The Bug Tracking System

Contents

[I. Project Details 3](#_Toc97220598)

[II. Structure of the system 4](#_Toc97220599)

[1. System Design 4](#_Toc97220600)

[2. Modules 5](#_Toc97220601)

[2.1 Customer Module: 5](#_Toc97220602)

[2.2 Manager Module: 6](#_Toc97220603)

[2.3 Employee Module: 6](#_Toc97220604)

[3. Components 7](#_Toc97220605)

[3.1 Sign-Up: 7](#_Toc97220606)

[3.2 Login Validation: 7](#_Toc97220607)

[3.3 Dashboard Display: 7](#_Toc97220608)

[3.4 Bug Creation: 7](#_Toc97220609)

[3.5 Bug View/Edit Option: 7](#_Toc97220610)

[3.6 Email Trigger: 8](#_Toc97220611)

[3.7 Project Report: 8](#_Toc97220612)

[4. User Roles and Authorities requirement 8](#_Toc97220613)

[4.1 Customer Authorities: 8](#_Toc97220614)

[4.2 Manager Authorities: 8](#_Toc97220615)

[4.3 Employee Authorities: 8](#_Toc97220616)

[III. Requirement Specification 9](#_Toc97220617)

[1. Functional Requirements 9](#_Toc97220618)

[1.1 Components Functional Requirement with respect to each module 9](#_Toc97220619)

[1.2 Database Requirement 13](#_Toc97220620)

[2. Non-Functional Requirements 14](#_Toc97220621)

[2.1 Security Requirements: 14](#_Toc97220622)

[2.2 Privacy Requirements: 14](#_Toc97220623)

[2.3 Scalability and performance Requirements: 14](#_Toc97220624)

[2.4 Usability Requirement: 14](#_Toc97220625)

[2.5 Compatibility Requirement: 14](#_Toc97220626)

[3. Interfaces (user, hardware, software, and/or communication) 14](#_Toc97220627)

[3.1 Hardware Interface Requirements: 14](#_Toc97220628)

[3.2 Software Interfaces Requirements: 15](#_Toc97220629)

[3.2 User Interface Requirement: 15](#_Toc97220630)

[3.3 Communication Interface Requirement: 20](#_Toc97220631)

[IV. Project Implementing Plan 20](#_Toc97220632)

[1. Development Phase -1: 20](#_Toc97220633)

[1.1 Developing Database tables (Bug and Login Tables). 20](#_Toc97220634)

[1.2 Developing the application layer for the webpage. 20](#_Toc97220635)

[1.3 Developing Webpages for all Modules. 20](#_Toc97220636)

[1.4 Integrating Database with the Front-End applications. 20](#_Toc97220637)

[2. Development Phase-2: 20](#_Toc97220638)

[2.1 Develop back-end code for Customer, Manager and Employee Modules. 20](#_Toc97220639)

[2.2 Integrate Components in the Customer, Manager and Employee Modules with the Front-End Applications and the Database. 20](#_Toc97220640)

[2.3 Perform Initial testing. 21](#_Toc97220641)

[3. Development Phase-3: 21](#_Toc97220642)

[3.1 Performing integrated system testing 21](#_Toc97220643)

[3.2 Preparing test cases and identifying Bugs. 21](#_Toc97220644)

[3.3 Fixing the identified bugs. 21](#_Toc97220645)

[3.4 Re-testing until all the requirements are met. 21](#_Toc97220646)

[V. Member contribution table 22](#_Toc97220647)

Table of Figures

[Figure 1: High Level Design of the Bug Tracking System 4](#_Toc97154878)

[Figure 2: System Structure Diagram of the Bug Tracking System. 5](#_Toc97154879)

[Figure 3: Sign-Up Page 15](#_Toc97154880)

[Figure 4: Login Page 15](#_Toc97154881)

[Figure 5:Dashboard Display Page in Customer View 16](#_Toc97154882)

[Figure 6: Dashboard Display Page in Manager View 16](#_Toc97154883)

