UNIVERSITY OF LEICESTER

School of Computing and Mathematical Sciences

CO7201 Individual Project

Final Report

Software Bug Tracking and Reporting Tool

Harishmitha Raja

hr200@student.le.ac.uk

239053031

Project Supervisor: Dr. James Hoey
Principal Marker: Dr. Newman Lau

Word Count: 13267 (Excluding Abstract, TOC, References and Appendices)

16/05/2025

DECLARATION

All sentences or passages quoted in this report, or computer code of any form whatsoever used and/or submitted at any stages, which are taken from other people's work have been specifically acknowledged by clear citation of the source, specifying author, work, date and page(s). Any part of my own written work, or software coding, which is substantially based upon other people's work, is duly accompanied by clear citation of the source, specifying author, work, date and page(s). I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this module and the degree examination as a whole.

Name: Harishmitha Raja
Date: 16/05/2025

ABSTRACT

The BugTrackR - Software Bug Tracking and Reporting Tool is a role-based web application developed to streamline the process of reporting, assigning and resolving software bugs. Existing bug tracking platforms are often complex (even simple bug reporting requires administrators to pre-configure workflows, issue types, field screens and permissions. This setup overhead slows down adopting the tool, especially for teams needing a fast, structured and role-specific bug tracking system) or too generalised (lacking built-in role-specific workflows for developers, testers and non-technical users). This system addresses those issues by creating a more structured and flexible solution designed specifically for managing bugs across multiple applications with dedicated roles and workflows for general users, developers, testers, team leads, and admins.

Key objectives include secure login with OTP verification for reset password, role-based dashboards tailored to each user type. Detailed bug reporting tailored to technical and non-technical users and role-based access ensures each user can only take actions appropriate to their role. Bugs can be manually assigned to developers and testers or automatically assigned, depending on workload, seniority and team to make sure it's fair. Bug reports can be tracked using a stepper interface which supports better visibility of bug status. Priority levels can be set manually or automatically classified. Manual and NLP-based contextual duplicate detection features are included to reduce repeated entries and improve productivity of developers and testers.

The system enables tech team collaboration via structured comments, with mention tagging and email alerts. Separate views for reported, assigned, mentioned bugs etc., are supported for better traceability. Developers and testers can request bug reallocation and bug reopen, and team leads or admins can approve or reject requests. Ability for critical bugs to be closed by admin or team lead is also implemented to prevent early closure of incorrectly fixed bug.

Advanced filtering and contextual search allow teams to efficiently locate bugs based on status, severity, or assigned developer/tester. Real-time scheduled alerts are triggered when bugs remain inactive or unassigned for too long. Email notifications are sent for key events like assignments, comments, and critical bug approval. Role-based analytics and export options allow team leads and admins to monitor team performance.

Optional features such as chat-based communication between reporters and tech users, and bug favouriting enhance collaboration and usability.

The project was developed using the Waterfall model, with some iterative improvements during implementation. The system ensures clarity in bug ownership, reduces redundancy, and improves communication between teams, resulting in a more organised and responsive bug tracking workflow.

Table of Contents

1.Introduction

1.1 Motivation

Efficient bug tracking is essential for maintaining the quality, reliability, and maintainability of software applications. However, in many real-world projects, the bug tracking process suffers from several inefficiencies that negatively affect developer productivity and user satisfaction. Studies show that **vague** or **incomplete** bug reports, **delayed assignments**, **redundant** submissions, and **lack of structured communication** are common issues that lead to wasted effort and slower resolution times [1][3][4].

Manual bug assignment, though often more accurate, is **time-consuming** and becomes difficult to manage when the number of bug reports increases. On the other hand, fully automated triaging (assignment) systems may **misallocate** bugs because they often **lack the contextual** understanding of developer workload, skill set, and team structure [2]. Duplicate bug reports are another major concern, they clutter the system, **increase backlog** count, and **consume unnecessary** developer time and reduce productivity [3]. As reported by Bettenburg et al., duplicates often go unnoticed due to poor detection mechanisms or limited support for contextual similarity checks.

In addition to these technical challenges, **communication gaps** within bug tracking systems remain a persistent issue. When collaboration is unstructured or happens **outside the system** (e.g., via email, third-party tools like MS Teams or Cisco Webex, or informal discussions), critical context and technical details are often lost. Research highlights that unclear, unfocused, or inaccessible comments significantly hinder the debugging process and contribute to delays in issue resolution [4].

Furthermore, as bug volumes grow in large or active projects, scalability and usability become critical. Many systems become **slower**, more **difficult to navigate**, or fail to maintain efficiency when subjected to high data loads, ultimately impacting both developer experience and decision-making quality [5].

These real-world problems motivated the design and development of BugTrackR (Bug Tracking + Reporting), a role-based bug tracking and reporting tool that aims to address the shortcomings observed in existing systems. By offering structured workflows, advanced duplicate detection techniques, intelligent assignment algorithm, and in-bug communication, BugTrackR seeks to reduce redundancy, improve collaboration, and provide a scalable, user-centered solution that supports both technical and non-technical users.

1.2 Aims and Objectives

The main aim of this project is to design and develop a role-based software bug tracking and reporting tool, BugTrackR, that **improves** how software defects are **reported**, **tracked**, and **resolved**. The system focuses on making bug reporting more **user-friendly** by **tailoring the** interface and form design to different user roles and by reducing the effort required to submit complete, high-quality reports. It also aims to minimize redundancy through **smarter duplicate detection**, improve collaboration through **integrated communication** features, and support better decision-making through analytics and structured workflows.

To achieve this aim, the following objectives were set:

- Support **structured**, **role-based workflows** by providing distinct dashboards and permissions for general users, developers, testers, team leads, and admins.
- Improve bug reporting quality through **guided input forms** tailored to technical and non-technical users, and automatic environment detection (e.g., browser, OS).
- Enable efficient bug allocation using an assignment algorithm that considers **developer/tester workload, seniority, team,** and bug **priority**.
- Minimize redundancy through both manual and automated duplicate detection, using **semantic similarity** techniques powered by NLP (Natural Language Processing).
- Enhance communication within the system by offering both an **internal comment** system for technical users and a **chat interface** for interactions between reporters and technical roles.
- Provide real-time **scheduled** updates and alerts through features like **email notifications** and **visual indicators** for **inactivity** or **pending updates**.
- Offer **insights** through analytics, such as **resolution trends**, **workload** summaries, and **performance** dashboards for developers, testers, team leads and admin.
- Ensure **data security** and **access control** by implementing secure **authentication**, **encrypted communication**, and **role-based access** enforcement.

These objectives guided the system throughout its development and informed many of the design decisions (see Section 6), ensuring that BugTrackR remains practical, scalable, and relevant to real-world software teams.

1.3 Challenges

- **New to NLP Techniques for Duplicate Detection**
  Implementing duplicate detection using NLP was one of the most challenging parts of the project, especially since I had no prior experience with Python-based NLP tools. I initially explored methods like WordNet, the Lesk algorithm, and Cosine similarity, but they didn’t produce accurate results for bug-related content. Learning and applying a more advanced solution required additional time and effort. See Section 9.4 for more information.
- **Limited API & Database Experience**
  Working with backend development using REST APIs and MongoDB was new to me, and I had to learn how to organize server code and manage user-role-based data properly. I used Express.js for route handling and Mongoose for defining schemas, validations, and structure. Separating route logic by feature and role helped me manage the backend more cleanly and reliably.
- **Ensuring Security & Access Control**
  Implementing secure role-based access required careful planning. I set up JWT authentication, password reset with OTP verification, and HTTPS communication. Middleware was added to restrict access to sensitive routes like assigning bugs and setting priority.
- **Balancing Manual & Automated Bug Assignment**
  Designing the bug assignment algorithm was technically demanding. It had to consider workload, priority, team, and seniority while supporting manual overrides. I implemented fallback logic and user-triggered reallocation workflows to handle edge cases. See Section 9.4 for more information.

- **Real-Time Collaboration**
  I had no prior experience with WebSockets or real-time communication. Implementing chat using Socket.io required learning how to manage connections, events and state. This was successfully used to support chat between reporters and tech users. See Section 6.9 and 7.6 for more information.

1.4 Risks

While planning and developing BugTrackR, several risks were identified that could potentially impact the quality or timely delivery of the project. These were considered early and monitored throughout the development process to minimize their effects.

- **Learning Curve for New Technologies:**
  One of the primary risks was the need to work with technologies and tools that I had limited experience with, such as Python-based NLP libraries, FastAPI, and FAISS for duplicate detection. This required additional learning time, and any unexpected issues could have delayed the implementation of key features.

- **Ambitious Scope:**
  The project included a wide range of essential and advanced features, which made it challenging to manage within the limited timeline. New features such as reopen requests, mentioned bugs view and a reporter-tech chat were added during development to cover practical gaps. To keep the project on track, a few planned features were later dropped/modified after reassessing their value and feasibility.

- **Integration Complexity:**
  BugTrackR includes multiple modules such as authentication, role-based dashboards, internal comments, real-time chat, duplicate detection using a separate Python microservice, email notification and more. Integrating these components smoothly was technically challenging, and there was a risk of runtime errors or incompatibility between services.

- **Testing Limitations:**
  Due to time constraints and the size of the project, thorough automated testing could not be fully implemented across all modules. To mitigate this, manual checks were done. End-to-end workflow run was also performed to ensure that interactions between components behaved as expected. During the testing phase, full manual testing and automation for challenging features were performed. Several issues were identified during this process and were resolved promptly.

- **Similarity Matching Performance**
  Semantic duplicate detection could become slow as data grew. To address this, FAISS was used for fast similarity search (see Section 6.4), and MongoDB indexing was applied to reduce delays in retrieving bug details (see Section 7.9).

1.5 System Overview

1.5.1 Workflow Summary

The workflow begins with bug reporting and continues through assignment, fixing, testing, and closure. Duplicate detection is run during and after submission. Users collaborate via comments and chat, and reallocation or reopen workflows support flexibility. Bug progress is tracked through clearly defined statuses, with alerts for critical actions or inactivity.

The below workflow diagram Figure 1, was created using Excalidraw [6] to give a general overview of the application flow.

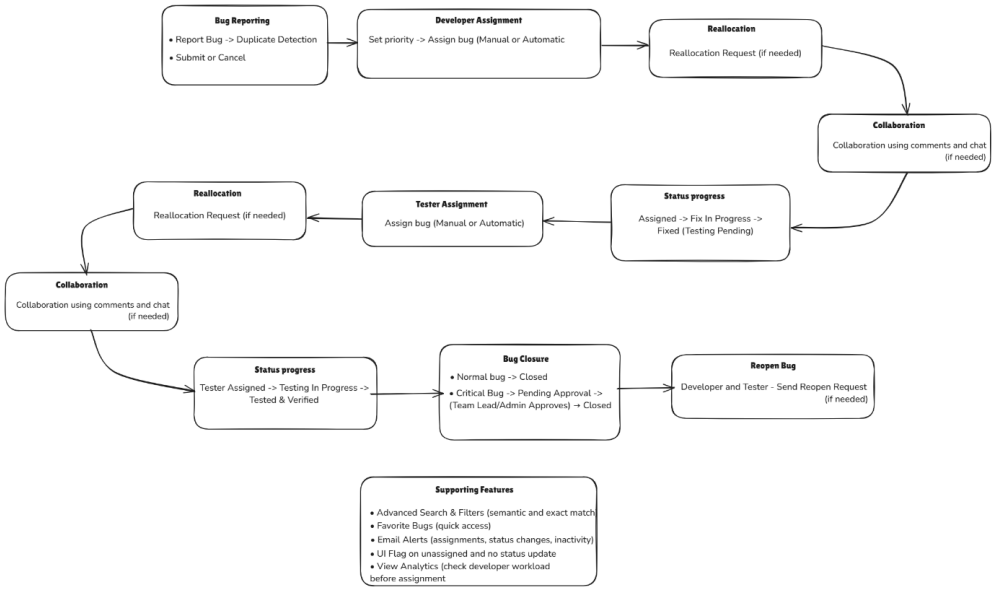


Figure 1:Workflow Diagram

2. Background Research

2.1. Technical Research and Technology Justification

The technologies used in BugTrackR were carefully chosen after comparing several popular options in terms of development time, performance, learning curve, flexibility, and how well they fit my project goals. Since this is a role-based system involving bug reporting, assignment, collaboration, and automation features like duplicate detection and priority classification, the chosen tech stack had to support flexibility, structured data handling, and fast development, while keeping things simple and maintainable.

Frontend: React, Material UI, and Tailwind CSS

For the frontend of BugTrackR, React was chosen after comparing it with other widely used frameworks such as Angular [7]. Based on some prior experience with Angular, React's simpler structure and smaller learning curve made it more manageable in an individual development context. Unlike Angular, which is a full-featured framework often used in enterprise environments, React offers greater flexibility in structuring the application without being constrained by boilerplate-heavy patterns, have to write and maintain a lot of extra setup

code (modules, decorators, service injections) even before start writing actual feature, whereas React lets us directly start building features with minimal structure requirements. [8]

Since the project involved building five different dashboards for distinct user roles the chosen frontend technology needed to be flexible, maintainable, and allow for rapid development of interactive features. React made it easy to create reusable components for features such as displaying header, sidebar, bug lists and details and viewing mentioned bugs. These components were reused across multiple dashboards with slight variations, which improved development speed and maintained consistency across the interface.

Another reason React was preferred was its virtual DOM, it enabled fast, smooth UI updates especially useful for dynamic tasks like toggling filters or updating bug statuses without full-page reloads. According to Stoyan Stefanov in React Up and Running [9], this approach leads to faster render times and better user experience by not reloading the entire page, especially in apps that have dynamic and state-heavy interfaces. Similarly, Freeman in Pro React 16 [10] notes that React's component-based structure and virtual DOM make it ideal for interactive platforms like issue tracking systems.

In terms of styling, Material UI (MUI) [11] was used to implement common UI elements like steppers (status tracking), tables (bugs list), modal (bug details), and buttons. It helped speed up the development of consistent and accessible interfaces. However, for more custom layout control and spacing tweaks, Tailwind CSS was used alongside MUI. This combination gave both the structure of a design system and the flexibility of utility-first (predefined) CSS.

Backend: Node.js, Express.js, and MongoDB with Mongoose

For the backend of BugTrackR, Node.js was chosen as the main runtime environment along with Express.js to create the APIs. One of the biggest reasons for choosing Node.js was that it allowed the entire application to be built using JavaScript, both on the frontend and backend. Since React with JavaScript was already used for the frontend, sticking with JavaScript across the whole project helped keep things consistent, faster during development and easier to manage.

Express.js is a lightweight framework that works with Node.js and makes it simple to create routes for all the features in the system. This includes submitting and managing bug reports, assignments, comments, and request workflows. The flexibility of Express.js also made it easier to organise logic cleanly, for example, by separating routes based on functionality or role and adding middleware functions that check permissions before certain actions are allowed.

While other options like Java with Spring Boot were considered but Node.js was more suitable for this project. It required less setup, supported non-blocking operations and was easier to work with as a solo developer. Since BugTrackR supports multiple users interacting with the system at once (e.g., reporting or updating bugs), Node.js helped keep everything responsive.

According to [12], Node.js is better suited for small-to-medium web apps where quick setup and flexibility are more important than enterprise-level configuration which is needed for Spring Boot.

For the database, MongoDB was selected due to its flexibility. Bug reports vary based on who submits them general users give basic details, while developers and testers may include logs or stack traces. In a relational database, this would need multiple linked tables, but MongoDB's document-based structure allowed all variations to be stored in a single collection with less complexity.

To add structure and validation, Mongoose was used, which let me define schemas for each collection and enforce field types (e.g., numbers for hours, strings for titles). Mongoose's middleware also helped with automating tasks, like setting bug IDs during submission without duplicating logic (for creating bug id) in the route handlers.

Overall, using Node.js, Express.js, MongoDB, and Mongoose made it possible to build a backend that is fast, easy to manage, and flexible enough to handle different types of data and user actions. It worked well for the needs of BugTrackR, which required handling a variety of user roles and types of bug reports while still being easy to maintain and scale.

NLP & Duplicate Detection: Python, FastAPI, Sentence Transformers, and FAISS

To enable duplicate detection in BugTrackR, a semantic similarity approach was used instead of traditional keyword matching. This allows the system to detect related bug reports even if they are phrased differently.

Sentence Transformers [14] were chosen to embed bug titles and descriptions into dense vectors that capture contextual meaning. Earlier techniques like TF-IDF or WordNet-based matching were explored but proved too shallow or inconsistent for this use case.

To compare embeddings efficiently, FAISS (Facebook AI Similarity Search)[15] was used. It supports fast, scalable similarity search and was more suitable than manually computing cosine similarity, especially as the data grows.

To host the detection service, FastAPI was selected over Flask due to its built-in async handling and faster performance under concurrent requests. Its integrated Swagger UI also made integration and checking easier during development.

A full explanation of the service flow, semantic similarity implementation and backend integration are included in Section 6.4 and 6.5

2.2 Review of Related Research

Research into key areas such as duplicate bug detection, automated triage/assignment, and user-centered design (UCD) has significantly influenced the planning and feature design of BugTrackR. These concepts directly address common challenges in bugs management, including issue duplication, triaging overhead, and usability concerns for technical users. These areas helped identify important challenges in bug tracking systems and informed the selection of appropriate technologies.

The way bug reports are written plays a major role in how quickly issues are resolved. A widely cited study [1] involving feedback from developers and bug reporters found that reproduction steps, stack traces, and system information (e.g., browser or OS) were especially helpful. However, for example fewer than 10% of bug reports included the OS details. This gap between

what developers need and what users typically provide highlighted the value of structured, guided forms. Based on this, BugTrackR's forms adapt by role, guiding general users with prompts and providing technical users with direct fields for stack traces, logs, and self-assignment of bug.

Duplicate detection is another critical area. According to [16], combining textual descriptions with execution logs can improve duplicate identification. Although BugTrackR does not use logs, the study reinforced the importance of semantic approaches over shallow keyword matching. This supported the decision to use Sentence Transformers for detecting contextually similar bug reports.

Another well-researched challenge is bug triage/assignment, where incoming issues must be routed to the appropriate developers or teams. Manual triage can be time-consuming, inconsistent, and prone to delays, especially in larger or more active systems. Another well-researched challenge is bug triage/assignment, where incoming issues must be routed to the appropriate developers or teams. Studies like [17] highlight how analyzing developer expertise and historical resolution patterns can support more efficient and consistent assignment. These ideas influenced BugTrackR's assignment logic, which focuses on factors such as workload, team specialization, priority, and seniority.

Research highlights the importance of user-centered design (UCD) in development tools, showing that structured, role-specific interfaces improve navigation, reduce cognitive load, and enhance productivity [18]. These insights directly influenced BugTrackR's interface, which includes role-based dashboards, advanced search, and streamlined workflows tailored to developers, testers, team leads, and admins.

Communication plays a key role in how quickly and effectively bugs get resolved. A study by Ramírez-Mora et al. [19] looked at over 500,000 comments from open-source issue trackers and found that referential comments ones that share clear, factual information were linked to faster and more successful bug fixes. It wasn't just about having comments, but the structure and clarity of those comments that made a difference. This influenced BugTrackR's communication features, which include a private comment section for technical users to collaborate and a chat feature to help reporters communicate directly with the tech team.

2.3 Evaluation of Existing Tools

To gain a clearer understanding of the current landscape of bug tracking systems, two well-established platforms, JIRA [20] and Bugzilla [21] were reviewed. These tools are widely used in both enterprise and open-source projects and have played a significant role in shaping many of the foundational design choices in BugTrackR.

Bugzilla, developed by Mozilla, is known for its structured workflows and lightweight setup. However, its user interface is outdated and heavily form-based, resembling a traditional database rather than a modern web application. As shown in Figure 2, the advanced search interface presents multiple filters, such as severity, component, milestone and change history, without visual grouping or guidance. This makes it difficult for non-technical users to report or find bugs. BugTrackR addresses this by adapting the UI based on user roles and hiding unnecessary options.

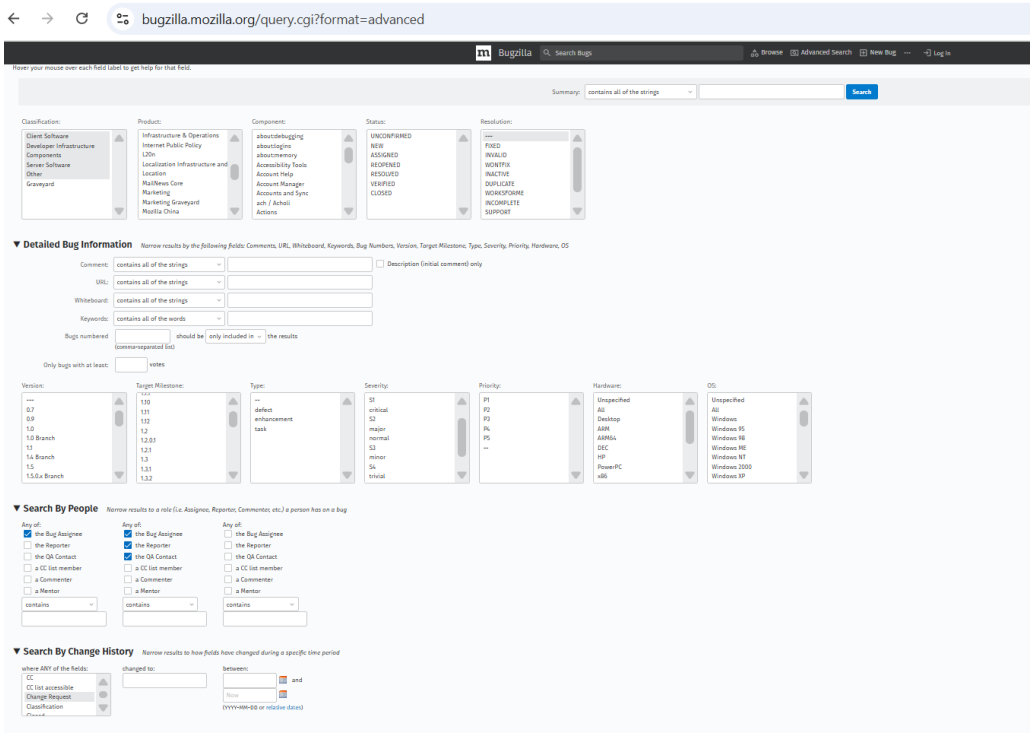


Figure 2:Bugzilla's database like UI.

JIRA, developed by Atlassian, offers a more modern, customizable interface with features like dashboards and integration with external tools. However, it requires significant setup effort, and many advanced features, such as automation and workflow configuration, depend on plugins or paid extensions, which adds to the system's complexity. To address this, BugTrackR introduces intelligent automation to reduce manual effort. For example, assigning bugs to developers or testers in JIRA often requires custom rule setup or paid plugins like JIRA Automation [22], and even then, assignments are usually based on fixed rules rather than dynamic factors. In contrast, BugTrackR includes a built-in auto-assignment system that considers real-time factors like developer workload, team specialization, priority, and seniority ensuring more balanced and efficient bug distribution without the need for external tools. See section 6.6 and 7.4 for more details.

Duplicate detection is another area where BugTrackR offers a significant improvement. JIRA's detection capabilities primarily rely on keyword-based matching in the summary field, often through plugins like "Find Duplicates" [23]. This approach may miss duplicates when bug descriptions use different wording. BugTrackR instead uses semantic similarity powered by Sentence Transformers and FAISS to detect duplicates based on meaning, making it more effective at reducing redundancy in large systems. See section 6.5 for more details.

Furthermore, as shown in Figure 3, JIRA's default bug creation form allows only a 255-character summary field, which often leads to vague reports and incomplete context. BugTrackR improves on this by requiring structured input tailored to each user role, ensuring better quality bug reports. Figure 4, highlights how the bug report details are cluttered in Jira.

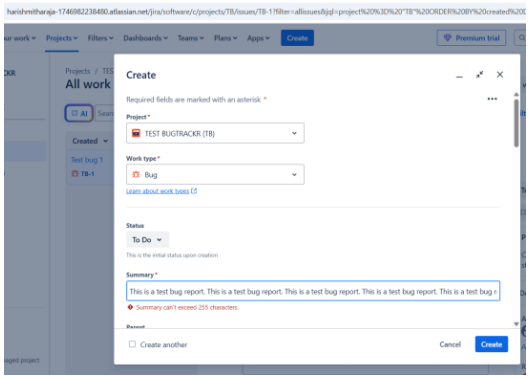


Figure 3: JIRA's summary-only bug form with char limit

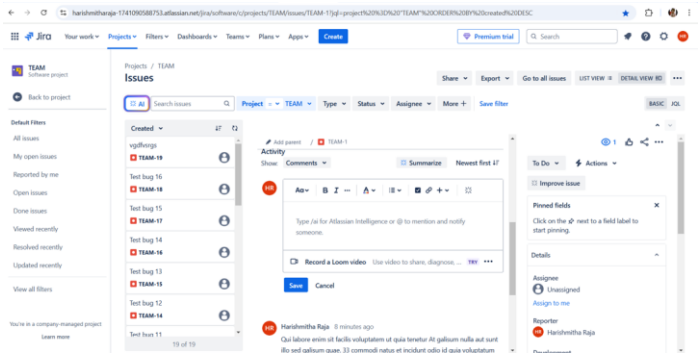


Figure 4: Cluttered JIRA issue view with multiple panels

Finally, BugTrackR also improves communication through a private, role-restricted comment section designed for technical users. While both JIRA and Bugzilla support general commenting, they do not provide structured role-based separation by default. In JIRA, for example, comment visibility can be restricted to certain roles or groups, but this must be manually set for each comment and requires additional configuration [24]. BugTrackR simplifies this by allowing developers, testers, team leads admin to collaborate privately within the system, improving focus in user-facing chats.

3.Requirements

3.1 Project Requirements

Essential Requirements

- User Authentication - The system can allow users to securely log in using their credentials.
- Password Reset - The system can allow users to request a password reset using email-based OTP verification.
- Role-Based Dashboards - The system can display a role-specific dashboard with relevant actions and data for each user type.
- Bug Reporting (General Users) - A general user can submit a bug report by providing basic information such as description, browser, OS, and issue type.
- Bug Reporting (Technical Users) - A developer, tester, team lead, or admin can report a bug with detailed technical information including logs, stack traces, replication steps and assignment.
- Bug Status Tracking - A user can view and track the progress and current status of bugs they reported.
- Bug Status Updates – Tech users can update the status of assigned bugs to reflect their progress.

- Bug Discussions - Tech users can comment on bug reports to facilitate internal collaboration.

- Additional Reporter Input - A user can provide extra details on a bug after submission if more information becomes available.

- Bug Assignment - A team lead or admin can manually assign or reassign a bug to a developer or tester.

- Priority Management - A team lead or admin can assign and update the priority level of a bug (Low, Medium, High, Critical).

- Duplicate Detection - The system can allow technical users to manually mark and undo duplicate and also show potential duplicates automatically during submission and on trigger.

- Email Notifications - The system can notify users via email when bugs are reported, assigned, commented on, marked as duplicate, or inactive for a long time.

- Advanced Search and Filtering - The system can support combination filters, sorting, and semantic search to help users retrieve relevant bug reports.

- Bug List Views - A user can view a list of bugs they reported, are assigned, and where mentioned based on their role.

Recommended

- Automated Bug Assignment - The system can automatically assign bugs to suitable developers/testers based on workload, expertise, seniority, and availability.

- Automated Priority Classification - The system can automatically assign a priority level to a new bug based on similarity with previous reports.

- Bug Reallocation Requests - A developer or tester can request the reassignment of a bug they cannot resolve, with justification.

- Bug Reopen Requests - A developer or tester can request the reopen of a bug they think the fix is not correct, with justification.

- Bug Resolution Analytics - The system can display tables and charts to provide analytics based on role (e.g., bug fix rates, workload).

- Custom Report Export - A user can export filtered bug reports in PDF or CSV format.

- Time-Based Alerts - The system can trigger alerts or visual warnings for bugs that remain inactive, unassigned, or unresolved for extended periods.

Optional

- Favourite Bugs - A user can mark bug reports as favourites for quick access.
- Unassigned Bug Queue - The system can allow admins to assign teams and optionally allocate developers or testers, while team leads can assign developers or testers within their team to unassigned bugs.
- Chat between Reporters and Tech Users - The system can support structured chat-like communication between reporters and technical users.

3.2 System-Level Requirements

The user-facing functionalities (functional requirements) of BugTrackR have been detailed in Section 3.1. This section focuses on the key system-level qualities that ensure the platform remains secure, scalable, and user-friendly across all workflows. While functional capabilities define what the system does, these requirements highlight how the system operates behind the scenes to deliver a smooth, efficient user experience.

3.2.1 Non-Functional Requirements

The non-functional requirements of BugTrackR focus on delivering a system that is secure, scalable, reliable, and user-friendly. **Security** is maintained through JWT-based authentication for session handling, role-based access control enforced via backend middleware, HTTPS-encrypted communication, and password hashing using bcrypt. **Performance** and responsiveness are supported through efficient API design with Express.js, modular backend architecture and real-time capabilities using WebSockets (Socket.io) and Node Cron for scheduled notifications. **Scalability** is achieved by using MongoDB, which supports flexible data structures and indexing for faster queries, along with FAISS for efficient semantic similarity searches during duplicate detection. **Usability** is addressed through a role-specific React frontend built with Material UI. Dashboards are tailored to each user role, and guided bug reporting forms display different fields based on whether the user is a general reporter or a technical user such as a developer or tester. This ensures that users of all technical backgrounds can interact with the system effectively and with noconfusion.

4. Technical Specification

Category		Technology & Version	Purpose
Frontend	Technology	React 18.2	JavaScript based library for frontend development
		JavaScript ES6	Modern JavaScript standard used for building dynamic and interactive web applications.
		HTML5	For structure and content of web applications
		CSS3	For styling web pages
	Syntax	JSX (JavaScript XML)	Syntax extension for combining HTML and JavaScript in React components
	Library/Package	Tailwind CSS 3.4.17	CSS framework for quick and efficient UI development
		React Router DOM	Handles frontend routing and navigation between views
		MUI (Material UI) 6.4.4	Prebuilt React UI components with accessibility and styling support
		FontAwesome	Library for adding scalable vector icons in the UI
		Recharts	Library for building charts and graphs for analytics (e.g., bug fix efficiency trends)
		React Scripts	Provides scripts for running, building, and testing React apps (via Create React App)
		jspdf	Creates PDF documents in frontend
		jspdf-autotable	Generates formatted tables inside PDFs
		file-saver	Allows saving files from browser (e.g., exported bug reports)
		react-chat-ui-kit	Provides prebuilt chat UI components for real-time communication
		react-dnd	Enables drag-and-drop functionality (e.g., bug status management)
		react-dropzone	Allows drag-and-drop file uploads (e.g., bug screenshots, logs)
		react-toastify	Displays toast notifications for user actions (e.g., success, error, info alerts).
Backend	Runtime Environment	Node.js 22.11.0	Runtime environment to run JavaScript in server
	REST API Framework	Express.js 4.21.2	Framework for creating and managing RESTful APIs
	Library/Package	JavaScript ES6	Modern JavaScript standard used for building dynamic and interactive web applications.

		Multer	Middleware for handling file uploads (e.g., screenshots, logs)
	Library/Package	JWT (JSON Web Token)	Authentication and authorization using tokens
		Bcrypt	Hashing and verifying user passwords securely
		Cors	Middleware for handling Cross-Origin Resource Sharing requests
		Nodemailer	Sends email notifications (e.g., bug updates, password reset)
Database & ODM	Database	MongoDB 8.0.4	NoSQL database for handling flexible, document-based data storage
	ODM	Mongoose	Object Data Modeling library for MongoDB to manage data efficiently
	GUI Tool	MongoDB Compass 1.46.1	GUI for viewing, querying, and debugging MongoDB documents and collections
Authentication & Security		JWT (JSON Web Token)	For secure authentication and user authorization
		SSL	Used for secure data communication between client and server
Microservice (NLP)		Python [36]	Programming language used for the NLP-based duplicate detection microservice
		FastAPI	Python framework used to quickly build and serve APIs
Version Control		GitLab	Platform for version control
Real-time Feature		Socket.io	Enables real-time features (e.g., chat system)
IDE		Visual Studio Code (VSCode) 1.99.3	Code editor
		Sourcetree	GUI tool for simplifying Git operations
API Validation		Postman	API testing tool
Testing Frameworks		Jest	Framework that is used for unit and integration testing
		Pytest	Framework that is used for unit and integration testing in Python.
NLP Libraries		Sentence Transformers	Converts text into vector embeddings for semantic similarity
		FAISS	Library for fast similarity search used for efficient duplicate bug detection.
Task Scheduling		Node Cron	Schedules and automates background tasks at regular intervals.
Operating System		Windows 11	Operating System used during development
Planning & Documentation		OneNote	Used to organise notes taken during meetings, track issues, and document feature ideas

	Microsoft Word	Used for drafting and formatting project reports
Other Utilities	Day.js	Lightweight JavaScript library for date and time formatting
	Instagantt [28]	Used to draw timeplan
	dotenv	Loads environment variables from .env files
	draw.io [30]	To draw design diagrams in Design Section
	excalidraw	To draw workflow diagram
	Flaticon [31]	Icons used in diagrams in Design Section
	TechIcons [32]	Icons used in diagrams in Design Section

5. Development Methodology

5.1 Overview of Development Methodology

The development of BugTrackR followed a Waterfall-based approach with some iterative refinements. The key stages included background research, requirements analysis, design, development, testing and report writing. While the process was largely sequential, minor adjustments were made during implementation, such as adding features like chat and reopen workflows, based on feedback and needs. These changes did not alter the overall structure but helped in building a well round application. A workflow check was also done during principal marker interview time with features done till then, and it highlighted a few gaps in user experience. Based on this, features like the “mentioned bugs” view were added.

5.2 Why Waterfall Was Chosen

Waterfall was selected for this project because it matched the nature of individual development and aligned well with the university’s assessment schedule. Each phase planning, design, development, testing could be completed in sequence, which supported focus and progress. Since the project had no external client or team, there was no need for Agile-style coordination. Although some minor adjustments were made based on supervisor/principal marker feedback, they did not require a full shift in methodology, making waterfall with iterative refinements a manageable and appropriate choice. See section 9 for more details.

6. Design

6.1 Architecture Design

6.1.1 Client-Server Architecture

The BugTrackR system follows a client-server architecture, Figure 5, where the frontend (client) communicates with the backend over the internet using REST APIs. The server processes the requests, it interacts/talks with the database and returns responses to the client.

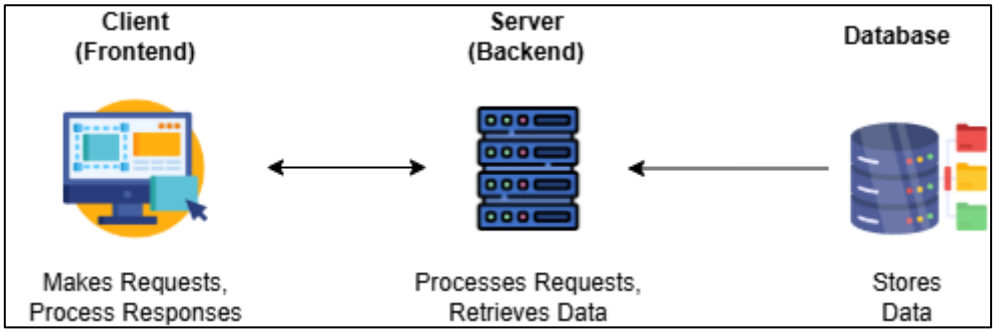


Figure 5: Client-Server Architecture Diagram

The diagram below, Figure 6, illustrates the overall architecture of the BugTrackR system, highlighting the core components and how they interact:

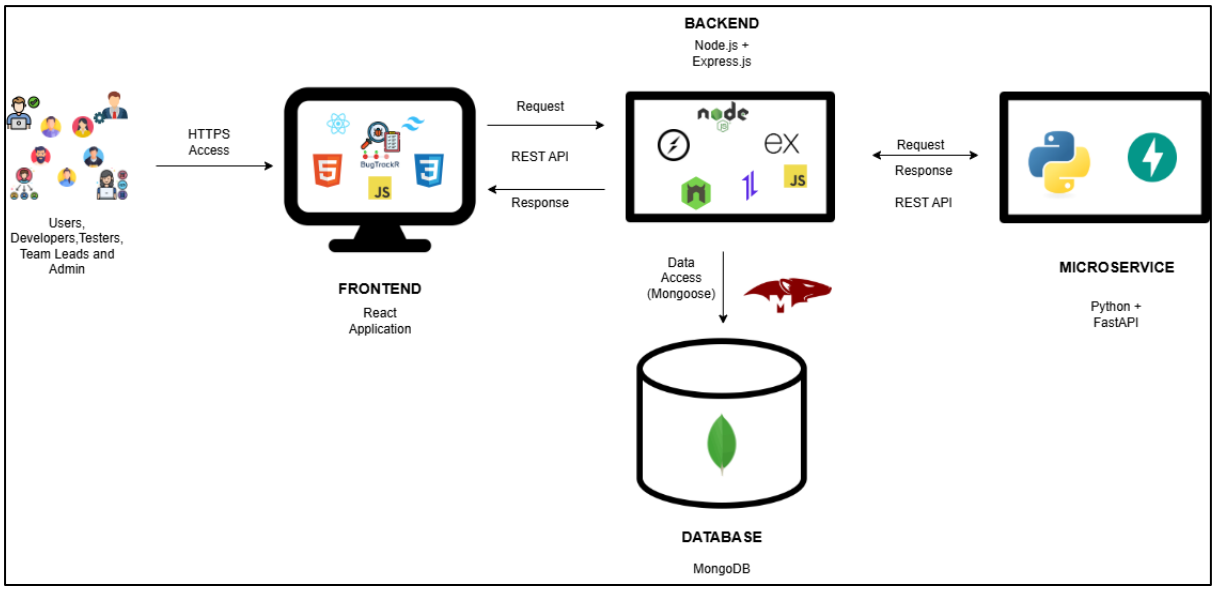


Figure 6: Architecture Diagram

Frontend (Client):
The client is a React-based single-page application accessed via HTTPS by different user roles such as general users, developers, testers, team leads, and admin. It includes user-facing features like bug submission, views and status tracking and communicates with the backend using REST API requests via Axios.

Backend (Server):
The backend is built using Node.js and Express.js. It acts as the central hub for processing client requests, implementing business logic, managing authentication and role-based access,

and coordinating all database and service interactions. It exposes RESTful API endpoints to the frontend.

Database:
The system uses MongoDB as the primary database, accessed using the Mongoose ODM (Object Data Modeling) library. All core collections (users, bug reports, comments, etc.,) are stored here. Data access happens through well-defined schemas and model methods within the backend.

Microservice:
A separate Python-based microservice built with FastAPI is responsible for semantic duplicate detection, semantic search and priority classification of bug reports. The Node.js backend sends bug titles and descriptions to this service and receives a ranked list of similar bugs via a REST API. This microservice is fully decoupled from the main system, making it reusable.

6.2 Database Design

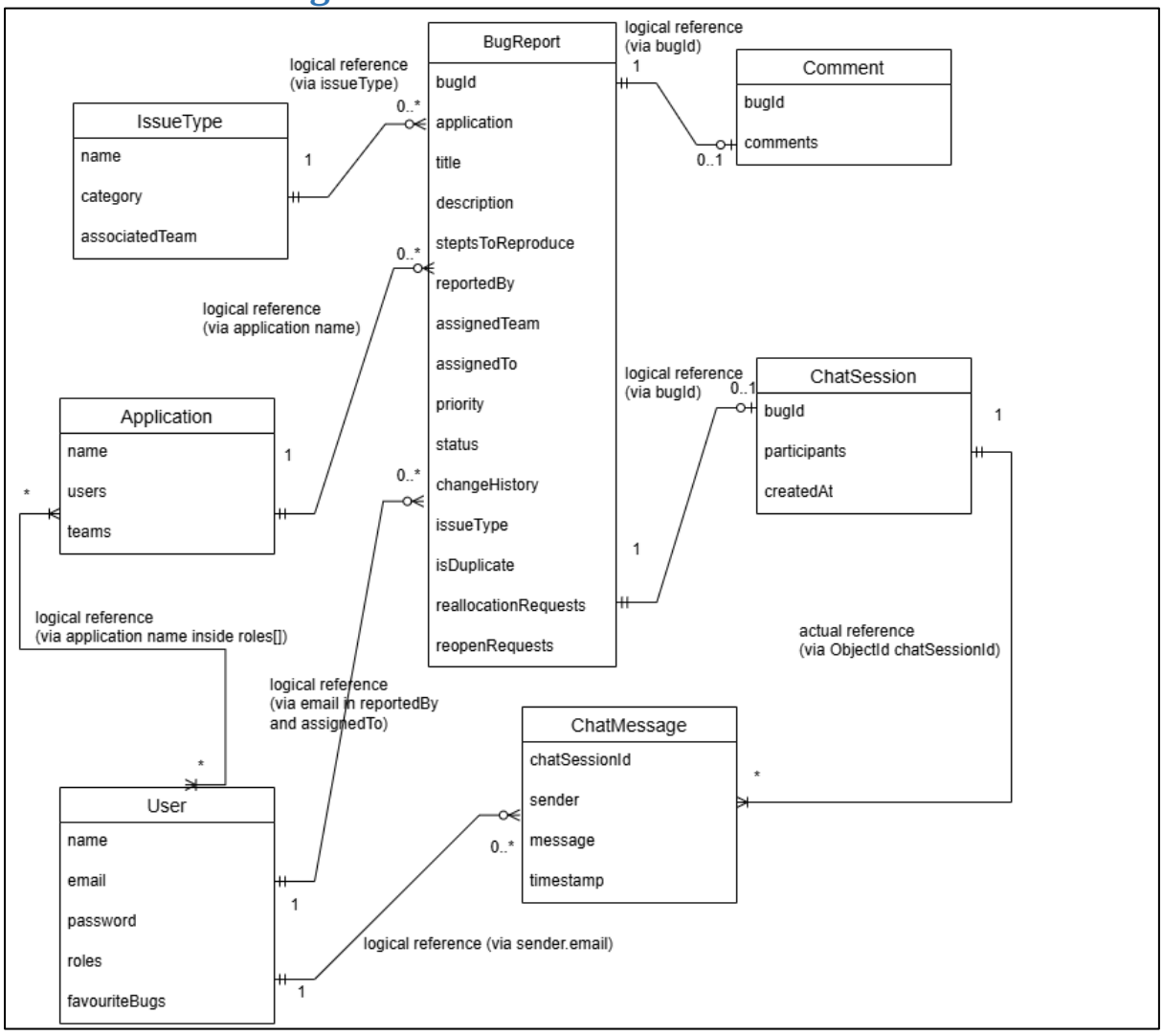


Figure 7: Entity-Relationship Diagram

The diagram Figure 7, presents the Entity-Relationship (ER) model used to represent the logical structure of the BugTrackR system's database, implemented in MongoDB using Mongoose schemas. While MongoDB is a NoSQL document-oriented database and does not enforce relational constraints (like foreign key concept, relational links etc., For example, when a user associated with a bug report is deleted MongoDB won't throw error like in relational databases), the ER diagram is still useful for visualizing the intended structure and data relationships within the system.

All relationships shown in the diagram are based on either:

- Actual references (e.g., ObjectId used with Mongoose ref)
- Logical references, where linking is done via string fields such as email, application name or bugId.

For instance, the BugReport collection references users (reporters, developers, testers) via email addresses, and links to its corresponding Application using the application name string. These are marked as logical references in the diagram.

The multiplicities are based on the real-world logic and data flow in the system. For example:

- A single application can have one or more users and a user can be associated with one or more than one application (in the case of general user)
- An application can have 0 or more bug reports and a bug report must be associated with exactly 1 application.
- A user may or may not report a bug or be assigned any bugs 0..*.
- Each bug can have 0 or 1 chat session but each chat session should be associated with exactly one bug report.
- A chat session can have many chat messages, each of which is linked to a sender via their email.

6.3 Use Case Diagram

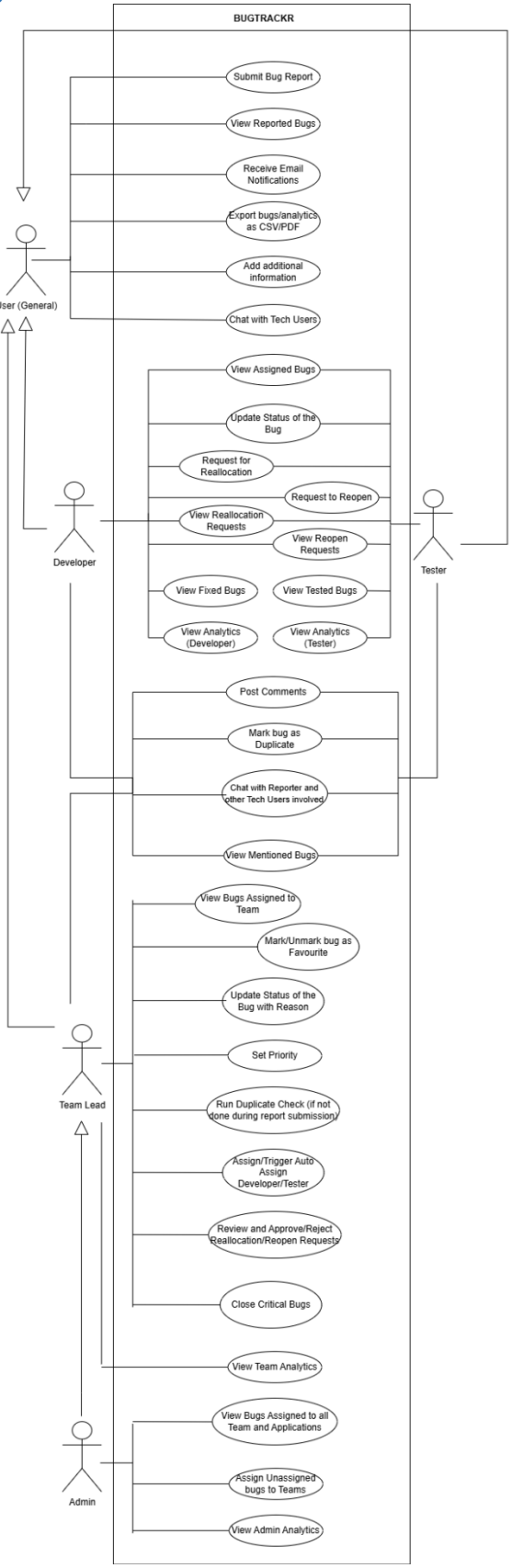


Figure 8: Use Case Diagram

The above diagram, Figure 8, illustrates the main use cases within the BugTrackR system, mapped against different types of users including General Users, Developers, Testers, Team Leads, and Admin. It provides a high-level overview of the interactions each actor has with the system.

BugTrackR supports five distinct roles and each role is granted access only to the functionalities relevant to their responsibilities. These use cases represent the system from the user's point of view and help define the scope of the interactions between actors and the application.

Developers, Testers, Team Leads, and Admins inherit all the capabilities of a General User. All the individual capabilities are as follows,

- General Users can report bugs, track their submissions, export bug data, receive notifications and chat with tech users.

- Developers can additionally update bug statuses, view bugs assigned to them, request reallocations or reopen bugs and view developer-specific analytics.

- Testers can validate bugs, view their tested bugs, request reallocations or reopen bugs, and view tester-specific analytics.

- Team Leads have a broader set of responsibilities. They can assign bugs to their team, set priorities, run duplicate checks (if not already done), and review and respond to reallocation or reopen requests and view team-specific analytics.

- Admins have the highest level of access and can perform all actions available to Team Leads. They can also manage bugs across all teams and applications, assign unassigned bugs and view admin- specific analytics.

Each use case is represented by an oval, while the users who can perform them are shown as stick-figure actors. This diagram helps visualize role-specific access and system functionality in a concise way and supports better understanding of how the BugTrackR system enforces access control.

