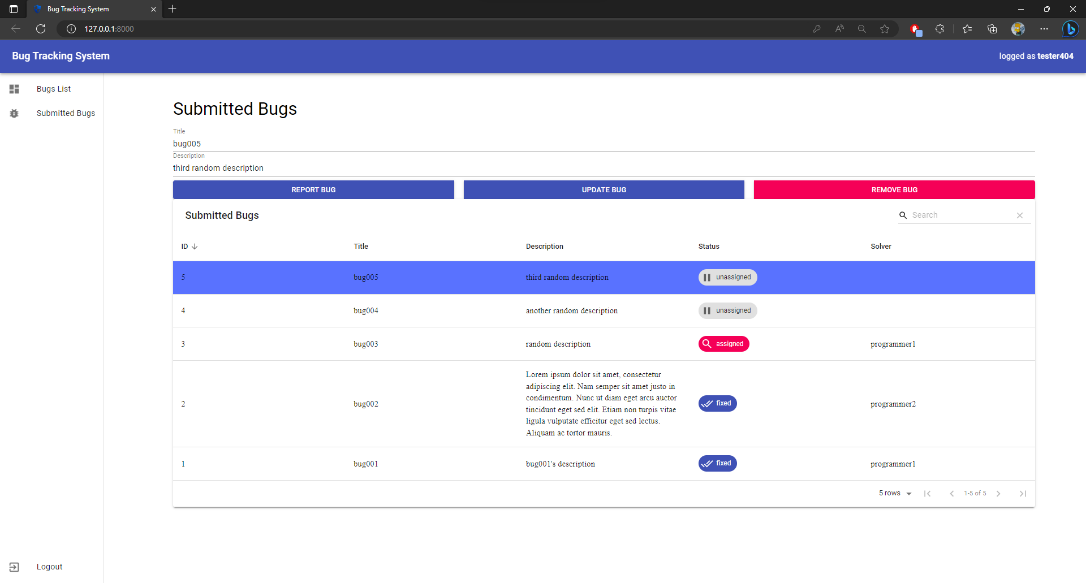


Figure 23. Submitted Bugs Page

The programmer will see the Your Assignments button, when they press it, they will be redirected to a page with all their assigned bugs. There, two buttons can be pressed, the mark as unassigned one, where the programmer remove their assignation, and the mark as fixed one, which is pressed when the programmer solved the bug. When one of the buttons is pressed and the operation fails, an error message will pop up, otherwise a success message will be shown.

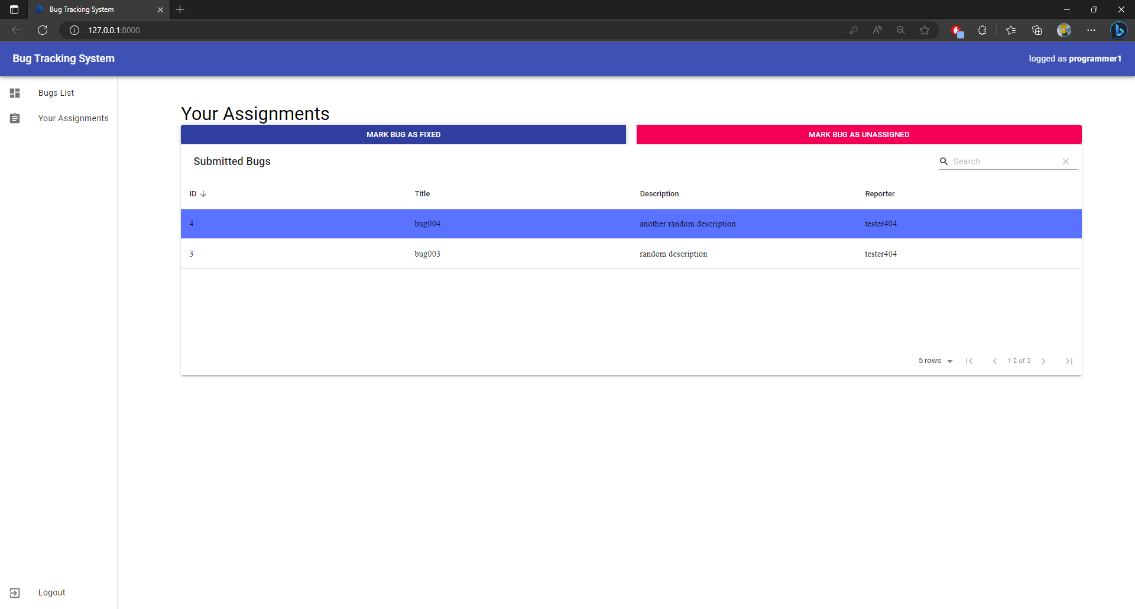


Figure 24. Your Assignments Page

# **Bibliography**

1. SQLite Documentation. <https://www.sqlite.org/docs.html>
2. Python3 Documentation. <https://docs.python.org/3/>
3. Czibula Istvan, Programming Fundamentals notes (python). <http://www.cs.ubbcluj.ro/~istvanc/fp/>
4. Czibula Istvan, Object-Oriented Programming notes. <http://www.cs.ubbcluj.ro/~istvanc/oop/>
5. Cojocar Grigoreta, System Designs and Implementation notes. <http://www.cs.ubbcluj.ro/~grigo/>
6. Django Documentation. <https://docs.djangoproject.com/en/4.2/>
7. Django + React Full Stack Web App Course. <https://www.classcentral.com/course/youtube-django-react-full-stack-web-app-tutorial-54816>
8. React 18 Documentation. <https://react.dev/>
9. Material-UI Documentation. <https://mui.com/>
10. StarUML Documentation. <https://docs.staruml.io/>
11. Petrașcu Vladiela, Software Engineering notes. <http://www.cs.ubbcluj.ro/~vladi/>
12. Sima Ioan, Final project structure. <https://simaioan.wordpress.com/2020/05/09/proiect-final-iss/>
13. Microsoft 365 Word Documentation. <https://support.microsoft.com/en-us/word>
14. DbSchema documentation. <https://dbschema.com/documentation/>