[Figure 7: Dashboard Display in Employee View 17](#_Toc97154884)

[Figure 8:Bug Creation Page in Customer View 17](#_Toc97154885)

[Figure 9: Bug page display 18](#_Toc97154886)

[Figure 10: Project Report Summary in Manager View 19](#_Toc97154887)

# Project Details

**Project Title:** The Bug Tracking System

**Group Name:** BUG MAFIA

**Group Members:**

1. Neha Goud Baddam (Team Leader)
2. Vamsi Sai Konidena
3. Reshmi Chowdary Divi
4. Prem Kumar Maddula
5. Rahul Mandaloju
6. Harika Uppalapati
7. Purandhara Maharshi Chidurala
8. Shiva Surya Vardhan Reddy

# **Structure of the system**

## System Design

The Users access the Bug Tracking System Web Page to track the bugs in their software.

Diagram

Description automatically generated

Figure 1: High Level Design of the Bug Tracking System

Below is the complete System Structure Diagram of the Bug Tracking System, that is explained in detail in the “2. Modules” and “3. Components” sections under “II. Structure of the system” heading of the document.

Diagram

Description automatically generated

Figure 2: System Structure Diagram of the Bug Tracking System.

## Modules

The overall system structure is described with respect to each module. Below are the three Main Modules:

### Customer Module:

1. Customer should type the URL and enters his/her login credentials, which will be validated.
2. If new user, they can register by clicking “Sign-Up” on the login page by selecting role as “Customer”.
3. Once logged in Dashboard displays all the tickets created by the user.
4. Customer can create new bug, or they can view/edit existing Bugs.
5. When Customer creates the ticket status is “To Do” by default.
6. Manager moves the status to “Closed” and assigns it back to the Customer. If the issue persists Customer can re-open the ticket and change the status from “Closed” to “Re-Open”.
7. Whenever the Bug ticket is assigned to the Manager, triggers an email to the assignee, saying that the ticket is assigned.

### Manager Module:

1. Manager should type the URL and enters his/her login credentials, Validation of credentials is done.
2. If new user, they can register by clicking “Sign-Up” on the login page by selecting role as “Manager”.
3. Dashboard displays all the tickets assigned to the manager by the customer.
4. A button “Project Report” to show number of bugs each employee in the project is working on is displayed on Dashboard.
5. Bug View will be same for every role but authorities to make changes to few fields will vary.
6. Manager should assign “To Do” or “Re-Open” status bugs to the employees.
7. After the Bug is resolved, employee assigns the Bug ticket back to the manager and the manager moves the status to “Closed” from the status “Resolved” and assigns it back to the Customer.
8. If the issue persists Customer can re-open the ticket. Change from status “Closed” to “Re-Open” and can assign the bug back to Manager.
9. Manager can check all the reports that are currently active in the project (tickets that are assigned to other managers that are in “To Do” or in “Re-Open” status / Tickets that are assigned to employees).
10. Report will have filters based on fields assignee, status, priority, and Type. Display valid fields like Bug Id, Summary, Assignee, Status, priority, and Type in the report, this report can be downloaded.
11. Whenever the Bug ticket is assigned to the Manager/Employee, it triggers an email to the assignee, saying that the ticket is assigned.

### Employee Module:

1. Employee should type the URL and enters his/her login credentials. Validation of credentials is done.
2. If new user, they can register by clicking “Sign-Up” on the login page with role as “Employee”.
3. Dashboard displays all the tickets the employee is currently working on.
4. Bug View will be same for every role but authorities to make changes to few fields will vary.
5. After the Bug is assigned to the employee, it is in “To Do” status.
6. Employee must change the ticket status in below SDLC flow: “To Do” or “Reopen” to “In Analysis” to “In Development” to “In QA” to “Resolved”
7. Whenever the Bug ticket is assigned to the Manager/Employee, triggers an email to the assignee, saying that the ticket is assigned.

## Components

The Bug tracking System consists of below components:

### Sign-Up:

Users should sign up with the email id and select their role (Customer/Manager/Employee).