Specific Use Case - Submit Bug Report (Admin)

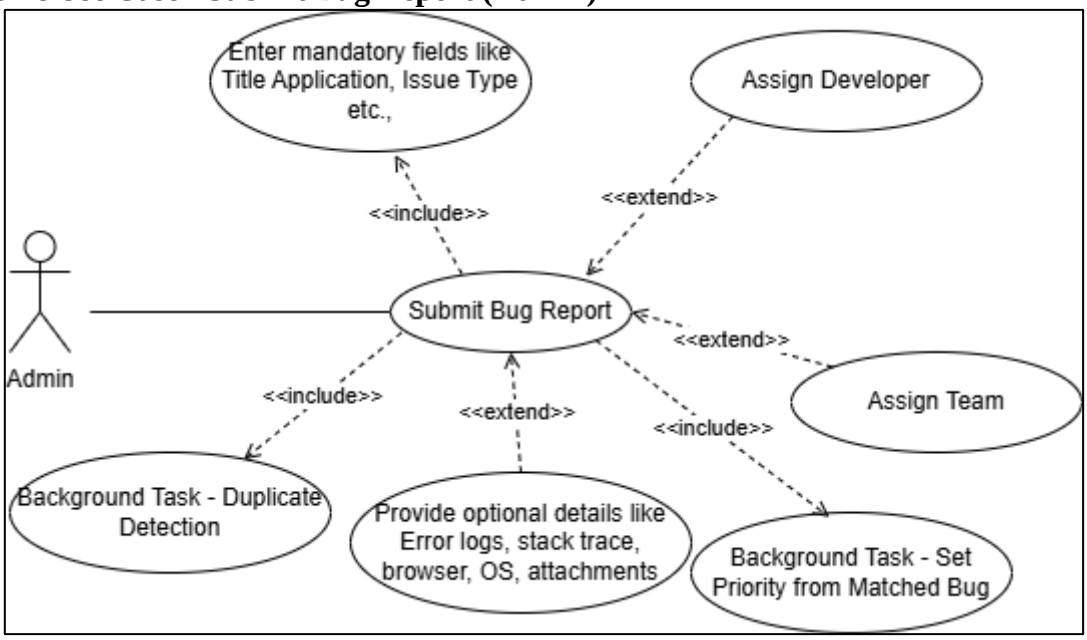


Figure9: Submit Bug Report - Admin Use Case

The above diagram, Figure 9, focuses on the Submit Bug Report functionality specifically from the perspective of an Admin. As shown in the diagram, the main use case is supported by several included and extended sub-use cases that reflect both mandatory and optional steps, along with background processes.

Included Use Cases (<<include>>)
These are actions that are always part of the bug submission process:

- **Enter Mandatory Fields**: This includes essential details such as bug title, description, application name, issue type and steps to reproduce the issue. The form will not submit without these values.
- **Run Duplicate Detection:** Automatically triggered after submit button is clicked to check for similar existing bugs using the background FAISS microservice.
- **Set Priority from Matched Bug:** If a similar bug is found, the system takes the priority of that bug.

Extended Use Cases (<<extend>>)
These are conditional/optional actions:

- **Provide Optional Technical Info:** Admins can choose to enter additional fields such as error logs, browser/OS details (auto detected but can be changed) and upload attachments.
- **Assign Developer:** If the Admin is confident about who should fix the issue, they may directly assign a developer.
- **Assign Team:** Similar to assigning a developer, Admins can assign the bug to the relevant technical team.

6.4 Initial Embedding for Semantic Features

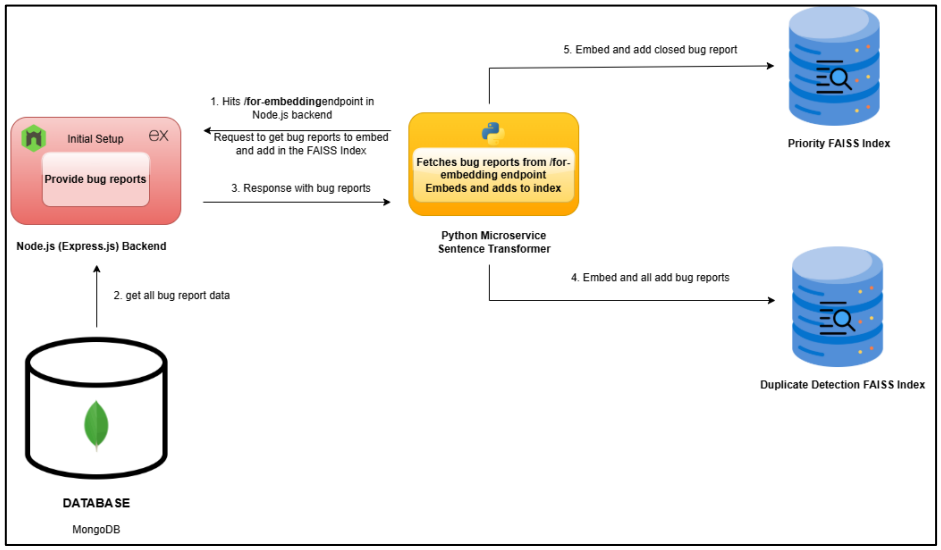


Figure 10: Initial embedding of bug reports into FAISS Index (for Duplicate Detection, Semantic Search and Priority Classification)

Before semantic similarity can be used for duplicate detection, priority classification or semantic search, the system performs an initial embedding step during bootstrap. This process converts all bug reports into vector form and stores them in separate FAISS indices for efficient retrieval, Figure 10. For specific design details, refer to Sections 6.5 (Duplicate Detection) and 6.7 (Priority Classification).

6.5 Duplicate Detection Design

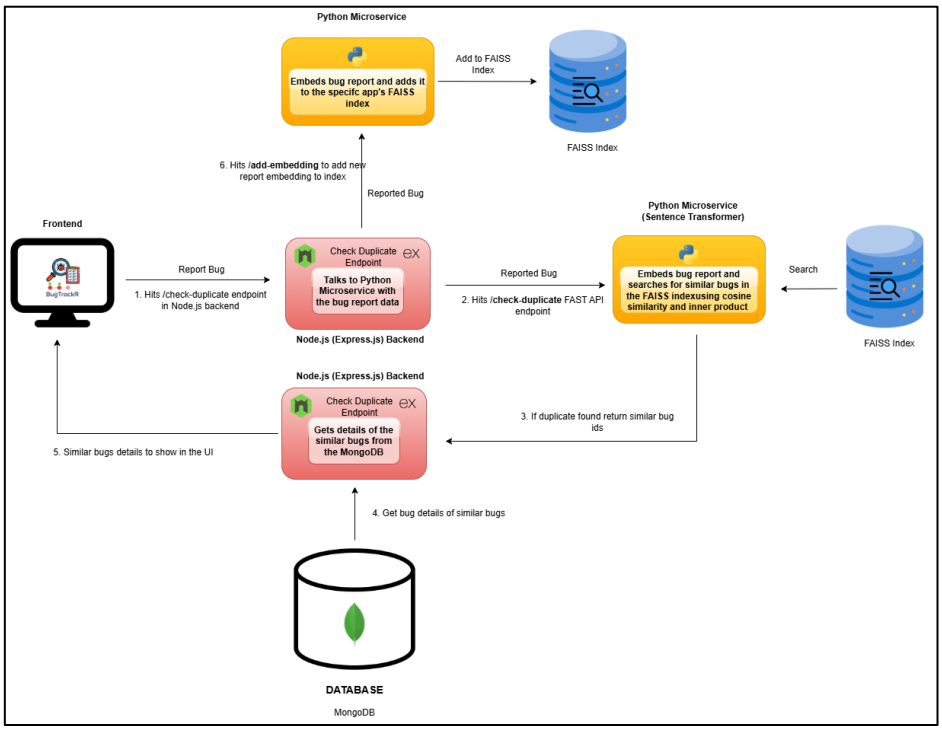


Figure 11: Duplicate Bug Detection Flow for New Bug Reports

The diagram (Figure 11) above illustrates the end-to-end duplicate detection workflow. When a user reports a new bug, the backend immediately sends this data to a Python microservice, which embeds the report and searches for similar cases in the FAISS index using cosine similarity. If a match is found, only the IDs are returned and the backend retrieves full bug details from MongoDB and sends to the frontend which shows the potential bugs and if submitted then the bug report is also embedded and added to the FAISS index for future detection.

BugTrackR adopts a semantic similarity-based approach rather than traditional keyword matching, allowing it to detect duplicates even when the same bug is described in different ways.

BugTrackR uses a semantic similarity approach for duplicate detection rather than traditional keyword matching, allowing it to detect duplicates even when the same bug is described in different ways. It is done by embedding bug titles, descriptions, and steps using the all-MiniLM-L6-v2 sentence transformer [14]. These embeddings are indexed with FAISS (Facebook AI Similarity Search) [25], which allows efficient similarity search.

Since cosine similarity better captures semantic meaning than Euclidean distance or inner product, all vectors are normalized, transforming inner product comparisons into cosine similarity. This enables accurate detection even when bugs are described differently.

Each application maintains a separate FAISS index to avoid misleading cross-application matches due to domain differences (e.g., bugs in "Marketplace" are compared only within that app).

Note: In development, the Python microservice accesses the HTTPS-secured Node.js backend to get bug reports to index using verify=False to bypass self-signed certificate errors. A custom header (x-similarity-key) is used to restrict access to internal endpoints in the Node.js backend, Figure 12. CORS is configured for browser access only and does not apply to backend-to-backend communication.

```python
#Bootstrap method to get data from the database through nodejs API to embed them and store in FAISS index
def bootstrap():
    try:
        #Get bugs from the nodejs backend API for embedding
        res = requests.get(
            "https://localhost:5000/api/bug-reports/for-embedding",
            headers={
                "x-similarity-key": os.getenv("SIMILARITY_API_KEY")
            },
            verify=False #Allow self-signed certs – needed because the nodejs backend uses https with a self-signed certificate
        )
        if res.status_code != 200:
            raise Exception("Failed to fetch bugs")
```

Figure 12:Python bootstrap call to Node.js backend with self-signed HTTPS and API key

6.6 Auto Assignment Design

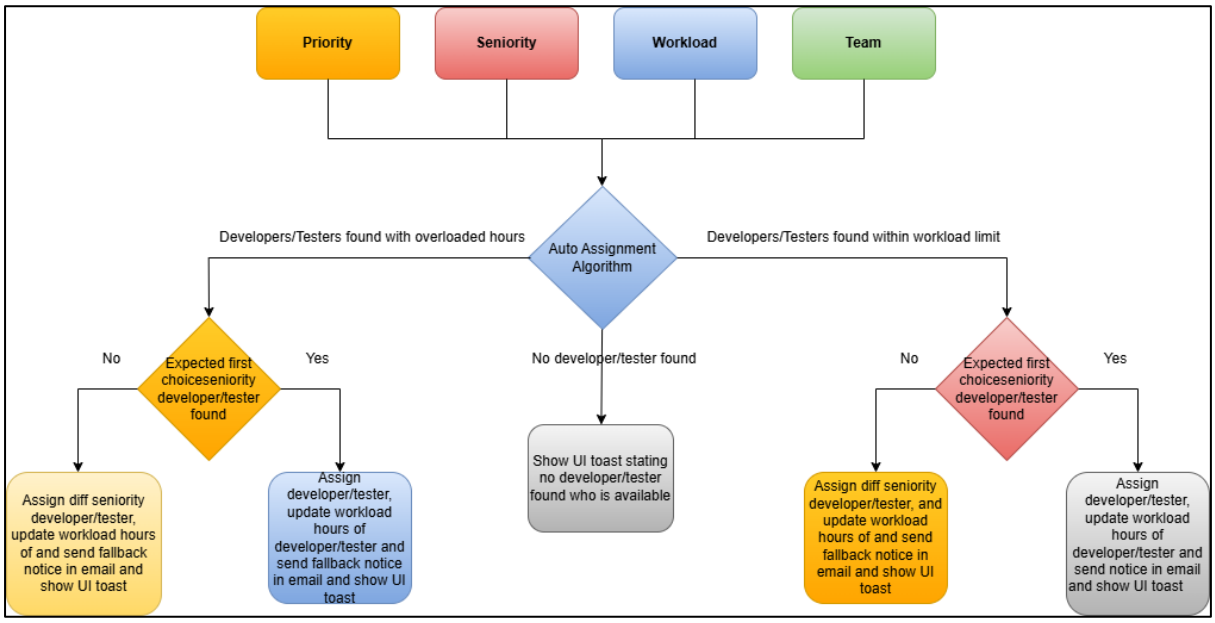


Figure 13:Auto-Assignment Flow

In BugTrackR, bugs can be automatically assigned to developers/testers based on a combination of factors such as **priority level, team, current workload,** and **developer/tester seniority**. The goal of this design is to ensure critical bugs are assigned quickly and fairly, while also balancing workload and preventing over-allocation. This logic is summarized in Figure 13. Fallback cases are highlighted with corresponding notifications.

To achieve this, the logic is split into two layers: a decision algorithm that selects an appropriate developer, and an API interface that manages role-based access, updates workloads, and triggers notifications. The system supports both auto-assignment (triggered by a team lead or admin) and automated retries via a scheduled background job.

The developer selection algorithm uses a priority-driven seniority preference model, where critical and high-priority bugs are assigned to senior developers first, while medium and low-priority bugs prefer mid-level or junior developers. A predefined mapping [37] assigns estimated effort in hours for each priority level. The system checks each eligible developer's workload, and if one is found within the 40-hour limit, they are assigned. If not, the algorithm looks for developers who are slightly over-allocated but still within an upper threshold (e.g., 45 hours for critical bugs). These fallback cases are tracked with a message explaining why the assignment was made.

Once a developer is selected, their workload hours are updated accordingly in the database. A notification email is also sent to the assigned developer. If the system makes a fallback assignment (e.g., to an overallocated developer or someone of lower seniority), a separate notice is sent to the team lead or admin who triggered the assignment, encouraging them to monitor or reassign if needed.

To prevent bugs from remaining unassigned, a background cron job runs twice daily during working hours. It checks for any bugs that have not been assigned to a developer and attempts

auto-assignment again. This helps ensure pending bugs are not overlooked if a developer was unavailable at the time of initial submission.

This design provides a balanced mix of automation and oversight. It ensures bugs are assigned thoughtfully rather than randomly, improves team efficiency and supports fair workload distribution. At the same time, it leaves room for human judgment through fallback notices and reassignment options.

6.7 Priority Classification Design

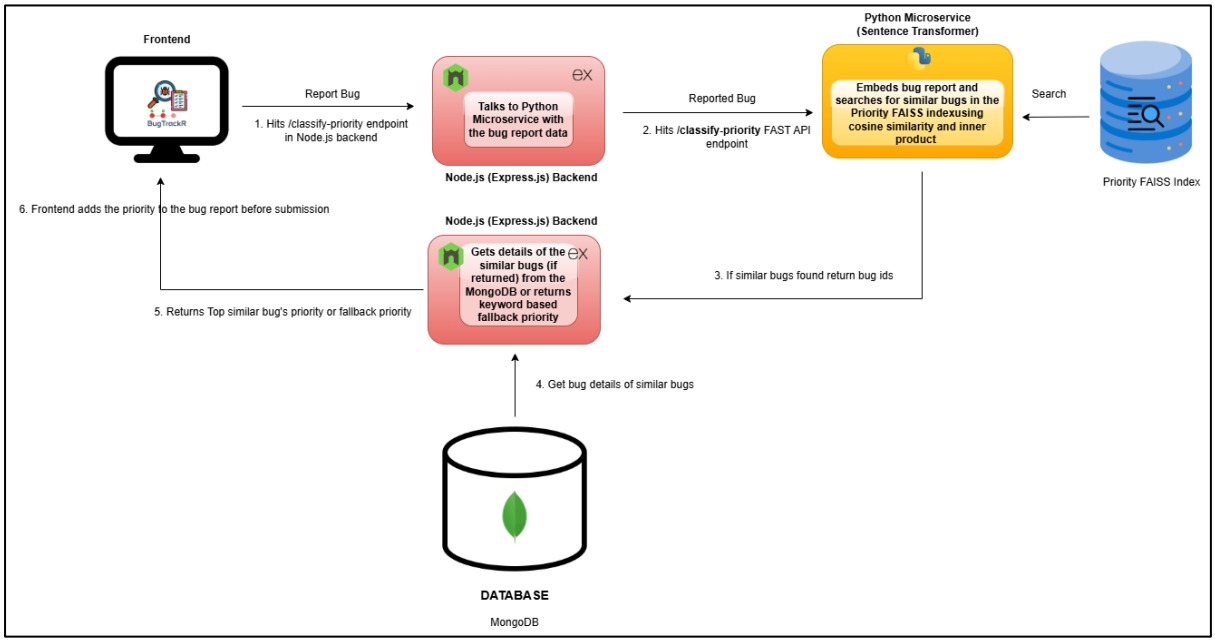


Figure 14:Priority classification flow - reuses priority from top similar closed bugs using FAISS

The diagram Figure 14, above shows the priority classification process. When a user submits a bug report, the system sends the bug data to the Python microservice that checks for similar past **closed** bugs using semantic similarity. If a match is found, its priority, or if not found a keyword-based approach using common terms like “crash,” “loss,” or “unable”,its priority or “Medium” priority (if no match is found), is returned to the frontend which attaches it to the bug report data and sends to backend for saving in to the database. This ensures consistent priority classification by learning from historical data. See Section 6.4 for more information.

To support human judgment, a notification is sent to the team lead and admin informing them that the priority was classified automatically. This gives them an opportunity to review and manually adjust the priority if the assigned value does not reflect the true urgency of the issue.

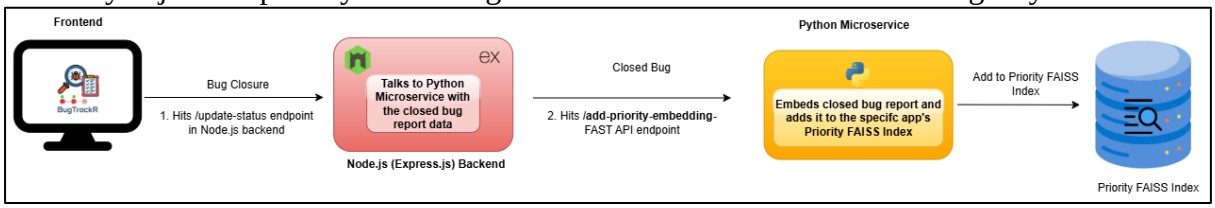


Figure 15: Closed bug embedding flow - adds priority-tagged reports to the FAISS index

The diagram Figure 15, above illustrates how BugTrackR updates the Priority FAISS index. When a bug is closed, its data is sent to the Python microservice, which embeds the report and stores it in the appropriate index for future classification.

6.8 Drag-and-Drop Status Tracking Design

BugTrackR supports intuitive bug status management via a drag-and-drop interface using react-dnd [26], primarily used by developers and testers. This feature was introduced based on supervisor idea and follows a visual board-style layout.

Users can drag bug cards across status columns (e.g., “Assigned” to “Fix In Progress”) to update their current status. This reduces clicks and improves clarity, especially when handling multiple bugs simultaneously. To promote accountability, the system flags bugs that have not had a status update in over 24 hours using a red clock icon, also an idea from supervisor.

For critical bugs, certain transitions are restricted (e.g., “Tested & Verified” to “Closed”) to ensure review is done by tech users. These rules are enforced at the UI level for immediate feedback, with validation also applied on the backend.

6.9 Chat Between Reporters and Tech Users Design

BugTrackR includes an in-app chat feature to help reporters and technical users (like developers and testers) communicate easily about specific bugs. This avoids confusion and delays that can happen when follow-up questions are sent through email.

The chat appears as a small floating button on the bug details page. When clicked, it opens a real-time chat window that shows the full message history. All users involved in the bug can send and receive messages here.

The chat works using Socket.IO for real-time communication and MongoDB to store chat sessions and messages. Each bug has its own chat room. When a user opens the chat, they join that room and can see past messages. Any new message is sent instantly to others in the room and also saved to the database.

The UI is built using the ChatScope React UI kit [27], which makes the messages easy to read and interact with. This chat feature keeps all bug-related conversations in one place and improves coordination between reporters and technical teams.

6.10 Advanced Search & Filtering

To help users find bugs more efficiently, BugTrackR includes a powerful search and filtering system. The design supports both exact and semantic search modes, along with multiple filters and date range selectors.

In exact match, user queries are split into individual words and a MongoDB $or condition is used to search across fields like bug ID, title, and description. This allows partial matches and the search works well for numeric queries too (e.g., “128” for “BUG-128”).

In semantic search mode, the search sentence is sent to a Python microservice (Uses the same approach used by Duplicate Detection (See Section 6.4 for details) that returns the most semantically similar bugs based on meaning, not keywords. The backend then fetches the full details of the returned bugs and displays them.

In addition to search modes, users can filter bugs by priority, status, issue type, assigned developer/tester, and reported date. The date pickers include min-max validation to prevent invalid input ranges (eg., choosing start date after to date). Filters and search can be combined in a single query, offering great flexibility.

The backend constructs a MongoDB query dynamically based on user inputs, ensuring correct application of all filters. This design keeps the interface responsive while enabling both precise and semanticsearches.

6.11 Logo and Branding

The BugTrackR logo, Figure 16, was designed using Figma [29], a widely used design tool for creating digital graphics and UI assets. The logo visually represents the core ideas of the application, including structured bug reporting, organized bug management, status tracking, and maintaining systematic bug records.

Figure 16: BugTrackR Logo

The primary colour scheme for BugTrackR is based on `#102B4E`, a deep blue tone selected to convey professionalism, that is crucial for a defect management platform. This colour is consistently applied across the application's headers, navigation bars, buttons and key interface elements to ensure visual coherence.

For typography, BugTrackR uses a clean and modern font family comprising Inter, Roboto, Arial, and sans-serif. This font stack was chosen to maintain readability and a professional appearance.

7. Implementation

7.1 Bug Reporting

Overview

The bug reporting feature allows users to submit structured and detailed reports, helping technical teams resolve issues efficiently without needing follow-up questions for basic information that can be brought in from the form. The form captures key information like title, description, application, issue type, steps to reproduce and optional like browser, OS, and optional attachments.

To suit different roles, the form varies slightly:

- **General users** report what they were doing and what went wrong, Figure 17.
- **Technical users** (e.g., developers) can include steps to reproduce directly, logs and stack traces, Figure 18.

Figure 17: Bug Report Form View General User

Figure 18:Bug Report Form View Tech User

Developers/Testers can assign themselves to a bug that is unassigned but falls under their team, Figure 19. The code Figure 20 and 21 shows that when the checkbox is clicked the developer's email is set to the form data in the assignTo.developer field.

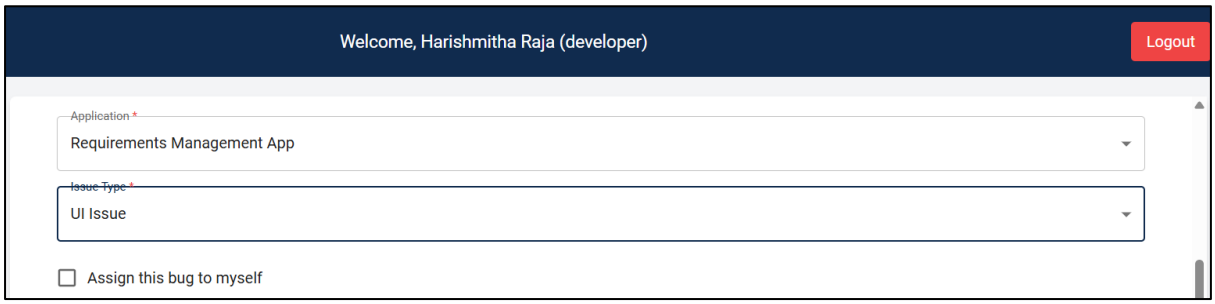


Figure 19:UI for self-assigning bug (Developer)

```
{/* Only visible if the logged in user is the developer of the selected app and if he/she is the developer of the
team associated with the issue type*/}
{isDeveloperOfSelectedApp &&
  teamAsscociatedToIssueType &&
  userRoles.some(
    (role) =>
      role.application === bugReportFormData.application &&
      role.role === "developer" &&
      role.team === teamAsscociatedToIssueType
  ) && (
    <FormControlLabel
      control={
        <Checkbox
          checked={selfAssign}
          onChange={handleCheckboxChange}
          color="primary"
        />
      }
      label="Assign this bug to myself"
    />
  )}
```

Figure 20: Code snippet to show check box to self-assign bug

```
//Self assignment checkbox change update
const handleCheckboxChange = (e) => {
  const { checked } = e.target;
  console.log(checked);

  setSelfAssign(checked);

  setBugReportFormData((prev) => ({
    ...prev,
    assignedTo: { developer: checked ? userEmail : "" },
    status: checked ? "Assigned" : "Open",
  }));
};
```

Figure 21: Code snippet to set developer email in form data

Team leads can assign any bug within their team to a suitable developer by selecting a developer from the dropdown list, Figure 22. The code conditionally for showing the developers list is shown in Figure 23.

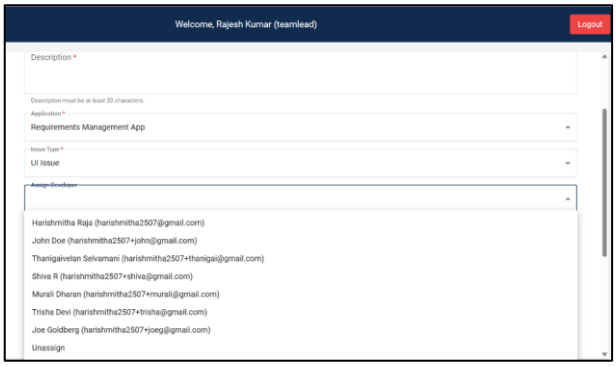


Figure 22: Dropdown populated with eligible team members

```
{/* Only visible if the logged in user is the team lead of the selected app and
if team of the lead and the issue type team match*/}
{isTeamLeadOfSelectedApp &&
  teamAsscociatedToIssueType &&
  teamAsscociatedToIssueType === developerTeam &&
  developersList.length > 0 && (
    <FormControl fullWidth>
      <InputLabel>Assign Developer</InputLabel>
      <Select
        name="assignedToDeveloper"
        value={bugReportFormData.assignedTo?.developer || ""}
        onChange={handleChange}
        displayEmpty
      >
        {developersList.map((dev) => (
          <MenuItem key={dev.email} value={dev.email}>
            {dev.fullName} ({dev.email})
          </MenuItem>
        ))}
        <MenuItem key="Unassign" value="Unassign">
          Unassign
        </MenuItem>
      </Select>
    </FormControl>
  )}
```

Figure 23: Code snippet for developers list dropdown

When the **admin**, first select issue type, the corresponding team is shown automatically and when they try to change the team unrelated to the issue, they see a warning regarding the same Figure 24. The bug can be assigned to a different team even if the issue type is different from their team.

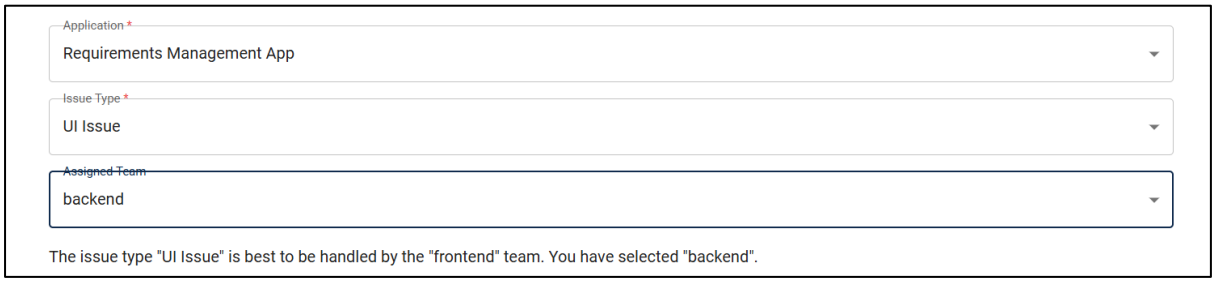


Figure 24: Assigned Team mismatch with issue type warning

When the team lead/admin chooses a developer from the list the check availability function is run which assumes the hours to be 6 (considering it can be critical or high priority bug and it's a changeable value) in the background to check if the developer is available, Figure 26. If not, the message is shown in the UI, Figure 25.

Figure 25: Selected developer not available

```
router.post('/check-availability', authenticateUser, async (req, res) => {
  try {
    const roleEntry = developer.roles.find(
      (r) =>
        r.role === "developer" &&
        r.application === application &&
        r.team === team
    );

    const currentHours = roleEntry?.workloadHours || 0;

    const isAvailable = currentHours + ASSUMED_HOURS <= 40;

    return res.status(200).json({
      available: isAvailable,
      currentWorkload: currentHours,
      assumedHours: ASSUMED_HOURS,
      maxLimit: 40,
      message: isAvailable
        ? "Developer is available (under assumed workload of 6 hours for the current bug being submitted.)."
        : "Selected developer has high workload and cannot be assigned right now.",
    });
  } catch (err) {
    console.error("Error ", err.message);
    return res.status(500).json({ error: "Internal server error." });
  }
});
```

Figure 26: Code Snippet to check availability of developer

Validation

Fields like Bug Title and Description have upper limits (e.g., 100 and 500 characters respectively). A live counter informs the user about base char count, Figure 28 and input is restricted beyond the limit. Core inputs like Bug Title, Application, Issue Type, and Description are required for submission. If submitted blank, red borders and inline messages guide the user to complete them, Figure 27.

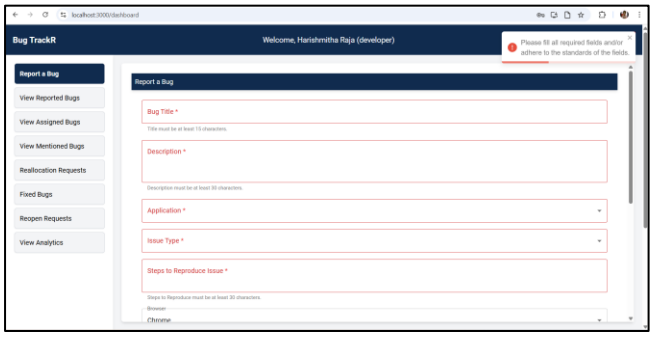


Figure 27: Mandatory fields validation

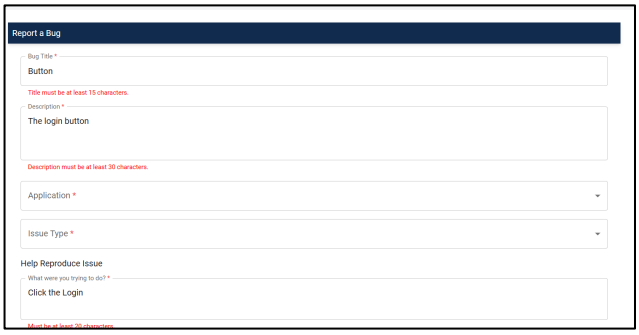


Figure 28:Character count validation

To support rich bug reporting, BugTrackR includes a drag-and-drop interface for attachments. Both general and technical users can upload upto 5 files, but with different allowed file types. General users are restricted to images and videos, Figure 29, while technical users can upload a wider range including .txt, .log, .pdf, .docx, and .csv files, Figure 30. The frontend uses react-dropzone and file type restrictions for user is shown in Figure 31.

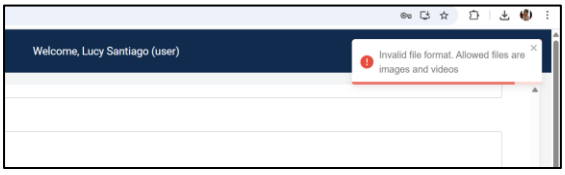


Figure 29: General user uploading restricted file added

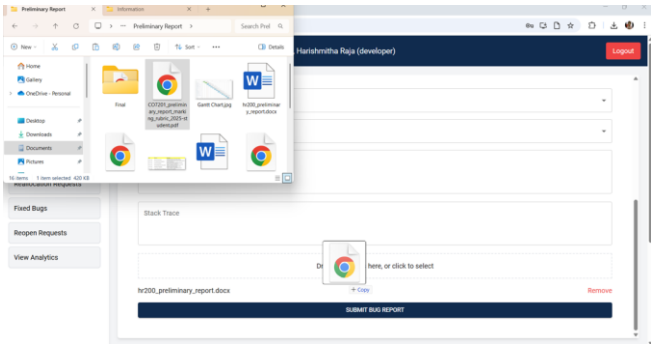


Figure 30: Tech user drag-and-drop interface showing unrestricted file upload

```
// Dropzone config - allows only images and videos for geenral users
const { getRootProps, getInputProps } = useDropzone({
  onDrop,
  onDropRejected,
  multiple: true,
  accept: {
    "image/*": [".png", ".jpg", ".jpeg", ".gif", ".bmp", ".webp"],
    "video/*": [".mp4", ".mov", ".avi", ".mkv", ".webm"],
  },
  maxFiles: FIELD_LIMITS.maxAttachments,
});
```

Figure 31: Code enforcing file type restrictions based on user role

On successful bug report submission an email is sent to the bug reporter, Figure 32.

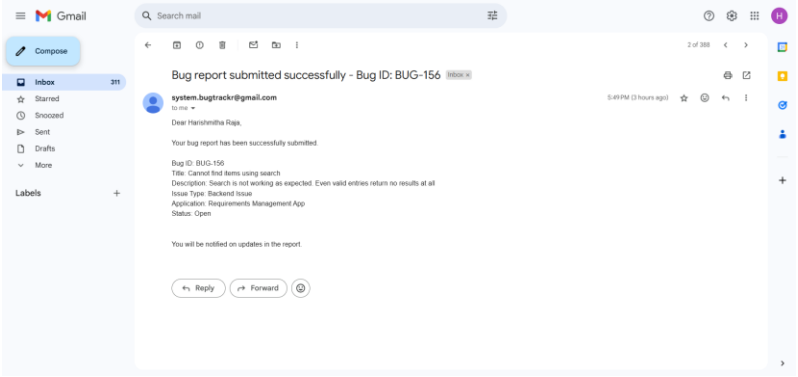


Figure 32: Email notification on bug report submission

7.2 Duplicate Detection

1. UI Trigger and Duplicate Suggestions
When a user submits a bug report, the frontend sends the bug title, description, and steps to the Node.js backend via the /check-duplicate endpoint. This logic is shown in Figure 33.

```
//API endpoint to check for duplicate
router.post(
  "/check-duplicate",
  authenticateUser,

  async (req, res) => {
    try {
      const {
        application,
        title,
        description,
        stepsToReproduce,
        userSteps,
        bugId,
      } = req.body;

      let response;
      try {
        if (bugId) {
          const bug = await BugReport.findOne({ bugId });

          if (!bug) {
            return res.status(404).json({ message: "Bug not found isjdksnd" });
          }

          //Not using thecheckIfBugIsClosed and checkIfBugIsDuplicate middlewares because when a new bug report is being reported
          //we won't have the bug id but it's expected in the middlewares
          //So manually checking
          if (bug.status === "Closed") {···
          }

          if (bug.status === "Duplicate") {···
          }

          //Only during these current status of the bug, duplicate check can be done if not done during submisison
          const validStatusesForDuplicateCheck = ["Open", "Assigned"];
          if (!validStatusesForDuplicateCheck.includes(bug.status)) {···
          }
        }

        //Send bug details to the Python duplicate detection service to check for similar existing bugs
        response = await axios.post("http://localhost:8000/check-duplicate", {
          application,
          title,
          description,
          stepsToReproduce,
          userSteps,
        });
```

Figure 33: Nodejs API endpoint /check-duplicate

The Node.js backend forwards this request to the respective endpoint for detecting duplicate. The Figure 34 shows code snippet which gets bug report details converts it into a single string and searches for similar bugs in the index. The Fast API is used to expose the microservice. Figure 35 and 36 shows sample request and response from the /check-duplicate endpoint. The flow is described in the Design section in detail (Section 6.5).

```python
#Endpoint to check if a bug is a possible duplicate
@app.post("/check-duplicate")
def check_duplicate(bug: BugInput):
    print(bug)
    if not bug.application:
        return {"error": "Application is required for duplicate check."}

    #Convert bug fields into a single string
    text = embeddings.build_text_input(
        bug.title,
        bug.description,
        bug.stepsToReproduce,
        bug.userSteps or {}
    )
    #Get similar bugs based on embedding similarity
    results = embeddings.search_similar(bug.application, text)

    #Return bug ids and similarity scores
    similar_bugs = []
    for r in results:
        similar_bugs.append({
            "bug_id": r[0],
            "score": r[1]
        })
    return {"similar_bugs": similar_bugs}
```

Figure 34: Python endpoint /check-duplicate

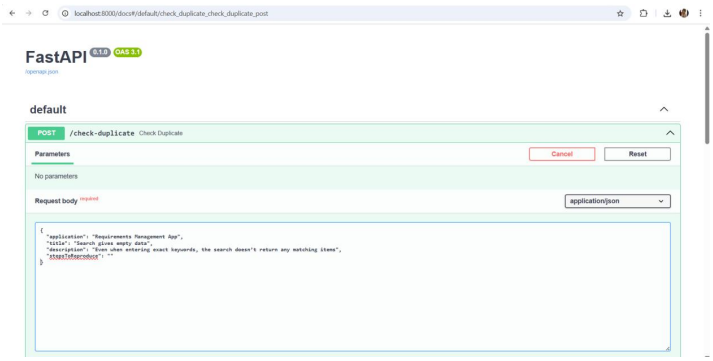


Figure 35:Fast API Request Body

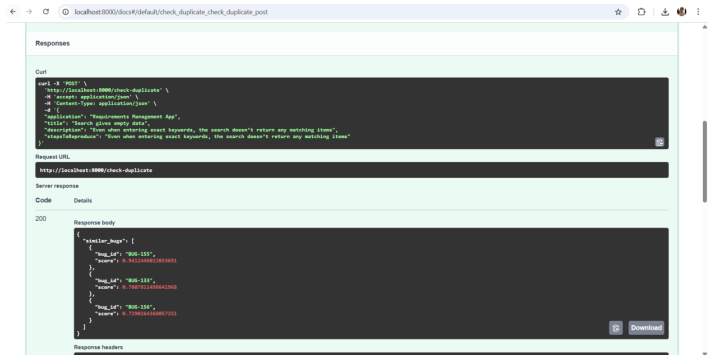


Figure 36:Fast API Response

Once the Node.js server receives the similar bug IDs, it queries the MongoDB database to fetch full bug details for each match. These are then returned to the frontend, which displays them in a modal, Figure 37. The reporter can cancel the submission or proceed anyway.

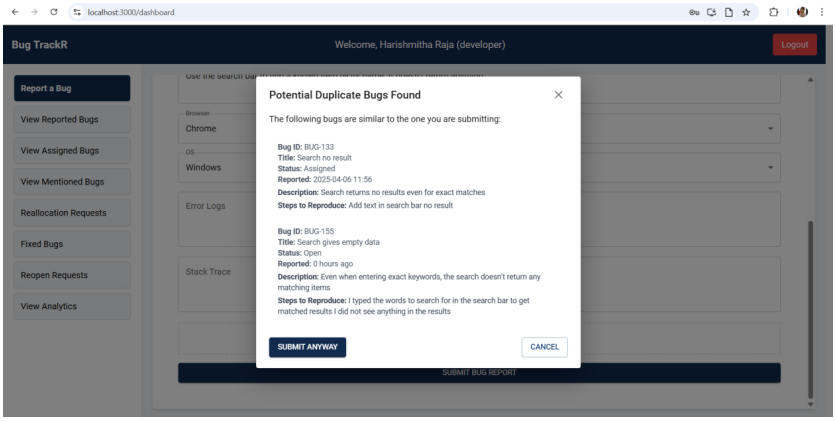


Figure 37: Potential Duplicates List During Submission

2. Similarity Search

The similarity check is handled by a separate method that receives the bug content and returns the top 3 semantically similar reports. The embedding logic (with normalisation) and similarity search implementation in the Python service are shown in Figure 38 and Figure 39.

```python
#Combine bug fields into a single string for embedding
def build_text_input(title, description, steps, user_steps):
    step1 = user_steps.get("step1", "")
    step2 = user_steps.get("step2", "")
    return f"{title} {description} {steps or ''} {step1} {step2}"

#Convert text into normalized vector for cosine similarity
def embed_text(text):
    return model.encode([text],normalize_embeddings=True)

#Set up FAISS index and bug id list for the app if not already done
def app_setup(app_name):
    if app_name not in app_indices:
        app_indices[app_name] = faiss.IndexFlatIP(embedding_dim)
        app_bug_ids[app_name] = []
```

Figure 38: Code Snippet for Embedding Text

```python
#Find top k similar bugs with similarity score greater than equal to min_score
def search_similar(app_name, text, k=3, min_score=0.6):
    #Return [] if the app has no index or no bugs stored
    if app_name not in app_indices or not app_bug_ids[app_name]:
        return []

    #Convert bug text to vector (list of numbers)
    vector = embed_text(text)

    #Search the FAISS index for top k similar bug vectors
    scores, indices = app_indices[app_name].search(np.array(vector), k)

    results = []
    for i in range(k):
        score = float(scores[0][i])
        index = indices[0][i]

        #if the index is valid and score is above threshold, add the result
        if index < len(app_bug_ids[app_name]) and score >= min_score:
            bug_id = app_bug_ids[app_name][index]
            results.append((bug_id, score))

    #Sort in descending order based on score
    results.sort(key=lambda x: -x[1])
    return results
```

Figure 39: Code Snippet for Searching Similar Bugs

3. Manual Trigger by Team Lead/Admin

If the duplicate check was skipped during submission due to the service was down, team leads and admins can manually run the check from the bug's detail view, Figure 40. This triggers the same Node.js API and uses the same flow as above. The results are shown in a collapsible panel below the bug description to all tech users associated with the bug Figure 41. This list is shown regardless of when the duplicate check was run either during submission automatically or triggered by team lead/admin

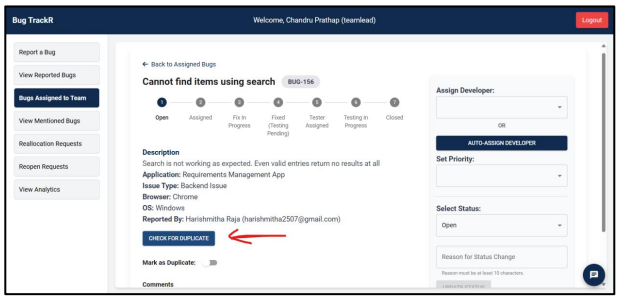


Figure 40: Check Duplicate Manual Trigger

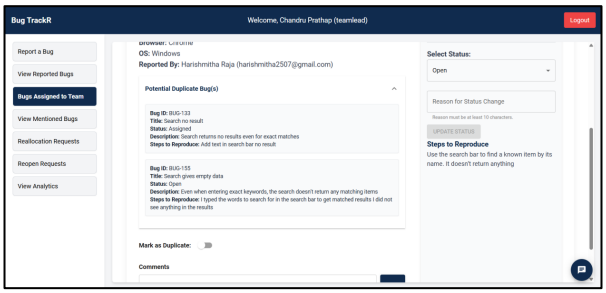


Figure 41: Potential Duplicate Bugs List in Bug Details View

4. Marking a Bug as Duplicate

If a bug is found to be a duplicate, tech users associated with the bug can mark it as such. Enabling the toggle reveals fields for the original bug ID and a required explanation. The frontend includes input validation to ensure correctness (Figure 42 shows the validation).

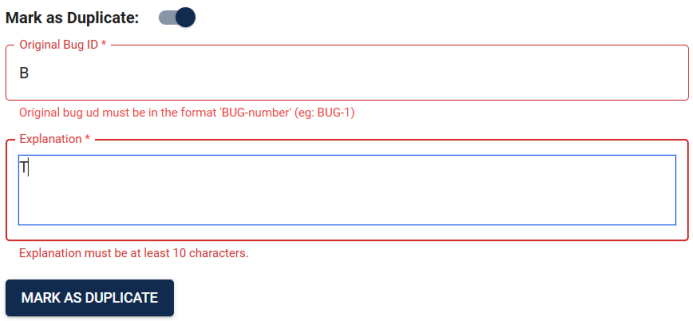


Figure 42: Mark Duplicate Validation

When valid input submitted, the Node.js backend updates the bug's record and triggers an email notification to both the team lead and admin, Figure 43 shows the resulting UI update, success toast message.

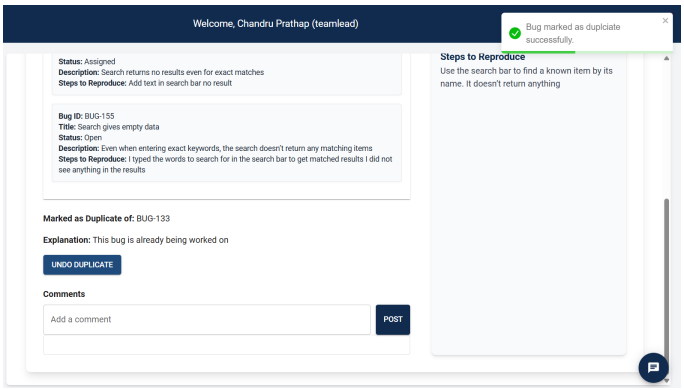


Figure 43:Marked Duplicate

5. Undo Duplicate

If needed, tech users associated with the bug can undo the duplicate marking by clicking the button shown in above Figure 43. The Undo UI update, shows success toast message like shown for mark duplicate and sends email notification.

7.3 Priority Classification using Similar Bugs

1. Background Trigger on Submission

When a bug report is submitted, the system automatically checks its priority in the background. The frontend sends the bug's title, description, and application to the backend. This process is not visible to the user, but run in the background, the network request can be seen in the browser's developer tools under the "/classify-priority" endpoint Figure 44. The figure shows check-duplicate and classify priority called one but user cancelled submission and other for the submitted anyways (submit endpoint)

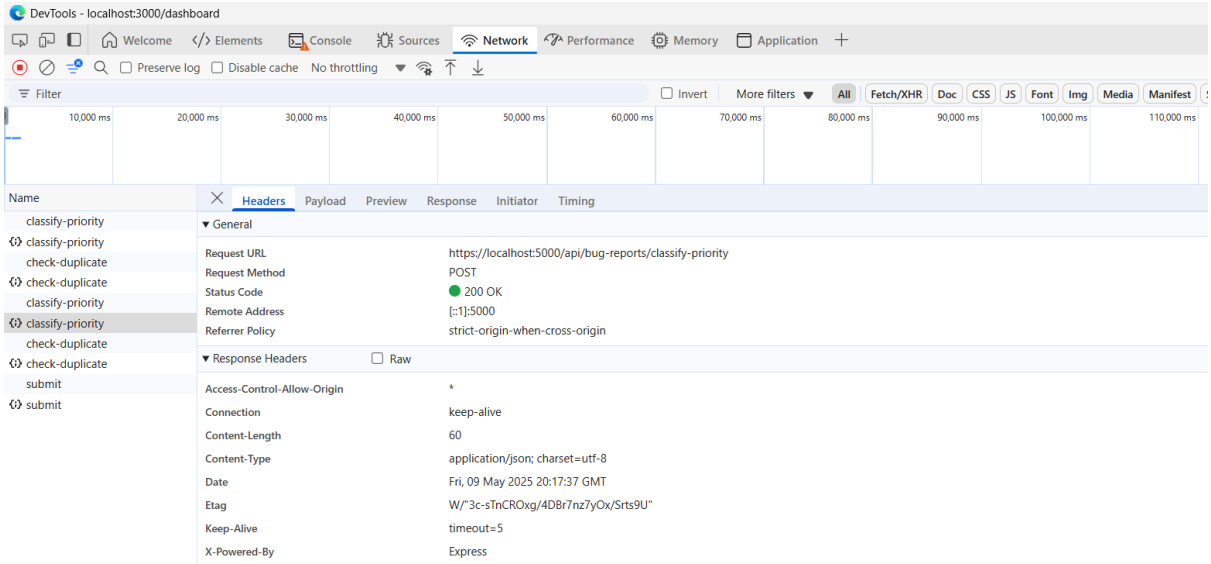


Figure 44: Network request triggered on bug submission (Browser Dev Tools)

The backend then forwards this data to a Python-based microservice, which performs semantic similarity matching with closed bugs. Based on the result, it either assigns the priority directly or falls back to keyword-based logic (Figure 45 shows the frontend code where this API is called in the background - in the handleSubmit method in the Form Page).

```
const BugReportFormUser = () => {
  const handleSubmit = async (e) => {

    const autoPriority = await classifyBugPriority(
      bugReportFormData.application,
      bugReportFormData.title,
      bugReportFormData.description
    );

    if (autoPriority) {
      bugReportFormDataToSend.append("priority", autoPriority);
    }
```

Figure 45: Frontend code calling the classify-priority

2. Node.js Backend

The backend receives the submitted bug data and contacts the Python microservice to perform a semantic similarity search. The logic follows the design described in Section 6.7. The priority response preview in the browser's developer tools can be seen in Figure 46. Once a priority is decided, it is included in the response sent back to the frontend.

