**Use Case Specification: Login**

**Brief Description:**

This use case describes how the User authenticates into the application.

**Basic Flows**

**B1. Login**

The use case begins when a User wants to Login.

1. The user opens the application.

2. The system displays the login form.

3. The user enters credentials.

4. The system deducts from the database if the credentials are correct.

5. The system displays the actions the user can further do.

6. The use case terminates.

**Alternate Flows**

**A1. Invalid credentials - reenter**

In step 4 the system displays that the credentials entered are wrong . The user can go back to step 3.

**A2. Invalid credentials – ask for account**

In step 4 the system displays that the credentials entered are wrong . The user can submit a request to the admin to create an account for him. After that, the user can go back to step 3.

**Preconditions**

-

**Postconditions**

Upon successful authentication, the tester sees the list of projects and chooses 1 of them to add bugs on it and the

**Use Case Specification: AddBug**

**Brief Description:**

This use case describes how the Tester adds a bug.

**Basic Flows**

**B1. AddBug**

The use case begins when a Tester wants to add a bug.

1. The tester sees the list of projects and chooses 1 of them.

2. The system displays the ‘add bug’ form.

3. The tester enters the bug information and submits the form.

4. The system saves the bug in DB and notifies the users.

5. The use case terminates.

**Alternate Flows**

**A1. Already existing bug**

After step 3, if the bug description already exists in the system, the tester will be notified and can start again.

**Preconditions**

The tester has to be logged in to the system.

**Postconditions**

Upon successful bug addition, the tester can add more bugs and the bug is saved in DB to be solved by programmers and the Programmer sees the list of projects and chooses 1 of them to solve.

**Use Case Specification: FixBug**

**Brief Description:**

This use case describes how the Programmer fixes a bug.

**Basic Flows**

**B1. FixBug**

The use case begins when a Programmer wants to fix a bug.

1. The Programmer sees the list of projects and chooses 1 of them.

2. The Programmer sees the list of bugs for that project and select 1 of them.

2. The system displays the bug information.

3. The user changes the bug state to FIXED.

4. The system saves the bug in DB and notifies the users.

5. The use case terminates.

**Alternate Flows**

**-**

**Preconditions**

The user has logged in to the system.

**Postconditions**

Upon successful bug fixing, the programmer can solve more bugs and the bug is saved in DB as fixed.