### Login Validation:

Users should type the URL and enters his/her login credentials, validation of credentials is done as a part of this component.

### Dashboard Display:

Displays all the tickets the user is currently working on.

### Bug Creation:

Bug Create component is only available for the Customers, to create a new bug.

### Bug View/Edit Option:

Bug View will be same for every role but authorities to edit few fields will vary with the user.

#### Activity Log:

All changes history that was made on the ticket will be recorded (simple record consisting of changes. Ex: User XXX added comment or User XXX changed assignee)

### Email Trigger:

Whenever the Bug ticket is assigned to the Manager/Employee, triggers an email to the assignee, saying that the ticket is assigned.

### Project Report:

This component is only available for the Manager, to track all the bugs and download report based on the filters available.

## User Roles and Authorities requirement

Below are the three roles:

### Customer Authorities:

* Customer has authority to create and assign ticket to the manager. (Cannot assign directly to employees)
* Customer can View/update existing bugs have created by the current user.
* If the issue/bug persists Customer can re-open the existing ticket by changing status from status “Closed” to “Re-Open”

### Manager Authorities:

* Once the ticket is assigned to manager, they assign it to any employee.
* Once the ticket is in “Resolved” status, manager assigns it back to the customer marking the status as “Closed”.

### Employee Authorities:

* Once the ticket is assigned to the employee, updates estimated completion date and moves the ticket from “To Do” or “Re-Open” to “In analysis” to “In Development” to “In QA” to “Resolved”.
* If unable to resolve the bug, employee updates the completion dates or adds comments or Can also re-assign ticket to another employee. (Cannot assign it back to Customer directly)
* Once the development and testing are completed, employee changes the ticket status to “Resolved” and assigns it back to the manager.

# Requirement Specification

## Functional Requirements

### Components Functional Requirement with respect to each module

Initially whenever the employee/Manager/Customer joins the organization, they are given user id and password.

#### Customer Module Requirements:

##### Sign-Up:

Customers can sign up with their organizations email id with Role as Customer.

##### Login Validation:

Customer should type the URL and enters his/her login credentials. Validation of credentials is done.

##### Dashboard Display:

Dashboards display all the tickets created by the user with a button “Create Bug” to create new Bug/Ticket.

##### Bug Creation:

Web page will have below fields in Bug creation page:

1. Type: Type of ticket must be mentioned i.e., either Bug / New (Bug-existing issue, New-new requirement).
2. Summary: Title for the Bug.
3. Description: Describe Bug.
4. Status: It is “To Do” by default while creating.
5. Priority: Set bug priority can be set (low/medium /high).
6. Comments: Comments can be added.
7. Attachments: Documents/Files related to bug can be uploaded.
8. Assignee: Assignee must be only the managers – tickets can be assigned only to people with manager authority.

Web page will populate below fields in the backend into the same record in the database:

1. Creation Date: Gets timestamp and populates this field.
2. Reporter: Current user will be the Bug reporter.
3. Last Updated Time: Timestamp of last updated changes.
4. Last Updated User: User who updated the changes.

##### Bug View/Edit Option:

* (Bug View will be same for every role but authorities to make changes to few fields will vary)

Customer can edit the fields at any point.

Customer creates the ticket and assigns it to the manager.

Can change below fields in Bug Page:

1. Type: Type of ticket must be mentioned i.e either Bug / New (Bug-existing issue, New-new requirement).
2. Summary: Title for the Bug.
3. Description: Describe Bug.
4. Status: It is “To Do” by default while creating.

Manager moves the status to “Closed” and assigns it back to the Customer. If the issue persists Customer can re-open the ticket. Change from status “Closed” to “Re-Open”.

1. Priority: Set bug priority can be set (low/medium /high).
2. Comments: Comments can be added.
3. Attachments: Documents/Files related to bug can be uploaded.
4. Assignee: Assignee must be only the managers – tickets can be assigned only to people with manager authority.