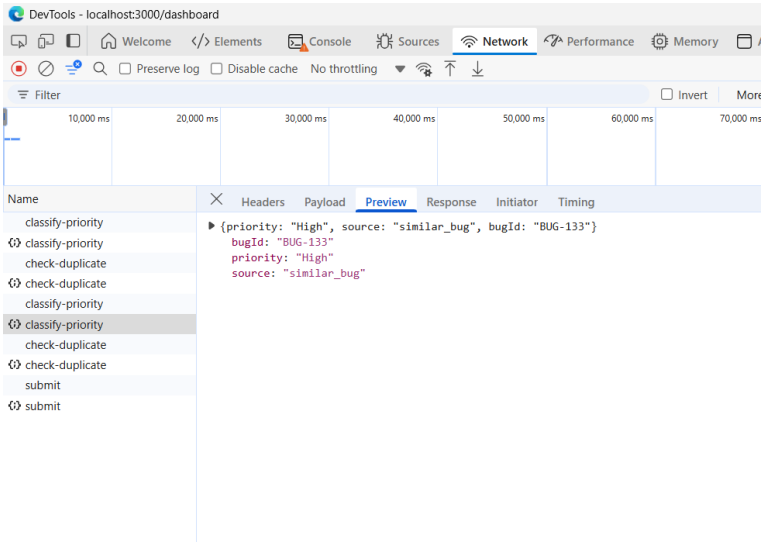


Figure 46: API response preview with classified priority

Note: The search similar bugs logic is exactly the same as for the Duplicate detection except for the index used and the bugs it has (see next point for more information).

3. Bootstrapping

Unlike duplicate detection, which indexes all bugs during the bootstrapping process, for priority classification, only indexes bugs with a **"Closed"** status. This ensures that priorities are learned from verified, resolved cases. The filtering logic during bootstrapping is shown in Figure 47.

```
#add each bug's embedding to the index one by one
embeddings.add_to_index(app_name, bug.get("bugId"), text)
#If bug status is "Closed" it goes to the priority classificarion index
if bug.get("status", "") in ["Closed"]:
    print(bug)
    embeddings.add_to_priority_index(app_name, bug.get("bugId"), text)
```

Figure 47: Filtering Closed status bugs to index

4. Notification

Once the priority is set, an email is sent to the team lead and admin informing them of the assigned priority and its source (semantic or fallback). This allows them to manually reclassify it if needed Figure 48.

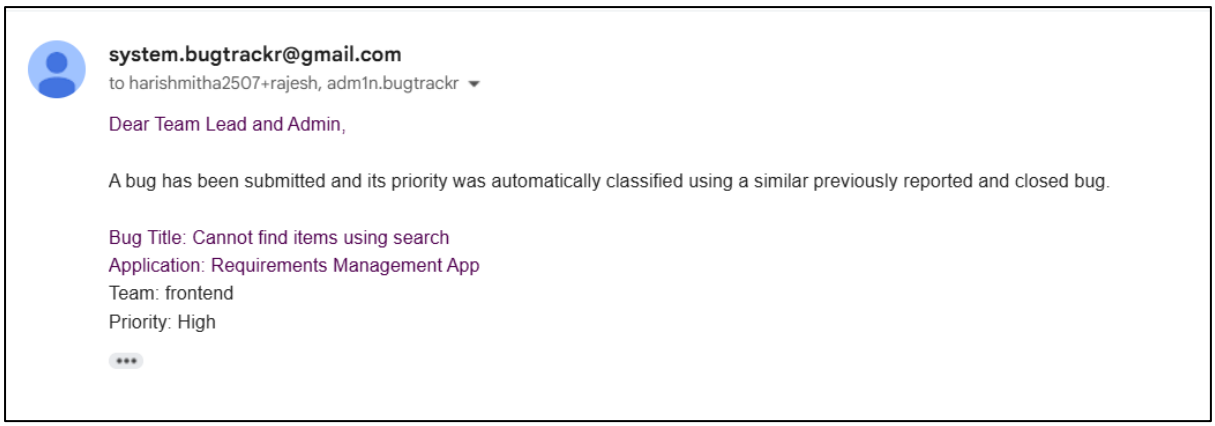


Figure 48:Email sent to team lead/admin after classification

7.4 Auto-Assignment of Developers/Testers

1. Trigger from the UI

Team leads and admins can trigger auto-assignment using the Auto-Assign Developer or Auto-Assign Tester buttons respectively in the bug details view Figure 49. This option is allowed only when the bug has a priority set. On successful assignment, the assigned user appears in the dropdown, and a success toast confirms the action Figure 50. If fallback logic is applied (either with strict check/within 40-hours limit or buffered/overloaded check), an additional info toast is displayed, Figure 51 and Figure 52.

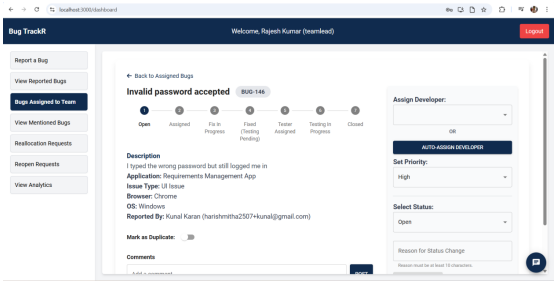


Figure 49: Auto-Assign Developer UI button in bug view

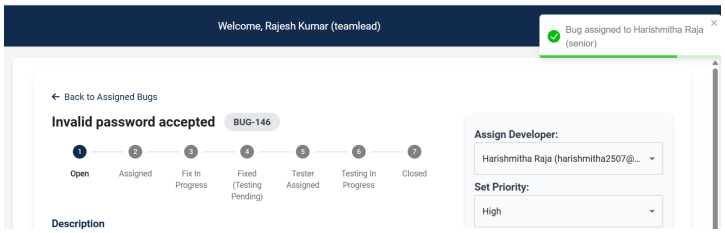


Figure 50: Success toast (developer assignment)

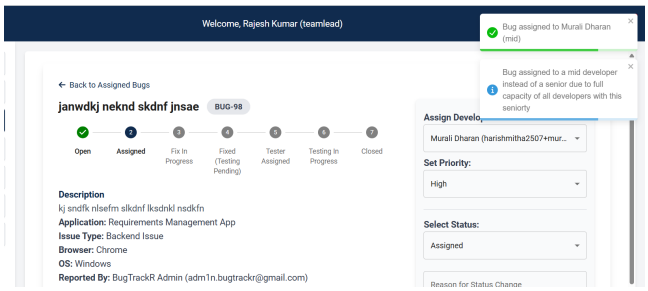


Figure 51: Success and Fallback notice toast (Strict 40-hours limit)

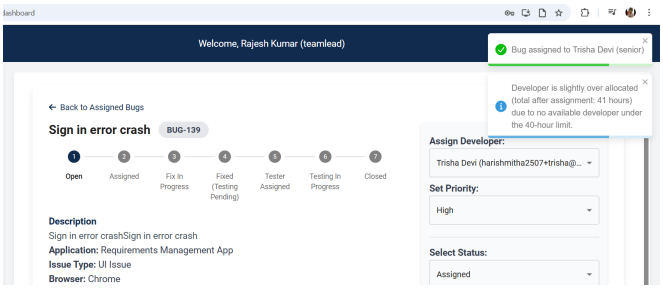


Figure 52: Success and Fallback notice toast (Overloaded)

2. Assignment Logic (Backend)

The design for assigning a developer/tester based on priority, workload, and seniority is explained in the Design section (see Section 6.6). The backend selects the most suitable person using that design and updates their workload accordingly. Figure 53 shows the estimated hours for each priority level, priority-wise assignment order and the max allowed hours. (can be modified as needed)

```
//Estimated development hours per priority (can be adjusted as needed)
const PRIORITY_DEVELOPER_HOURS = {
  Critical: 6,
  High: 9,
  Medium: 3,
  Low: 1,
};

//Priority assignment order
const PRIORITY_SENIORITY_ORDER = {
  Critical: ["senior", "mid", "junior"],
  High: ["senior", "mid", "junior"],
  Medium: ["mid", "junior", "senior"],
  Low: ["junior", "mid", "senior"],
};

//Max buffer hours
const PRIORITY_MAX_HOURS = {
  Critical: 45,
  High: 43,
  Medium: 42,
  Low: 40,
};
```

Figure 53: Auto-Assignment Estimates for Developer

Figure 54 and Figure 55 shows the selection function which explains the algorithm in Section 6.6, and Figure 56 and Figure 57 shows the backend route which calls the assignDeveloper method (algorithm) and updates the bug and email notification.

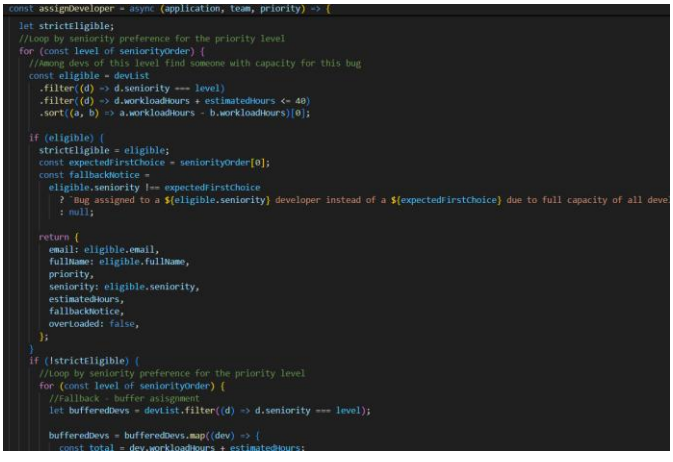


Figure 54: Code Snippet for Developer selection logic

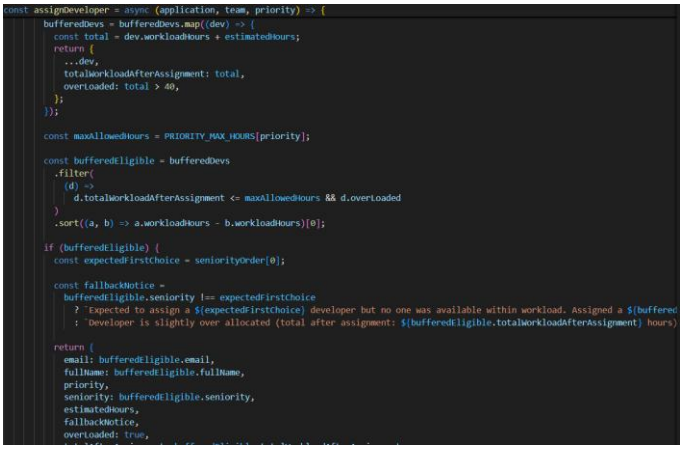
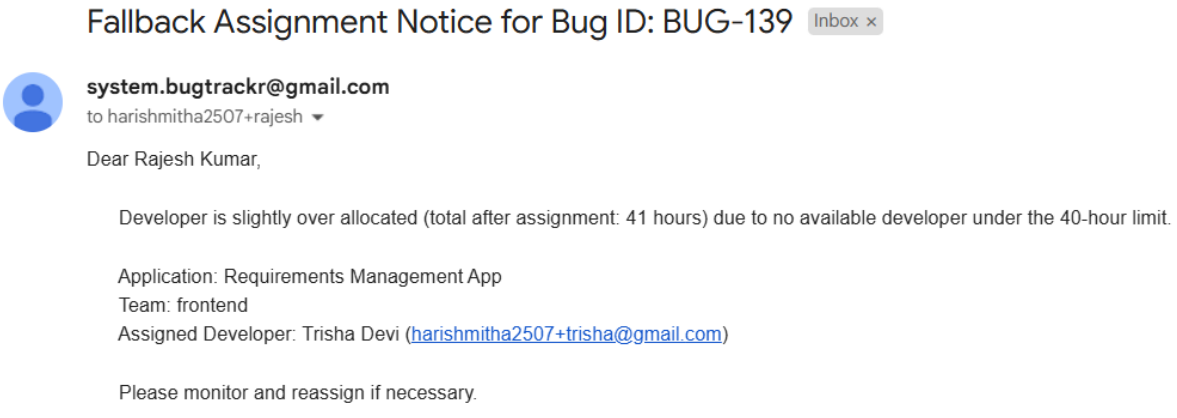


Figure 55: Code Snippet for Developer selection logic continuation

```
gReportRoutes.js > router.put('/auto-assign-developer') callback
sync (req, res) => {
  const dev = await assignDeveloper(application, assignedTeam, priority);

  if (!dev) {...
  }

  bug.assignedTo.developer = dev.email;
  bug.status = "Assigned";
  (bug.statusLastUpdated = new Date()),
    bug.changeHistory.push({
      type: "Assignment",
      developer: dev.email,
      changedOn: new Date(),
      changedBy: userEmail,
      changedByRole: isAdmin ? "admin" : "teamlead",
      reason: "Auto-assigned using system logic",
    });

  await bug.save();

  //Updae the workload hours of the matched developer

  if (dev.totalAfterAssignment > 40) {
    await User.updateOne(
      {
        email: dev.email,
        "roles.application": application,
        "roles.team": assignedTeam,
        "roles.role": "developer",
      },
      { $inc: { "roles.$.workloadHours": dev.estimatedHours } },
      {
        $set: { "roles.$.overLoaded": true },
      }
    );
  } else {
    await User.updateOne(
      {
        email: dev.email,
        "roles.application": application,
        "roles.team": assignedTeam,
        "roles.role": "developer",
      },
      {
        $inc: { "roles.$.workloadHours": dev.estimatedHours },
      }
    );
  }

  const transporter = nodemailer.createTransport({
    service: "gmail",
    auth: {
      user: process.env.EMAIL_USER,
      pass: process.env.EMAIL_PASS,
    },
  });

  if (dev.fallbackNotice) {
    const mailOptions = {
      from: process.env.EMAIL_USER,
      to: userEmail, //Send to the one (team lead or admin) who triggered assignment
      subject: `Fallback Assignment Notice for Bug ID: ${bug.bugId}`,
      text: `Dear ${user.fullName},

  ${dev.fallbackNotice}
```

Figure 56: Backend auto-assign route



Fallback Assignment Notice for Bug ID: BUG-139 Inbox ×

system.bugtrackr@gmail.com
to harishmitha2507+rajesh

Dear Rajesh Kumar,

Developer is slightly over allocated (total after assignment: 41 hours) due to no available developer under the 40-hour limit.

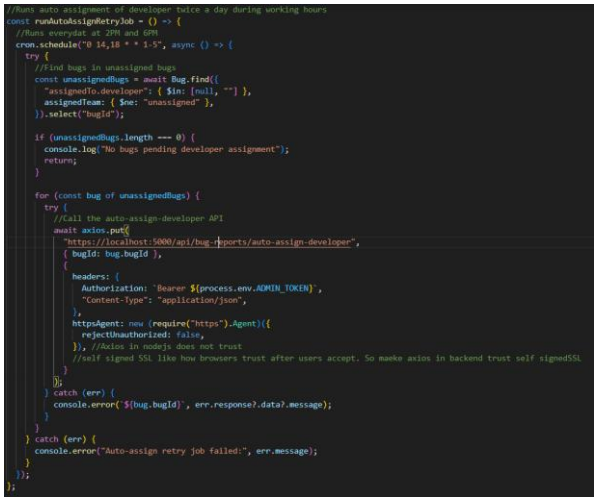
Application: Requirements Management App
Team: frontend
Assigned Developer: Trisha Devi (harishmitha2507+trisha@gmail.com)

Please monitor and reassign if necessary.

Figure 57: Email Notification on overallocation

3. Scheduled Retry

To ensure bugs don't remain unassigned, a background job runs twice daily on working days and reattempts assignment for any unassigned bugs Figure 58. The job uses the same API as manual triggers and is helpful when no developer or tester was initially available.



```
//Runs auto assignment of developer twice a day during working hours
const runAutoAssignRetryJob = () => {
  //Runs everydat at 2PM and 6PM
  cron.schedule("0 14,18 * * 1-5", async () => {
    try {
      //Find bugs in unassigned bugs
      const unassignedBugs = await Bug.find({
        "assignedTo.developer": { $in: [null, ""] },
        assignedTeam: { $ne: "unassigned" },
      }).select("bugId");

      if (unassignedBugs.length === 0) {
        console.log("No bugs pending developer assignment");
        return;
      }

      for (const bug of unassignedBugs) {
        try {
          //Call the auto-assign-developer API
          await axios.put(
            "https://localhost:5000/api/bug-reports/auto-assign-developer",
            { bugId: bug.bugId },
            {
              headers: {
                Authorization: `Bearer ${process.env.ADMIN_TOKEN}`,
                "Content-Type": "application/json",
              },
              httpsAgent: new (require("https").Agent)({
                rejectUnauthorized: false,
              }), //Axios in nodejs does not trust
              //self signed SSL like how browsers trust after users accept. So maeke axios in backend trust self signedSSL
            }
          );
        } catch (err) {
          console.error(`${bug.bugId}`, err.response?.data?.message);
        }
      }
    } catch (err) {
      console.error("Auto-assign retry job failed:", err.message);
    }
  });
};
```

Figure 58: Cron job retrying assignment for unassigned bugs

Note: The same logic and interface are used for auto-assigning testers once the bug is marked as fixed. The backend reuses the same algorithm and API structure, with tester-specific validations applied based on the bug status.

7.5 Drag-and-Drop Status Tracking

1. Column-Based Status View

The developer and tester assigned list view use a column layout Figure 59, where each column represents a bug status such as "Assigned" , "Fix In Progress" etc., Bugs are displayed as cards under their current status. This helps users easily see their workload.

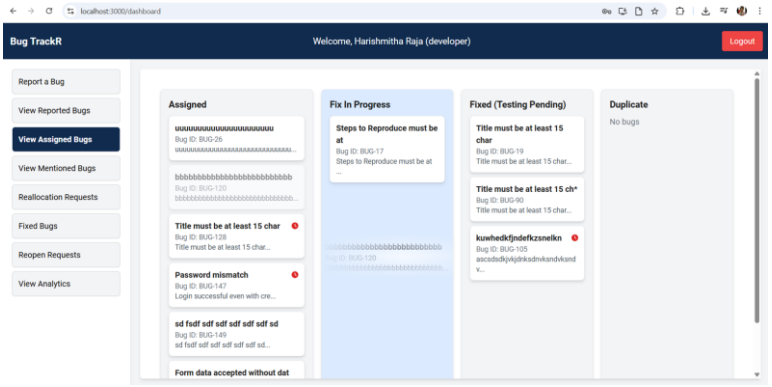


Figure 59: Drop target highlight while dragging a bug.

2. Drag-and-Drop Status Update

Bugs can be moved between statuses using drag-and-drop. When a bug card is dropped in a different column, moveBug() is called which updates the status by calling the API update-status, Figure 60.

```
const BugColumn = ({ status, bugs, moveBug, setSelectedBug }) => {
  console.log("Inside bug column");
  const [{ isOver }, drop] = useDrop(() => ({
    accept: "BUG",
    drop: async (item) => {
      const { bugId, currentStatus, priority } = item;
      //Restrict critical bug direct closure
      if (
        priority === "Critical" &&
        currentStatus === "Tested & Verified" &&
        status === "Closed"
      ) {
        toast.error(
          "Critical bugs must be moved to 'Ready for Closure' first."
        );
        return;
      }

      await moveBug(bugId, status);
    },
    collect: (monitor) => ({
      isOver: monitor.isOver(), //isOver to update UI styles during dragging of bug card
    }),
  }));

  return (
    <div
      ref={drop}
      className={`w-1/3 p-4 border ${
        isOver ? "bg-blue-100" : "bg-gray-100"
      } rounded-lg shadow-md`}
    >
      <h2 className="text-lg font-bold mb-2">{status}</h2>{" "}
      {bugs.length > 0 ? (
        bugs.map((bug) => (
          <BugCard
            key={bug.bugId + bug.statusLastUpdated}
            bug={bug}
            setSelectedBug={setSelectedBug}
          />
```

Figure 60: Bug Column Drop and Validation Logic

3. Restriction and flags enforced

- Critical bugs cannot be moved directly to “Closed” from “Tested & Verified” (shown via a toast message in Figure 61. It needs to be moved to “Ready For Closure” and the team lead or admin can close it.

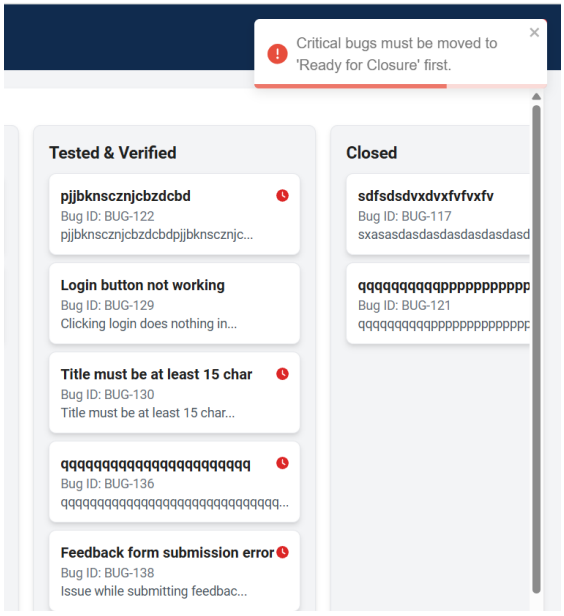


Figure 61:UI alert preventing direct closure of critical bugs

- A clock icon is shown on cards where status has not been updated in over 24 hours Figure 62, helping users identify bugs with no update.

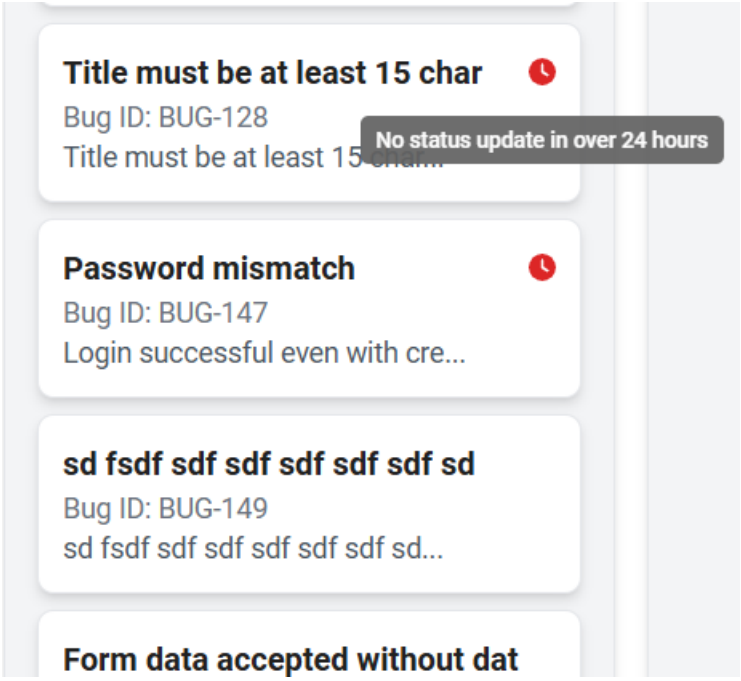


Figure 62:Red icon showing delayed status update (over 24 hours)

4. Modal View and Manual Update Support

Clicking a bug card opens a detailed modal view where users can also update status using a dropdown. These changes are reflected in the dashboard using the updateBugLocally() function which updates the state with the updates, ensuring that the correct column is updated without a page refresh Figures 63 and 64.

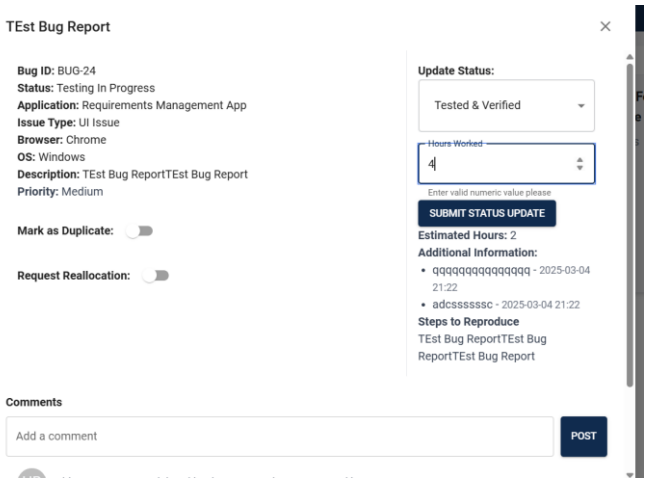


Figure 63: Status update and hours input in modal

```
const DeveloperAssignedBugsList = () => {
  //To make sure the card is displayed inthe correct column even if the status is changed using the dropdown in the details page
  //and the bug details are updated
  const updateBugLocally = (bugId, updates) => {
    setBugs((prevBugs) =>
      prevBugs.map((bug) =>
        bug.bugId === bugId ? { ...bug, ...updates } : bug
      )
    );
  };
```

Figure 64: Code Snippet in frontend to update bug status locally

7.6 Chat Between Reporters and Tech Users

1. Floating Chat UI

The chat can be opened from the bug details page using a floating icon Figure 65. It expands into a compact chat window where users can send and receive messages in real time. The interface provides a message layout, user avatars and timestamps Figure 66.

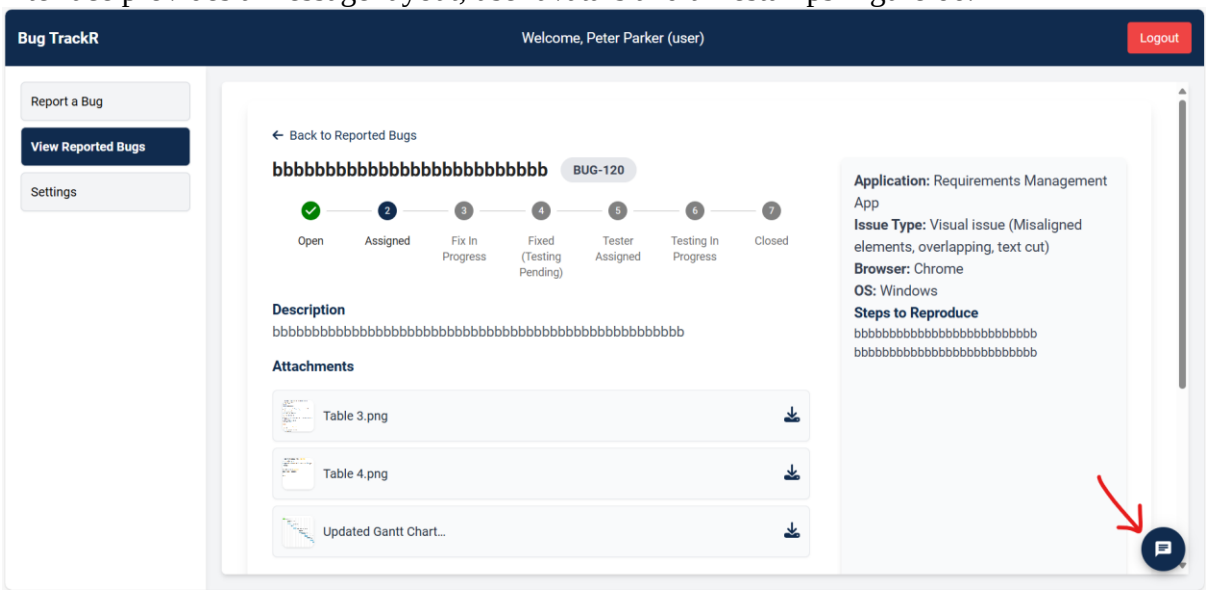


Figure65: Floating icon for opening chat

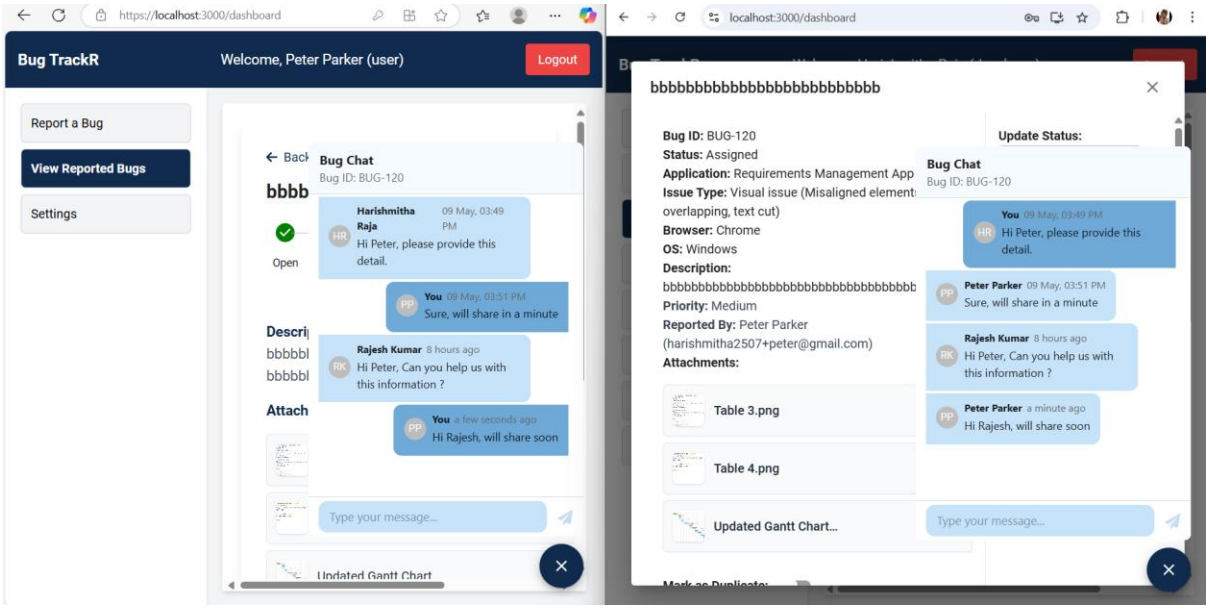


Figure 66:Chat window with real-time conversation display (between three users)

2. Real-Time Socket Communication

The backend uses Socket.IO to support real-time communication. As shown in Figure 67, the server listens for connections and joins users to bug-specific rooms. Messages sent to a room are instantly broadcast to all participants via the send_message and receive_message events.

```
//Create socket server with CORS config (only frotned is allowed)
const io = new Server(server, {
  cors: {
    origin: "https://localhost:3000",
    methods: ["GET", "POST"],
  },
});

//Listener for connection from frontend
io.on("connection", (socket) => {
  console.log("conneted");

  //Listener when a user joins the bug chat room
  socket.on("join_bug_room", (bugId) => {
    socket.join(`bug_${bugId}`);
  });

  //Listener when a user sends a message in the bug chat room
  socket.on("send_message", ({ bugId, message }) => {
    io.to(`bug_${bugId}`).emit("receive_message", message);
  });

  //Listener when a user disconnects from the bug chat room
  socket.on("disconnect", () => {
    console.log("User disconnected");
  });
});
```

Figure 67: Socket.io setup for real-time chat per bug

3. Message Storage and Session Creation

To ensure persistence, every bug chat is stored in MongoDB. Figure 68 shows the backend logic for storing chat messages. When a user sends a message, the system checks if a chat session exists for the bug. If not, one is created. Messages are saved with sender data.

```
//Send message to the chat
router.post("/:bugId/send", async (req, res) => {
  const { sender, message } = req.body;
  const { bugId } = req.params;
  if (!sender?.email || !sender?.name || !sender?.role) {
    return res.status(400).json({ error: "Sender data is incomplete" });
  }

  let session = await ChatSession.findOne({ bugId });

  //If not session found for the bug, create one
  if (!session) {
    session = await ChatSession.create({
      bugId,
      participants: [sender],
    });
  } else {
    //Add new participant to the list of participants
    const exists = session.participants.some((p) => p.email === sender.email);
    if (!exists) {
      session.participants.push(sender);
      await session.save();
    }
  }
  //Create new message
  const newMsg = await ChatMessage.create({
    chatSessionId: session._id,
    sender,
    message,
  });

  res.json(newMsg);
});
```

Figure 68: Creating chat sessions and saving messages

7.7 Time-Based Alerts and Notifications

1. UI Alerts for Stale and Unassigned Bugs
Clock-based status alerts are shown on both the admin and team lead dashboards to highlight bugs that haven't been updated recently:

- Status Alert: Team leads and admin are shown yellow/orange/red icons for bugs with no update in 6+, 12+ or 24+ hours (Figure 69).
- Unassigned Alert: Admins are shown red badges if a bug remains unassigned for over 2 hours (Figure 70).

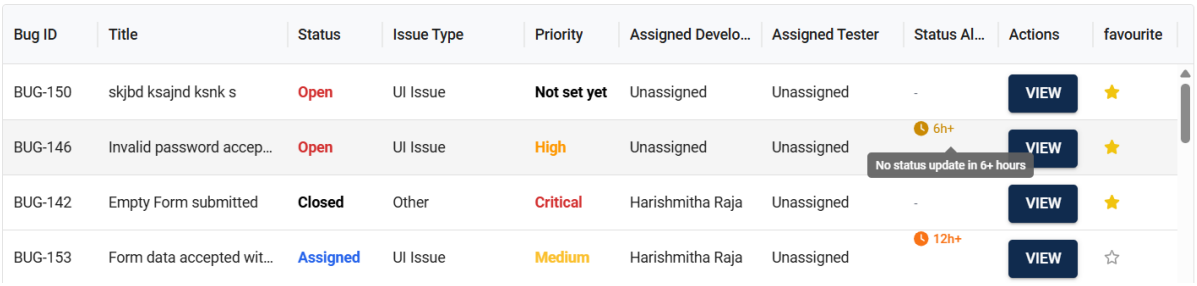


Figure 69: Clock badge for no bug status update (Team Lead View)

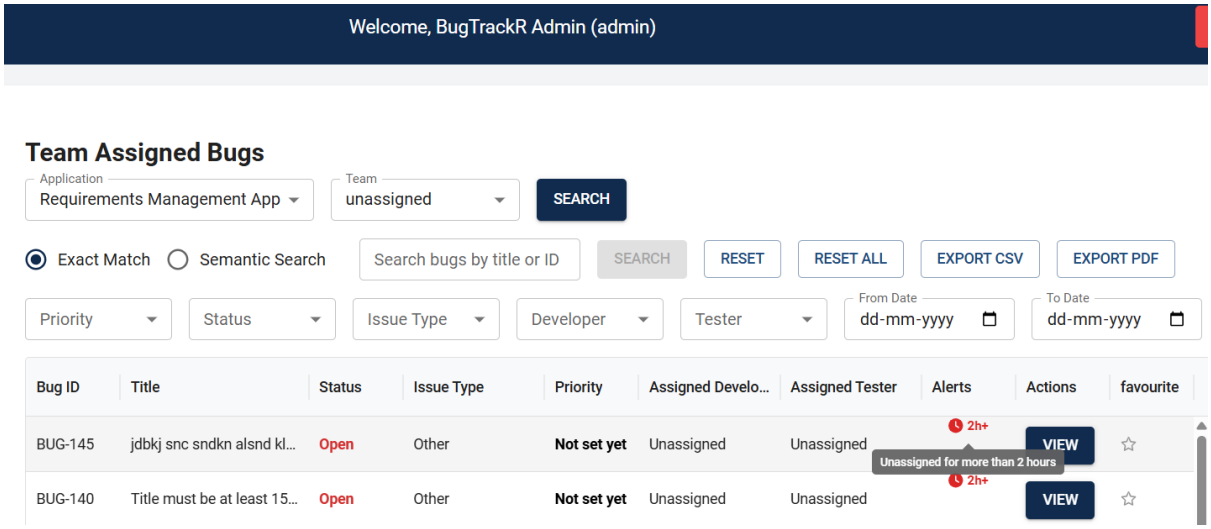


Figure 70: Alerts for unassigned bugs (Admin View)

These alerts are built using configurable time thresholds and rendered dynamically in the frontend, Figure 71.

```
{
  field: "statusAlert",
  headerName: "Status Alert",
  flex: 1,
  renderCell: (params) => {
    const bug = params.row;
    if (bug.statusLastUpdated) {
      const hours = getHoursSinceStatusUpdate(bug.statusLastUpdated);

      if (
        bug.status !== "Closed" &&
        bug.status !== "Duplicate" &&
        bug.statusLastUpdated
      ) {
        if (hours > STATUS_UPDATE_THRESHOLD.HIGH) {
          return (
            <Tooltip title="No status update in over 24 hours" arrow>
              <span className="flex items-center gap-1 text-red-600 font-semibold">
                <FontAwesomeIcon icon={faClock} />
                <span className="text-xs">24h+</span>
              </span>
            </Tooltip>
          );
        } else if (hours > STATUS_UPDATE_THRESHOLD.MEDIUM) {
          return (
            <Tooltip title="No status update in 12+ hours" arrow>
              <span className="flex items-center gap-1 text-orange-500 font-medium">
                <FontAwesomeIcon icon={faClock} />
                <span className="text-xs">12h+</span>
              </span>
            </Tooltip>
          );
        } else if (hours > STATUS_UPDATE_THRESHOLD.LOW) {
          return (
            <Tooltip title="No status update in 6+ hours" arrow>
              <span className="flex items-center gap-1 text-yellow-600 font-normal">
                <FontAwesomeIcon icon={faClock} />
                <span className="text-xs">6h+</span>
              </span>
            </Tooltip>
          );
        }
      }
```

Figure 71: Code to render time-based status alerts dynamically

2. Scheduled Email Alerts

BugTrackR includes background cron jobs to ensure bugs are not overlooked due to inactivity or missed assignments. These jobs are implemented in Node.js using the node-cron library and run automatically at fixed intervals during working hours.

The following alerts are supported:

1. Unassigned Bugs Alert

If a bug remains unassigned to a team for more than 2 hours, an alert is emailed to the admin. Figure 72 shows an example alert email sent to the admin.

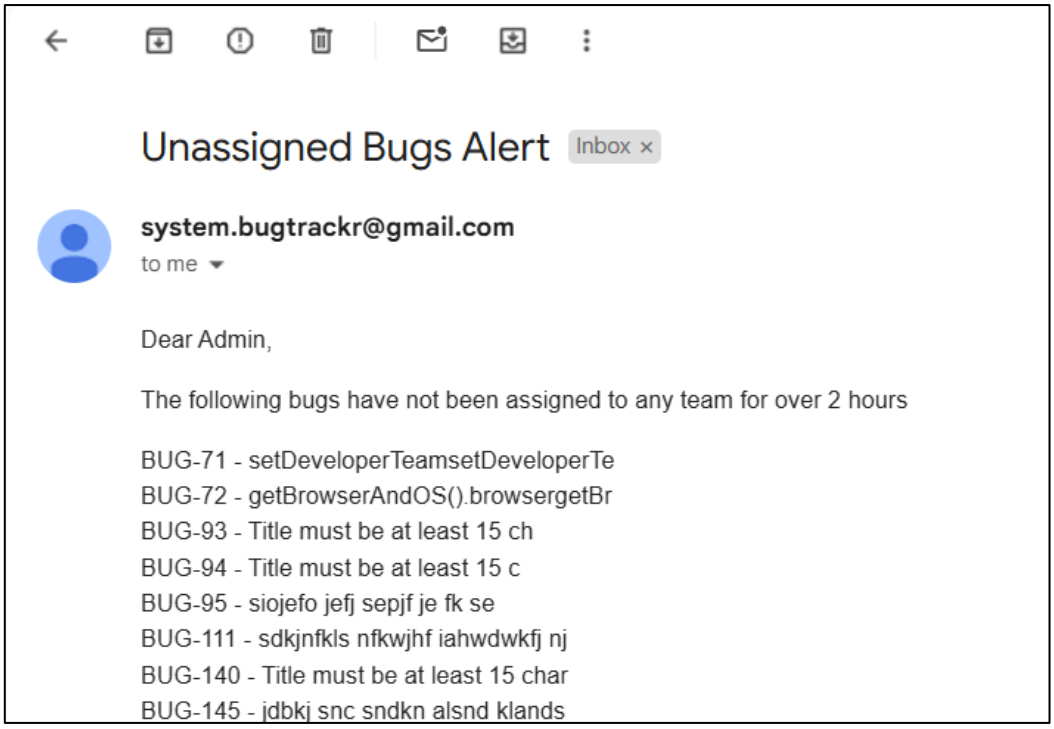


Figure 72: Unassigned bugs alert email

2. No Status Update Alert

If a bug has not had a status update in the last 24 hours and is not in a state like "Closed" or "Duplicate", an alert is sent to the assigned developer, tester, team lead, and admin.
Figure 73 displays a sample status update reminder email.

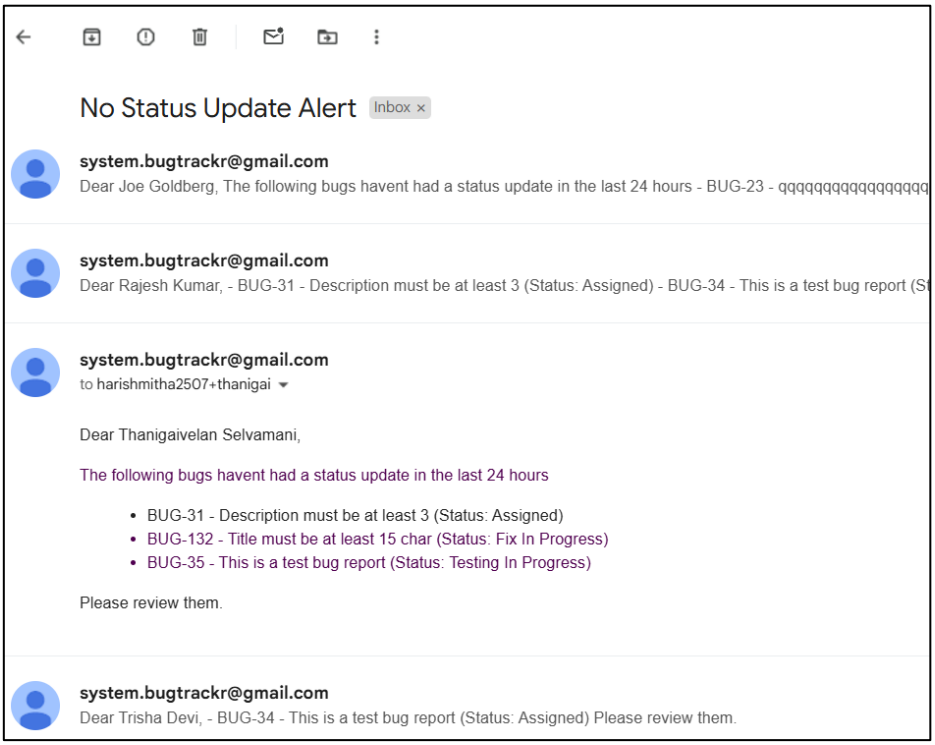


Figure 73: No status update alert email

3. Critical Bugs Waiting for Closure

When a critical bug is in the "Ready for Closure" status for more than 2 hours without any action, the system notifies the respective team lead and admin. Figure 74 illustrates this alert.

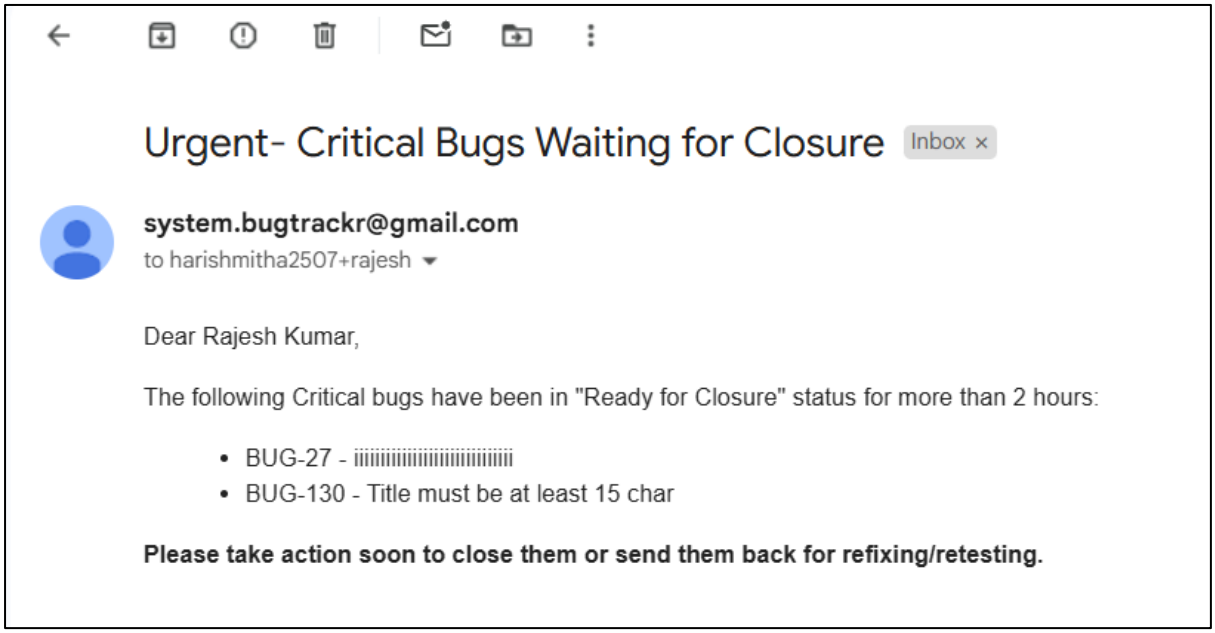


Figure 74: Critical bug closure reminder email

Figure 75 shows the backend logic for the No Status Update Alert, which serves as a representative example.

```
//Bugs not updated for 24 hours will be alerted
const ALERT_AFTER_HOURS = 24;

const runNoStatusAlertJob = () => {
  //Runs every day at 10am
  cron.schedule("0 10 * * 1-5", async () => {
    try {
      //Get the time that was 24 hours ago
      const deadlineTime = new Date(
        Date.now() - ALERT_AFTER_HOURS * 60 * 60 * 1000
      );

      //Find bugs that are not updated recently and not already closed or marked as duplicate
      const oldBugs = await Bug.find({
        statusLastUpdated: { $lt: deadlineTime },
        status: { $nin: ["Closed", "Duplicate"] },
      });
```

Figure 75: Cron job for status alert logic

7.8 Advanced Search & Filtering

1. UI Controls and Filter Inputs

Users can switch between Exact Match and Semantic Search modes and apply filters for priority, status, issue type, developer, tester and a date range, Figure 76. Date fields validate min/max limits Figure 77 (UI) and Figure 78 (Code containing limit).

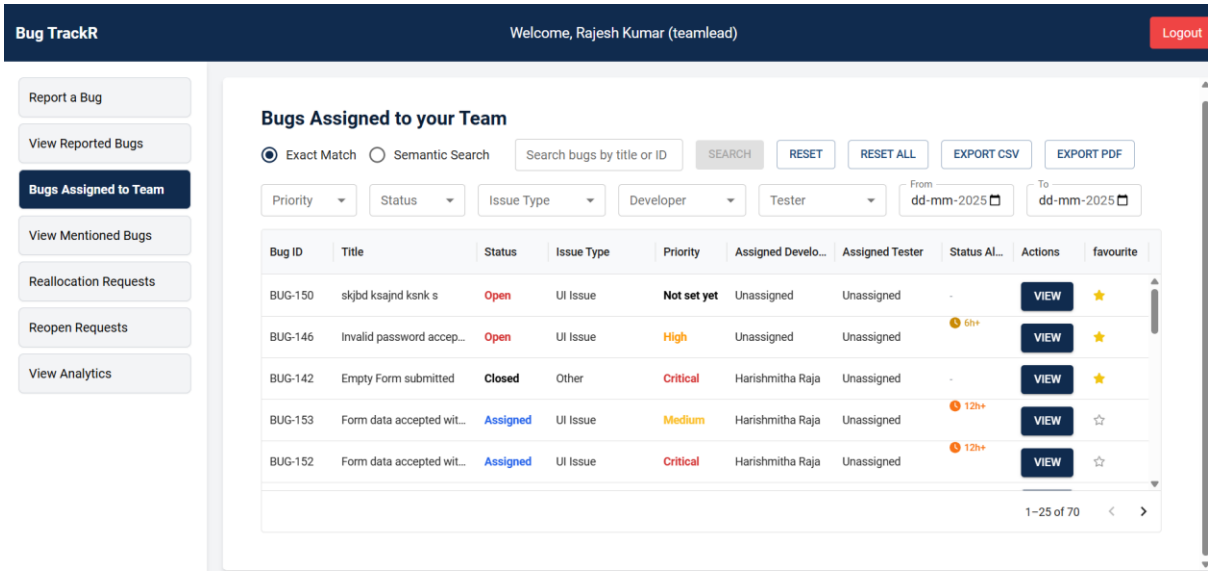


Figure 76:Filter options with exact/semantic and dropdown for other fields

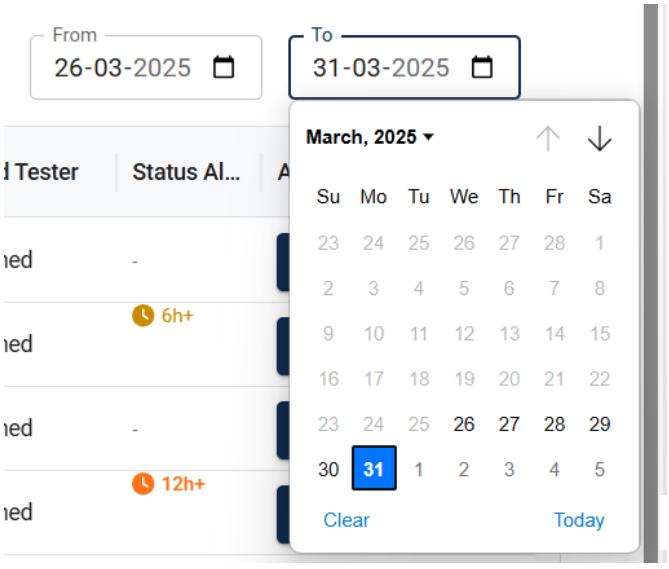


Figure 77:Date range picker

```
<TextField
  size="small"
  type="date"
  label="From"
  InputLabelProps={{ shrink: true }}
  value={startDate}
  onChange={(e) => setStartDate(e.target.value)}
  inputProps={{
    min: "2025-02-17",
    max: endDate || dayjs().format("YYYY-MM-DD"),
  }}
/>
<TextField
  size="small"
  type="date"
  label="To"
  InputLabelProps={{ shrink: true }}
  value={endDate}
  onChange={(e) => setEndDate(e.target.value)}
  inputProps={{
    min: startDate || "2025-02-17",
    max: dayjs().format("YYYY-MM-DD"),
  }}
/>
```

Figure 78: Date limit

2. Exact Search (Node.js Backend)

When Exact Match is selected, the query is split into words and an $or condition is built to match any word across bugId, title, and description (Figure 79) (See Section 6.10 for more details). Combined filters and date ranges are also possible, Figure 80.

bugReportRoutes.js > router.get("/assigned/team/:application/:team") callback

```
async (req, res) => {

  const filter = {
    application,
    assignedTeam: team,
  };

  if (priority) filter.priority = priority;
  if (status) filter.status = status;

  if (developer === "Unassigned") {
    filter["assignedTo.developer"] = null;
  } else if (developer) {
    filter["assignedTo.developer"] = developer;
  }

  if (tester === "Unassigned") {
    filter["assignedTo.tester"] = null;
  } else if (tester) {
    filter["assignedTo.tester"] = tester;
  }

  if (startDate || endDate) {
    filter.createdAt = {};
    if (startDate) filter.createdAt.$gte = new Date(startDate);
    if (endDate) {
      const toDate = new Date(endDate);
      toDate.setDate(toDate.getDate() + 1); //includes endDate day
      filter.createdAt.$lte = toDate;
    }
  }

  //Single/combined or because if two ors are used then one will replace the other according to MongoDB rule
  const or = [];

  if (issueType) {
    const mappedKeywords = issueTypeKeywordMap[issueType] || [];
    or.push(
      { issueType: issueType },
      { issueType: { $in: mappedKeywords } },
      { otherIssueTypeDescription: { $in: mappedKeywords } }
    );
  }

  if (searchQuery && searchMode === "exact") {
    const words = searchQuery.toLowerCase().split(" ");
    words.forEach((word) => {
      or.push(
        { bugId: { $regex: word, $options: "i" } },
        { title: { $regex: word, $options: "i" } },
        { description: { $regex: word, $options: "i" } }
      );
    });
  }
```

Figure 79: MongoDB query logic with filters and search condition in bugReportRoutes.js

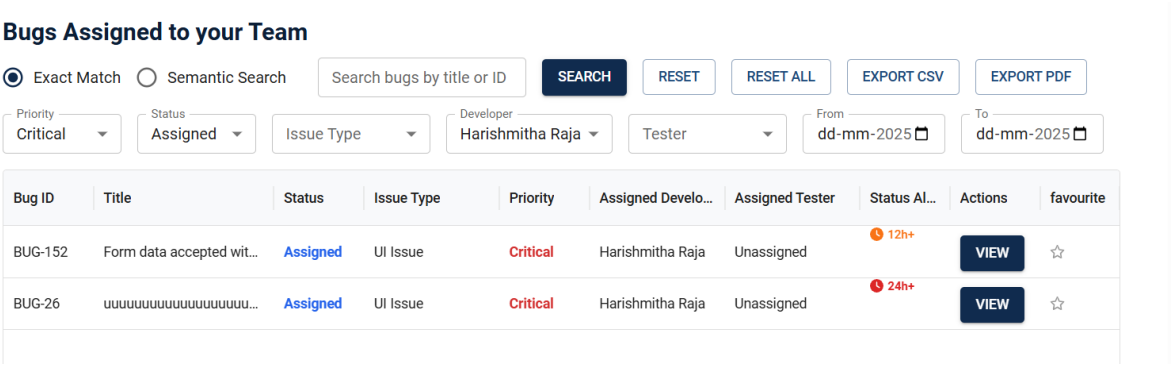


Figure 80: Combination Filter Result

3. Semantic Search

The backend sends the query text to the microservice API. This uses the same embedding and search technique described earlier in the Design section (Section 6.5). It returns the top similar bug IDs if their cosine similarity score is >= 0.4, where 1.0 means exact match. This threshold balances strictness and flexibility, low enough to catch meaningful similarities, but high enough to avoid irrelevant matches, Figure 81 (UI) and Figure 82 (code snippet).

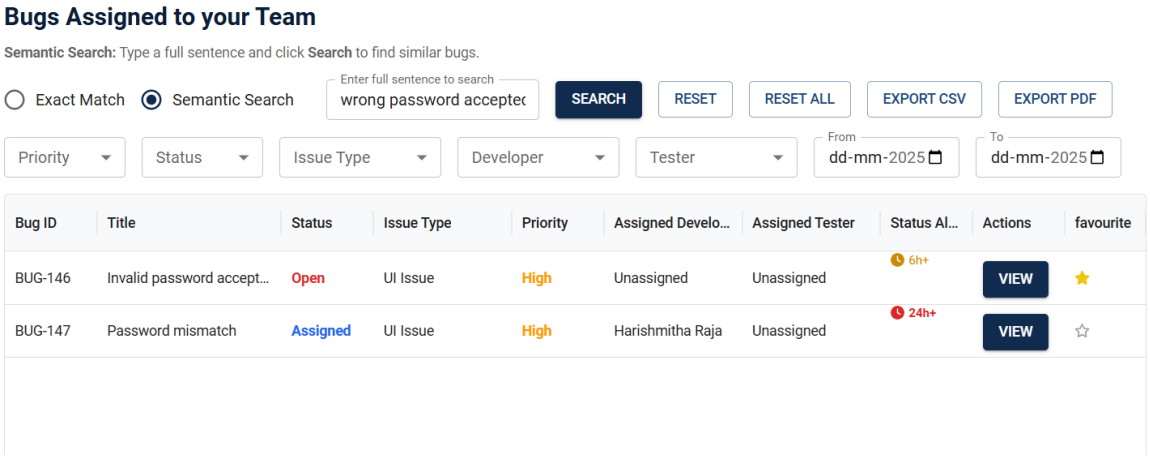


Figure 81: Semantic search results

```python
#Endpoint to seach the bug database semantically(based on meaning rather than exact words)
@app.post("/search/semantic")
def semantic_search(query_data: SearchQuery):
    if not query_data.application or not query_data.query:
        return {"error": "Application and query fields are required"}

    query_text = query_data.query.strip()

    #Perform semantic similarity search
    results = embeddings.search_similar(query_data.application, query_text, min_score=0.4)

    similar_bugs = [
        {"bug_id": bug_id, "score": score}
        for bug_id, score in results
    ]

    return {"similar_bugs": similar_bugs}
```

Figure 82: Python endpoint for semantic bug matching

7.9 Pagination, Indexing and Batch Loading

1. Server-Side Pagination

MongoDB queries use .skip() (already loaded bugs list) and .limit(), along with .sort() by createdAt latest first to return only the bugs needed for the current page, Figure 83. The frontend DataGrid component is configured to work in "server" pagination mode and fetches and shows data page by page, Figure 84.

```javascript
  if (or.length > 0) {
    filter.$or = or;
  }

  console.log(filter);
  //skips already fetched results and using parseInt coz the query will be passed as string
  const skip = (parseInt(page) - 1) * parseInt(limit);

  //Get bugs and total docs to show in the UI like (Eg: of 120)
  const [bugs, total] = await Promise.all([
    BugReport.find(filter)
      .sort({ createdAt: -1 })
      .skip(skip)
      .limit(parseInt(limit)),
    BugReport.countDocuments(filter),
  ]);

  res.status(200).json({
    bugs,
    total,
    page: parseInt(page),
  });
```

Figure 83: Backend pagination logic

```
<DataGrid
  style={{ height: 380 }}
  rows={sortedRows}
  columns={columns}
  getRowId={(row) => row.bugId}
  paginationMode="server"
  paginationModel={{ page: page - 1, pageSize: limit }}
  onPaginationModelChange={({ page }) => {
    setPage(page + 1);
    setApplyFilters(true);
  }}
  rowCount={total}
  pageSizeOptions={[limit]}
  disableRowSelectionOnClick
/>
```

Figure 84: Frontend DataGrid config

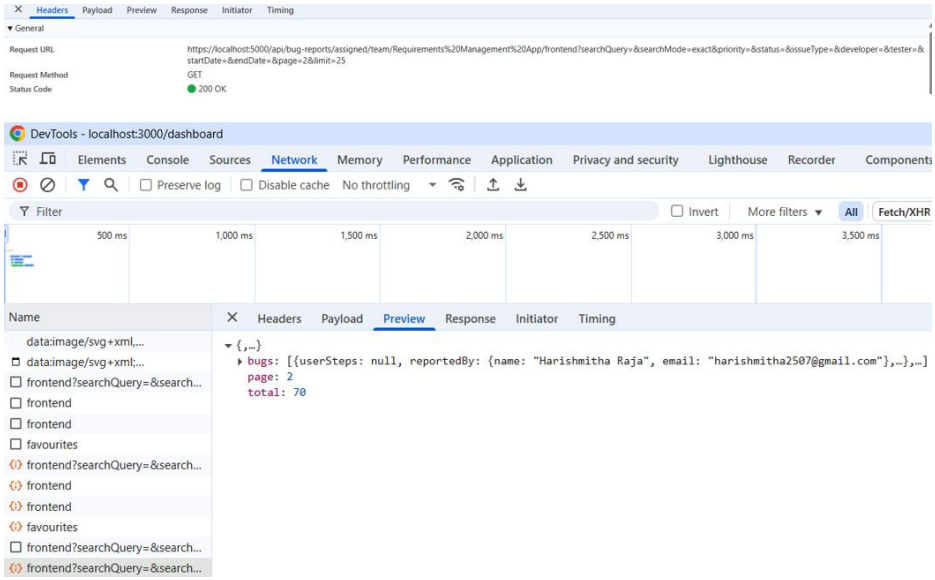


Figure 85: Paginated API response

2. Backend Indexing for Fast Queries

MongoDB indexes are created on frequently filtered fields like status, priority, assignedTo.developer, etc., Compound indexes are also used for combinations like { priority, status } or { application, assignedTeam } to optimize multi-field filtering , Figure 86.

```
BugReportSchema.index({ title: "text", description: "text" });
BugReportSchema.index({ "assignedTo.developer": 1 });
BugReportSchema.index({ "assignedTo.tester": 1 });
BugReportSchema.index({ application: 1 });
BugReportSchema.index({ createdAt: -1 });
BugReportSchema.index({ status: 1 });
BugReportSchema.index({ assignedTeam: 1 });

//Combinations can be added based on most used combos for filtering
BugReportSchema.index({
  application: 1,
  assignedTeam: 1,
});
BugReportSchema.index({
  priority: 1,
  status: 1,
});

BugReportSchema.index({
  priority: 1,
  "assignedTo.developer": 1,
  status: 1,
  "assignedTo.tester": 1,
});
```

Figure 86: MongoDB index definitions

3. Index Check with MongoDB Shell

The .explain("executionStats") method is used to verify the effectiveness of indexes. Queries with indexes return quickly and examine fewer documents, Figures 87 and 88. In contrast, unindexed queries take longer and scan more records Figures 89 and 90.

```
>_MONGOSH
> use bug_tracking
< switched to db bug_tracking
> db.bugreports.find(
    { status: "Open" }
  ).hint( { status: 1 } ).explain("executionStats")
< {
```

Figure 87: Indexed status query

```
>_MONGOSH
    executionSuccess: true,
    nReturned: 68,
    executionTimeMillis: 3,
    totalKeysExamined: 68,
    totalDocsExamined: 68,
    executionStages: {
```

Figure 88: Status query stats

```
> db.bugreports.find(
    { "reportedBy.email": "harishmitha2507@gmail.com" }
  ).explain("executionStats")
< {
```

Figure 89:Unindexed Query

```
executionStats: {
  executionSuccess: true,
  nReturned: 65,
  executionTimeMillis: 2,
  totalKeysExamined: 0,
  totalDocsExamined: 147,
```

Figure 90:Email query stat

7.10 API List and Database Overview

1. Backend API Coverage

The system is built with a modular Node.js Express backend that exposes RESTful endpoints, Figures 91 and 92. These APIs support all core features such as bug reporting, assignment, and priority setting, auto-assignment and duplicate detection, commenting, mentions, and bug chat, advanced search (semantic+exact), role-based views and approval workflows (reopen, reallocation).

Each endpoint is secured with role-based access. For example, only admins and team leads can assign bugs, while general users can only view or report them.

API Routes/Endpoints	Method	Purpose	Role Access
Authentication			
/auth/login	POST	User login	All Roles
/auth/send-otp	POST	Send OTP to reset password	All Roles
/auth/verify-otp	POST	Verify OTP for password reset	All Roles
/auth/reset-password	POST	Reset user password	All Roles
Bug Reports			
/bug-reports/submit	POST	Submit a new bug report	All Roles
/bug-reports/reported	GET	Fetch bugs reported by logged-in user	General User
/bug-reports/reported/:bugId	GET	Fetch specific reported bug by ID	General User
/bug-reports/reported/:bugId/add-info	POST	Add more information to a bug report	General User, Developer, Tester, Admin
/bug-reports/assigned/team/:application/:team	GET	Fetch bugs assigned to a team	Team Lead, Admin
/bug-reports/assigned/team/:bugId	GET	Fetch details of bug assigned to a team	Team Lead, Admin
/bug-reports/assigned/developer/:developerEmail	GET	Fetch bugs assigned to a developer	Developer, Team Lead, Admin
/bug-reports/assigned/tester/:testerEmail	GET	Fetch bugs assigned to a tester	Tester, Team Lead, Admin
/bug-reports/assign-team	PUT	Assign bug to a team	Admin
/bug-reports/assign-developer	PUT	Assign bug to a developer	Team Lead, Admin
/bug-reports/assign-tester	PUT	Assign bug to a tester	Team Lead, Admin
/bug-reports/set-priority	PUT	Set priority for a bug	Team Lead, Admin
/bug-reports/update-status	PUT	Update the status of a bug	Developer, Tester, Team Lead, Admin
/bug-reports/mark-duplicate	PUT	Mark bug as duplicate	Developer, Tester, Team Lead, Admin
/bug-reports/undo-duplicate	PUT	Undo duplicate mark	Developer, Tester, Team Lead, Admin
/bug-reports/request-reallocation	PUT	Request bug reallocation	Developer, Tester
/bug-reports/reallocation-requests	GET	Fetch all reallocation requests	Team Lead, Admin
/bug-reports/approve-reject-reallocation	PUT	Approve or reject reallocation request	Team Lead, Admin
/bug-reports/reopen-bug	PUT	Request to reopen a closed bug	Developer, Tester, Team Lead, Admin
/bug-reports/approve-reject-reopen	PUT	Approve or reject reopen request	Team Lead, Admin

Figure 91: API list with route, method, purpose, and role access (1)

API Routes/Endpoints	Method	Purpose	Role Access
/bug-reports/reopen-requests	GET	Fetch all reopen requests	Team Lead, Admin
/bug-reports/assigned/developer/:developerEmail/fixed-bugs	GET	Fetch bugs fixed by developer	Developer, Team Lead, Admin
/bug-reports/assigned/tester/:testerEmail/tested-bugs	GET	Fetch bugs tested by tester	Tester, Team Lead, Admin
/bug-reports/for-embedding	GET	Python microservice uses for embedding	System/internal
/bug-reports/check-duplicate	POST	Check if bug is a duplicate	All Roles
/bug-reports/classify-priority	POST	Classify priority of bug based on similar bugs	All Roles
/bug-reports/search/semantic	POST	Semantic search of bugs	All Roles
/bug-reports/auto-assign-developer	PUT	Auto-assign bug to developer	Admin, Team Lead
/bug-reports/auto-assign-tester	PUT	Auto-assign bug to tester	Admin, Team Lead
Applications			
/applications/	GET	Fetch all applications	All Roles
/applications/:appName/details	GET	Fetch details of a specific application	All Roles
/applications/:appName/mentionable-users	GET	Fetch mentionable users in comment for an app	All Roles
Developers			
/developers/:appName/team	GET	Fetch developers of a team in an app	Team Lead, Admin
/developers/check-availability	POST	Check if developer is available	Team Lead, Admin
Testers			
/testers/:appName/team	GET	Fetch testsers of a team in an app	Team Lead, Admin
Issue Types			
/issue-types/	GET	Fetch issue types based on user role	All Roles
Chat			
/chat/:bugId/messages	GET	Fetch all chat messages for a bug	Developer, Tester, General User
/chat/:bugId/send	POST	Send a message in bug chat	Developer, Tester, General User
Comments			
/comments/add	POST	Add a comment to a bug report	Developer, Tester
/comments/:bugId/details	GET	Fetch comments and bug details	All Roles
/comments/edit/:commentId	PUT	Edit a comment within 15 minutes	Developer, Tester
/comments/delete/:commentId	DELETE	Archive a comment	Developer, Tester
/comments/mentioned	GET	Fetch bugs where user was mentioned in comments	All Roles
/comments/mentioned/:bugId	GET	Fetch comment details for mentioned bug	All Roles
Favourites			
/favourites/add	POST	Add a bug to favourites	All Roles
/favourites/remove	POST	Remove a bug from favourites	All Roles
/favourites	GET	Get all favourite bugs	All Roles

Figure 92: API list with route, method, purpose, and role access (2)

In addition, a Python FastAPI microservice supports duplicate detection, priority classification, and semantic search, Figure 93.

API Endpoint	Method	Purpose	Role Access
/check-duplicate	POST	Check if a bug is a duplicate using semantic similarity	Used by all roles
/search/semantic	POST	Perform semantic bug search using embedded query	Used by all roles
/similar-bugs-for-classify-priority	POST	Find similar bugs to classify priority level	Used by all roles
/add-embedding	POST	Add new bug embedding to the FAISS index	Internal/System

Figure 93: API list in Python Microservice

2. Key API Groups

- /auth/* - User login and password reset
- /bug-reports/* - Main bug handling and advanced logic
- /developers/*, /testers/*, /applications/* - Team and app metadata
- /comments/* and /chat/* - Collaboration and Real-time communication
- /favourites/* - Favourite features

3. Database Collections

The backend uses MongoDB with the following main collections: (Figure 94)

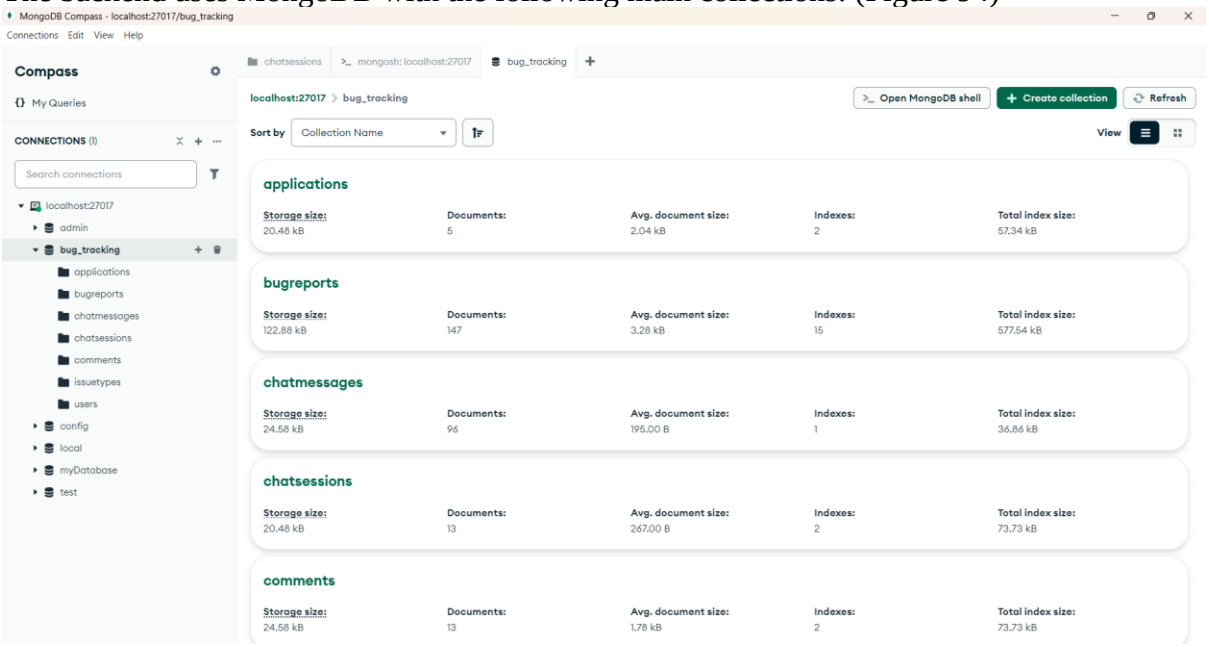


Figure 94: MongoDB Collections

- BugReport - Stores all bug data including assignment, status, timestamps (Figures 95 and 96)
- User - All user accounts, roles, workload hours (Figure 97)
- Comment - Comments added to bug reports (Figure 98)
- ChatMessage and ChatSession - Stores chat messages per bug
- Application, IssueType, Favourite

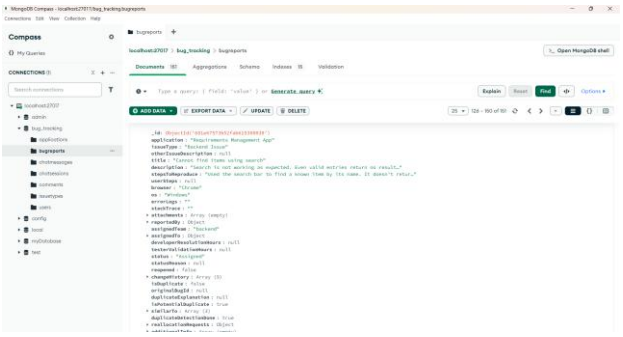


Figure 95: BugReport document structure

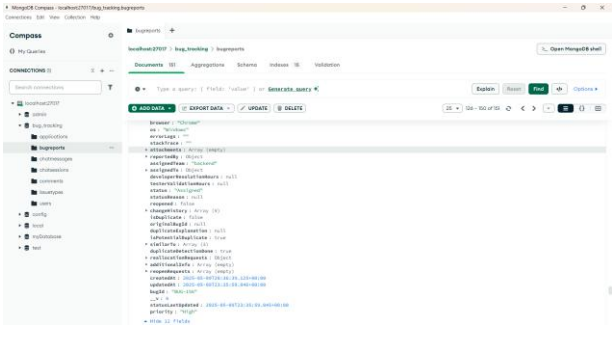


Figure 96: BugReport document structure continuation

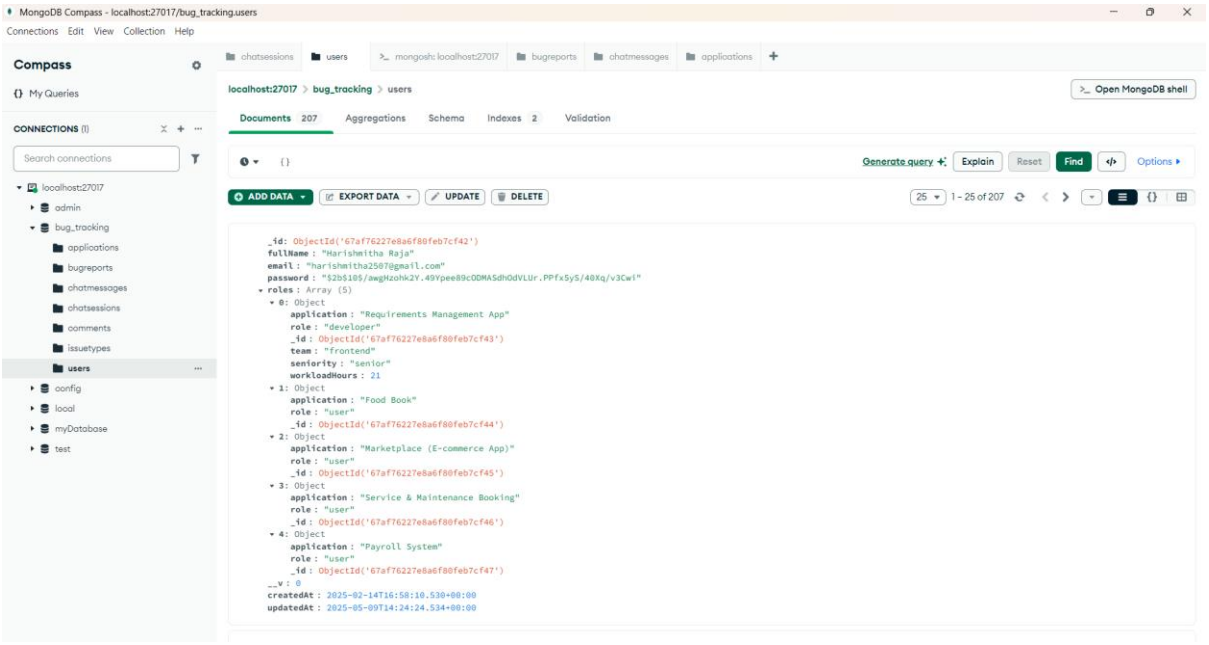


Figure 97: User document structure

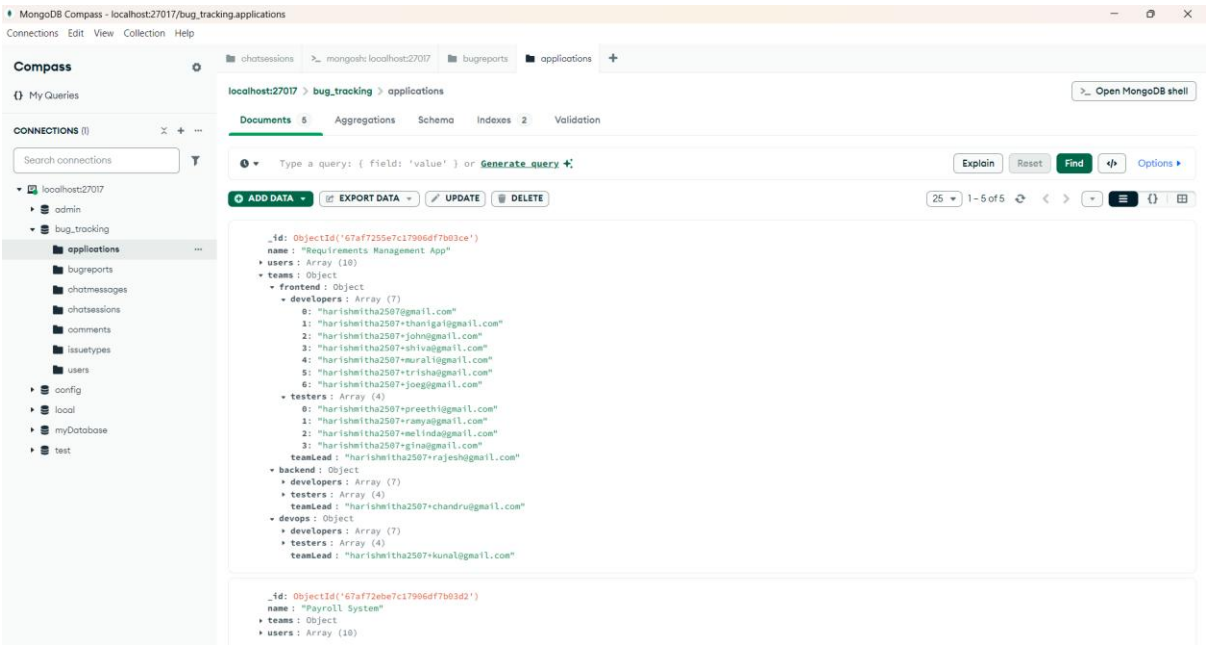


Figure 98: Application document structure

7.11 Running the Application

To run the BugTrackR system locally, the project repository should be cloned from the university's GitLab using either SSH or HTTPS, Figure 99. The system consists of three components: the frontend, backend, and duplicate detection (python microservice).

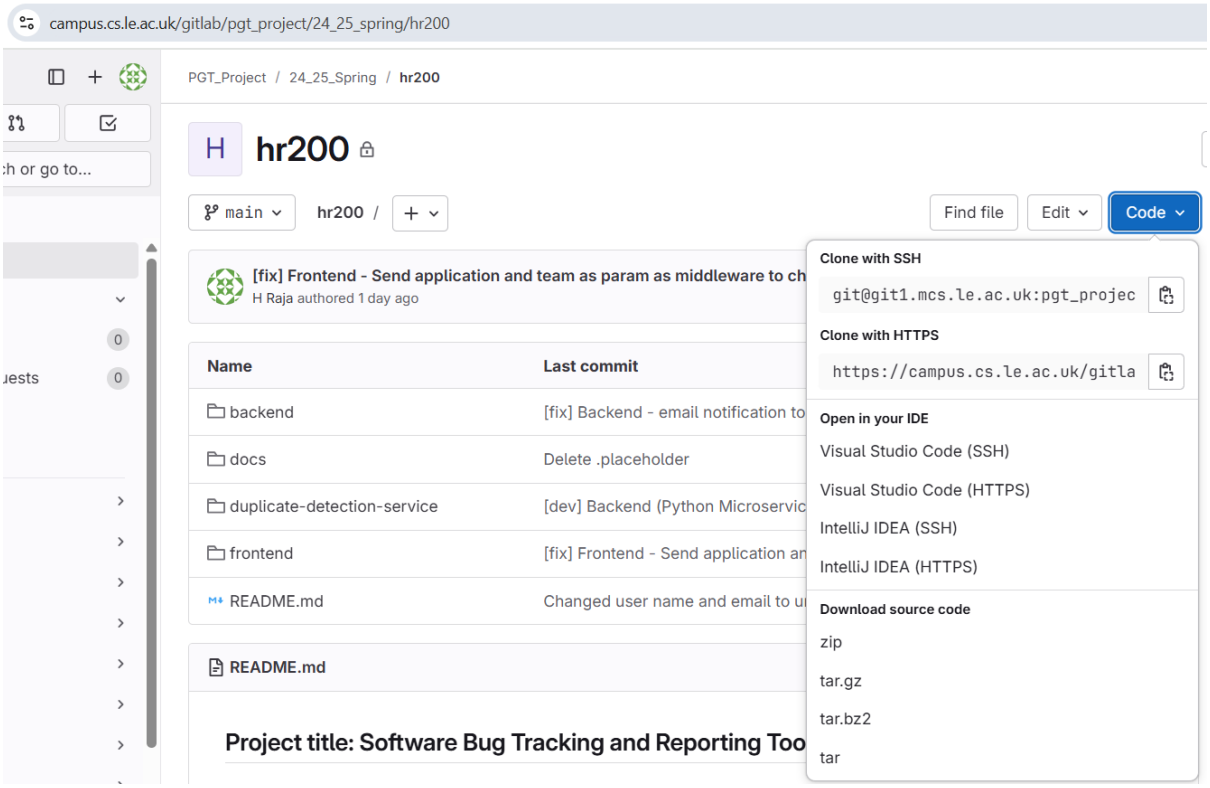


Figure 99: GitLab

1. Clone the Repository

- `git clone https://campus.cs.le.ac.uk/gitlab/pgt_project/24_25_spring/hr200.git`

2. Frontend

- Navigate to the frontend directory via command prompt or in Visual Studio Code [33] by clicking File -> Open Folder and open terminal and install dependencies:
  `npm install`
- If any version conflicts arise, use:
  `npm install --legacy-peer-deps`
- Enabling HTTPS in Frontend
  To run the app securely on https://localhost:3000, a self-signed SSL certificate is used. The certificate and key are generated using the OpenSSL command below, and stored in a certs directory at the root of the project.

```
openssl req -x509 -newkey rsa:2048 -nodes -keyout localhost-key.pem -out localhost.pem -days 365
```

These files are then used in the package.json start script. The **.pem** and **key.pem** files can be named anything as needed, in my applicationI have used localhost+1 for .pem and localhost+1-key for key.pem. (Figure 100)

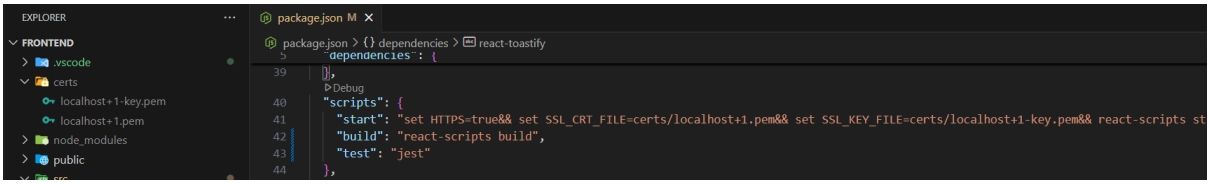


Figure 100: Certs folder and script in package.json in frontend

- Then start the development server:
  `npm start`

- This allows the development server to use HTTPS with live reload (react-scripts has live reload by default) .

3. Backend

- Navigate to the backend folder and install dependencies:
  `npm install`
- If any version conflicts arise, use:
  `npm install --legacy-peer-deps`
- Enabling HTTPS in Backend

The same SSL certificate (`localhost+1.pem`) and key (`localhost+1-key.pem`) can be reused by the backend to run securely at https://localhost:5000. These are read using Node's `fs` module from the certs folder containing the certificate and passed to the HTTPS server (Figure 101).

```
const options = {
  key: fs.readFileSync("certs/localhost+1-key.pem"),
  cert: fs.readFileSync("certs/localhost+1.pem"),
};
```

Figure 101: Certs read by fs module in backend server.js

- Then start the server using:
  `npm run dev`
  Backend runs on https://localhost:5000

- This uses nodemon, which automatically restarts the server on file changes, improving development productivity by avoiding the need for manual restarts after code updates.

4. Python Microservice (FastAPI)

- Navigate to the root and set up a virtual environment:
  ```
  python -m venv venv
  venv\Scripts\activate
  ```
- Install required packages:
  ```
  pip install fastapi uvicorn sentence-transformers faiss-cpu numpy requests python-dotenv
  ```
- Run the microservice
  ```
  uvicorn app.main:app --reload --port 8000
  ```

- Once running, the following URL can be used for testing http://127.0.0.1:8000/docs

5. MongoDB Setup

- Ensure that MongoDB [34] is installed and running locally. The backend connects to MongoDB using the URI stored in the .env file

(e.g., MONGO_URI=mongodb://localhost:27017/bug_tracking).

- Collections such as bugreports, chatmessages and comments are automatically created by Mongoose based on usage.

- Use MongoDB Compass [35] for visual inspection and query testing during development (see Section 7.10 for schema views).

Environment Configurations

The application uses a .env file to manage sensitive settings:

- MONGO_URI: MongoDB connection string.
- JWT_SECRET: A randomly generated string used to sign and verify authentication tokens.
- EMAIL_USER and EMAIL_PASS: App-specific Gmail credentials for sending notification emails.
- SIMILARITY_API_KEY: A key used by the nodejs backend to validate incoming requests from the microservice before generating embeddings.
- ADMIN_TOKEN: A pre-generated admin access token used for backend authentication during auto assign developer and tester scheduled jobs.

8. Testing

8.1 Requirements Evaluation Plan

Unit and integration testing was done for the technically complex and critical components. These included the duplicate detection microservice, semantic search, auto-assignment logic, and priority classification. For the remaining areas, manual functional and workflow-based testing was carried out to ensure the system behaved as expected across user roles and key features. This approach ensured that core logic, role-based flows, and advanced features were tested, while maintaining broad coverage across the system.

8.2 Unit and Integration Testing

I wrote unit tests for the microservice as mentioned above Section 8.1. These tests helped validate the core logic, input handling and expected outputs using various real and edge-case scenarios. I also performed integration testing on these services to check the end-to-end flow through their APIs.

Run the test by navigating to the microservice root folder and running commands below:

```
pytest tests/test_main.py
pytest --cov=app tests/
pytest -s --log-cli-level=INFO tests/test_main.py
```

Additionally, I tested the auto-assignment logic in the backend. This included checking whether bugs were assigned correctly based on different criteria.
Run the test by navigating to the backend root folder and running commands below:

```
npm test
npm run test:coverage
```

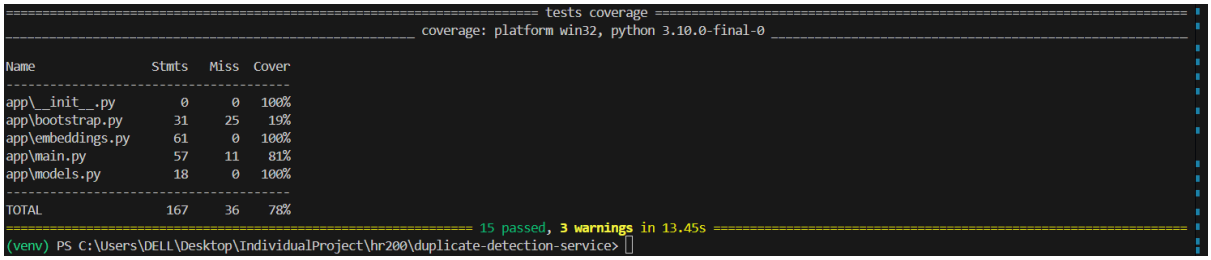


Figure 102: Unit and Integration test coverage of microservice

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Line #s
All files	97.77	88.63	90	97.61	
models	100	100	100	100	
UserSchema.js	100	100	100	100	
utils	97.7	88.63	90	97.53	
assignDeveloper.js	97.67	95	90	97.5	114
assignTester.js	97.72	83.33	90	97.56	119

Figure 103: Unit test coverage for auto assignment

8.3 Functional and Workflow Testing

Most of the system was tested manually through functional and workflow testing, focusing on use cases such as bug reporting, advanced search and filtering, duplicate check etc., Testcases (Figures 104 and 105), listing each feature tested, the description, role and result. A few minor edge cases were not tested in depth due to time constraints, but these do not affect core functionality and can be addressed with some investigation. Screenshots were taken during test execution. See Appendices. This helped me ensure that most features were working as expected and that workflows were logically consistent.

TC No	Module	Role	Testcase Description	Test Status
1	Report a Bug	All Roles	A user can log in with valid credentials	Pass
2	Report a Bug	All Roles	A user cannot log in with incorrect credentials	Pass
3	Report a Bug	All Roles	A user can reset password using OTP verification sent to email	Pass
4	Report a Bug	All Roles	A user can resend OTP after 30 seconds	Pass
5	Report a Bug	All Roles	A user canot reuse the old OTP sent before resending a new one	Pass
6	Report a Bug	All Roles	A user can report a bug with title, description, application, and issue type	Pass
7	Report a Bug	General User	A user can follow guided steps to describe how to reproduce the issue	Pass
8	Report a Bug	Developer, Tester, Team Lead, Admin	A technical user can provide steps to reproduce without any guidance	Pass
9	Report a Bug	All Roles	A user can select or auto-detect their browser and OS	Pass
10	Report a Bug	General User	A user can attach (file select and drag and drop) up to 5 images or videos when reporting a bug	Pass
11	Report a Bug	Developer, Tester, Team Lead, Admin	A technical user attach up to 5 images, videos, txt, logs, xlsx, word and pdf files when reporting a bug	Pass
12	Report a Bug	All Roles	The system can validate and restrict file uploads to 5 files	Pass
13	Report a Bug	Developer, Tester, Team Lead, Admin	A technical user can add error logs and stack trace	Pass
14	Report a Bug	All Roles	The system can alert the user if required fields are missing during bug submission	Pass
15	Report a Bug	All Roles	The system can alert the user if required character count is missed in title, description, steps to reproduce, error logs and stack trace	Pass
16	Report a Bug	All Roles	The system can show suggestions for possible duplicate bugs	Pass
17	Report a Bug	All Roles	A user can proceed with submission even if a similar bug is suggested or cancwl	Pass
18	View Reported Bugs	All Roles	A user can view the list of bugs they have reported	Pass
19	View Reported Bugs	All Roles	A user can view detailed information of a reported bug	Pass
20	View Reported Bugs	All Roles	A user can track the bug status using a visual stepper	Pass
21	View Reported Bugs	All Roles	A user can add additional information after submitting a bug	Pass
22	View Reported Bugs	All Roles	The system can send an email confirmation after a bug is submitted	Pass
23	View Reported Bugs	All Roles	The system can notify the user when bug status is updated	Pass
24	View Reported Bugs	All Roles	The system can notify the user when a technical comment is added to their bug	Pass
25	View Reported Bugs	All Roles	A user can chat with technical users and reporters through the bug detail page	Pass
26	View Reported Bugs	All Roles	A user can view only their own reported bugs in a table	Pass
27	View Reported Bugs	All Roles	A user cannot edit the bug report after submission	Pass
28	View Reported Bugs	All Roles	A user can export their bug reports to PDF format	Pass
29	View Reported Bugs	All Roles	A user can export their bug reports to CSV format	Pass
30	View Reported Bugs	All Roles	A user can filter reported bugs by priority, status and date range	Pass
31	View Reported Bugs	All Roles	A user can sort reported bugs in ascending and descending order	Pass
32	View Reported Bugs	All Roles	A user can switch between exact and semantic search modes in reported bugs page	Pass
33	View Reported Bugs	All Roles	A user can reset all filters added in reported bugs page	Pass
34	View Reported Bugs	All Roles	A user can click on the View button to view the details of the reported bug	Pass
35	View Reported Bugs	All Roles	A user cannot see internal comments from the technical team	Pass
36	View Reported Bugs	All Roles	A user can access the chat to talk to tech team members	Pass
37	View Assigned Bugs	Developer, Tester	A developer, tester can view all bugs assigned to them	Pass
38	View Assigned Bugs	Developer, Tester	A developer, tester receives an email when a bug is assigned to them	Pass
39	View Assigned Bugs	Developer, Tester	A developer, tester can view the full details of an assigned bug	Pass
40	View Assigned Bugs	Developer, Tester, Team Lead, Admin	A tech user can add, edit, and delete a comment within 15 minutes within their access scope	Pass
41	View Assigned Bugs	Developer, Tester, Team Lead, Admin	A tech user can mention other users in a comment using @mention	Pass
42	View Assigned Bugs	Developer	The system can send an email to mentioned users when tagged in a comment	Pass
43	View Assigned Bugs	Developer	A developer receives email when a bug is assigned to them	Pass
44	View Assigned Bugs	Developer	A developer can change the bug status using dropdown in the details page	Pass
45	View Assigned Bugs	Developer, Tester	The system can show flags for bugs with no status update after threshold	Pass
46	Update Status	Developer	A developer can change bug status to 'Fix In Progress'	Pass
47	Update Status	Developer	A developer can update bug status to 'Fixed (Testing Pending)'	Pass
48	Update Status	Developer	A developer can revert status update within 15mins of change	Pass
49	Update Status	Developer	A developer can update the status of an assigned bug	Pass
50	Update Status	Developer	A developer can change the bug status using a drag-and-drop interface	Pass
51	Update Status	Developer, Tester	A developer, tester can only update bugs that are not closed or marked duplicate	Pass
52	View Mentioned Bugs	Developer, Tester, Team Lead, Admin	A tech user can view bugs where they were mentioned	Pass
53	View Mentioned Bugs	Developer, Tester, Team Lead, Admin	A tech user can search and filter the mentioned bugs by status or priority	Pass
54	View Mentioned Bugs	Developer, Tester, Team Lead, Admin	A tech user can export mentioned bug reports to PDF or CSV	Pass
55	View Mentioned Bugs	Developer, Tester, Team Lead, Admin	A user can filter reported bugs by priority, status and date range	Pass
56	View Mentioned Bugs	Developer, Tester, Team Lead, Admin	A tech user can sort reported bugs in ascending and descending order	Pass
57	View Mentioned Bugs	Developer, Tester, Team Lead, Admin	A tech user can switch between exact and semantic search modes in mentioned bugs page	Pass
58	View Mentioned Bugs	Developer, Tester, Team Lead, Admin	A tech user can reset all filters added in mentioned bugs page	Pass
59	View Developer Analyti	Developer	A developer can access personalized analytics dashboard	Pass
60	View Developer Analyti	Developer	A developer can view workload summary, fix effieicency over time, average fix time by priority, SLA breaches, bug status overview	Pass
61	View Developer Analyti	Developer	A developer can use relevant filters in each metric/insight	Pass
62	View Developer Analyti	Developer	A developer can export metrics/insights as CSV	Pass
63	View Fixed Bugs	Developer	A developer can view all fixed bugs	Pass
64	View Fixed Bugs	Developer	A developer can export fixed bug reports to PDF or CSV	Pass
65	View Fixed Bugs	Developer	A developer can filter fixed bugs by priority, status and date range	Pass
66	View Fixed Bugs	Developer	A developer can sort fixed bugs in ascending and descending order	Pass
67	View Fixed Bugs	Developer	A developer can switch between exact and semantic search modes in fixed bugs page	Pass
68	View Fixed Bugs	Developer	A developer can reset all filters added in fixed bugs page	Pass
69	Request Reallocation	Developer, Tester	A developer, tester can request bug reallocation with reason	Pass
70	Request Reallocation	Developer, Tester	A developer, tester can view all bug reallocation requests and their status	Pass
71	Request Reallocation	Developer, Tester	A developer, tester can search and filter the reallocation requests by status or priority	Pass
72	Request Reallocation	Developer, Tester	A developer, tester can filter reallocation requests by priority, status, request status and date range	Pass
73	Request Reallocation	Developer, Tester	A developer, tester can sort reallocation requests in ascending and descending order	Pass
74	Request Reallocation	Developer, Tester	A developer, tester can switch between exact and semantic search modes in reallocation requests page	Pass
75	Request Reallocation	Developer, Tester	A developer, tester can reset all filters added in reallocation requests page	Pass
76	Reopen Request	Developer, Tester	A developer, tester can raise a reopen request with reason	Pass
77	Reopen Request	Developer, Tester	A developer, tester can view all bug reopen requests and their status	Pass
78	Reopen Request	Developer, Tester	A developer, tester can search and filter the reopen requests by status or priority	Pass
79	Reopen Request	Developer, Tester	A developer, tester can filter reopen requests by priority, status, request status and date range	Pass
80	Reopen Request	Developer, Tester	A developer, tester can sort reopen requests in ascending and descending order	Pass