Web page will populate below fields in the backend into the same record in the database whenever the changes are made to the Bug ticket:

1. Creation Date: Gets timestamp and populates the field,
2. Reporter: Current user will be the Bug reporter.
3. Last Updated Time: Timestamp of last updated changes.
4. Last Updated User: User who updated the changes.

##### Activity Log:

It is a sub-component within the Bug View. All changes history that was made on the ticket will be recorded (simple record consisting of changes.

Ex: User XXX added comment or User XXX changed assignee)

##### Email Triggers:

Whenever the Bug ticket is assigned to the Manager, trigger an email to the assignee, saying that the ticket is assigned.

#### Manager Module Requirement:

##### Sign-Up:

Manager can sign up with their organizations email id with role as Manager.

##### Login Validation:

Manager should type the URL and enters his/her login credentials. Validation of credentials is done.

##### Dashboard Display:

Dashboard display all the tickets assigned to the manager by the client. Also has a button “Project Report” to show number of bugs each employee in the project is working on.

##### Bug View/Edit Option:

Bug View will be same for every role but authorities to make changes to few fields will vary. Manager can edit the fields at any point, make sure to log the changes.

Manager can change below fields in Bug Page:

1. Type: Type of ticket must be mentioned i.e either Bug / New (Bug-existing issue, New-new requirement)
2. Summary: Title for the Bug.
3. Description: Describe Bug.
4. Status:

Manager should assign “To Do” or “Re-Open” status bugs to the employees. After the Bug is resolved, employee assigns the Bug ticket back to the manager and the manager moves the status to “Closed” from the status “Resolved” and assigns it back to the Customer. If the issue persists Customer can re-open the ticket. Change from status “Closed” to “Re-Open”

1. Priority: Set bug priority can be set (low/medium /high).
2. Comments: Comments can be added.
3. Attachments: Documents/Files related to bug can be uploaded.
4. Assignee: Manager can assign ticket to anyone in the project (both employees and Customers).

Web page will populate below fields in the backend into the same record in the database whenever the changes are made to the Bug ticket:

1. Last Updated Time: Timestamp of last updated changes.
2. Last Updated User: User who updated the changes.

##### Activity Log:

It is a sub-component within the Bug View. All changes history that was made on the ticket will be recorded (simple record consisting of changes. Ex: User XXX added comment or User XXX changed assignee)

##### Project Report:

* Manager can check all the reports that are currently active in the project (tickets that are assigned to other managers that are in “To Do” or in “Re-Open” status / Tickets that are assigned to employees).
* Report should have filters based on fields assignee, status, priority, and Type. Display valid fields like Bug Id, Summary, Assignee, Status, priority, and Type in the report.
* There is also an option to download the report.

##### Email Triggers:

Whenever the Bug ticket is assigned to the Manager/Employee, trigger an email to the assignee, saying that the ticket is assigned.

#### Employee Module Requirement:

##### Sign-Up:

Employee can sign up with their organizations email id with Role as Employee.

##### Login Validation:

Employee should type the URL and enters his/her login credentials. Validation of credentials is done.

##### Dashboard Display:

Display all the tickets the employee is currently working on.

##### Bug View/Edit Option:

Bug View will be same for every role but authorities to make changes to few fields will vary

Employee can change below fields in Bug Page:

1. Status: “To Do” or “Reopen” to “In Analysis” to “In Development” to “In QA” to “Resolved”

After the Bug is assigned to the employee, it is in “To Do” status.

Employee must change the ticket status in below SDLC flow:“To Do” or “Reopen” to “In Analysis” to “In Development” to “In QA” to “Resolved”

1. Comments: Comments can be added.
2. Attachments: Documents/Files related to bug can be uploaded.
3. Assignee: Employee can assign Bug tickets to other employees and the manager but not to the Customer.

Web page will populate below fields in the backend into the same record in the database whenever the changes are made to the Bug ticket:

1. Last Updated Time: Timestamp of last updated changes
2. Last Updated User: User who updated the changes

##### Activity Log:

It is a sub-component within the Bug View. All changes history that was made on the ticket will be recorded (simple record consisting of changes. Ex: User XXX added comment or User XXX changed assignee)

##### Email Trigger:

Whenever the Bug ticket is assigned to the Manager/Employee, trigger an email to the assignee, saying that the ticket is assigned.

### 1.2 Database Requirement

#### 1.2.1 Bug Database Requirement:

Database will have **records of all the tickets/bugs** created by the Customer. (A unique Bug ID number will be assigned to every ticket that is created)

Bug Database will have below fields:

1. Bug ID – Unique number automatically created in program
2. Type: Type of ticket must be mentioned i.e either Bug / New (Bug-existing issue, New-new requirement)
3. Summary: Title for the Bug
4. Description: Describe Bug
5. Status: “To Do” – by default while creating
6. Priority: set bug priority (low/medium /high)
7. Comments: add comments
8. Attachments: adding documents
9. Assignee: Assignee must be only the managers – tickets can be assigned only to people with manager authority
10. Creation Date: get timestamp and populate the field
11. Reporter: Current user will be the Bug reporter.
12. Last Updated Time: Timestamp of last updated changes
13. Last Updated User: User who updated the changes

#### Login Database Requirement:

Login Database will have below fields:

1. Username: Username will be given to each employee/Customer/Manager by the organization
2. Password: Password will be given to each employee/Customer/Manager by the organization.
3. Role: Role “Manager”/ “Customer” / “Employee” will be given by the organization.
4. Last Updated Time: Timestamp of record addition/updation.

## Non-Functional Requirements

### Security Requirements:

We will be using JavaScript functions for User Authentication and securing user data. Even during DOS attacks, the software must be available and behave consistently. The integrity of the customer account information must be ensured by the software. Any browser that the server cannot authenticate should not be served a restricted web page.

### Privacy Requirements:

The application doesn’t show any personal information of users apart from the email id that will be used while registering. We will be using Java Script for Authenticating the login credentials.

### Scalability and performance Requirements:

All the employees in the organization will be working on the bug tracking system, as a part of their daily job. Hence the web page needs to be highly responsive and should perform at a greater speed, we will be using XAMPP server that is highly scalable.

### Usability Requirement:

Usability plays an important role as this application works as a bridge between the user interface should be well-designed. We will be using jQuery and Ajax programming for making web pages user friendly. For the users to learn quickly and understandably, we'll take them on a tour of the options and actions available to them.

### Compatibility Requirement:

Our website should be compatible in working in the following browsers

1. Chrome Browser Version 98.0.0.0 or above
2. Microsoft Edge Version 90 or above
3. Safari Browser Version 13.1.2 or above

## Interfaces (user, hardware, software, and/or communication)

### Hardware Interface Requirements:

This application would need a browser installed on **Laptop / PC**.

### Software Interfaces Requirements:

1. **HTML**: We will be using HTML for website Designing.
2. **JQUERY:** We will be using jQuery for styling the website Design.
3. **AJAX:** We will be using AJAX dynamically loading the webpages.
4. **JavaScript**: We will be using JavaScript for making webpages interactive and to connect to the database and server. Also, for email verification, Registration and Login Java Script is used.
5. **PHP:** PHP is the language that will be used for developing the web application.
6. **SQL:** We will be using SQL foe accessing the database.
7. **Visual Studio:** We will be using Visual Studio as Integrated Development Environment for the Project
8. **XAMPP:** XAMPP will be used as a server for deploying this project.

### User Interface Requirement:

#### Sign-Up Interface:

In Sign-Up interface, Users should sign in with the email id and select their role (Customer/Manager/Employee).

A picture containing text, computer, keyboard, computer

Description automatically generated

Figure 3: Sign-Up Page

#### Login Interface:

In login interface, Users should type the URL and enters his/her login credentials, validation of credentials is done as a part of this component.