Figure 104: Testcases (1)

81	Reopen Request	Developer, Tester	A developer, tester can switch between exact and semantic search modes in reopen requests page	Pass
82	Reopen Request	Developer, Tester	A developer, tester can reset all filters added in reopen requests page	Pass
83	Chat with Reporter/Tec	Developer, Tester, Team Lead, Admin	A tech user can access chat to communicate with the reporter	Pass
84	Update Status	Tester	A tester can change bug status to 'Testing In Progress'	Pass
85	Update Status	Tester	A tester can update bug status to 'Tested & Verified'	Pass
86	Update Status	Tester	A tester can update bug status to 'Fix In Progress'	Pass
87	Update Status	Tester	A tester can revert status update within 15mins of change	Pass
88	Update Status	Tester	A tester can update bug status to 'Ready For Closure' if the bug is critical	Pass
89	Update Status	Tester	A tester cannot update bug status to 'Closed' if the bug is critical	Pass
90	Update Status	Tester	A tester can update bug status to 'Closed' if the bug is not critical	Pass
91	Update Status	Tester	A tester can access the drag-and-drop interface to update status	Pass
92	View Tested Bugs	Tester	A tester can view the list of bugs tested by them	Pass
93	View Tested Bugs	Tester	A tester can view the list of bugs tested by them can search and filter the tested bugs by status or priority	Pass
94	View Tested Bugs	Tester	A tester can export tested bug reports to PDF or CSV	Pass
95	View Tested Bugs	Tester	A tester can filter tested bugs by priority, status and date range	Pass
96	View Tested Bugs	Tester	A tester can sort tested bugs in ascending and descending order	Pass
97	View Tested Bugs	Tester	A tester can switch between exact and semantic search modes in tested bugs page	Pass
98	View Tested Bugs	Tester	A tester can reset all filters added in fixed bugs page	Pass
99	View Tester Analytics	Tester	A tester can access personalized analytics dashboard	Pass
100	View Tester Analytics	Tester	A tester can view workload summary, validation effieicency over time, average test time by priority, SLA breaches, bug status overvie	Pass
101	View Tester Analytics	Tester	A tester can use relevant filters in each metric/insight	Pass
102	View Tester Analytics	Tester	A tester can export metrics/insights as CSV	Pass
103		Developer	A developer receives an email when reopen request is approved	Pass
104		Tester	A tester receives an email when reopen request is approved	Pass
105		Developer	A developer receives an email when reallo request is approved	Pass
106		Tester	A tester receives an email when reallo request is approved	Pass
107	Assign Bugs	Team Lead, Admin	A team lead, admin can manually assign bugs to developers or testers	Pass
108	Assign Bugs	General User, Developer, Tester	A general user, developer, teser cannot assign/reassign bugs	Pass
109	Auto Assign Bugs	Team Lead, Admin	A team lead, admin can trigger auto assign bugs to developers or testers	Pass
110	Auto Assign Bugs		The system runs a cron job to auto assign developer at 2 PM to 6 PM (Mon-Fri)	Pass
111	Auto Assign Bugs		The system runs a cron job to auto assign tester at 3 PM to 7 PM (Mon-Fri)	Pass
112	View Team Assigned Bu	Team Lead, Admin	A team lead, admin can view all bugs reported in their team (team lead) or all teams acorss applications (admin)	Pass
113	View Team Assigned Bu	Team Lead, Admin	A team lead, admin can batch load all bugs reported in their team (team lead) or all teams acorss applications (admin) bug using pagir	Pass
114	View Team Assigned Bu	Team Lead, Admin	A team lead, admin can export team assigned bug reports to PDF or CSV	Pass
115	View Team Assigned Bu	Team Lead, Admin	A team lead, admin can filter team assigned bugs by priority, status, issue type, developer, tester and date range	Pass
116	View Team Assigned Bu	Team Lead, Admin	A team lead, admin can sort team assigned bugs in ascending and descending order	Pass
117	View Team Assigned Bu	Team Lead, Admin	A team lead, admin can switch between exact and semantic search modes in team assigned bugs page	Pass
118	View Team Assigned Bu	Team Lead, Admin	A team lead, admin can reset all filters added in team assigned bugs page	Pass
119	View Team Assigned Bu	Team Lead, Admin	The system can show flags for bugs with no status update after threshold, show flag for team unassigned bugs (admin)	Pass
120	Mark Favourite	Team Lead, Admin	A team lead, admin can mark a bug as favourite in team assigned bugs page	Pass
121	Unmark Favourite	Team Lead, Admin	A team lead, admin can unmark a favourite bug in team assigned bugs page	Pass
122	Set Priority	Team Lead, Admin	A team lead, admin can set or update (admin) bug priority	Pass
123	Set Priority	General User, Developer, Tester	A general user, developer, teser cannot set priority	Pass
124	Update Status	Team Lead, Admin	A team lead, admin can update status with reason	Pass
125	Update Status	Team Lead, Admin	The system can show alert in UI if the reason is not provided or character count is not within the limit for status update	Pass
126	Run Duplicate Check	Team Lead, Admin	A team lead, admin can run duplicate check in the team assigned detail page to check for duplicate bugs (if not already done by the sy	Pass
127	Run Duplicate Check	Team Lead, Admin	A team lead, admin can see the potential duplicate bugs after running the duplicate check	Pass
128	Mark Duplicate	Developer, Tester, Team Lead, Admin	A tech user can mark a bug as duplicate by providing the bug id and the reason	Pass
129	Mark Duplicate	Developer, Tester, Team Lead, Admin	The system can show alert in UI if the original bug id is not provided and reason is not provided or character count is not within the lir	Pass
130	Reallocation Requests	Team Lead, Admin	A team lead, admin can review reallocation requests and approve or reject the same	Pass
131	Reallocation Requests	Team Lead, Admin	The system can show only the developers/testers expect the one who raised the request for reallocation when team lead or admin a	Pass
132	Reopen Requests	Team Lead, Admin	A team lead, admin can review reopen requests and approve or reject the same	Pass
133	Reopen Requests	Team Lead, Admin	The system can show confirmation dialog with yes or no while processing reopen request	Pass
134	View Team Lead Analyt	Team Lead	A team lead can access personalized analytics dashboard	Pass
135	View Team Lead Analyt	Team Lead	A team lead can view current workload summary of developers and testers in team, bug distribution, fix effieicency (developer),valida	Pass
136	View Team Lead Analyt	Team Lead	A team lead can use filters like priority, status, developer, tester, stuck threshold and date range to view the metrics/insights	Pass
137	View Team Lead Analyt	Team Lead	A team lead can export the metrics/insights as CSV	Pass
138	Critical Bug Closure	Team Lead, Admin	The system can send email alerts when critical bug closure is requested	Pass
139	Critical Bug Closure	Team Lead, Admin	A team lead, admin can view email and marked critical bug as closed or change status	Pass
140	Critical Bug Closure	Team Lead, Admin	The system runs a cron job to send email at 2 hours interval from 9 AM to 5 PM (Mon-Fri) to team lead and admin about the critical b	Pass
141	Update Status, Assignm	Team Lead, Admin	A team lead, admin can override bug status or assignment in case of conflict	Pass
142	Unassigned Queue	Admin	An admin can view unassigned bugs in an application	Pass
143	Unassigned Queue	Admin	An admin can manually assign unassigned bugs to a specific team	Pass
144	Unassigned Queue	Admin	The system runs a cron job to send email at 2 hours interval from 9 AM to 5 PM (Mon-Fri) to admin about the unassigned bugs to tear	Pass
145	No Status Update	Developer, Tester, Team Lead, Admin	The system runs a cron job to send email at 10 AM (Mon-Fri) to developer, tester, team lead and admin about the no status update	Pass

Figure 105: Testcases (2)

8.4 API Testing

As part of backend testing, API testing was performed using Postman to validate key endpoints. Although this was not part of the original test plan, it became a crucial step in ensuring the correctness of backend logic and access control. The tested APIs included bug reporting, bug assignment and role-based access checks. For example, while testing the assign-developer API, I verified both successful and unauthorized attempts to confirm that permissions were enforced correctly (Figures 106 and 107). These tests helped validate consistent behaviour across different roles and contributed to overall system reliability.

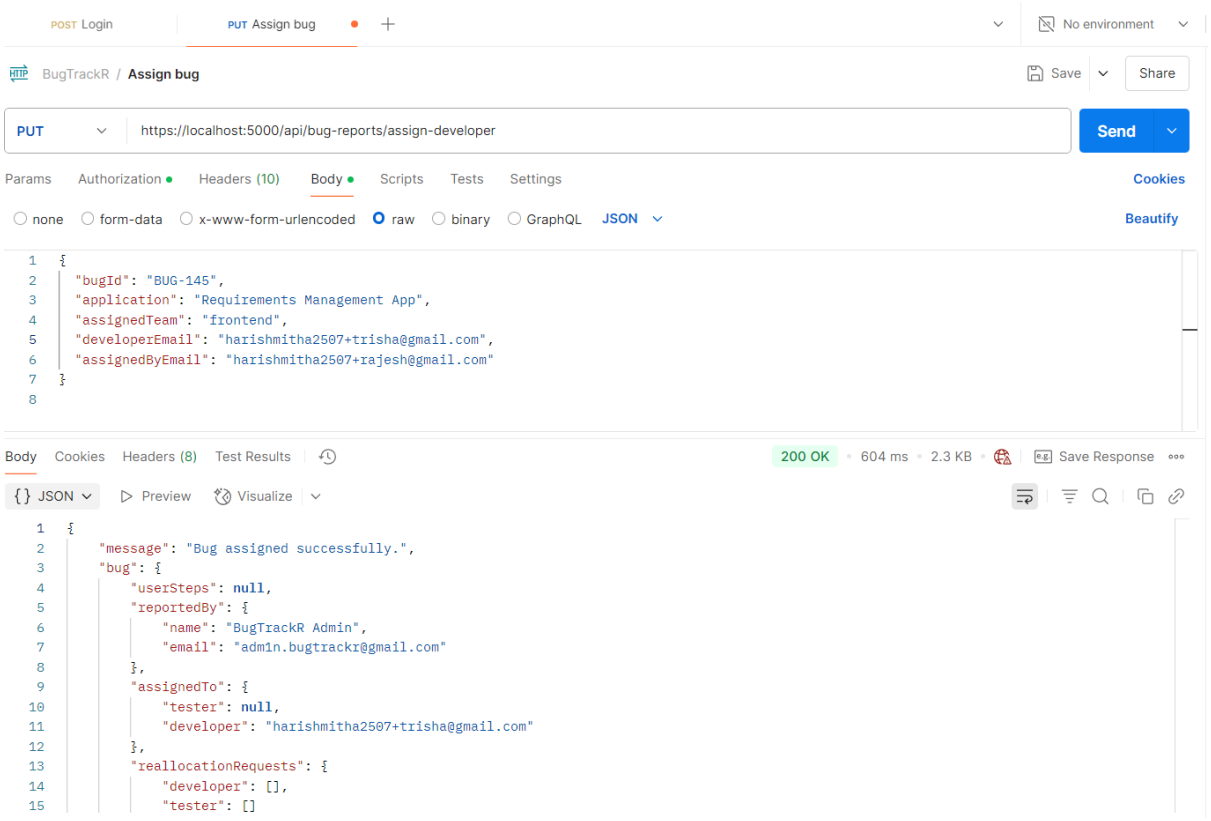


Figure 106: API Testing Success Case

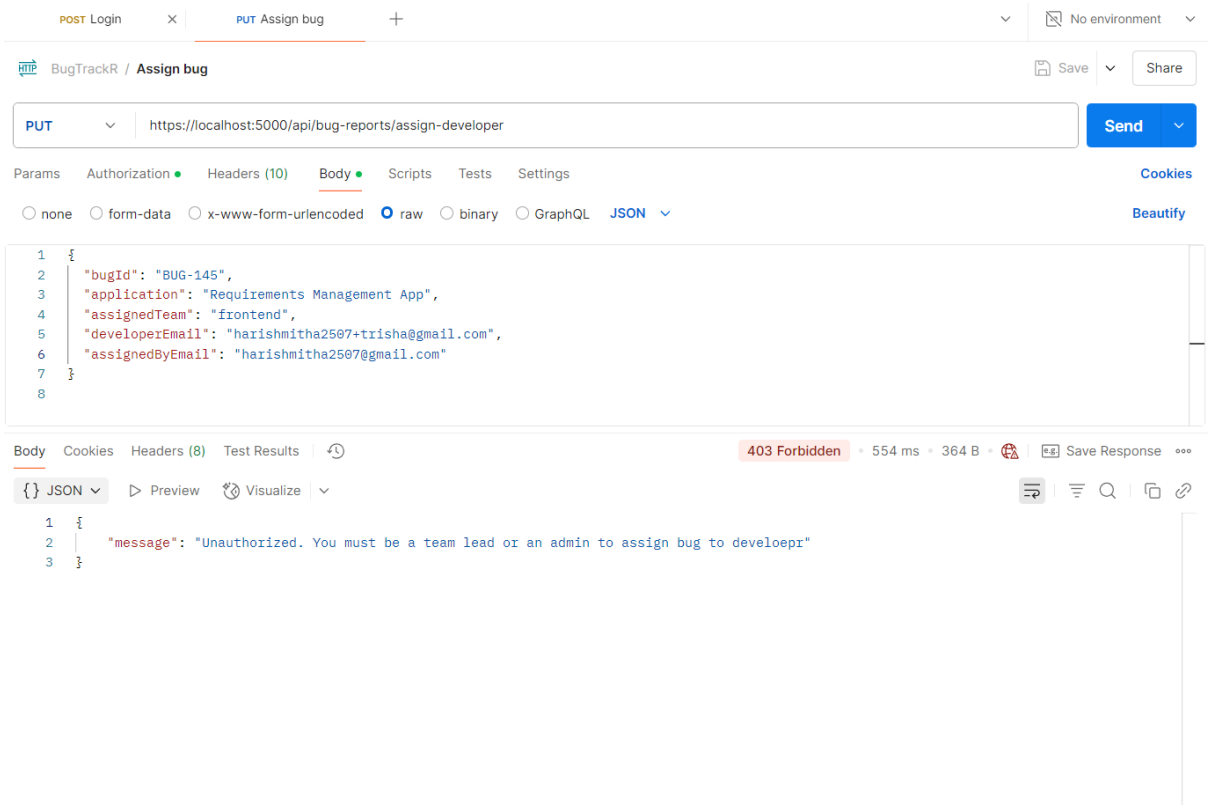


Figure 107: API Testing Unauthorised Case

9. Critical Reflection/Evaluation

9.1 Changes to Requirements During Development

Additions and Adjustments
Reported and Assigned Bugs Listing:
I realized during development that it would be more helpful to show reported and assigned bugs in separate views. This made it easier for users to track the bugs they had reported and those assigned to them. This helped make the overall workflow smoother.

Mentioned Bugs View:
After implementing the feature where users can mention others in comments, I noticed during workflow check that mentioned users didn't have a way to find the bugs they were tagged in. To solve this, I added a separate view for "Mentioned Bugs" where users can see and comment on bugs they were mentioned in without being able to change the core details. This helped improve collaboration without giving them too much access or editing rights.

Reopen Requests:
While working on reallocation requests, I realized that developers and testers might need to reopen bugs if they believed the issue wasn't resolved properly. This led to the addition of a reopen request workflow, where bugs could be requested for reopen with approval from team leads or admin.

Chat Between Reporters and Tech Users:
Originally, I had planned to include basic UI customisation options, like themes and fonts. However, as I progressed, it became clear that communication between reporters and tech users was more important. Reporters had no structured way to talk to the tech team once a bug was submitted. To solve this, I introduced a chat feature that allows direct, real-time conversation within the bug report. This helped reduce the need for email and improved the traceability of discussions.

Critical Bug Approval:
During testing, I realised that critical bugs should not be closed immediately. To prevent accidental closures, I added a workflow where team leads or admin have to review and close the bugs themselves. If they don't act in time, reminder notifications are also triggered.

Change History (Backend Only):
Although I thought of showing change history in the frontend as an additional feature, I wasn't able to due to time constraint. However, I implemented the backend logic to track key changes like assignments, priority updates and duplicate markings.

Dropped or Refined Features
In-App Notifications:
Since email notifications already covered all key events like assignments and status changes, I felt that in-app notifications would be unnecessary now so dropped it.

UI Customisation
This was dropped in favour of building features that added more value to communication and workflow. The chat feature replaced this, as it provided more benefit to the user experience.

Unassigned Bug Queue for Team Leads
Initially, I wanted team leads to allocate bugs marked as "Other" to their teamfrom an unassigned queue. But during access control planning, I decided to limit this to admins to avoid conflicts and confusion. Admins now handle these bugs centrally. In the future, this could be expanded to team leads if needed.

GitHub Integration
Although this was listed as an optional feature, I dropped it during development. The two-way integration with GitHub would have required a lot more time and testing, so I decided to prioritise workflows like reallocation, reopen requests and communication, which I felt were more impactful.

9.2 Justification of Methodology

I chose the Waterfall model for this project because it matched the structure of both the project work and the MSc project timeline. The university's fixed deadlines, like the preliminary, interim, and final reports aligned well with this approach. Each report reflected a specific stage in the project: the preliminary report focused on planning and research, the interim covered design and early implementation, and the final report with testing, evaluation and reflection.

Since I was working independently without a team or client, Agile methods like Scrum didn't really apply. Practices such as daily standups, sprint planning and retrospectives are useful in collaborative settings, but they weren't relevant here. Waterfall gave me a clearer structure to follow and allowed me to plan long-term from the start.

I didn't follow Waterfall in a completely rigid way. While I didn't fully go back and revise earlier stages, I did make small refinements while progressing through development. For example, the changes in Section 9.1 were made after the main planning phase, but they were done without needing to revisit or restructure earlier work. This wasn't a formal Iterative Waterfall [38] approach either, since I didn't loop back to redesign stages. The core phases stayed as originally planned.

9.3 Time Plan Changes

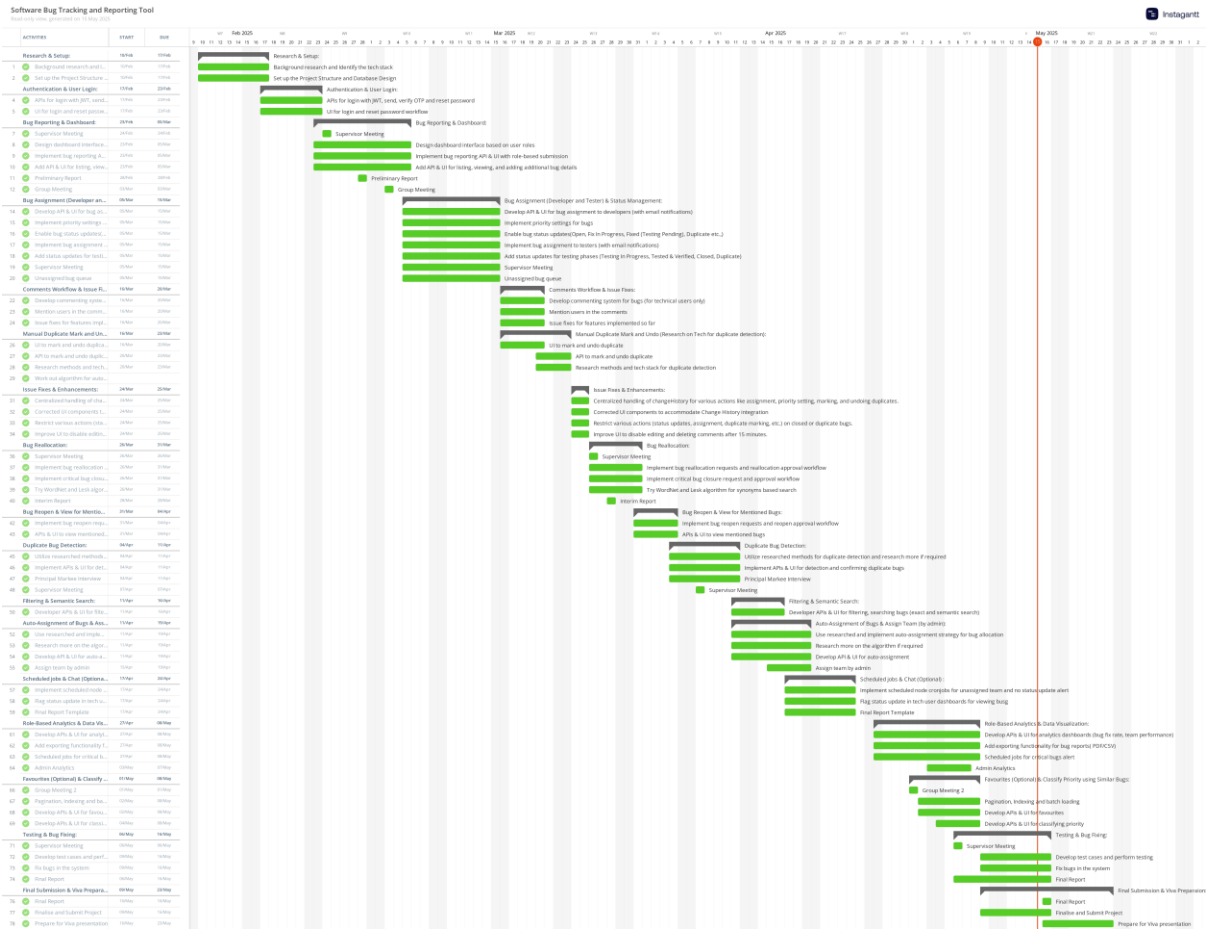


Figure 108: Timeplan

The project mostly followed the original time plan, but a few tasks had to be adjusted as the development progressed. For example, scheduled alerts and inactivity notifications were missed in the plan, but they became important once the core workflows were in place. I added them with role-based triggers to make the system more responsive and keep users informed.

Some features, like role-based analytics, took longer than expected because generating and displaying accurate metrics for each user type turned out to be more complex and tedious than I first thought. Also, duplicate detection was originally planned as one feature, but I ended up separating it into manual and automated workflows during the mid of the project. This made the implementation cleaner and improved the results but also increased the number of days for the same.

Features like mentioned bugs and reopen requests were also added due to which some adjustments were made to the timeplan.

These changes were part of adapting to real development needs and helped make the system more complete and reliable in the end. Figure 109 has information on changes, explained from the interim.

Date	Feature grouping in Gantt Chart	Explanation	Original Duration	Actual Duration
31st Mar – 4th Apr	Bug Reopen & View for Mentioned Bugs	These features were new additions not present in the interim plan. Implemented admin/team lead reopen logic, and created a separate view for mentioned bugs.	Not planned	5 days
4th Apr – 11th Apr	Duplicate Bug Detection	Originally estimated for 4 days in the interim report, but extended to 7 days due to underatnding Python and concepts in NLP model used and checking.	4 days	7 days
11th Apr – 16th Apr	Filtering & Semantic Search	Was grouped with bug reallocation in the interim plan (total 5 days), but reallocation alone took the full duration. This search module was separated and done after, with added exact/semantic search toggle and filters.	Grouped under 5 days	5 days
12th Apr – 19th Apr	Auto-Assignment of Bug & Assign Team (Admin)	No change from interim plan. Developed and checked auto-assignment logic using multiple constraints (priority, workload, team, seniority). Admin-level team assignment also included. (Unassigned Queue (Optionl feature)). No changes in schedule.	1 week	1 week
17th Apr – 24th Apr	Scheduled Jobs & Chat (Optional)	Scheduled jobs for unassigned bugs and delayed updates were missed in interim planning. Included flag icon in UI and implemented chat.	Not planned	1 week
27th Apr – 8th May	Role-Based Analytics & Data Visualization	Originally planned for 5 days but extended to around 11 days due to additional CSV/PDF export features and continuous refinement of /team lead/developer/tester dashboards.	5 days	11 days
1st May – 8th May	Favourites (Optional) & Classify Priority	Worked back and forth along with admin analytics to avoid blockers. Also added batch loading, pagination, and indexing for scalability based on principal marker feedback. These were not originally planned in interim report.	Favourites: 3 days	8 days (combined)
9th May – 16th May	Testing & Bug Fixing	Issue fixes and testing . Code clean. Worked on final report template (half of final report)	1 week	1 week
6th May – 16th May	Final Report	Started from supervisor feedback on the final report template. Progressively updated all sections while working on final fixes and testing. Dedicated time allocated for drwaing design diagrams, refining and proofreading. No changes in schedule.	1 week	11 days
9th May – 23rd May	Final Submission & Viva Preparation	Report finalization, final fixes and viva slide preparation done during this time. No changes in schedule.	1 week	1 week

Figure 109: Time plan changes explanation

9.4 Challenges Faced and Lessons Learned

Duplicate Detection

I started with WordNet [37] and Lesk to search and detect duplicates based on synonyms. However, based on principal marker feedback to use more contextual and scalable approach and validation of WordNet, gave unrelated meanings (e.g., “crash” matching “car crash” or “financial crash”), and Lesk depended on exact overlaps in dictionary definitions, which didn’t work well with real bug reports. I also explored TF-IDF and cosine similarity, but it lacked semantic understanding and was slower than FAISS (handles upto 1 billion vectors) [25] respectively.

I switched to using Sentence Transformers to generate contextual embeddings and used FAISS for faster search. Setting up FastAPI for this gave me practical experience with Python service integration.

Auto-Assignment Algorithm

One of the most complex parts of the project was designing the logic for automatically assigning bugs to developers. My supervisor suggested factoring in real-world constraints like workload hours, bug priority, team type, and developer seniority.
Edge cases, like multiple critical bugs reported at once, were especially tricky. I had to make sure bugs could still be distributed fairly without overloading anyone. I also added reallocation support in case a developer couldn’t handle a bug after assignment. This challenge taught me how to design a realistic decision-making algorithm under constraints.

Real-Time Chat

I initially implemented chat by polling (fetching) chat messages from the backend every few seconds using setInterval(). Although this worked functionally, it created unnecessary load on the server due to constant repeated API calls. It also didn’t feel real-time, as there was always a slight delay in message updates. To solve this, switched to Socket.io, which allowed real-time communication using WebSocket connections. This made the chat more responsive and efficient.

State Management and Role-Based Page Structure

Initially, I planned to reuse the same pages across different roles by conditionally rendering content. However, this led to complex logic and inconsistent state behaviour, which made debugging difficult. For example, in the bug report forms, certain useEffect hooks would run unnecessarily when unrelated fields were updated, due to shared state and dependencies.

To simplify state handling and reduce unintended side effects, I eventually created separate pages for each role, with their own components and logic. I still reused common components where appropriate, but avoided shared pages where it added more confusion than benefit.

Although this approach involved some trade-offs in code reuse, it significantly improved clarity, reduced the likelihood of errors and helped me manage the project more effectively given the timeframe.

Priority Classification with Machine Learning

Implementing ML was completely new to me. I started by learning key concepts like features, labels, TF-IDF, and evaluation metrics such as accuracy, precision and recall. I used cleaned bug report data (title and description), converted the text to numerical vectors using TF-IDF, and experimented with both Logistic Regression and Naive Bayes models.

To improve results, I gave more weight to Critical and High classes during training, as they are more important for the system. This helped Logistic Regression achieve a good F1-score of 0.78 for Critical, but the scores for High (0.11) and Medium (0.30) were still poor. I also tried using class_weight='balanced', but overall accuracy stayed around 63%, Figure 110. Naive Bayes gave a similar result (61% accuracy), Figure 111, which suggests that TF-IDF alone couldn’t capture the contextual overlap between Medium and High classes.

Adjusting class weights helped for Critical bugs, but the overall performance was limited. Improving this would require more time to explore models like BERT (Bidirectional encoder representations from transformers). A contextual model like BERT could better understand the meaning of bug descriptions and distinguish between closely related priorities like Medium and High.

So I implemented priority classification using the similar bugs approach explained in Section 6.7 and 7.3.

	precision	recall	f1-score	support
Low	0.44	0.47	0.45	1772
Medium	0.38	0.24	0.30	1872
High	0.52	0.06	0.11	694
Critical	0.72	0.86	0.78	6183
accuracy			0.63	10521
macro avg	0.52	0.41	0.41	10521
weighted avg	0.60	0.63	0.60	10521

Figure 110: Classification report for Logistic Regression

	precision	recall	f1-score	support
Critical	0.48	0.21	0.29	1772
High	0.51	0.05	0.10	1872
Low	0.50	0.00	0.00	694
Medium	0.63	0.97	0.76	6183
accuracy			0.61	10521
macro avg	0.53	0.31	0.29	10521
weighted avg	0.57	0.61	0.51	10521

Figure 111: Classification report for Naïve Bayes

9.5 Comparison with Existing Tools

The limitations identified in Section 2.3 were directly addressed in BugTrackR's design and implementation. Compared to Bugzilla's static, form-heavy UI for search, BugTrackR introduced role-based adaptive interfaces, where users only see fields relevant to their role, Figure 112 and 113.

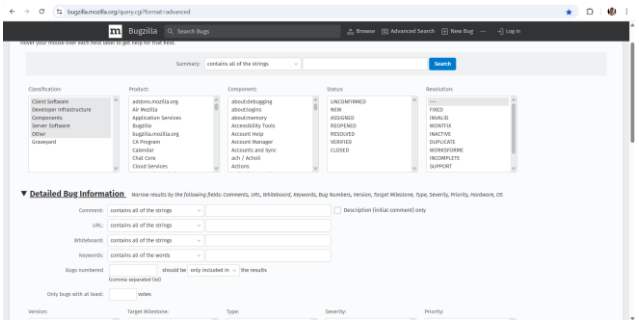


Figure 112: Search view in Bugzilla

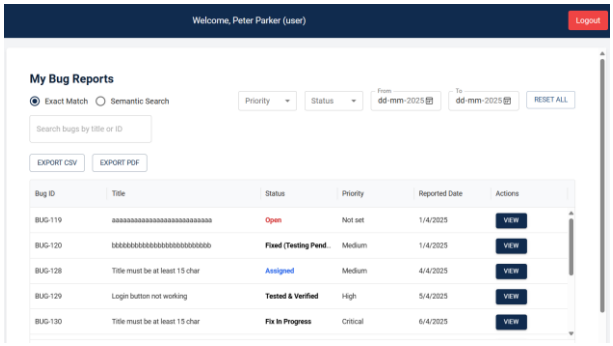


Figure 113: Search view in BugTrackR

Unlike JIRA, which relies on keyword matches in the summary field, BugTrackR uses semantic similarity to suggest duplicates based on overall meaning, even when the phrasing is different. The comparison below Figures 114, 115 and 116 highlights how each system presents potential duplicates.

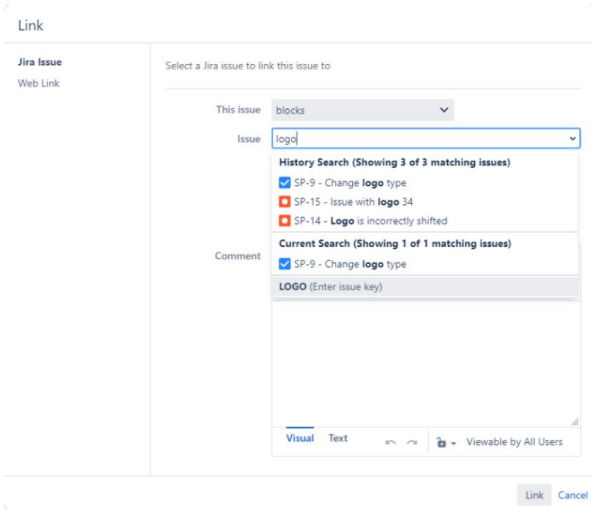


Figure 114: Duplicate results from JIRA [23]

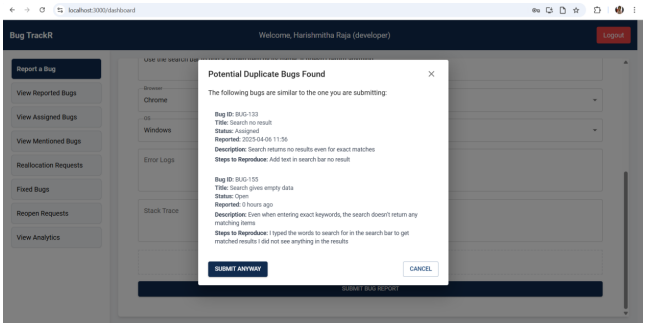


Figure 115: Duplicate results from BugTrackR (see 7.2 for more details)

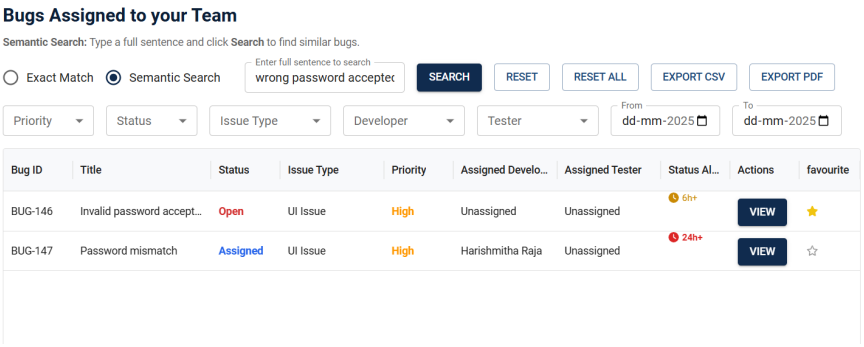


Figure 116: Semantic Search Results from BugTrackR

Unlike JIRA's single summary field, Figure 117, BugTrackR uses structured, role-specific forms to ensure more complete and useful bug reports, Figure 118.

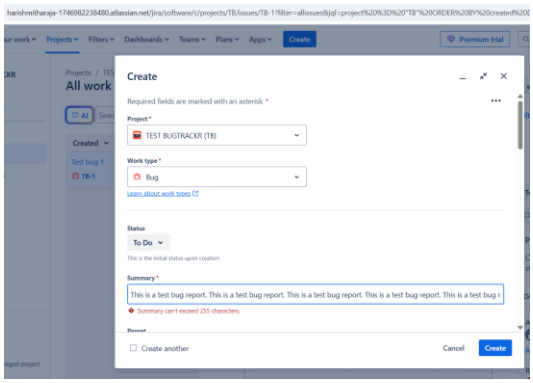


Figure 117: Bug Report form view in Jira

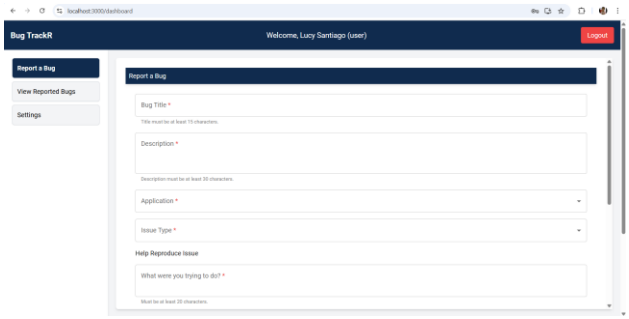


Figure 118: Bug Report form view in BugTrackR

10. Conclusion

In conclusion, BugTrackR successfully met all project objectives by delivering a scalable and user-centric bug tracking system tailored to real-world team workflows. The system supports structured bug reporting, automated assignment based on seniority, team, workload and priority, and duplicate detection using semantic search. By combining role-specific dashboards with integrated communication spaces, BugTrackR improves coordination across technical and non-technical users, reducing delays and redundancy in the bug resolution process.

The platform's modular design and use of flexible technologies make it well-suited for future scaling, including integration with third-party tools and the addition of ML-powered features. Overall, BugTrackR offers a complete and practical solution to common limitations found in existing tracking systems, and stands as a strong foundation for further innovation in collaborative software quality management.

11. Recommendations for Future Work

To avoid meaningless or gibberish bug reports, a multi layered validation mechanism could be added as a future enhancement. This would include checking semantic score (similar or unrelated) between the title, description, and steps using Sentence Transformers, detecting gibberish by calculating the real-word ratio which checks how many words in each field are valid English words using a dictionary and using language detection to ensure the content is in English. Bug reports failing these checks could be rejected with a clear message, improving report quality.

In future, the system could be updated to support different time-based iterations. Once updated, bugs could be linked to defined iterations so that actions are tied to a specific phase.

Now, only admins manage the team unassigned queue. In future, team leads could be allowed to pick bugs from the unassigned queue and tag them under their teams. Similarly, developers and testers could be allowed to self-assign bugs from their team's unassigned queue, giving more flexibility and reducing delays in triage.

In future, syncing BugTrackR with GitHub could allow automatic issue creation and status sync. This would especially benefit teams that use GitHub for source control and issue tracking.

12. References

[1] Zimmermann, T., Premraj, R., Bettenburg, N., Just, S., Schroter, A. and Weiss, C. What Makes a Good Bug Report? IEEE Transactions on Software Engineering, 36(5), pp.618–643. https://doi.org/10.1109/tse.2010.63

[2] Bhattacharya, P., & Neamtiu, I. "Fine-Grained Incremental Learning and Multi-Feature Tossing Graphs to Improve Bug Triaging." Proceedings of the IEEE International Conference on Software Maintenance (ICSM), 1–10. https://dl.acm.org/doi/10.1109/ICSM.2010.5609736

[3] Bettenburg, N & Premraj, R. & Zimmermann, T & Kim, S. Duplicate Bug Reports Considered Harmful ... Really? IEEE International Conference on Software Maintenance, ICSM.337-345.10.1109/ICSM.2008.4658082. https://www.researchgate.net/publication/224343297_Duplicate_Bug_Reports_Considered_Harmful_Really

[4] Hooimeijer, P., & Weimer, W. "Modeling Bug Report Quality." Proceedings of the 22nd IEEE/ACM International Conference on Automated Software Engineering (ASE), 34-43. https://dl.acm.org/doi/10.1145/1321631.1321639

[5] Mockus, A., & Herbsleb, J. D. "Expertise Browser: A Quantitative Approach to Identifying Expertise." Proceedings of the 24th International Conference on Software Engineering(ICSE),503-512. https://www.researchgate.net/publication/2543802_Expertise_Browser_A_Quantitative_Approach_to_Identifying_Expertise

[6] Excalidraw. "Excalidraw.", https://excalidraw.com/

[7] "Angular." Angular.io, 2025, https://v17.angular.io/guide/developer-guide-overview

[8] "Angular vs ReactJS : Which One to Choose as Frontend Development Framework in 2024." GeeksforGeeks, 29 May 2019, www.geeksforgeeks.org/angular-vs-reactjs/.

[9] Stefanov, S. (2016). React Up and Running: Building Web Applications. O'Reilly Media. https://dl.ebooksworld.ir/books/React.Up.and.Running.2nd.Edition.Stoyan.Stefanov.OReilly.9781492051466.EBooksWorld.ir.pdf

[10] Freeman, A. (2018). Pro React 16. Apress. https://dl.ebooksworld.ir/motoman/Pro.React.16.Adam.Freeman.Apress.www.EBooksWorld.ir.pdf

[11] "Overview - MUI System." Mui.com, 2025 https://mui.com/system/getting-started/

[12] Khatri, Agastya. "Node.js vs Spring Boot- Which Should You Choose? - Javarevisited - Medium." Medium, Javarevisited, 20 Dec. 2023, https://medium.com/javarevisited/node-js-vs-spring-boot-which-should-you-choose-c86de8d36d07

[13] MongoDB Docs - Horizontal Scalability and Performance Features, https://www.mongodb.com/docs/manual/sharding/

[14] "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," Hugging Face. [Online]. https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

[15] "Faiss/README.md at Main · Facebookresearch/Faiss." GitHub, 2017, https://github.com/facebookresearch/faiss/blob/main/README.md

[16] Wang, Xiaoyin, et al. "An Approach to Detecting Duplicate Bug Reports Using Natural Language and Execution Information." Proceedings of the 13th International Conference on SoftwareEngineering-ICSE'08,2008, https://www.researchgate.net/publication/221554559_An_approach_to_detecting_duplicate_bug_reports_using_natural_language_and_execution_information

[17] Kumari, Neha, et al. "Automating Bug Triage: Unleashing the Power of Machine Learning and Natural Language Processing." Research Square (Research Square), 22 July 2024 https://www.researchgate.net/publication/382455930_Automating_Bug_Triage_Unleashing_the_Power_of_Machine_Learning_and_Natural_Language_Processing

[18] Kualitee. User-Centric Bug Resolution: Best Practices with Bug Tracking Software. https://www.kualitee.com/blog/bug-management/user-centric-bugresolution-best-practices-with-bug-tracking-software/

[19] Ramírez-Mora, Sandra L., et al. "Exploring the Communication Functions of Comments during Bug Fixing in Open Source Software Projects." Information and Software Technology, vol. 136, Aug. 2021, p. 106584, https://www.sciencedirect.com/science/article/pii/S0950584921000665

[20] "Jira Documentation. Atlassian Support. Atlassian Documentation." https://confluence.atlassian.com/jira

[21] "Features." Bugzilla, 2025, https://www.bugzilla.org/about/features.html

[22] Admin. "Automatically Assign Issues - JIRA Automation - Tech Agilist." Tech Agilist, 4 Mar. 2023, www.techagilist.com/agile/jira/automatically-assign-issues-jira-automation/.

[23] Nataliya Vasyliv. "How to Find Duplicates in Jira?" Reliex.com, Reliex, 2023, https://reliex.com/blog/how-to-find-duplicates-in-jira

[24] Campbell, Sarah. "New! Restrict Your Internal Notes to Project Roles or Groups." Atlassian Community, 22 May 2024, https://community.atlassian.com/forums/Jira-Service-Management-articles/New-Restrict-your-internal-notes-to-project-roles-or-groups/ba-p/2704909

[25] "How to Build a Semantic Search Engine with Transformers and Faiss | towards Data Science." Towards Data Science, 9 Nov. 2020 https://towardsdatascience.com/how-to-build-a-semantic-search-engine-with-transformers-and-faiss-dcbea307a0e8/

[26] “React DnD.” React-Dnd.github.io, https://react-dnd.github.io/react-dnd/about

[27] “GitHub - Chatscope/Chat-Ui-Kit-React: Build Your Own Chat UI with React Components in Few Minutes. Chat UI Kit from Chatscope Is an Open Source UI Toolkit for Developing Web Chat Applications.” GitHub, 3 Mar. 2024. https://github.com/chatscope/chat-ui-kit-react

[28] “Instagantt. Online Gantt Chart Software. Asana Integration.” https://app.instagantt.com/

[29] “Figma: The Collaborative Interface Design Tool.” Figma, 2024, www.figma.com/.

[30] draw.io. “Flowchart Maker & Online Diagram Software.” App.diagrams.net, 2025. https://app.diagrams.net

[31] Flaticon. “Flaticon, the Largest Database of Free Vector Icons.” Flaticon, 2010, www.flaticon.com/.