A picture containing text, computer, keyboard, electronics

Description automatically generated

Figure 4: Login Page

#### Dashboard Interface:

Dashboard interface displays all the tickets the user is currently working on.

* Web page will display a “create bug” button in Customer View.

Graphical user interface, application

Description automatically generated

Figure 5:Dashboard Display Page in Customer View

* Web page will display a “Project Report” button in Manager View.

Graphical user interface, application

Description automatically generated

Figure 6: Dashboard Display Page in Manager View

* All tickets assigned to Employee are shown

Graphical user interface, application, website

Description automatically generated

Figure 7: Dashboard Display in Employee View

#### Bug Creation:

Bug Create interface is only available for the Customers, to create a new bug.

Graphical user interface, application

Description automatically generated

Figure 8:Bug Creation Page in Customer View

#### Bug View/Edit Option:

Bug View/Edit Interface will be same for every role but authorities to edit few fields will vary with the user.

Graphical user interface, application, website

Description automatically generated

Figure 9: Bug page display

#### Project Report:

This interface is only available for the Manager, to track all the bugs and download report based on the filters available.

Graphical user interface, application

Description automatically generated

Figure 10: Project Report Summary in Manager View

### Communication Interface Requirement:

#### Email Trigger:

Whenever the Bug ticket is assigned to the Manager/Employee, triggers an email to the assignee, saying that the ticket is assigned.

# Project Implementing Plan

Below are the three development phases for developing the project:

## 1. Development Phase -1:

### Developing Database tables (Bug and Login Tables).

**Requirements:**

We will be using MySQL for Database creation.

### Developing the application layer for the webpage.

**Requirements:**

We will be using JAVASCRIPT for Application layer development.

### Developing Webpages for all Modules.

**Requirements:**

We will be using HTML, AJAX and JQUERY for the development of Webpages.

### Integrating Database with the Front-End applications.

**Requirements:**

We will be using HTML, AJAX, JQUERY and JAVASCRIPT for integration, also for login and registration validation.

## 2. Development Phase-2:

### Develop back-end code for Customer, Manager and Employee Modules.

**Requirements:**

We will be using PHP and JAVASCRIPT for coding each component with respect to Customer, Manager and Employee Roles and authorities.

### Integrate Components in the Customer, Manager and Employee Modules with the Front-End Applications and the Database.

**Requirements:**

We will be using Visual Studio as Integrated development Environment.

### Perform Initial testing.

**Requirements:**

The application will be deployed through XAMPP server, once done few test cases can be prepared and tested as a part of initial testing.

## 3. Development Phase-3:

### Performing integrated system testing

**Requirements:**

Test cases can be prepared to test the complete code and must be documented in the Microsoft Word.

### Preparing test cases and identifying Bugs.

**Requirements:**

Performing repeated testing and resolving bugs and documenting the results by making use of MS Office.

### Fixing the identified bugs.

**Requirements:**

PHP and JavaScript can be used in fixing the bugs in the back end.

### Re-testing until all the requirements are met.

**Requirements:**

XAMPP server is used to deploy the final application.

# Member contribution table

|  |  |  |  |
| --- | --- | --- | --- |
| **Member Name** | **Contribution Description** | **Overall Contribution (%)** | **Note (If Applicable)** |
| **Neha Goud Baddam** | The overall structure of the system, including diagram. Also defined User Interfaces. | 14 |  |
| **Vamsi Sai Konidena** | Project implementation Plan Phase-1 | 12 |  |
| **Reshmi Chowdary Divi** | Project implementation Plan Phase-2 | 12 |  |
| **Prem Kumar Maddula** | Project implementation Plan Phase-3 | 12 |  |
| **Rahul Mandaloju** | Deliverable-2 document and member contribution table. | 12 |  |
| **Harika Uppalapati** | Interfaces requirement (software, and communication) | 12 |  |
| **Purandhara Maharshi Chidurala** | Functional requirement specification | 13 |  |
| **Shiva Surya Vardhan Reddy** | Non-functional requirement specification | 13 |  |