[32] IcePanel Technologies Inc. “TechIcons.” Techicons.dev, 2024. https://techicons.dev/

[33] Microsoft. “Visual Studio Code.” Visualstudio.com, Microsoft, 2024. https://code.visualstudio.com/

[34] “Install MongoDB - MongoDB Manual.” Www.mongodb.com, https://www.mongodb.com/docs/manual/installation/

[35] “MongoDB Compass.” MongoDB, www.mongodb.com/products/tools/compass

[36] W3Schools. “Python Tutorial.” W3schools.com, 2019. www.w3schools.com/python/

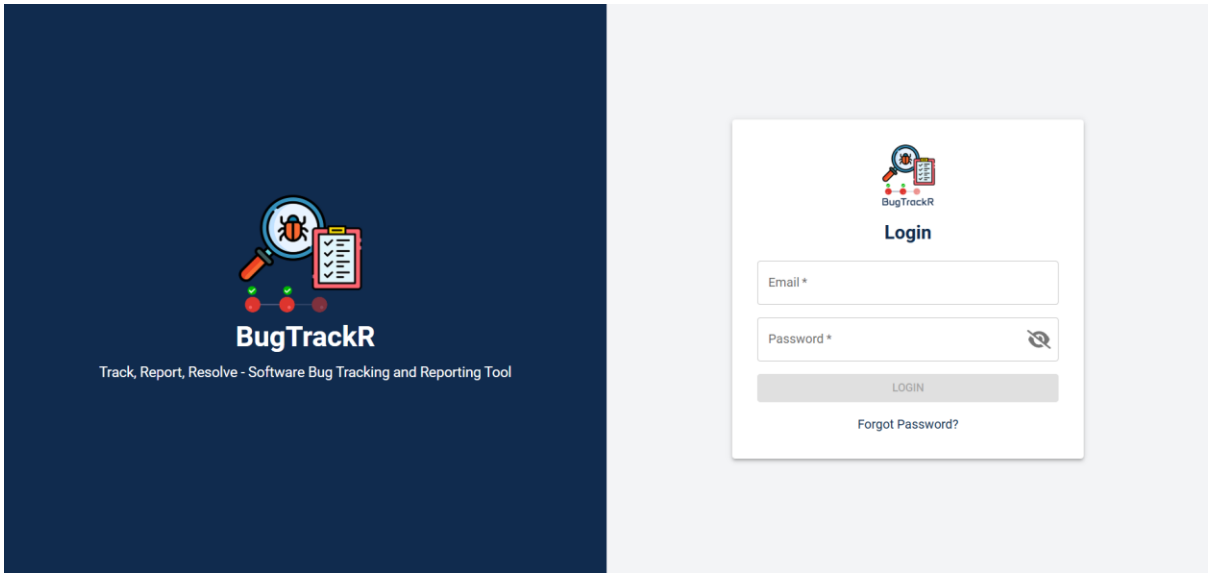
[37] Bird, Jim. “Why Are Some Bugs Harder to Fix than Others?” Blogspot.com, 2017, https://swreflections.blogspot.com/2012/07/why-are-some-bugs-harder-to-fix-than.html

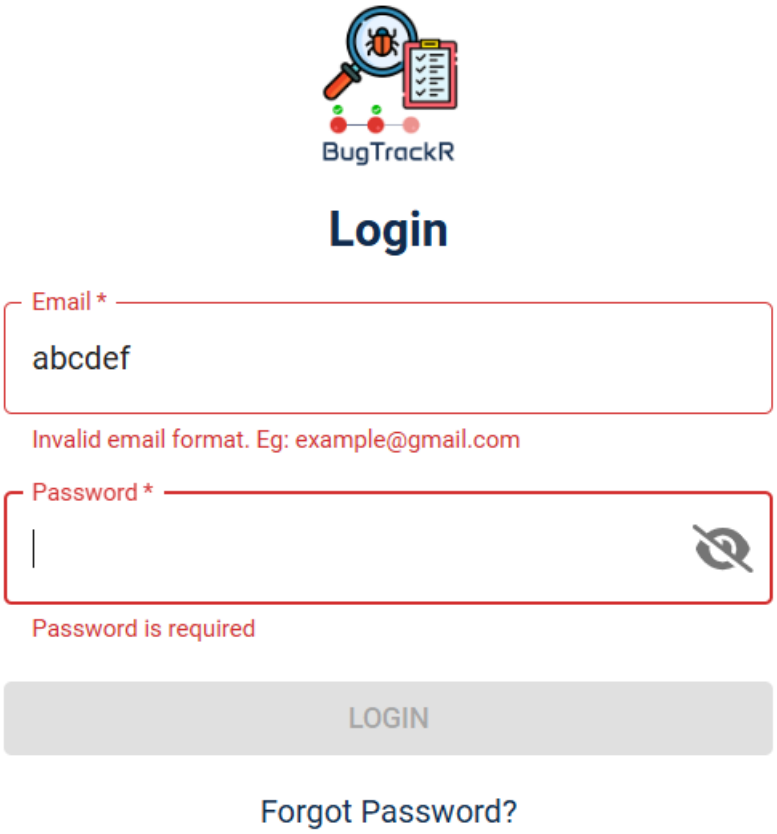
[38] Princeton University "About WordNet." WordNet. Princeton University. 2010. https://wordnet.princeton.edu/citing-wordnet

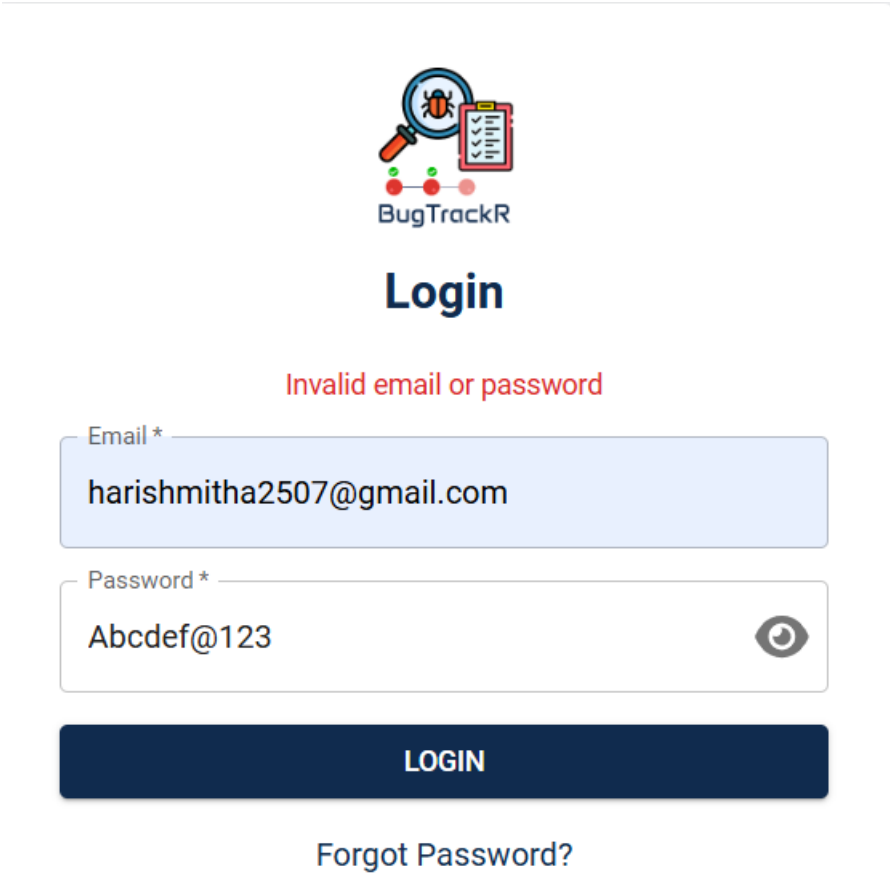
[39] GeeksforGeeks. “Iterative Waterfall Model Software Engineering.” GeeksforGeeks, 18 Mar. 2018, https://www.geeksforgeeks.org/software-engineering-iterative-waterfall-model/

13. Appendices

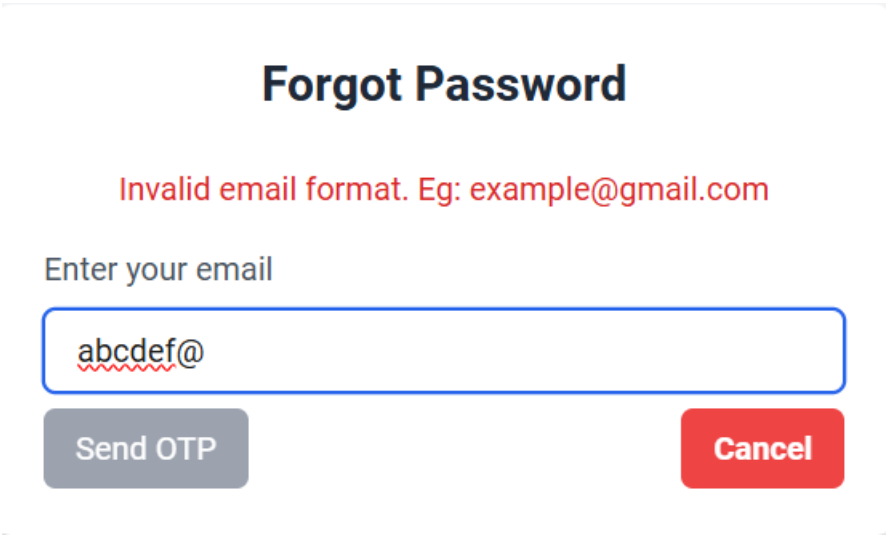
APPENDIX A – Login

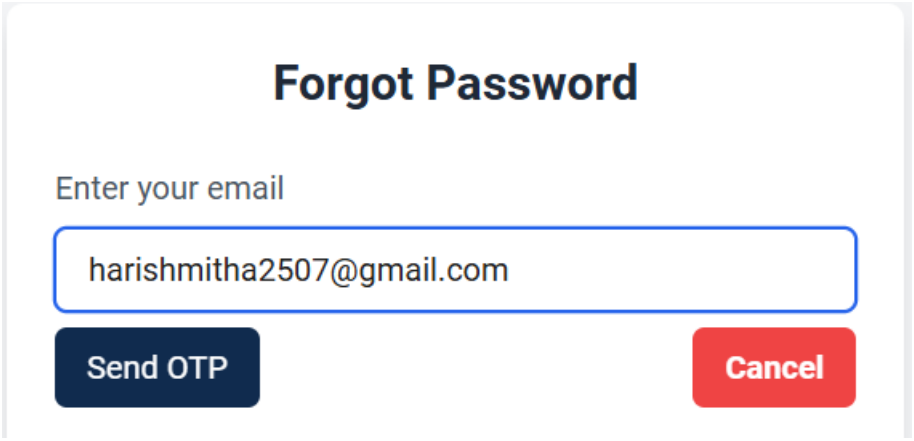

BugTrackR
Track, Report, Resolve - Software Bug Tracking and Reporting Tool
BugTrackR
Login
Email *
Password *
LOGIN
Forgot Password?

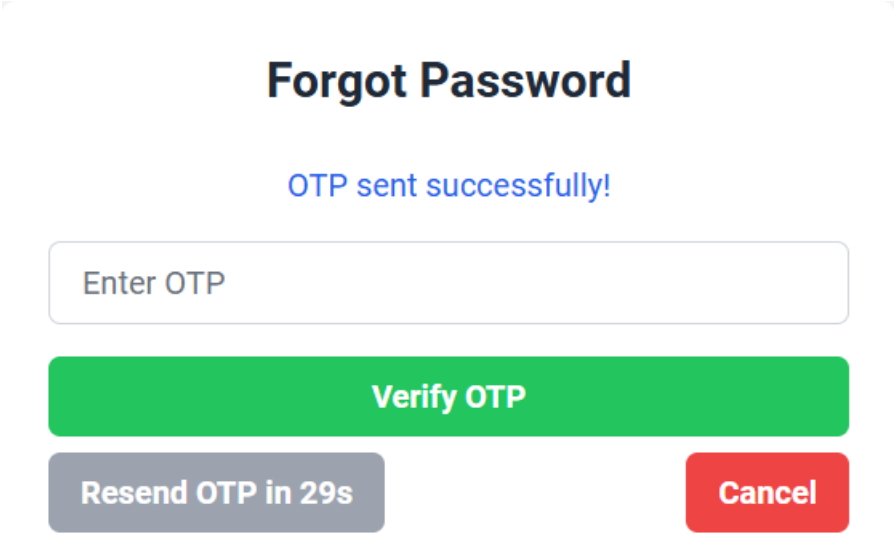

BugTrackR
Login
Email *
abcdef
Invalid email format. Eg: example@gmail.com
Password *
Password is required
LOGIN
Forgot Password?

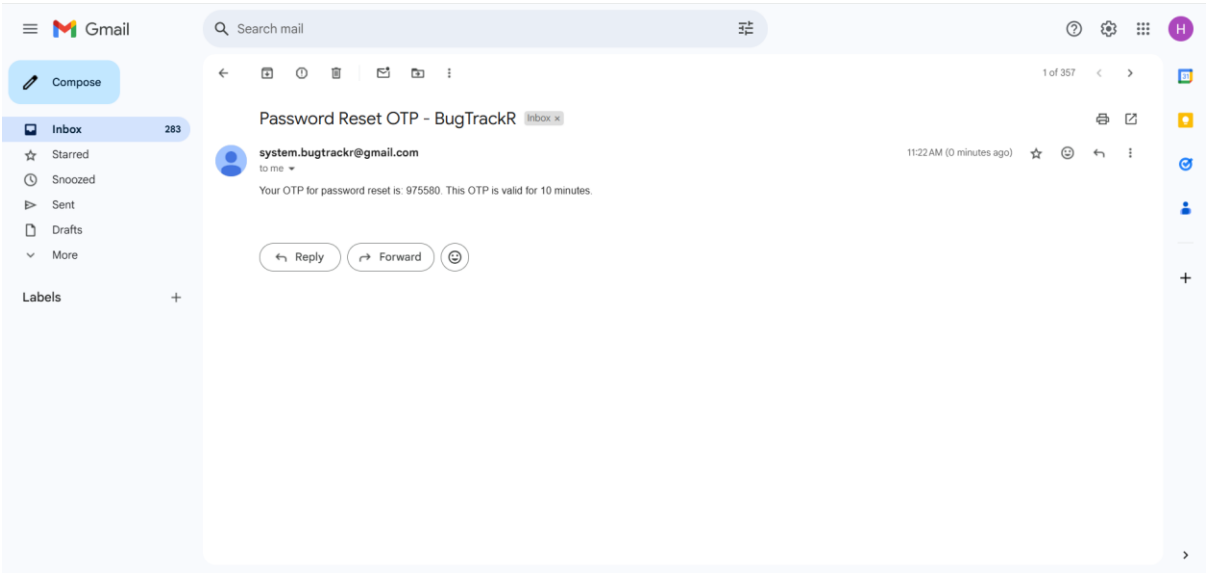


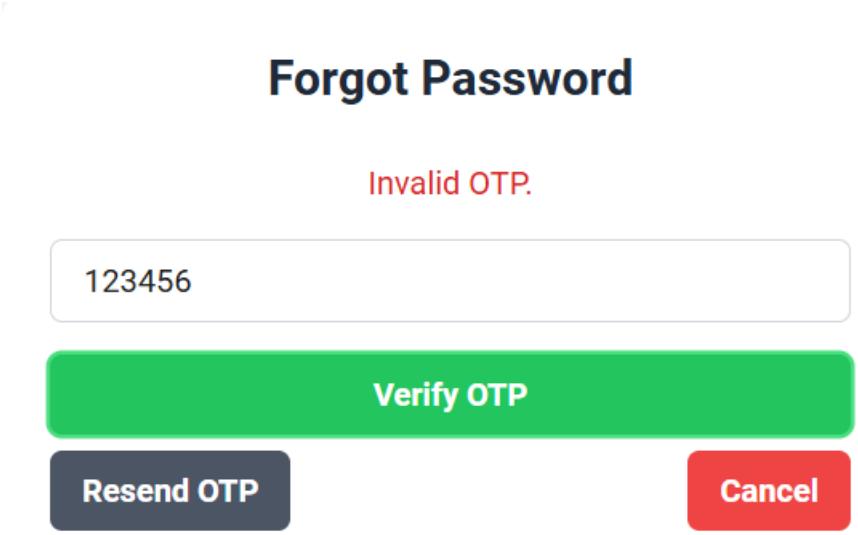
APPENDIX A – Forgot PAssword

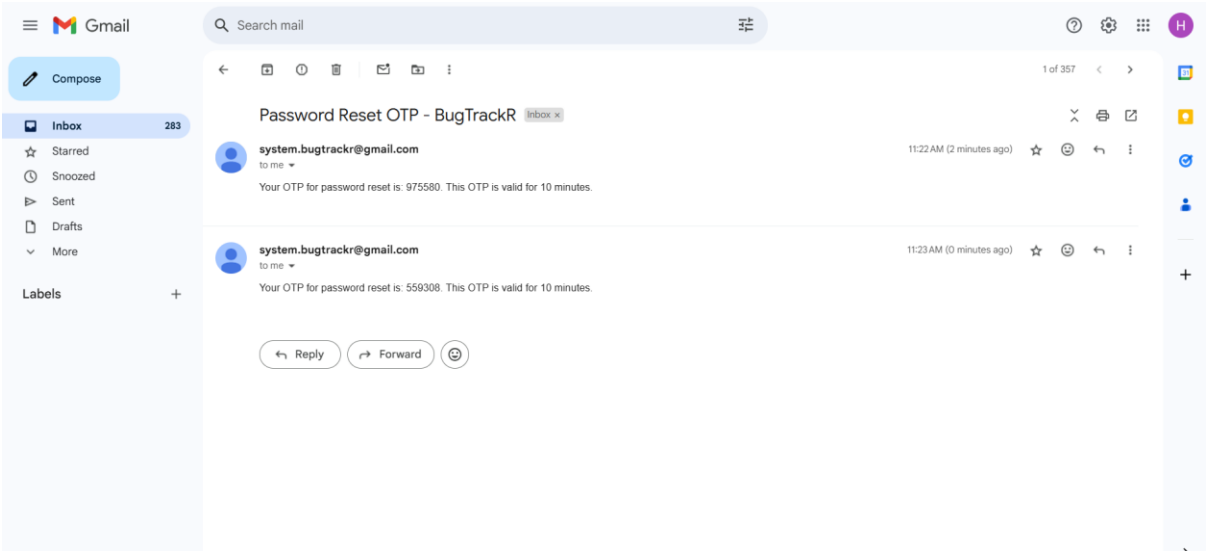


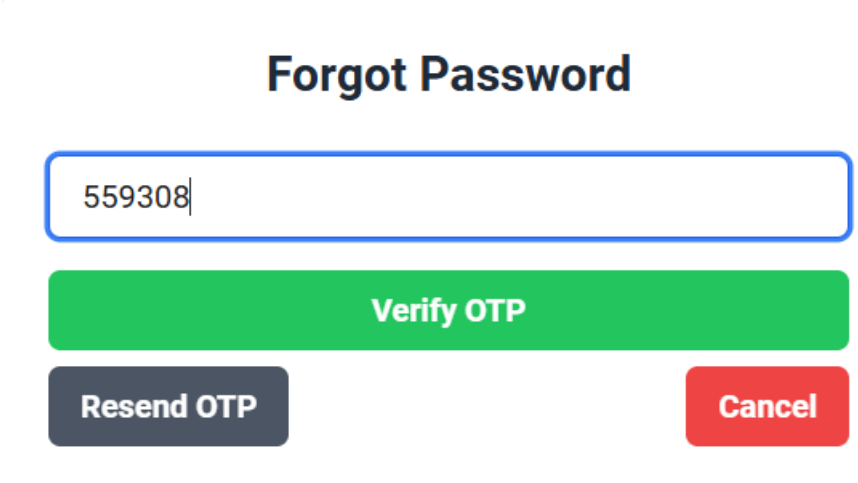

Forgot Password
Enter your email
harishmitha2507@gmail.com
Send OTP
Cancel

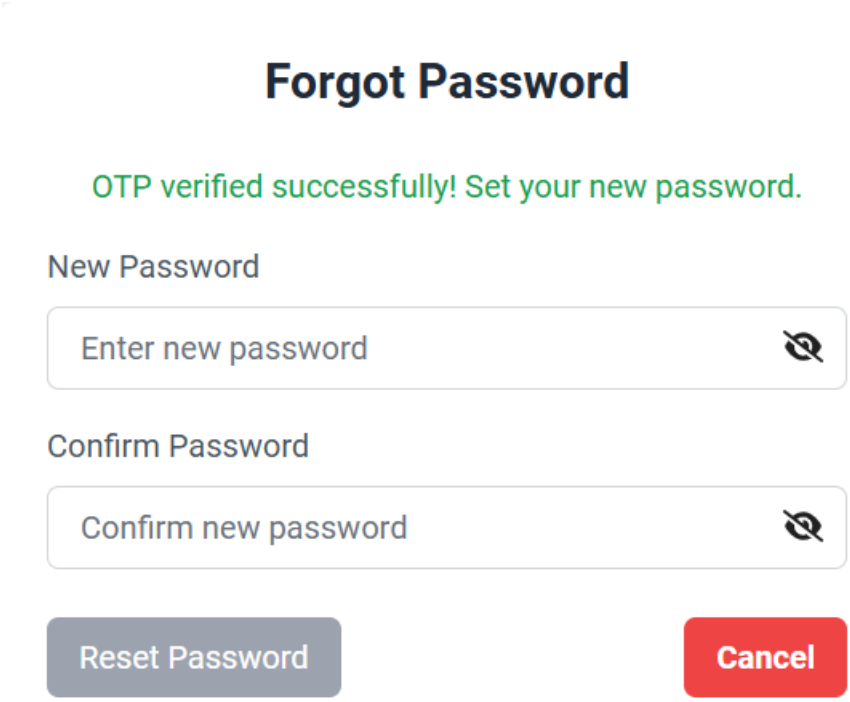

Forgot Password
OTP sent successfully!
Enter OTP
Verify OTP
Resend OTP in 29s
Cancel

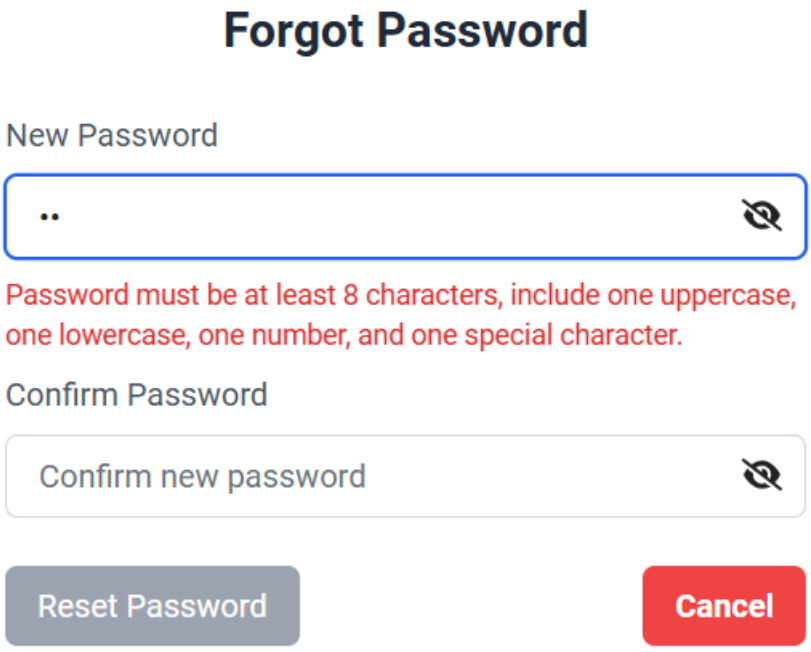

Gmail
Search mail
Compose
Inbox 283
Starred
Snoozed
Sent
Drafts
More
Labels
Password Reset OTP - BugTrackR
system.bugtrackr@gmail.com
to me
11:22 AM (0 minutes ago)
Your OTP for password reset is: 975580. This OTP is valid for 10 minutes.
Reply
Forward

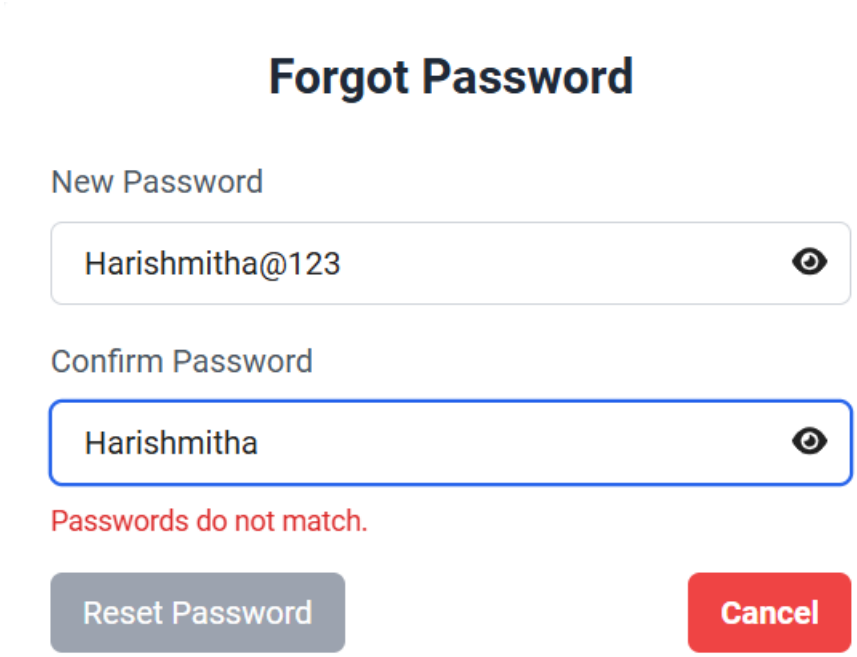

Forgot Password
Invalid OTP.
123456
Verify OTP
Resend OTP
Cancel

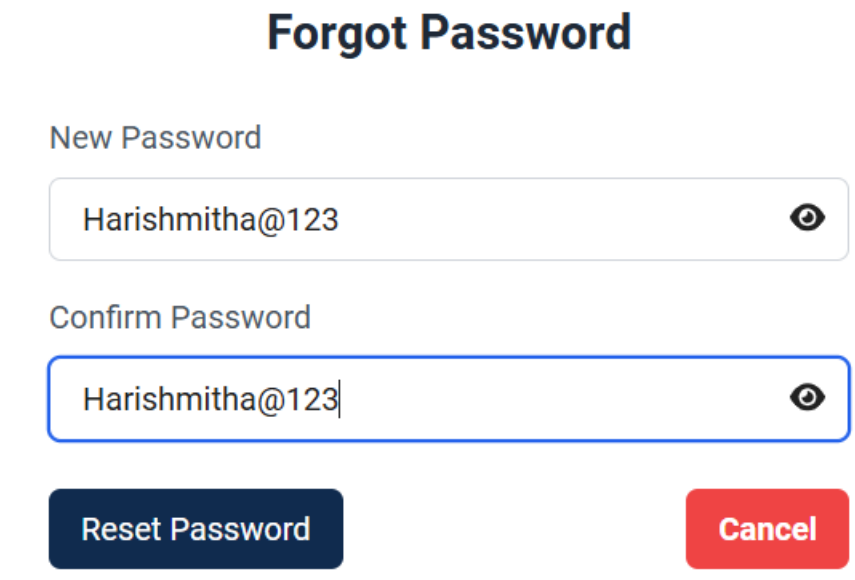

Gmail
Search mail
Compose
Inbox 283
Starred
Snoozed
Sent
Drafts
More
Labels
1 of 357
Password Reset OTP - BugTrackR
Inbox
system.bugtrackr@gmail.com
to me
11:22 AM (2 minutes ago)
Your OTP for password reset is: 975580. This OTP is valid for 10 minutes.
system.bugtrackr@gmail.com
to me
11:23 AM (0 minutes ago)
Your OTP for password reset is: 559308. This OTP is valid for 10 minutes.
Reply
Forward

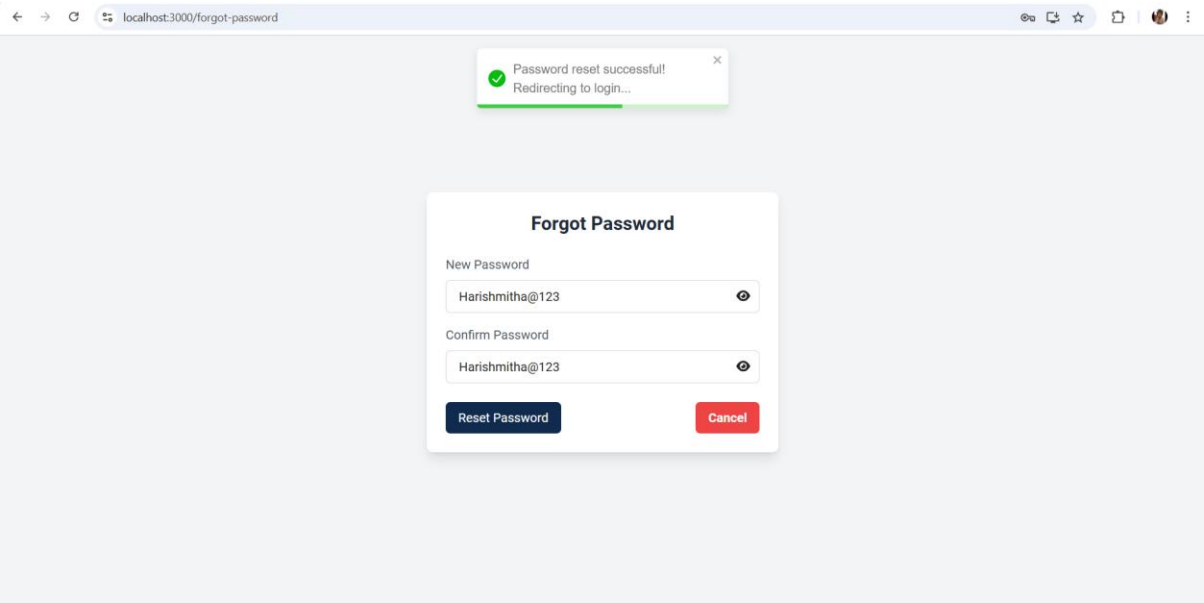

Forgot Password
559308
Verify OTP
Resend OTP
Cancel


Forgot Password
OTP verified successfully! Set your new password.
New Password
Enter new password
Confirm Password
Confirm new password
Reset Password
Cancel


Forgot Password
New Password
••
Password must be at least 8 characters, include one uppercase, one lowercase, one number, and one special character.
Confirm Password
Confirm new password
Reset Password
Cancel

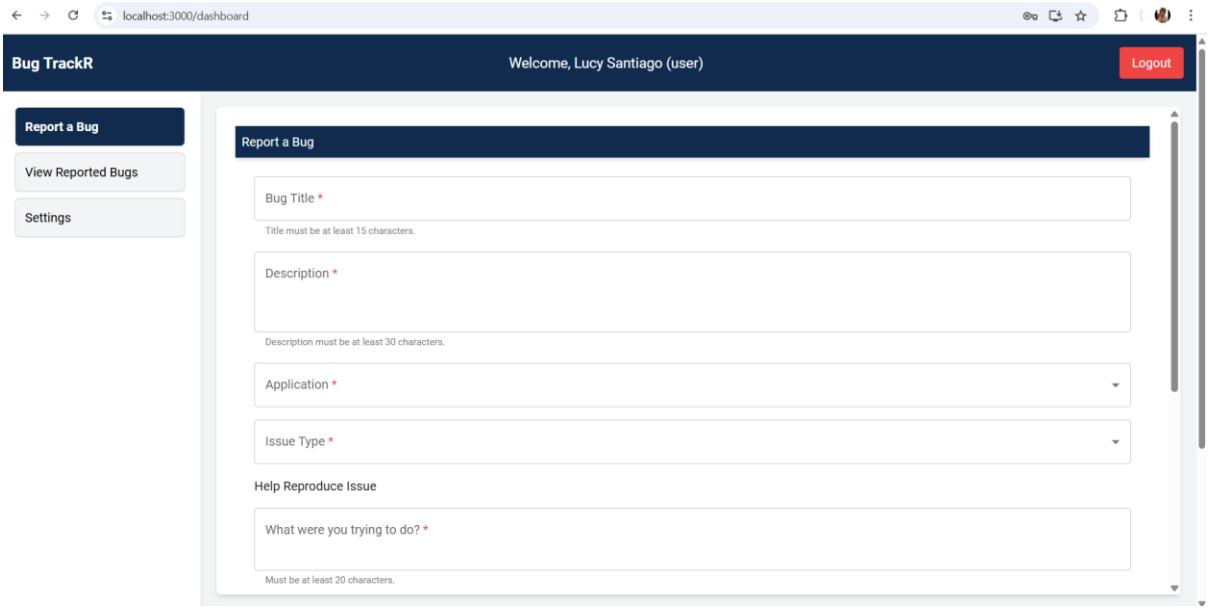

Forgot Password
New Password
Harishmitha@123
Confirm Password
Harishmitha
Passwords do not match.
Reset Password
Cancel

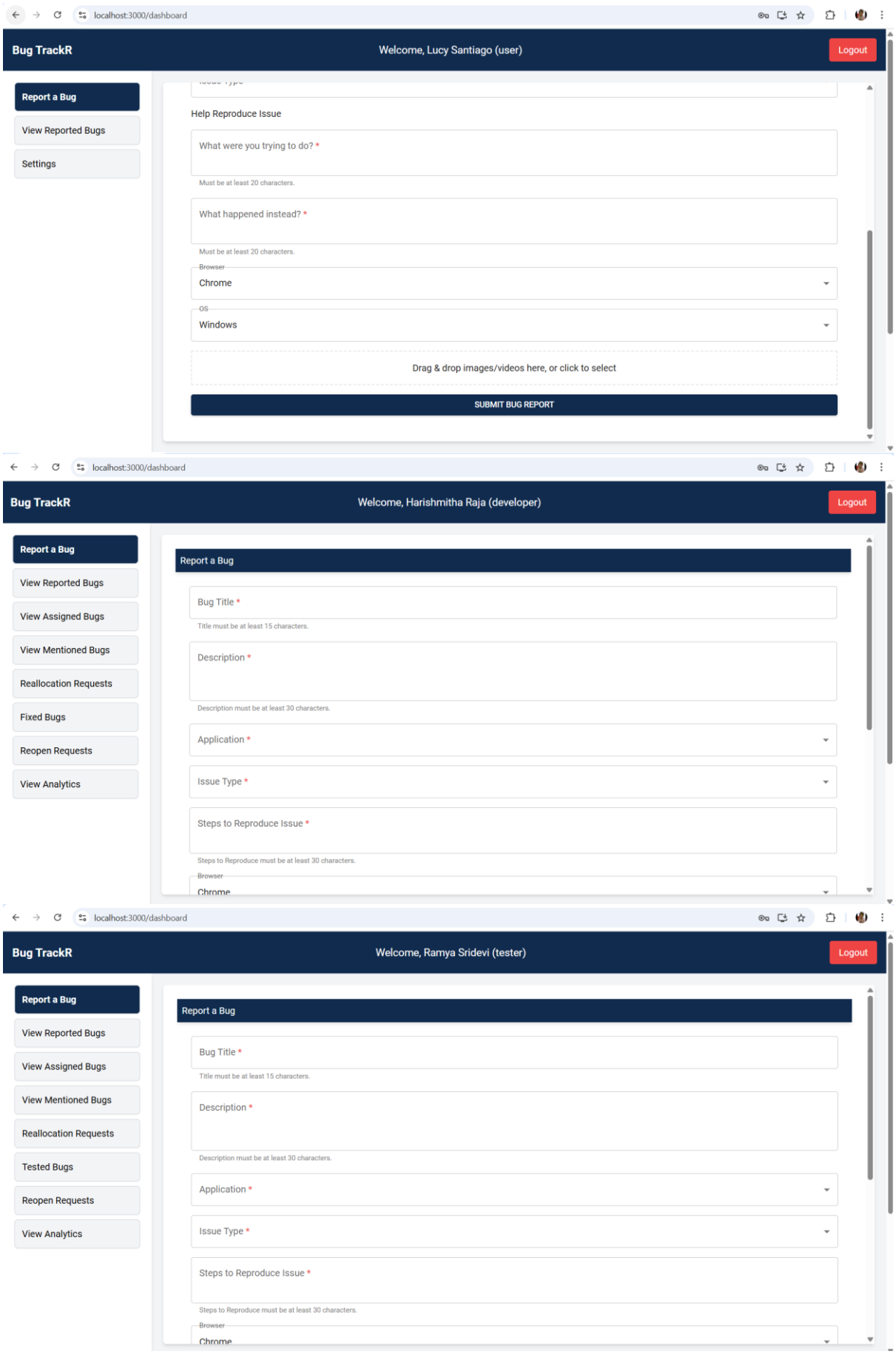

Forgot Password
New Password
Harishmitha@123
Confirm Password
Harishmitha@123
Reset Password
Cancel

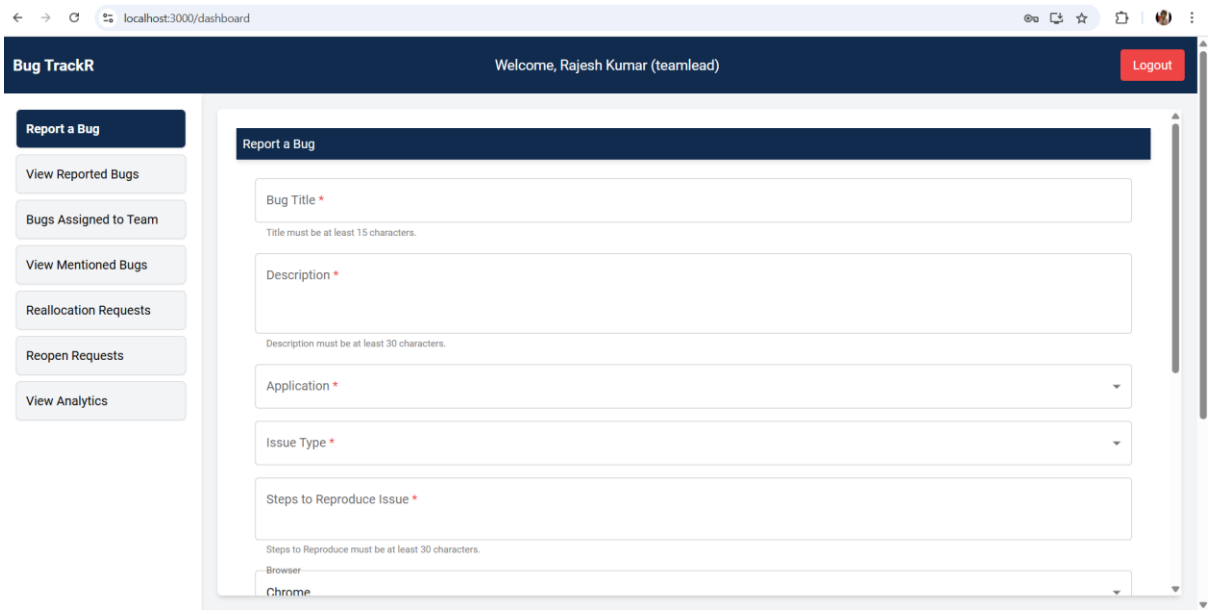


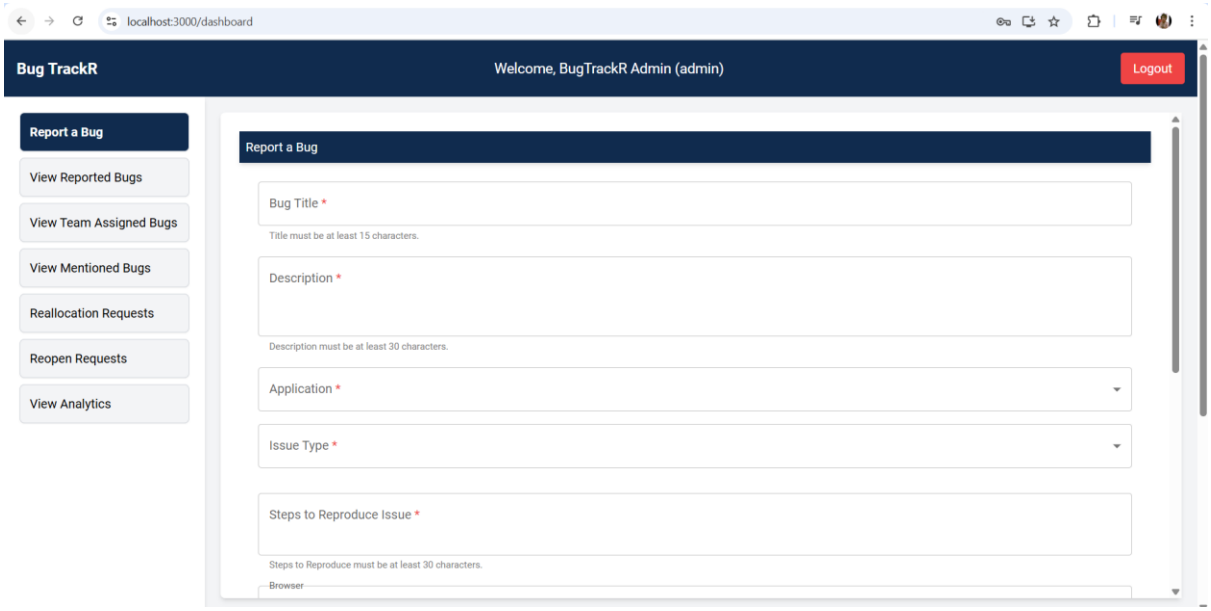
Note: Role is available on the header of the pages – next to the name (for reference)

APPENDIX C - Dashboard

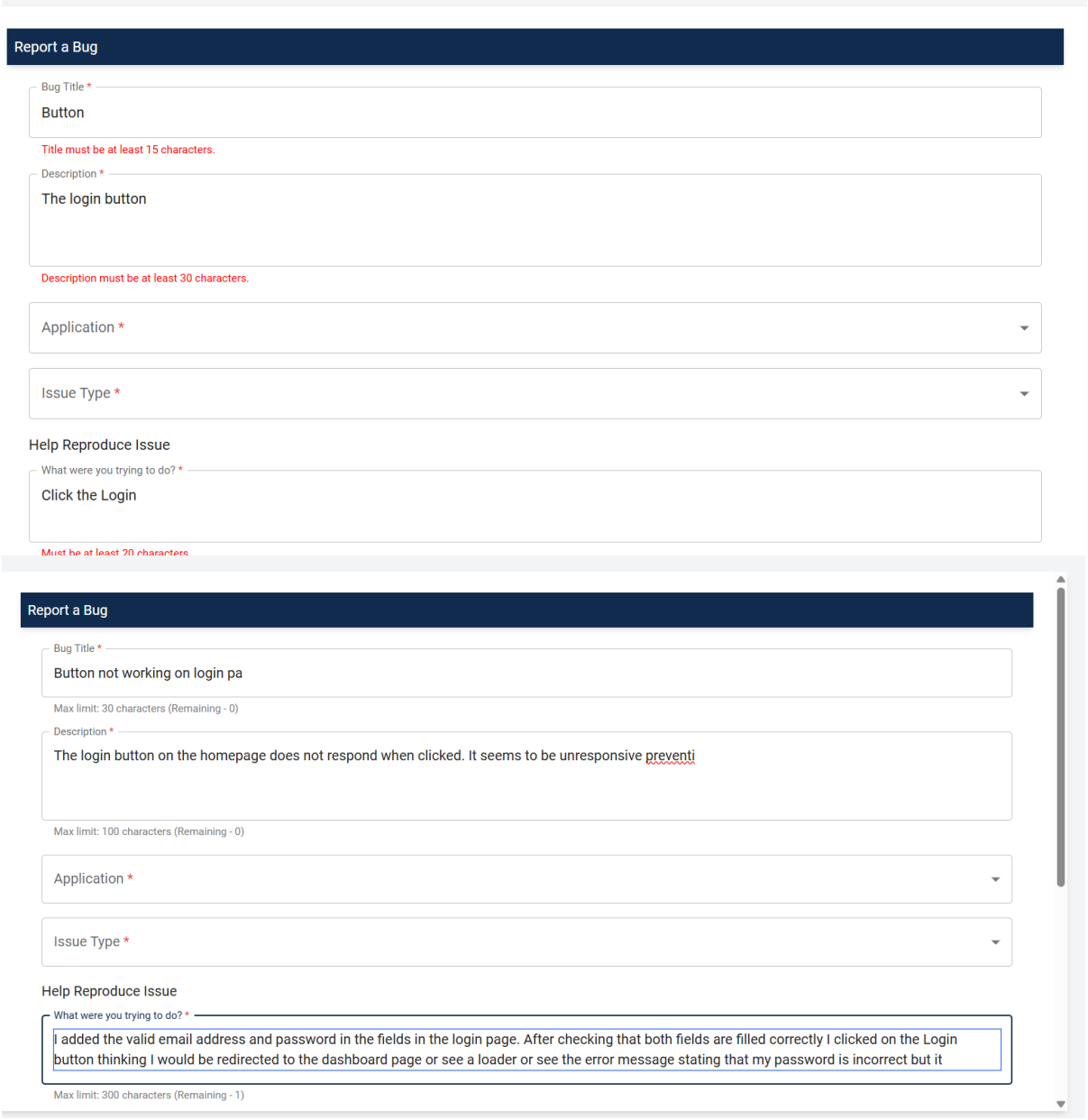


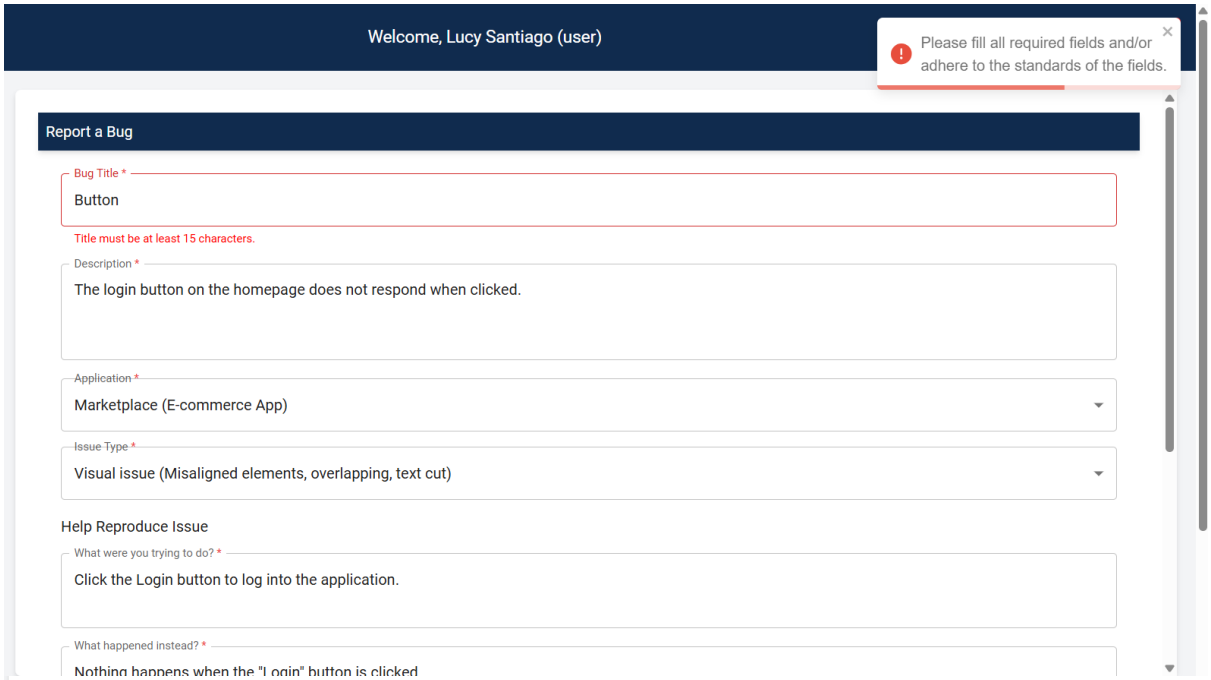

localhost:3000/dashboard
Bug TrackR
Welcome, Lucy Santiago (user)
Logout
Report a Bug
View Reported Bugs
Settings
Help Reproduce Issue
What were you trying to do? *
Must be at least 20 characters.
What happened instead? *
Must be at least 20 characters.
Browser
Chrome
OS
Windows
Drag & drop images/videos here, or click to select
SUBMIT BUG REPORT
localhost:3000/dashboard
Bug TrackR
Welcome, Harishmitha Raja (developer)
Logout
Report a Bug
View Reported Bugs
View Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Fixed Bugs
Reopen Requests
View Analytics
Report a Bug
Bug Title *
Title must be at least 15 characters.
Description *
Description must be at least 30 characters.
Application *
Issue Type *
Steps to Reproduce Issue *
Steps to Reproduce must be at least 30 characters.
Browser
Chrome
localhost:3000/dashboard
Bug TrackR
Welcome, Ramya Sridevi (tester)
Logout
Report a Bug
View Reported Bugs
View Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Tested Bugs
Reopen Requests
View Analytics
Report a Bug
Bug Title *
Title must be at least 15 characters.
Description *
Description must be at least 30 characters.
Application *
Issue Type *
Steps to Reproduce Issue *
Steps to Reproduce must be at least 30 characters.
Browser
Chrome

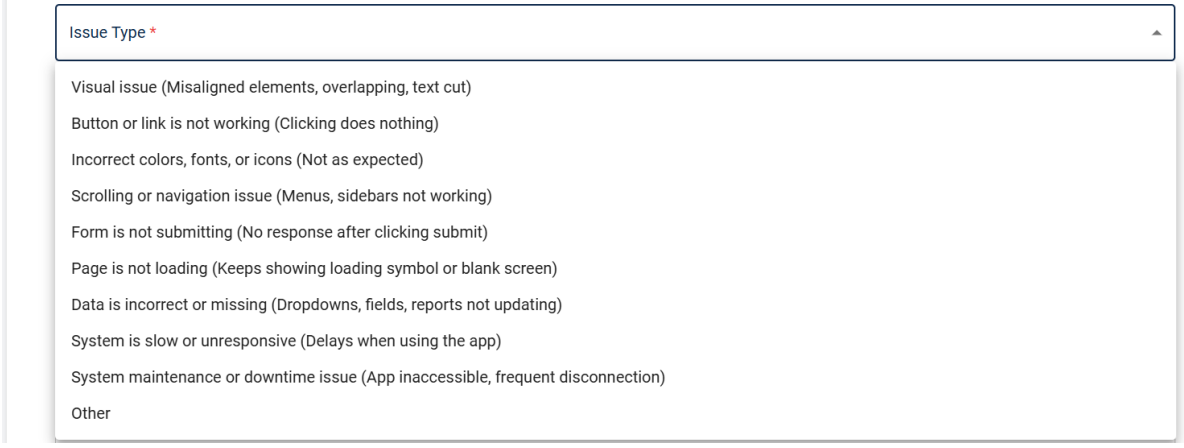

localhost:3000/dashboard
Bug TrackR
Welcome, Rajesh Kumar (teamlead)
Logout
Report a Bug
View Reported Bugs
Bugs Assigned to Team
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
Report a Bug
Bug Title *
Title must be at least 15 characters.
Description *
Description must be at least 30 characters.
Application *
Issue Type *
Steps to Reproduce Issue *
Steps to Reproduce must be at least 30 characters.
Browser
Chrome

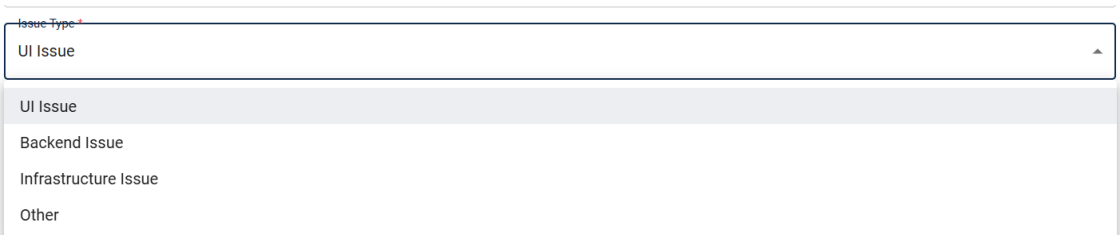

localhost:3000/dashboard
Bug TrackR
Welcome, BugTrackR Admin (admin)
Logout
Report a Bug
View Reported Bugs
View Team Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
Report a Bug
Bug Title *
Title must be at least 15 characters.
Description *
Description must be at least 30 characters.
Application *
Issue Type *
Steps to Reproduce Issue *
Steps to Reproduce must be at least 30 characters.
Browser

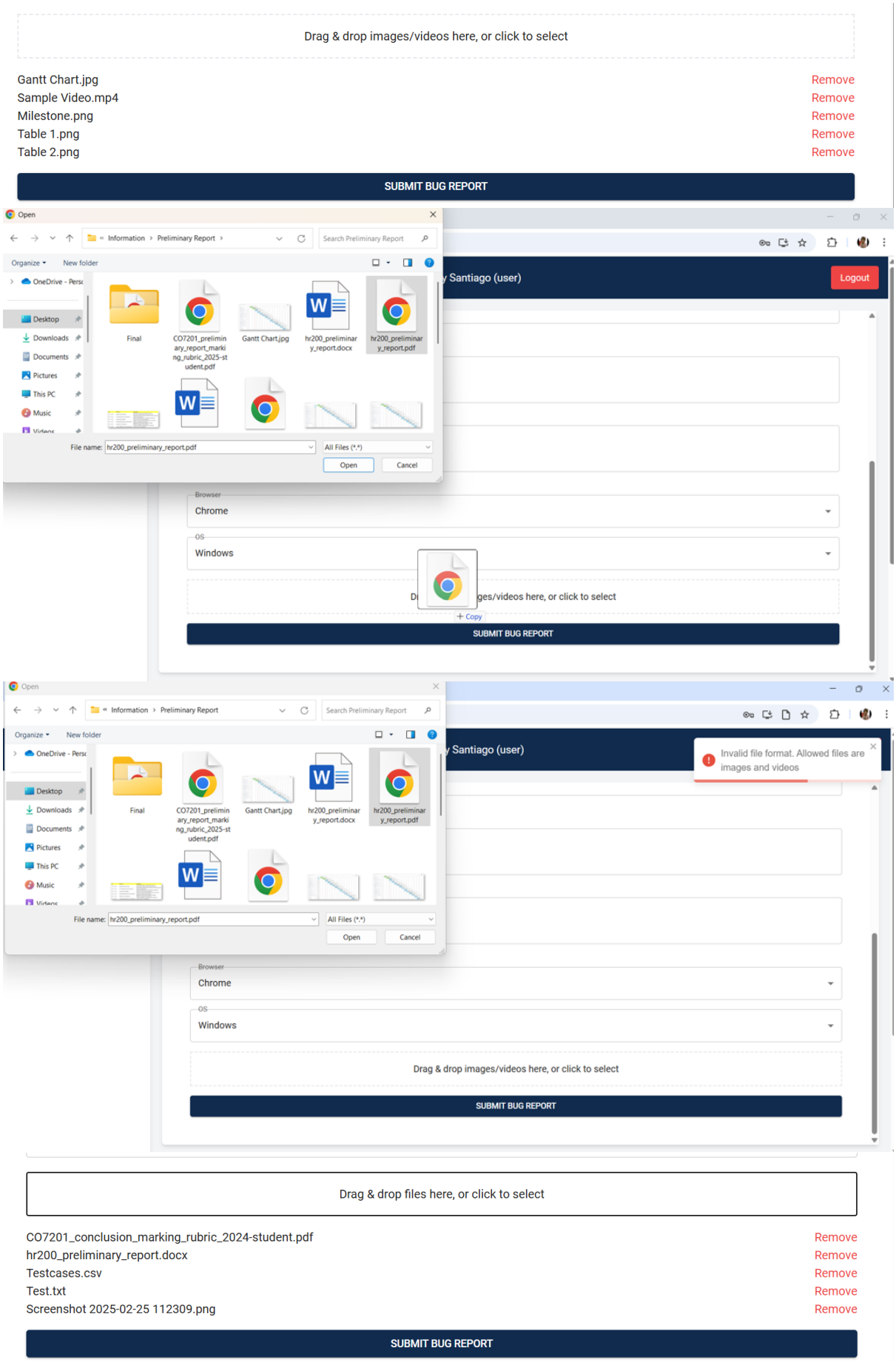
APPENDIX D – Bug Report Form (Validation + All roles view)




Welcome, Lucy Santiago (user)
Please fill all required fields and/or adhere to the standards of the fields.
Report a Bug
Bug Title *
Button
Title must be at least 15 characters.
Description *
The login button on the homepage does not respond when clicked.
Application *
Marketplace (E-commerce App)
Issue Type *
Visual issue (Misaligned elements, overlapping, text cut)
Help Reproduce Issue
What were you trying to do? *
Click the Login button to log into the application.
What happened instead? *
Nothing happens when the "Login" button is clicked


Issue Type *
Visual issue (Misaligned elements, overlapping, text cut)
Button or link is not working (Clicking does nothing)
Incorrect colors, fonts, or icons (Not as expected)
Scrolling or navigation issue (Menus, sidebars not working)
Form is not submitting (No response after clicking submit)
Page is not loading (Keeps showing loading symbol or blank screen)
Data is incorrect or missing (Dropdowns, fields, reports not updating)
System is slow or unresponsive (Delays when using the app)
System maintenance or downtime issue (App inaccessible, frequent disconnection)
Other

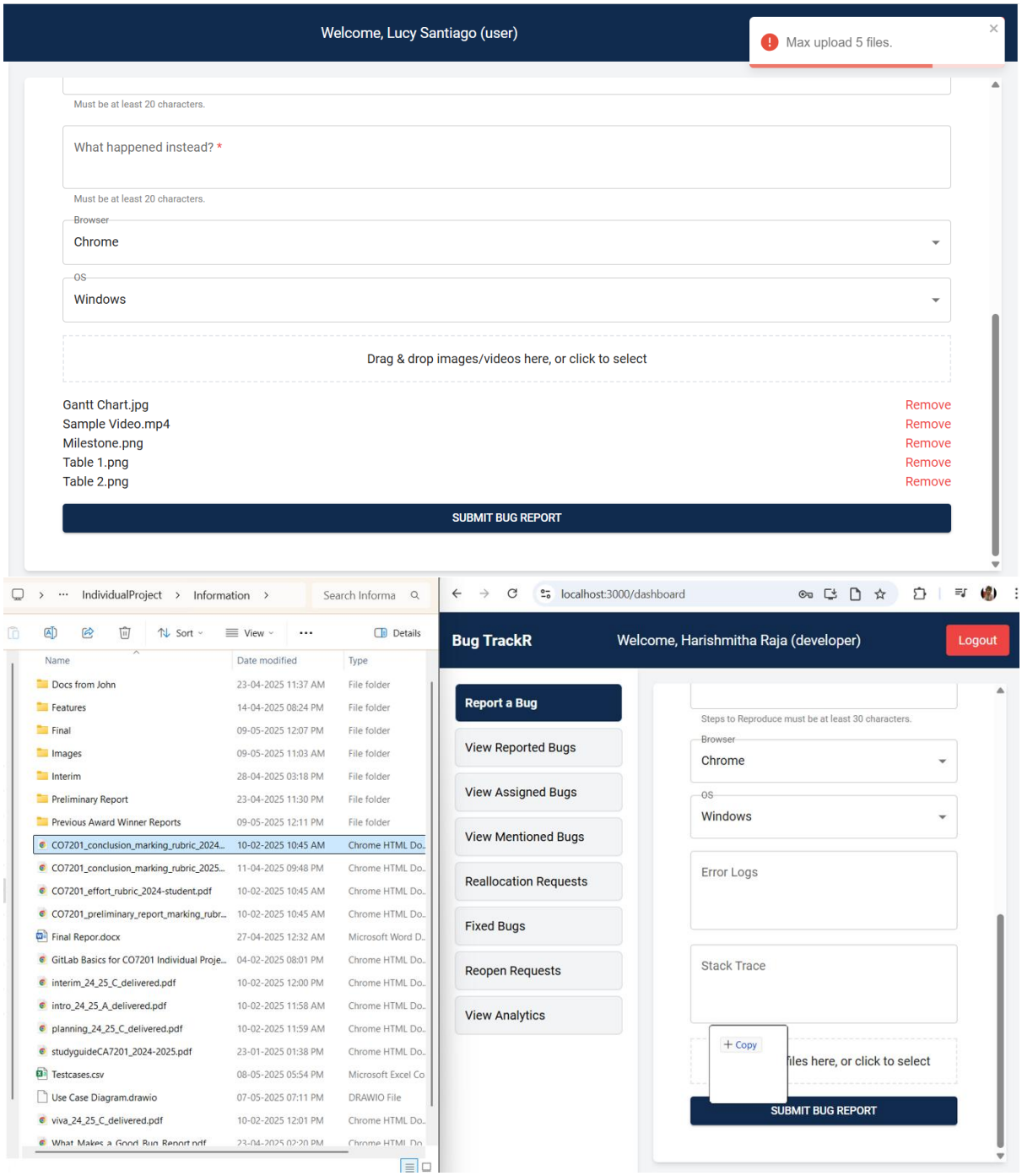

Issue Type *
UI Issue
UI Issue
Backend Issue
Infrastructure Issue
Other

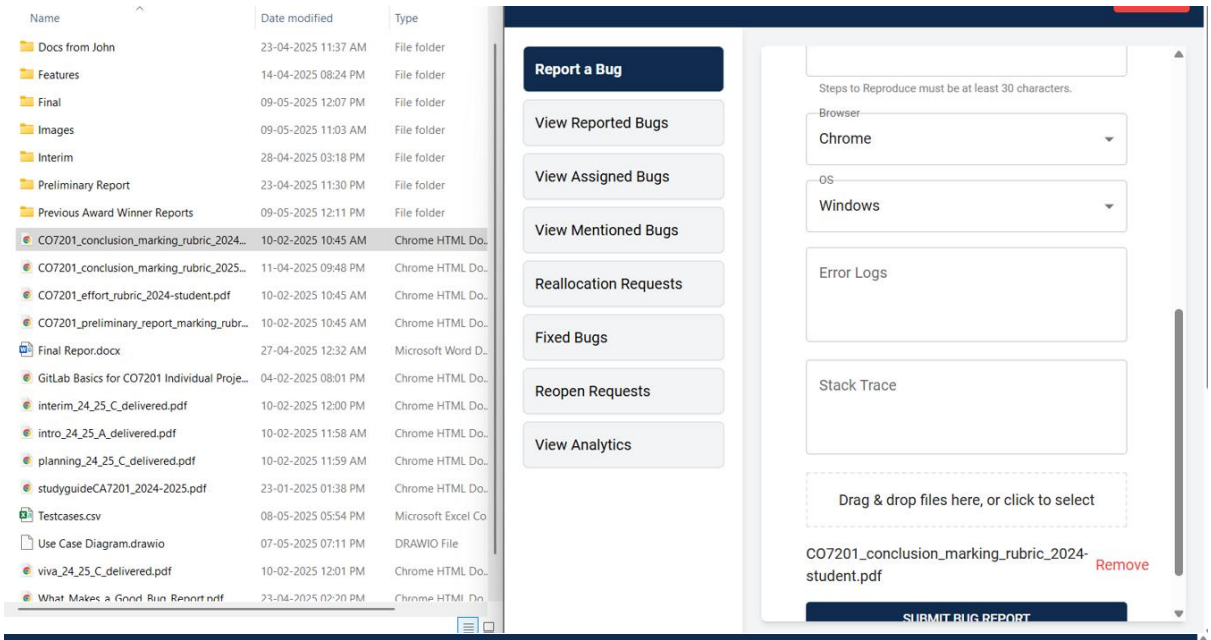


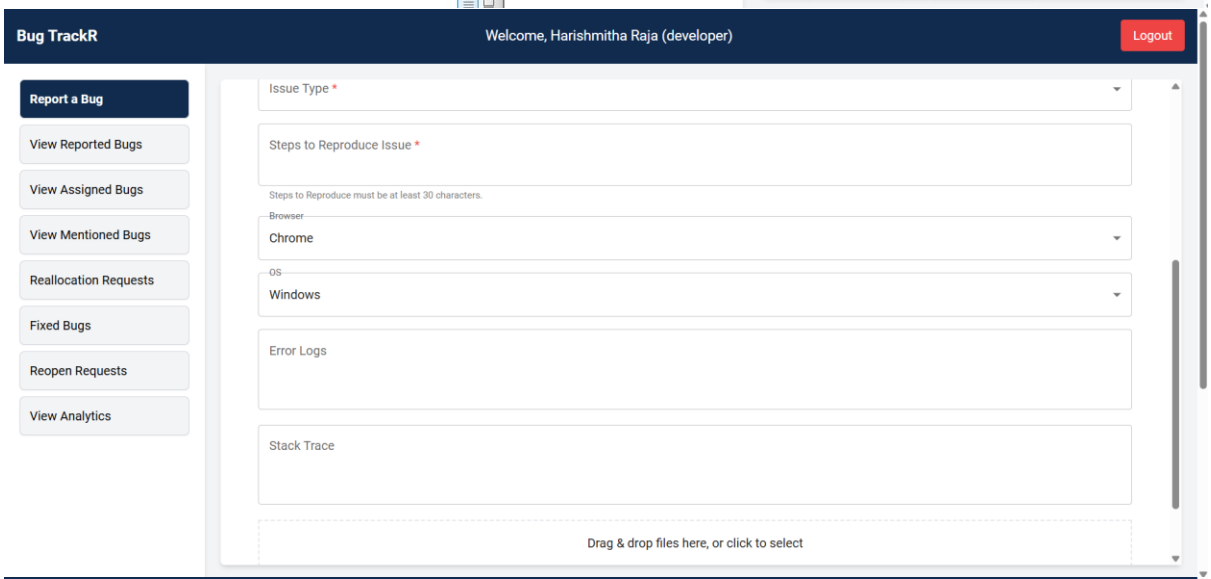
Drag & drop files here, or click to select

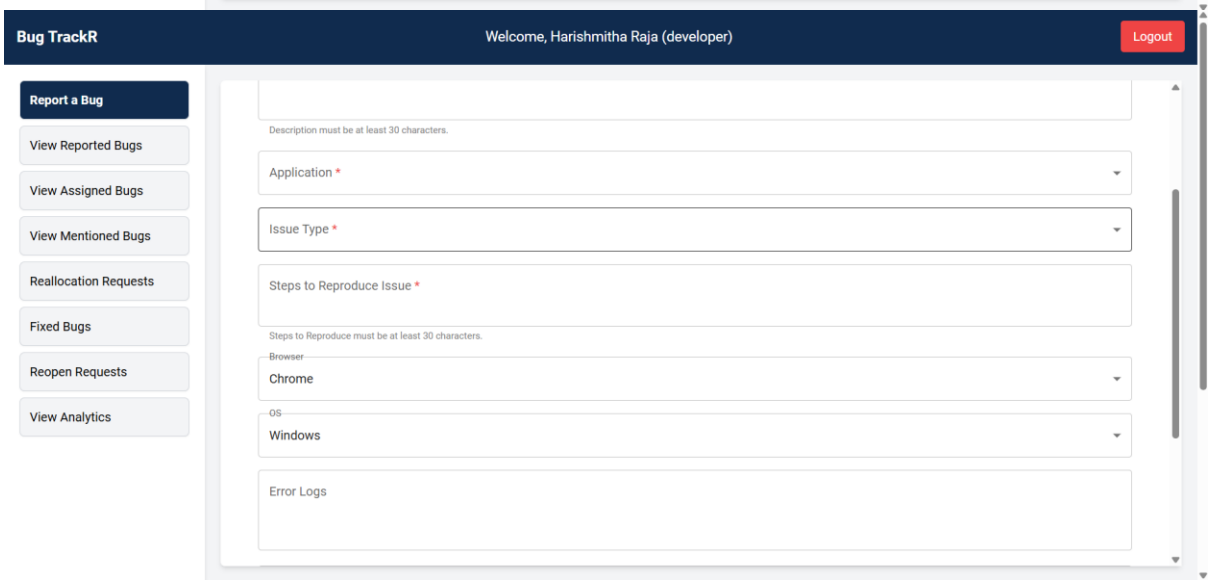
CO7201_conclusion_marking_rubric_2024-student.pdf Remove
hr200_preliminary_report.docx Remove
Testcases.csv Remove
Test.txt Remove
Screenshot 2025-02-25 112309.png Remove

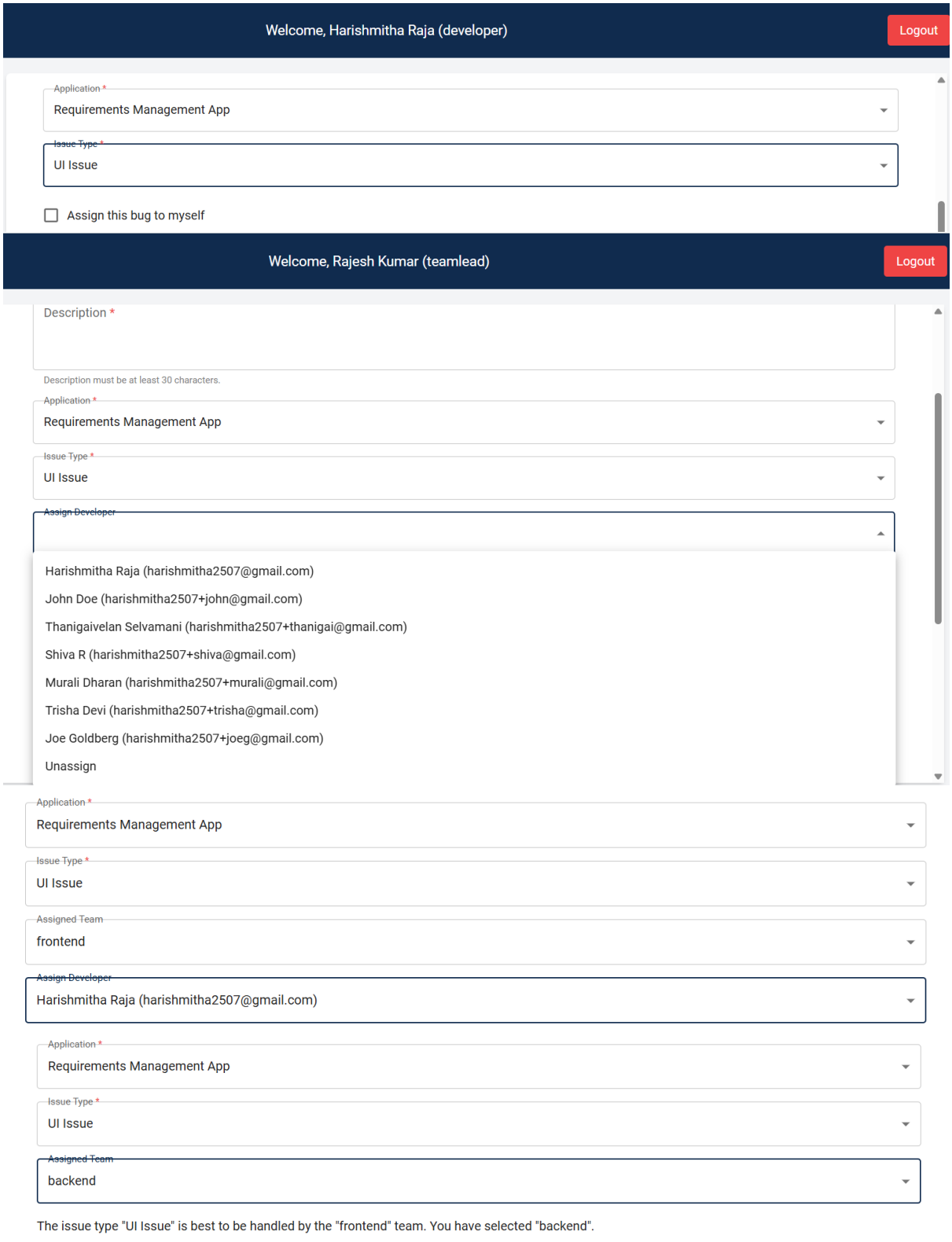
SUBMIT BUG REPORT

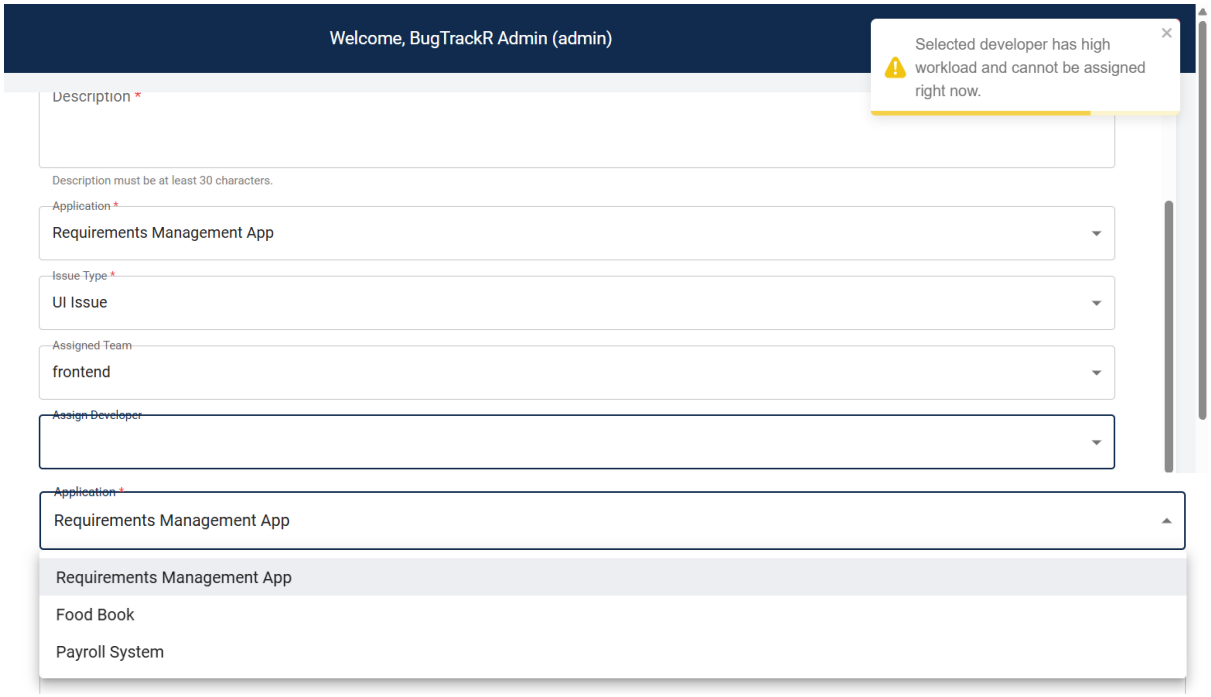

Welcome, Lucy Santiago (user)
Max upload 5 files.
Must be at least 20 characters.
What happened instead? *
Must be at least 20 characters.
Browser
Chrome
OS
Windows
Drag & drop images/videos here, or click to select
Gantt Chart.jpg
Sample Video.mp4
Milestone.png
Table 1.png
Table 2.png
Remove
SUBMIT BUG REPORT
IndividualProject
Information
Search Informa
Sort
View
Details
Name
Date modified
Type
Docs from John
Features
Final
Images
Interim
Preliminary Report
Previous Award Winner Reports
Final Repor.docx
Use Case Diagram.drawio
Testcases.csv
localhost:3000/dashboard
Bug TrackR
Welcome, Harishmitha Raja (developer)
Logout
Report a Bug
View Reported Bugs
View Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Fixed Bugs
Reopen Requests
View Analytics
Steps to Reproduce must be at least 30 characters.
Browser
Chrome
OS
Windows
Error Logs
Stack Trace
Copy
SUBMIT BUG REPORT


Name
Date modified
Type
Docs from John 23-04-2025 11:37 AM File folder
Features 14-04-2025 08:24 PM File folder
Final 09-05-2025 12:07 PM File folder
Images 09-05-2025 11:03 AM File folder
Interim 28-04-2025 03:18 PM File folder
Preliminary Report 23-04-2025 11:30 PM File folder
Previous Award Winner Reports 09-05-2025 12:11 PM File folder
CO7201_conclusion_marking_rubric_2024... 10-02-2025 10:45 AM Chrome HTML Do...
CO7201_conclusion_marking_rubric_2025... 11-04-2025 09:48 PM Chrome HTML Do...
CO7201_effort_rubric_2024-student.pdf 10-02-2025 10:45 AM Chrome HTML Do...
CO7201_preliminary_report_marking_rubr... 10-02-2025 10:45 AM Chrome HTML Do...
Final Repor.docx 27-04-2025 12:32 AM Microsoft Word D...
GitLab Basics for CO7201 Individual Proje... 04-02-2025 08:01 PM Chrome HTML Do...
interim_24_25_C_delivered.pdf 10-02-2025 12:00 PM Chrome HTML Do...
intro_24_25_A_delivered.pdf 10-02-2025 11:58 AM Chrome HTML Do...
planning_24_25_C_delivered.pdf 10-02-2025 11:59 AM Chrome HTML Do...
studyguideCA7201_2024-2025.pdf 23-01-2025 01:38 PM Chrome HTML Do...
Testcases.csv 08-05-2025 05:54 PM Microsoft Excel Co
Use Case Diagram.drawio 07-05-2025 07:11 PM DRAWIO File
viva_24_25_C_delivered.pdf 10-02-2025 12:01 PM Chrome HTML Do...
Report a Bug
View Reported Bugs
View Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Fixed Bugs
Reopen Requests
View Analytics
Steps to Reproduce must be at least 30 characters.
Browser
Chrome
OS
Windows
Error Logs
Stack Trace
Drag & drop files here, or click to select
CO7201_conclusion_marking_rubric_2024-student.pdf
Remove
SUBMIT BUG REPORT

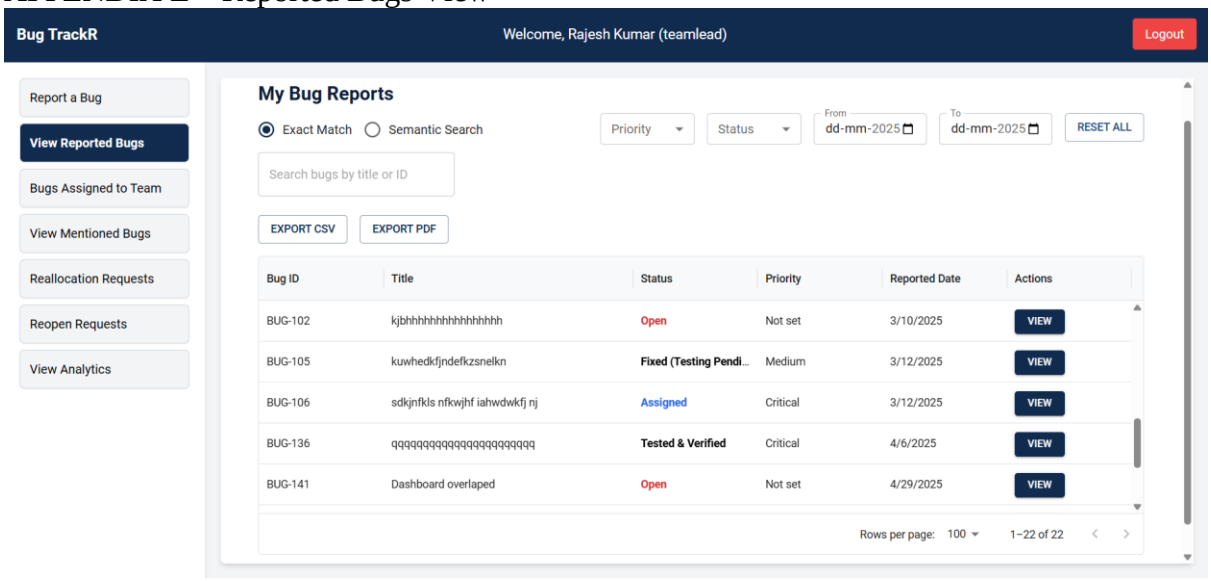

Bug TrackR
Welcome, Harishmitha Raja (developer)
Logout
Report a Bug
View Reported Bugs
View Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Fixed Bugs
Reopen Requests
View Analytics
Issue Type *
Steps to Reproduce Issue *
Steps to Reproduce must be at least 30 characters.
Browser
Chrome
OS
Windows
Error Logs
Stack Trace
Drag & drop files here, or click to select


Bug TrackR
Welcome, Harishmitha Raja (developer)
Logout
Report a Bug
View Reported Bugs
View Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Fixed Bugs
Reopen Requests
View Analytics
Description must be at least 30 characters.
Application *
Issue Type *
Steps to Reproduce Issue *
Steps to Reproduce must be at least 30 characters.
Browser
Chrome
OS
Windows
Error Logs

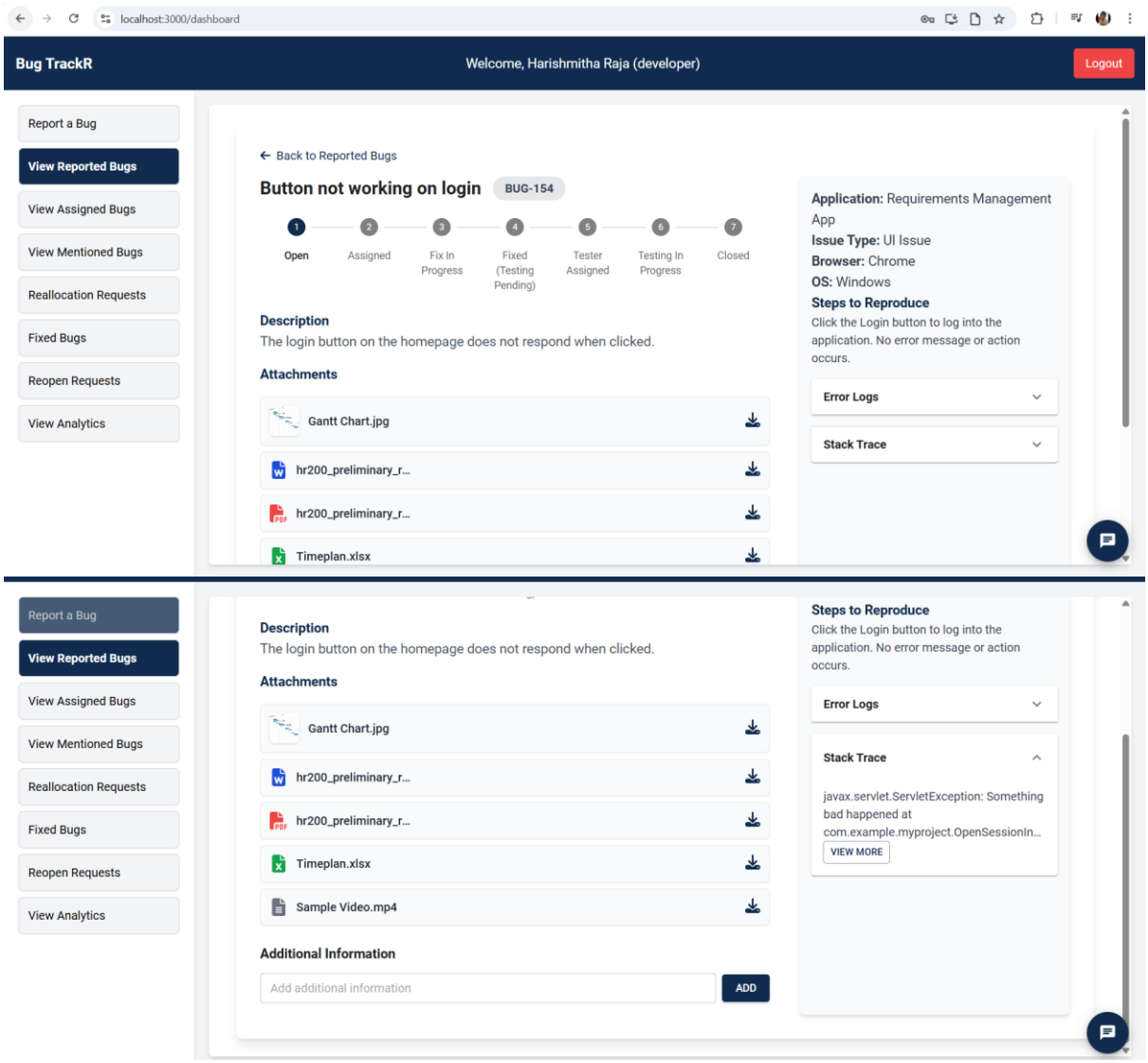

Welcome, Harishmitha Raja (developer)
Logout
Application *
Requirements Management App
Issue Type *
UI Issue
Assign this bug to myself
Welcome, Rajesh Kumar (teamlead)
Logout
Description *
Description must be at least 30 characters.
Application *
Requirements Management App
Issue Type *
UI Issue
Assign Developer
Harishmitha Raja (harishmitha2507@gmail.com)
John Doe (harishmitha2507+john@gmail.com)
Thanigaivelan Selvamani (harishmitha2507+thanigai@gmail.com)
Shiva R (harishmitha2507+shiva@gmail.com)
Murali Dharan (harishmitha2507+murali@gmail.com)
Trisha Devi (harishmitha2507+trisha@gmail.com)
Joe Goldberg (harishmitha2507+joeg@gmail.com)
Unassign
Application *
Requirements Management App
Issue Type *
UI Issue
Assigned Team
frontend
Assign Developer
Harishmitha Raja (harishmitha2507@gmail.com)
Application *
Requirements Management App
Issue Type *
UI Issue
Assigned Team
backend
The issue type "UI Issue" is best to be handled by the "frontend" team. You have selected "backend".

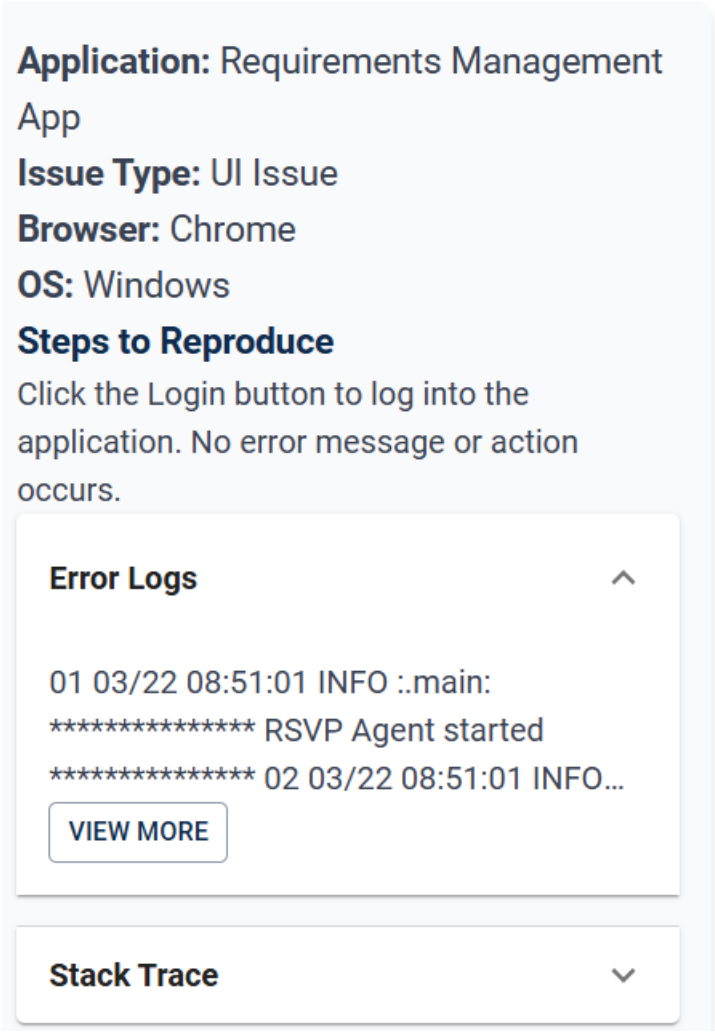


APPENDIX E – Reported Bugs View

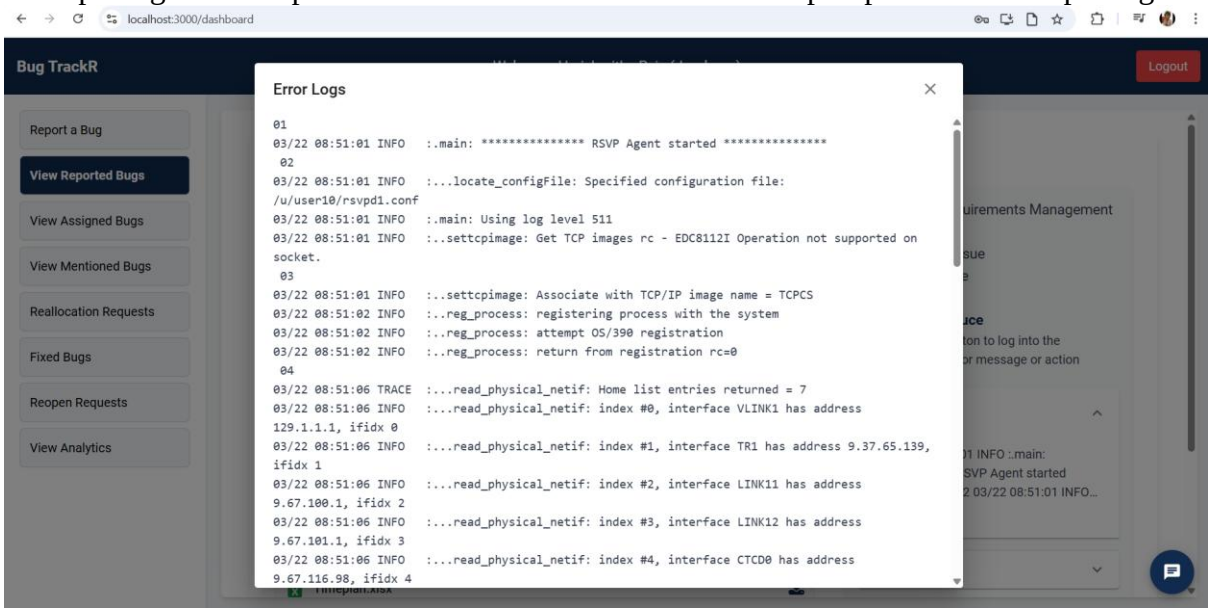


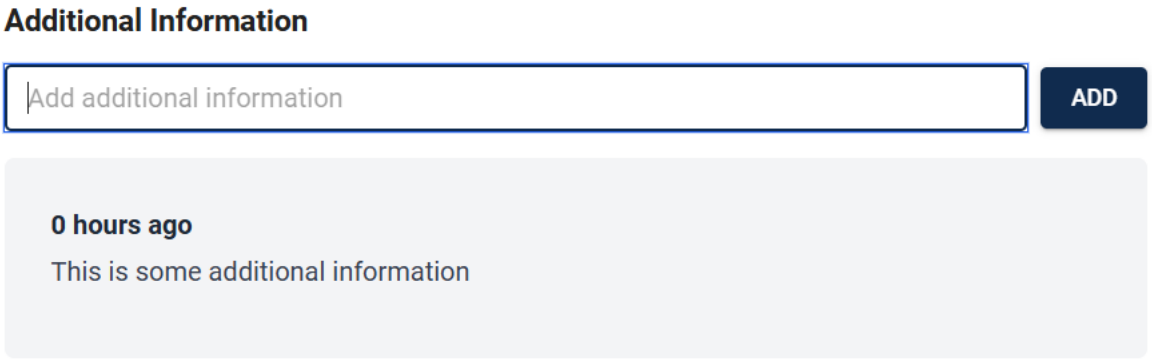
APPENDIX F – Reported Bug Details


localhost:3000/dashboard
Bug TrackR
Welcome, Harishmitha Raja (developer)
Logout
Report a Bug
View Reported Bugs
View Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Fixed Bugs
Reopen Requests
View Analytics
Back to Reported Bugs
Button not working on login
BUG-154
Open
Assigned
Fix In Progress
Fixed (Testing Pending)
Tester Assigned
Testing In Progress
Closed
Application: Requirements Management App
Issue Type: UI Issue
Browser: Chrome
OS: Windows
Steps to Reproduce
Click the Login button to log into the application. No error message or action occurs.
Error Logs
Stack Trace
Description
The login button on the homepage does not respond when clicked.
Attachments
Gantt Chart.jpg
hr200_preliminary_r...
hr200_preliminary_r...
Timeplan.xlsx
Sample Video.mp4
javax.servlet.ServletException: Something bad happened at com.example.myproject.OpenSessionIn...
VIEW MORE
Additional Information
Add additional information
ADD

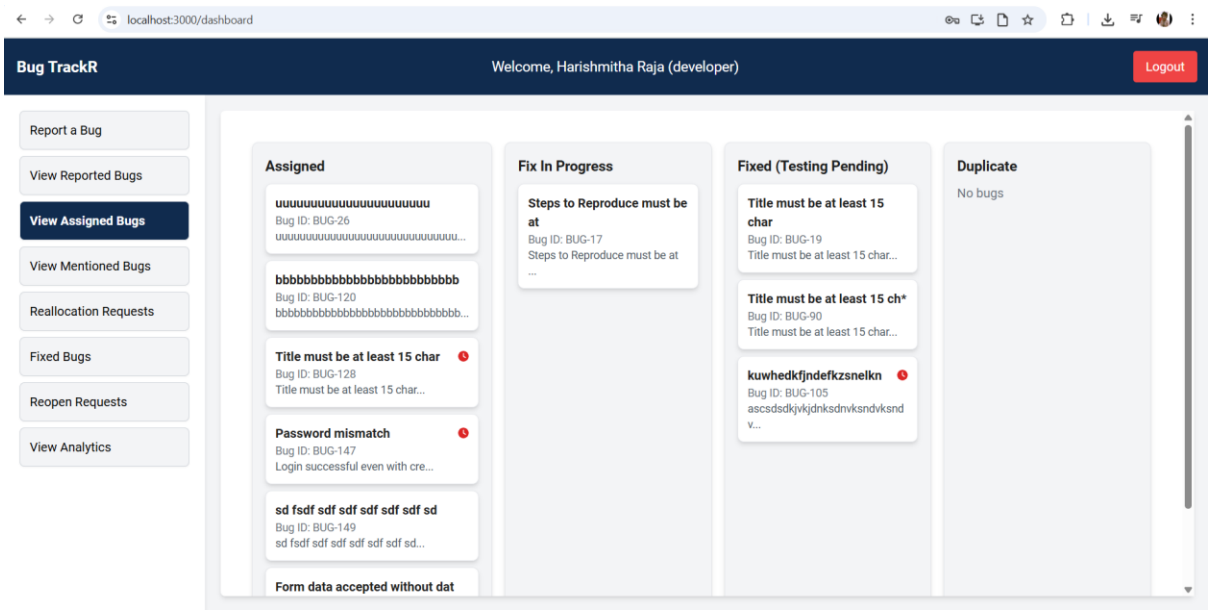


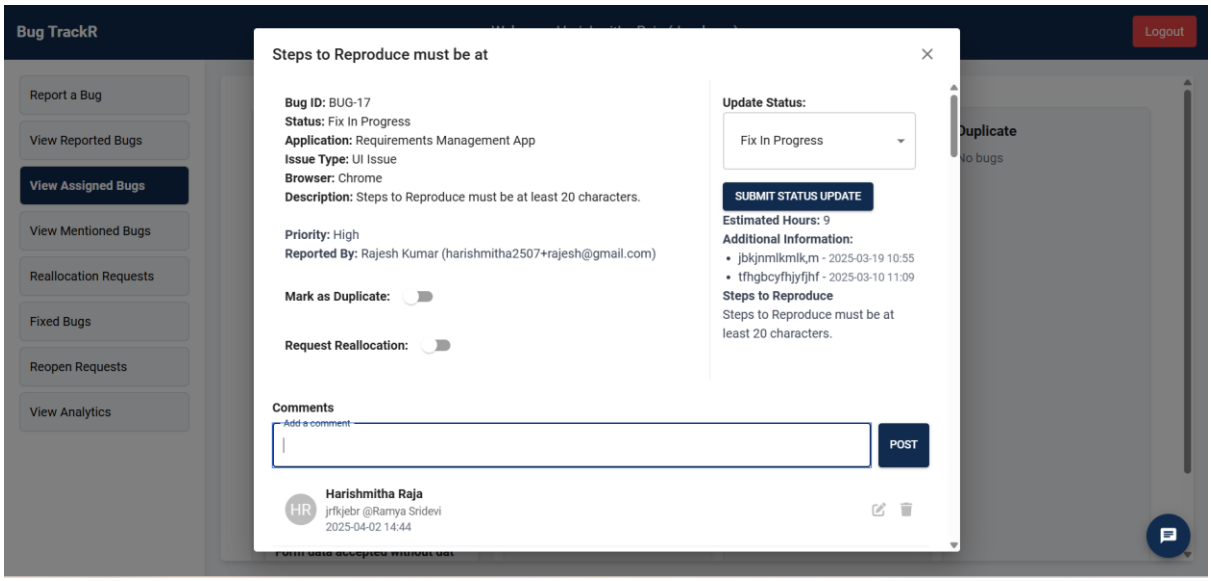
Example log from : https://www.ibm.com/docs/en/zos/2.4.0?topic=problems-example-log-file



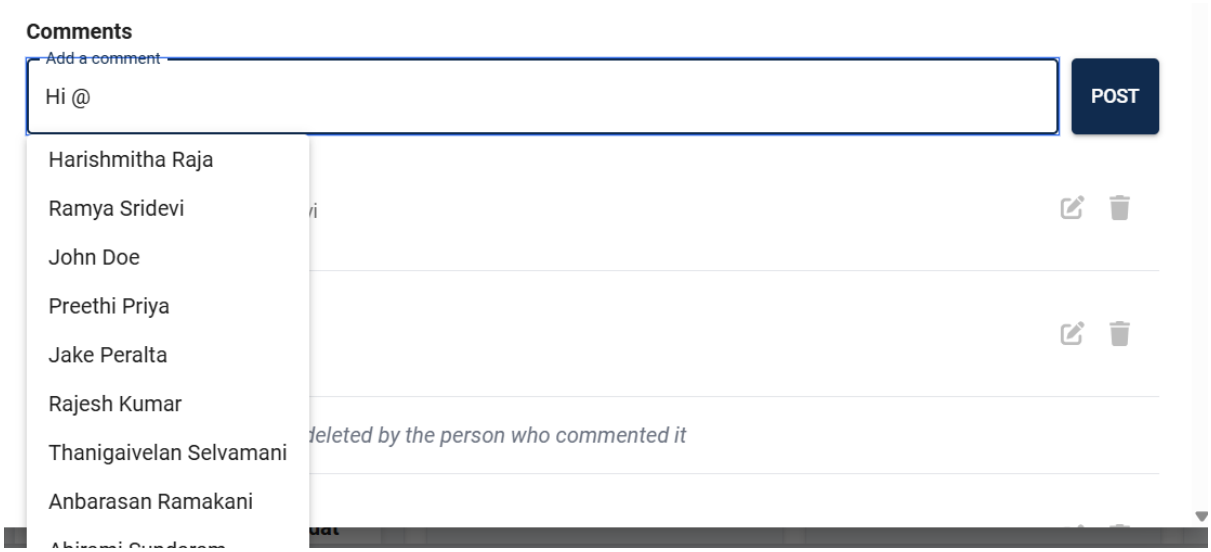


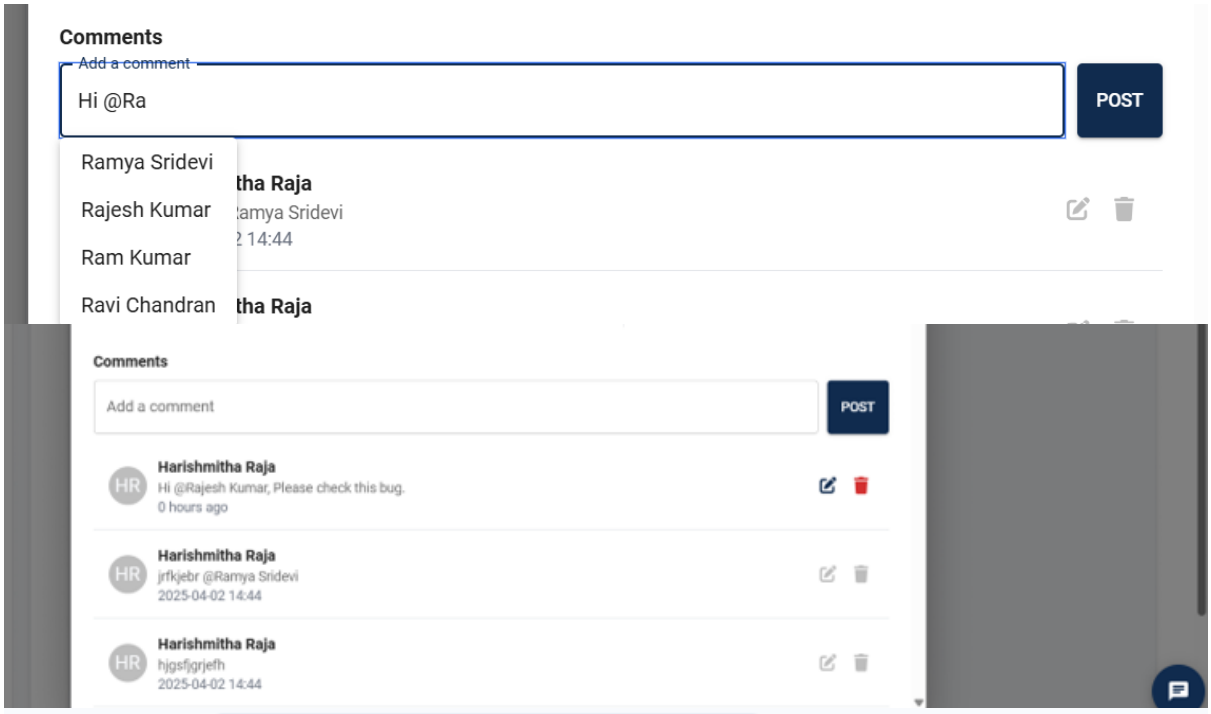
APPENDIX G – Assigned Bugs View (Developer/Tester – similar)

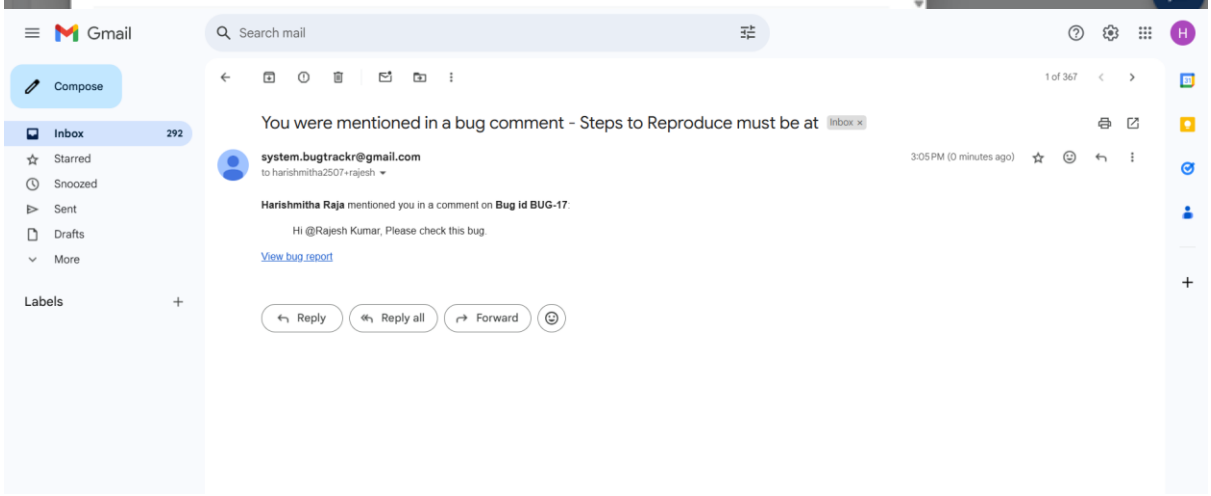


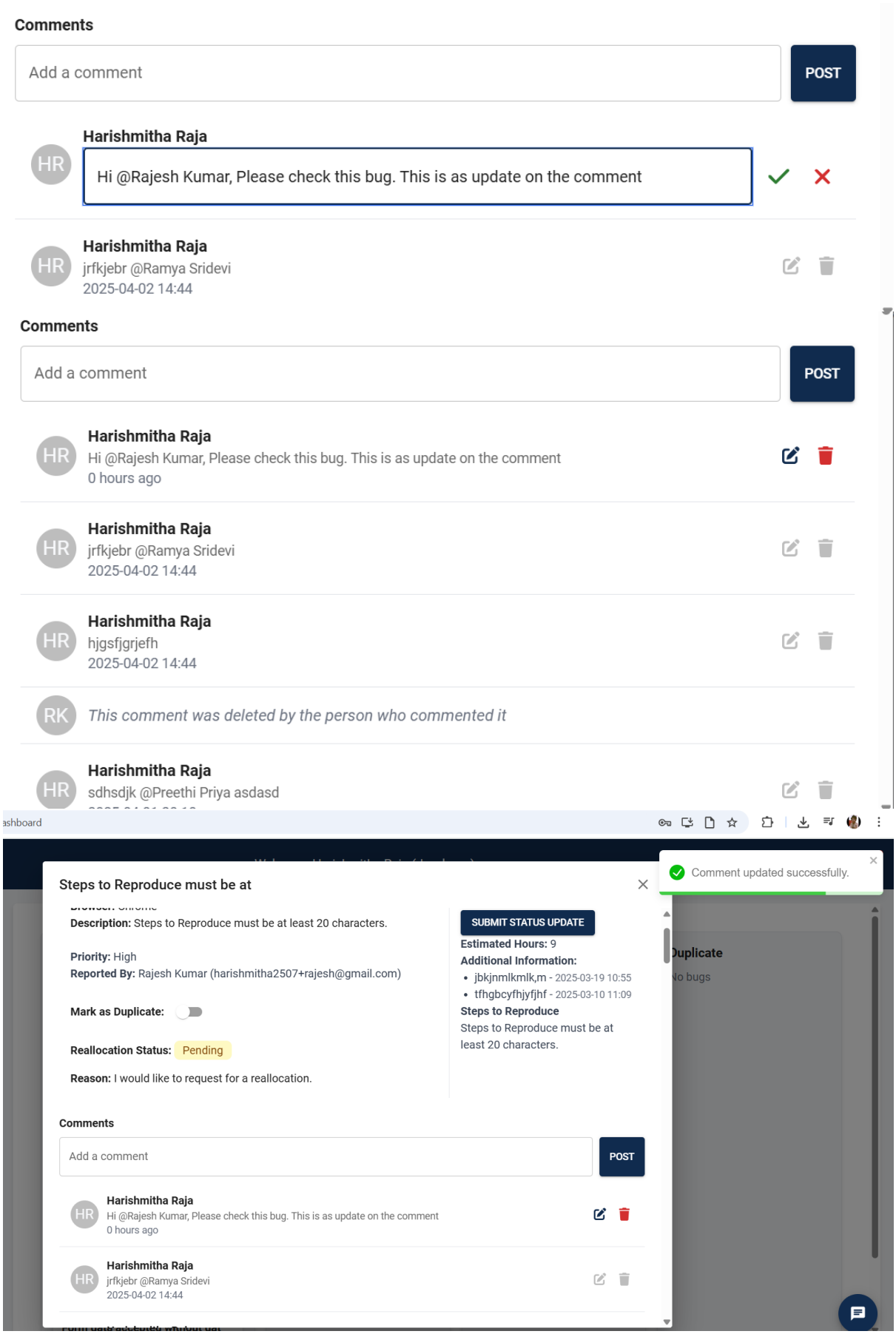


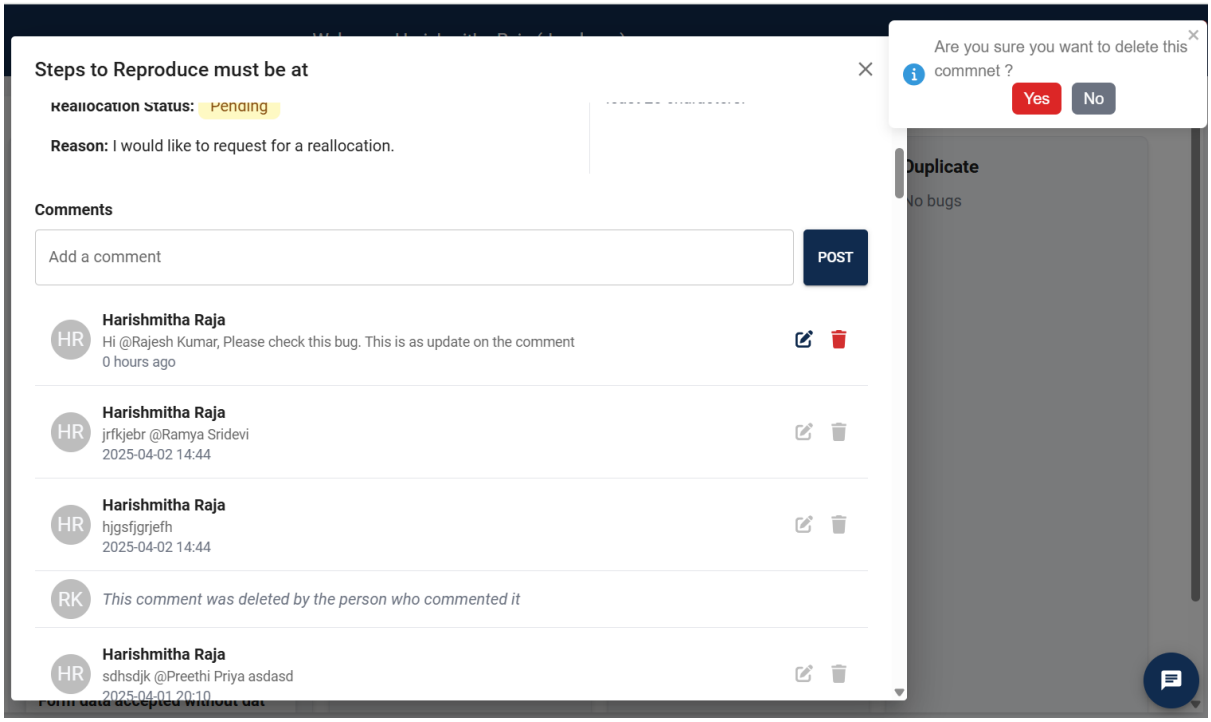
APPENDIX H – Comments

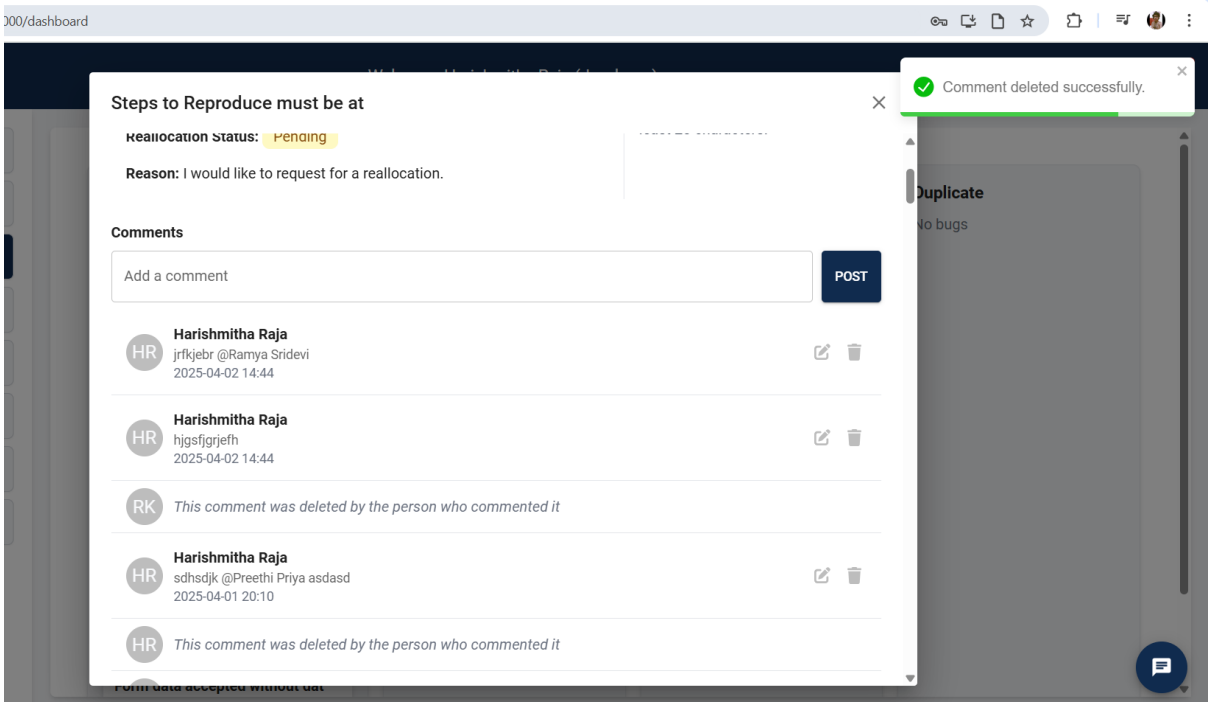


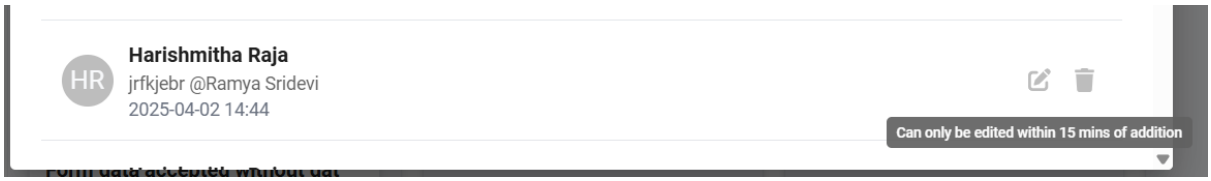


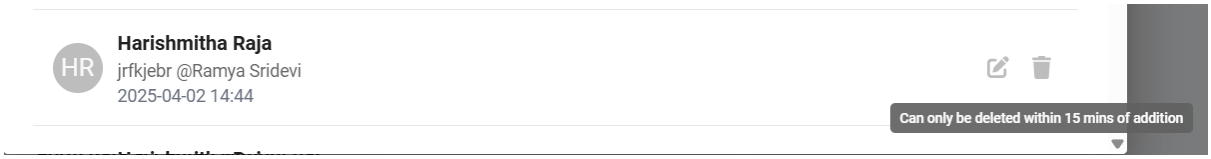



Comments
Add a comment
POST
Harishmitha Raja
Hi @Rajesh Kumar, Please check this bug. This is as update on the comment
Harishmitha Raja
jrfkjebr @Ramya Sridevi
2025-04-02 14:44
Comments
Add a comment
POST
Harishmitha Raja
Hi @Rajesh Kumar, Please check this bug. This is as update on the comment
0 hours ago
Harishmitha Raja
jrfkjebr @Ramya Sridevi
2025-04-02 14:44
Harishmitha Raja
hjgsfjgrjefh
2025-04-02 14:44
This comment was deleted by the person who commented it
Harishmitha Raja
sdhsdjk @Preethi Priya asdasd
Steps to Reproduce must be at
Description: Steps to Reproduce must be at least 20 characters.
Priority: High
Reported By: Rajesh Kumar (harishmitha2507+rajesh@gmail.com)
Mark as Duplicate:
Reallocation Status: Pending
Reason: I would like to request for a reallocation.
SUBMIT STATUS UPDATE
Estimated Hours: 9
Additional Information:
jbkjnmlkmlk,m - 2025-03-19 10:55
tfhgbcyfhjyfjhf - 2025-03-10 11:09
Steps to Reproduce
Steps to Reproduce must be at least 20 characters.
Comment updated successfully.
Comments
Add a comment
POST
Harishmitha Raja
Hi @Rajesh Kumar, Please check this bug. This is as update on the comment
0 hours ago
Harishmitha Raja
jrfkjebr @Ramya Sridevi
2025-04-02 14:44

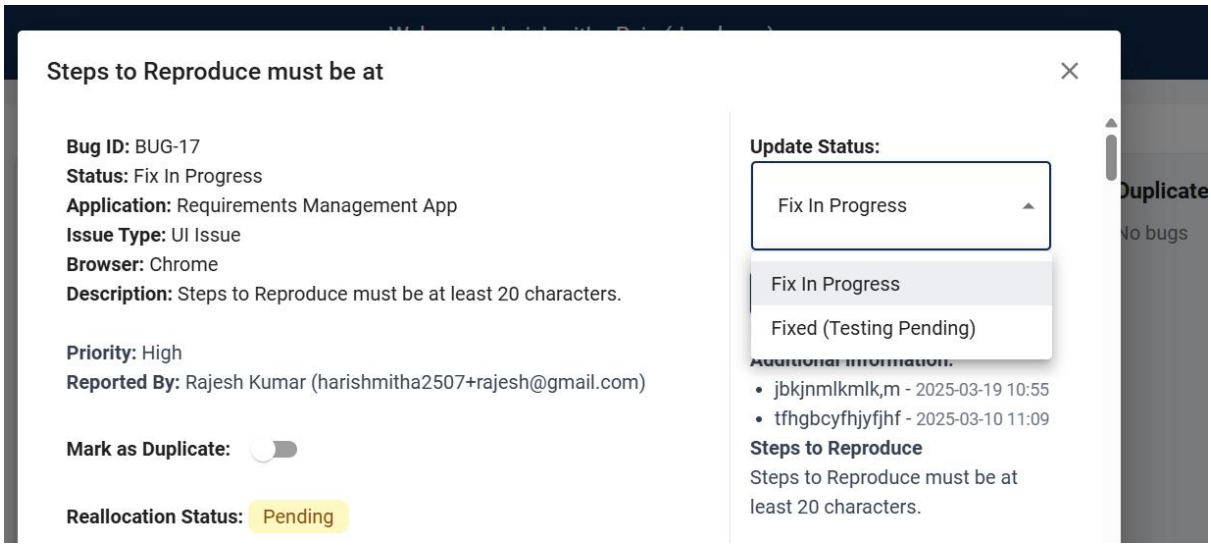

Steps to Reproduce must be at
Reallocation Status: Pending
Reason: I would like to request for a reallocation.
Comments
Add a comment
POST
Harishmitha Raja
Hi @Rajesh Kumar, Please check this bug. This is as update on the comment
0 hours ago
Harishmitha Raja
jrfkjebr @Ramya Sridevi
2025-04-02 14:44
Harishmitha Raja
hjgsfjgrjefh
2025-04-02 14:44
This comment was deleted by the person who commented it
Harishmitha Raja
sdhsdjk @Preethi Priya asdasd
2025-04-01 20:10
Are you sure you want to delete this commnet ?
Yes
No
Duplicate
No bugs

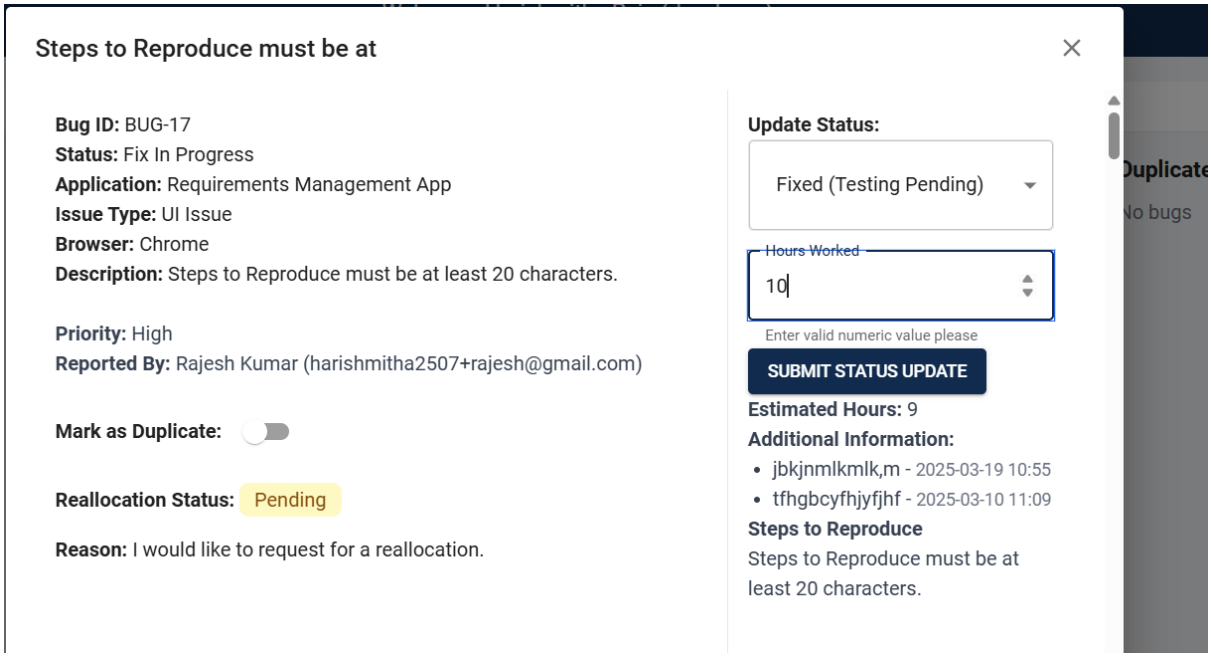

000/dashboard
Steps to Reproduce must be at
Reallocation Status: Pending
Reason: I would like to request for a reallocation.
Comments
Add a comment
POST
Harishmitha Raja
jrfkjebr @Ramya Sridevi
2025-04-02 14:44
Harishmitha Raja
hjgsfjgrjefh
2025-04-02 14:44
This comment was deleted by the person who commented it
Harishmitha Raja
sdhsdjk @Preethi Priya asdasd
2025-04-01 20:10
This comment was deleted by the person who commented it
Comment deleted successfully.
Duplicate
No bugs


Harishmitha Raja
jrfkjebr @Ramya Sridevi
2025-04-02 14:44
Can only be edited within 15 mins of addition

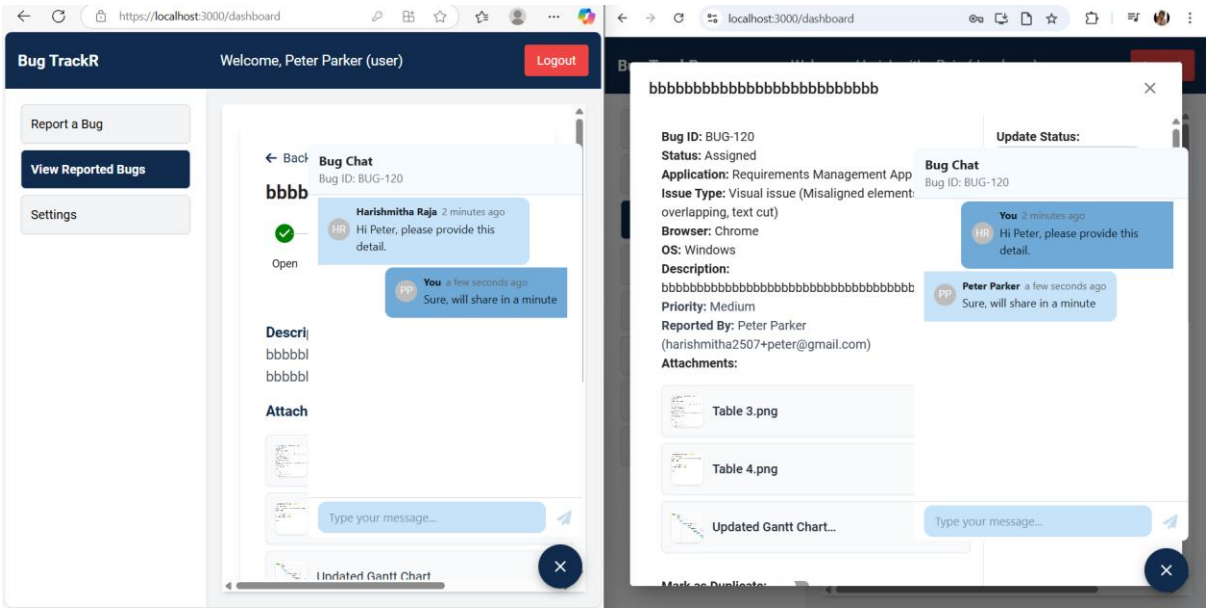


APPENDIX I – Assigned Bug Details View - Status Update(Developer/Tester – similar)

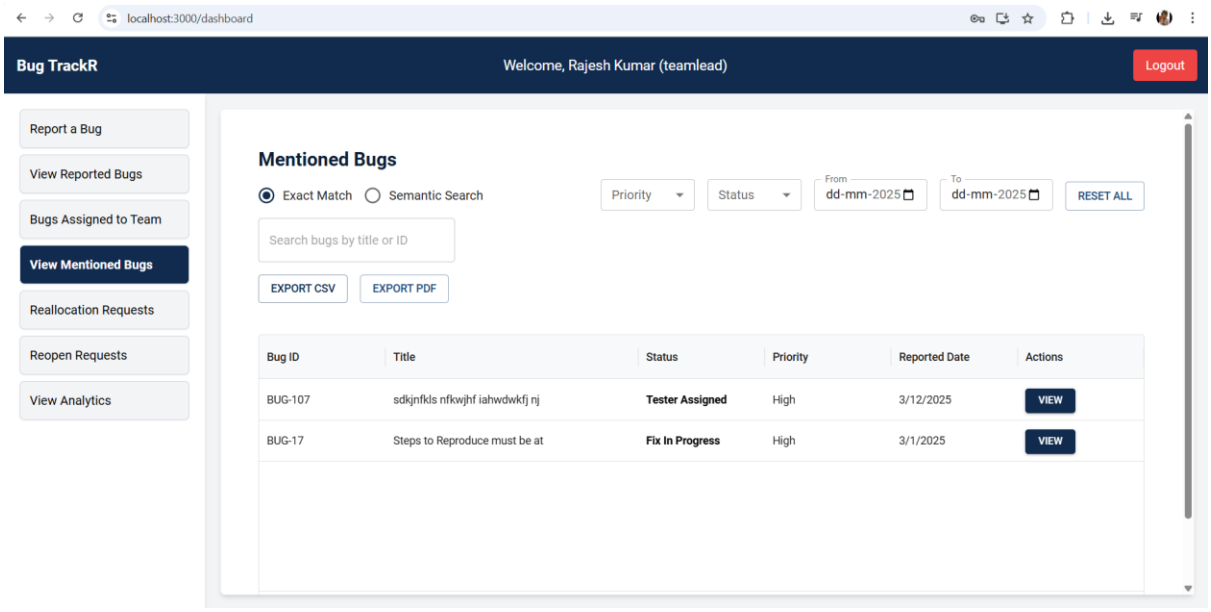




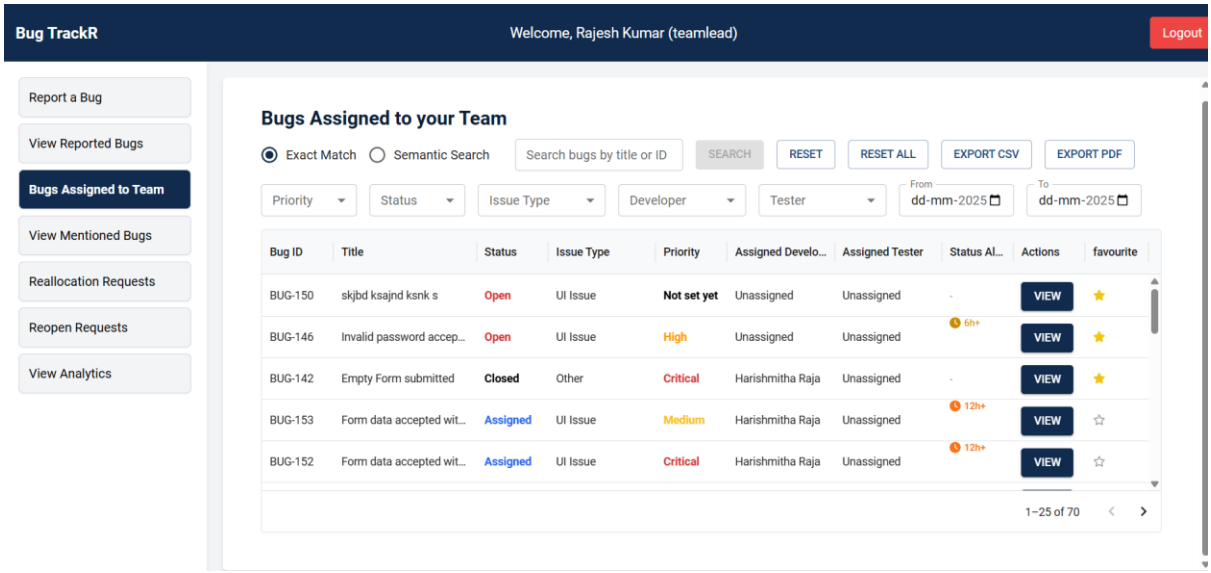
APPENDIX J – Chat between tech users



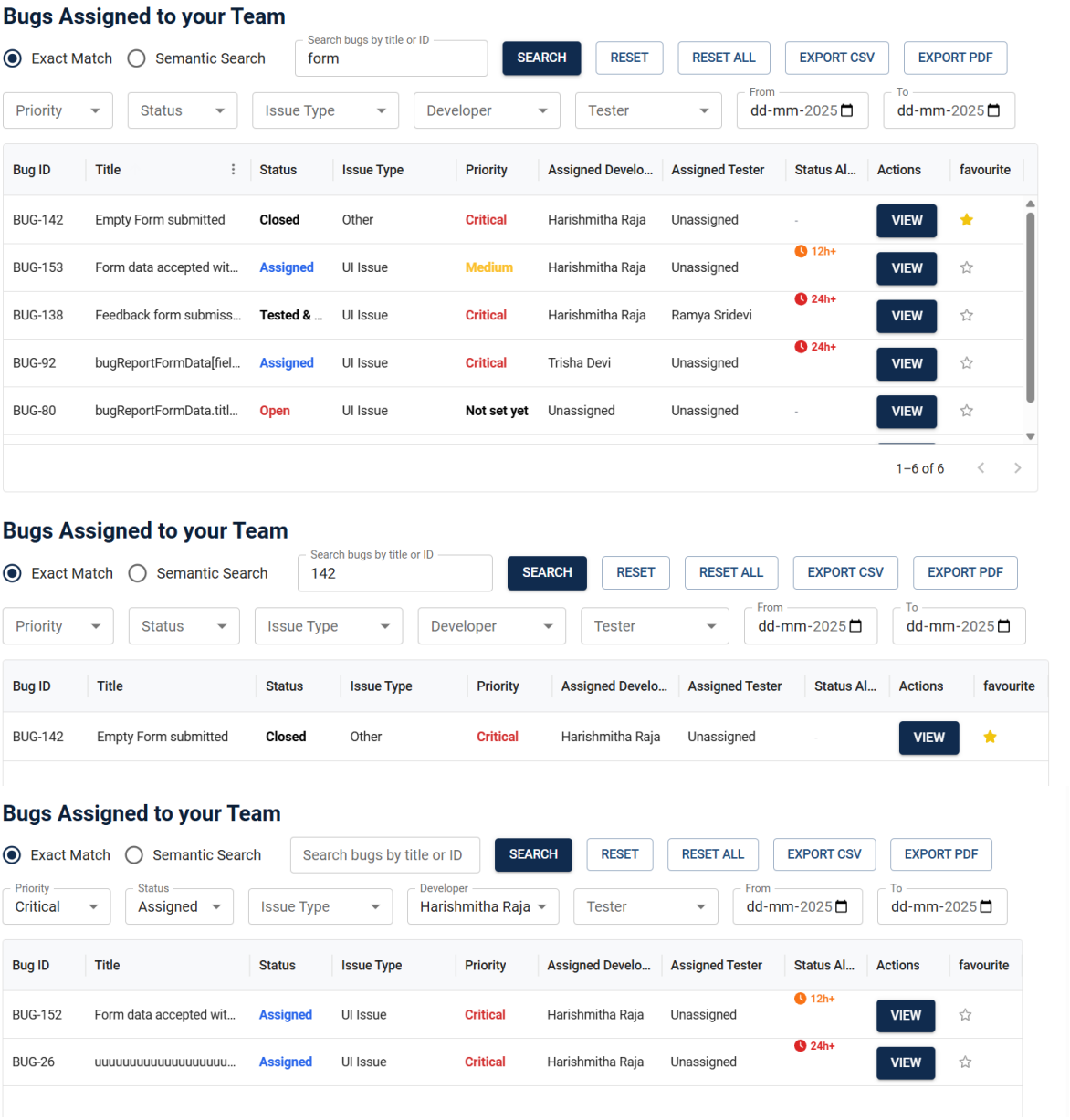
APPENDIX K – Mentioned Bugs

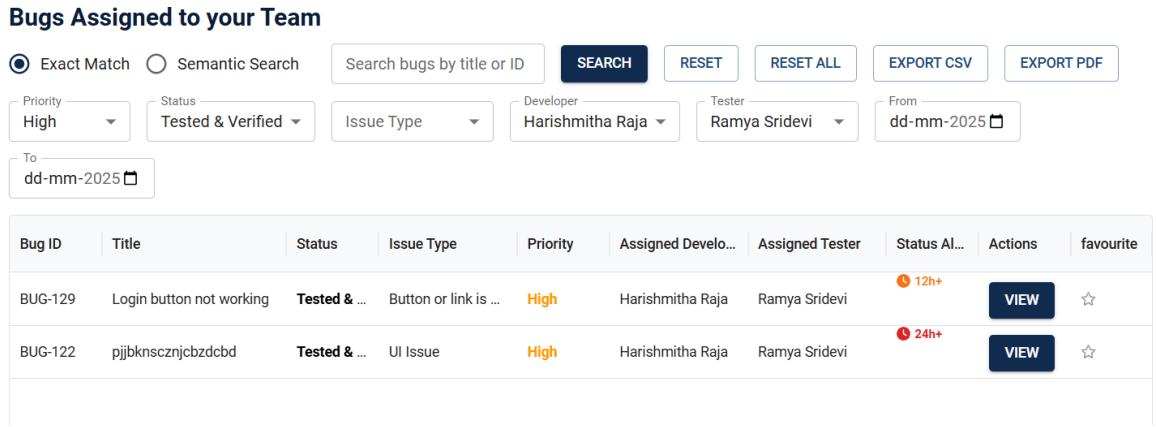


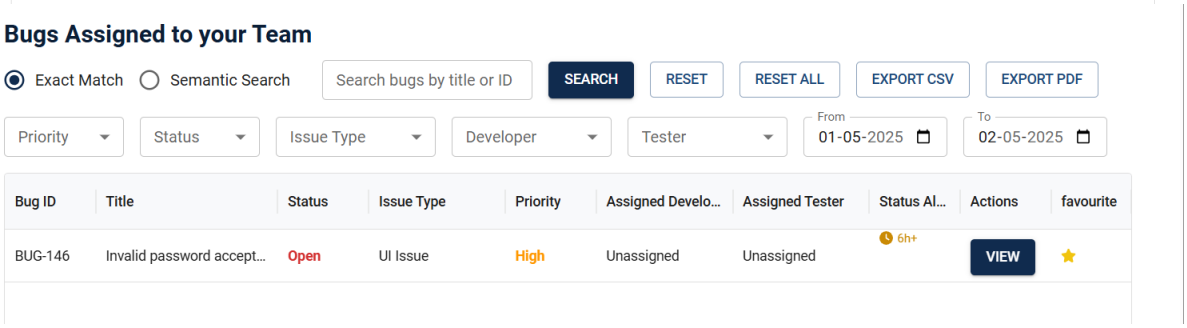
APPENDIX L – Team Lead Team Assigned Bugs List + Favourites (similarly for admin)

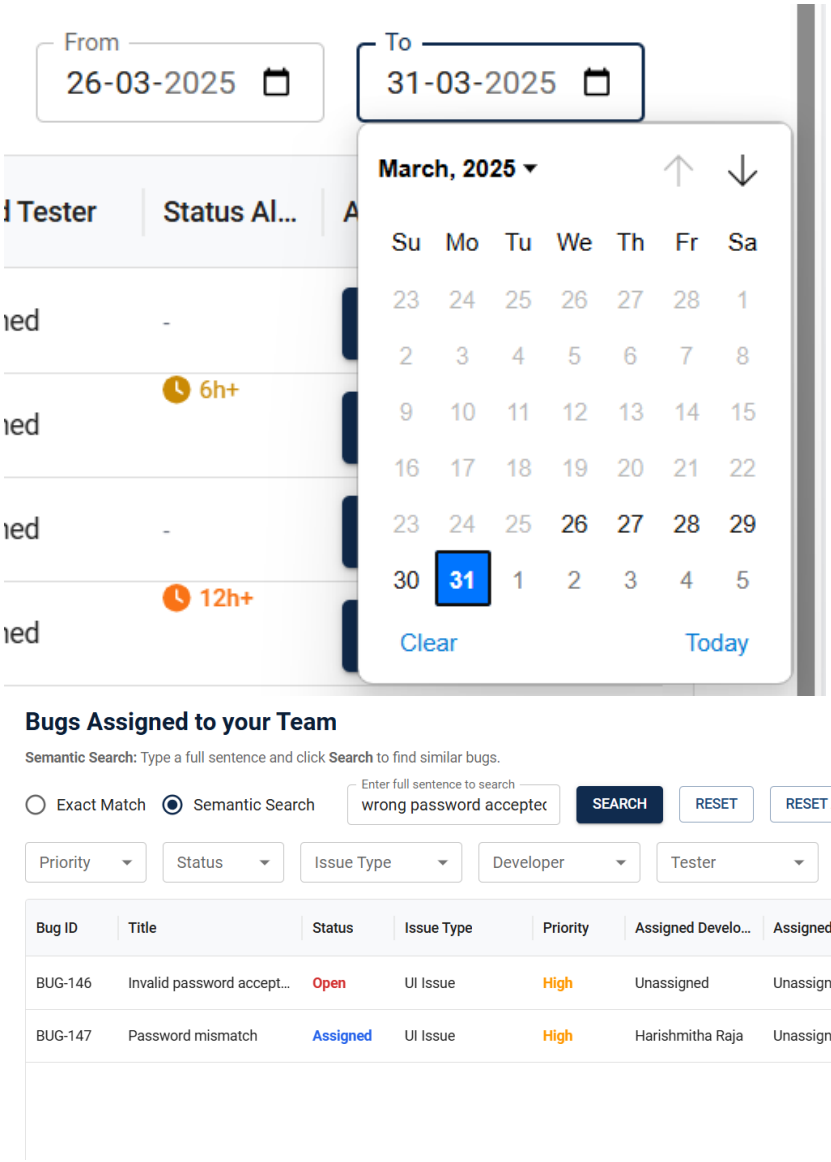


APPENDIX M – Team Lead Team Assigned Bugs List Advanced Search and Filtering (similarly for admin)



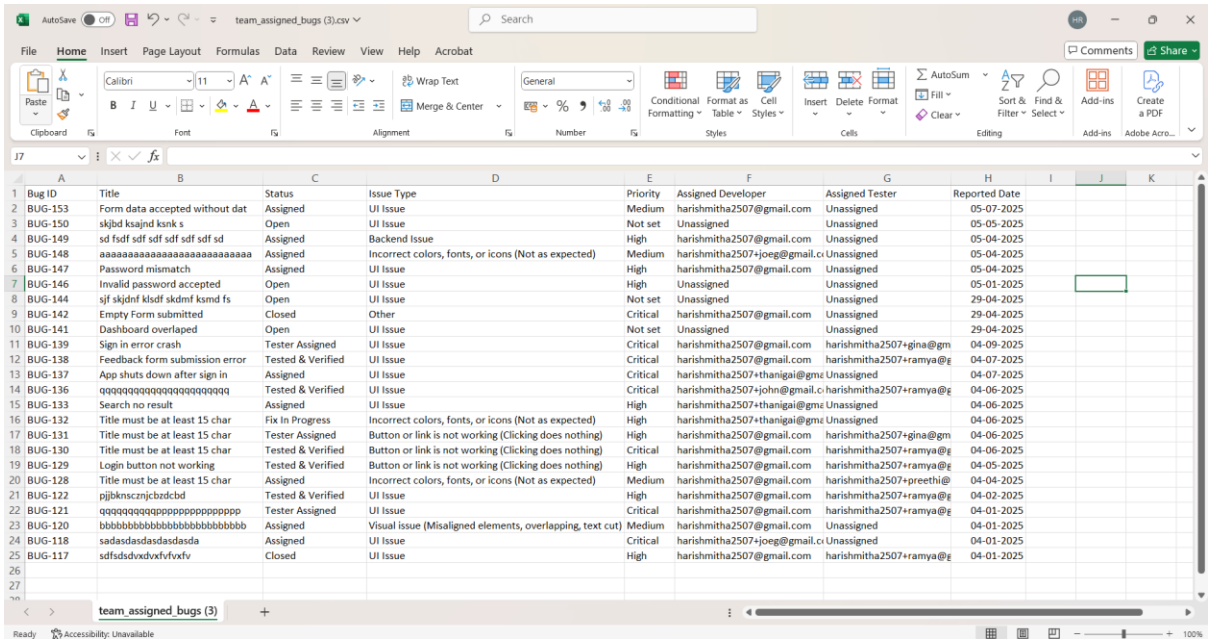

Bugs Assigned to your Team
Exact Match
Semantic Search
Search bugs by title or ID
SEARCH
RESET
RESET ALL
EXPORT CSV
EXPORT PDF
Priority
High
Status
Tested & Verified
Issue Type
Developer
Harishmitha Raja
Tester
Ramya Sridevi
From
dd-mm-2025
To
dd-mm-2025
Bug ID
Title
Status
Issue Type
Priority
Assigned Develo...
Assigned Tester
Status Al...
Actions
favourite
BUG-129
Login button not working
Tested & ...
Button or link is ...
High
Harishmitha Raja
Ramya Sridevi
12h+
VIEW
BUG-122
pjjbknscznjcbzdcbd
Tested & ...
UI Issue
High
Harishmitha Raja
Ramya Sridevi
24h+
VIEW


Bugs Assigned to your Team
Exact Match
Semantic Search
Search bugs by title or ID
SEARCH
RESET
RESET ALL
EXPORT CSV
EXPORT PDF
Priority
Status
Issue Type
Developer
Tester
From
01-05-2025
To
02-05-2025
Bug ID
Title
Status
Issue Type
Priority
Assigned Develo...
Assigned Tester
Status Al...
Actions
favourite
BUG-146
Invalid password accept...
Open
UI Issue
High
Unassigned
Unassigned
6h+
VIEW

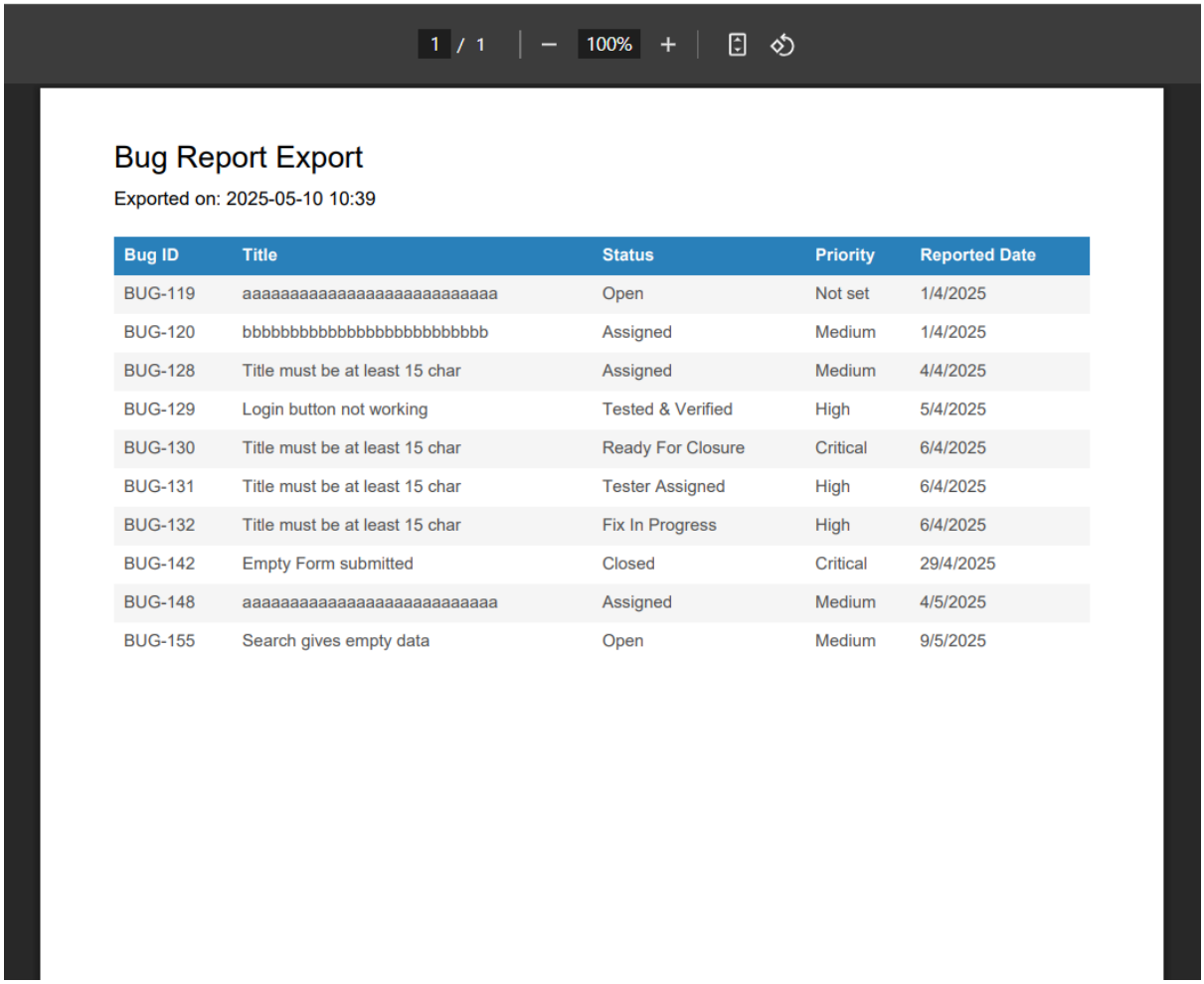

From
26-03-2025
To
31-03-2025
March, 2025
Su Mo Tu We Th Fr Sa
23 24 25 26 27 28 1
2 3 4 5 6 7 8
9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30 31 1 2 3 4 5
Clear
Today
Tester
Status Al...
6h+
12h+

Bugs Assigned to your Team
Semantic Search: Type a full sentence and click Search to find similar bugs.
Exact Match
Semantic Search
Enter full sentence to search
wrong password acceptec
SEARCH
RESET
RESET ALL
EXPORT CSV
EXPORT PDF
Priority
Status
Issue Type
Developer
Tester
From
dd-mm-2025
To
dd-mm-2025
Bug ID
Title
Status
Issue Type
Priority
Assigned Develo...
Assigned Tester
Status Al...
Actions
favourite
BUG-146
Invalid password accept...
Open
UI Issue
High
Unassigned
Unassigned
6h+
VIEW
BUG-147
Password mismatch
Assigned
UI Issue
High
Harishmitha Raja
Unassigned
24h+
VIEW

APPENDIX N – Export CSV/PDF



Bug ID	Title	Status	Issue Type	Priority	Assigned Developer	Assigned Tester	Reported Date
BUG-153	Form data accepted without dat	Assigned	UI Issue	Medium	harishmitha2507@gmail.com	Unassigned	05-07-2025
BUG-150	skjbd ksajnd ksnk s	Open	UI Issue	Not set	Unassigned	Unassigned	05-05-2025
BUG-149	sd fsdf sdf sdf sdf sdf sd	Assigned	Backend Issue	High	harishmitha2507@gmail.com	Unassigned	05-04-2025
BUG-148	aaaaaaaaaaaaaaaaaaaaaaaaa	Assigned	Incorrect colors, fonts, or icons (Not as expected)	Medium	harishmitha2507+joeg@gmail.c	Unassigned	05-04-2025
BUG-147	Password mismatch	Assigned	UI Issue	High	harishmitha2507@gmail.com	Unassigned	05-04-2025
BUG-146	Invalid password accepted	Open	UI Issue	High	Unassigned	Unassigned	05-01-2025
BUG-144	sjf skjdnf klsdf skdmf ksmd fs	Open	UI Issue	Not set	Unassigned	Unassigned	29-04-2025
BUG-142	Empty Form submitted	Closed	Other	Critical	harishmitha2507@gmail.com	Unassigned	29-04-2025
BUG-141	Dashboard overlaped	Open	UI Issue	Not set	Unassigned	Unassigned	29-04-2025
BUG-139	Sign in error crash	Tester Assigned	UI Issue	Critical	harishmitha2507@gmail.com	harishmitha2507+gina@gm	04-09-2025
BUG-138	Feedback form submission error	Tested & Verified	UI Issue	Critical	harishmitha2507@gmail.com	harishmitha2507+ramya@g	04-07-2025
BUG-137	App shuts down after sign in	Assigned	UI Issue	Critical	harishmitha2507+thanigai@gma	Unassigned	04-07-2025
BUG-136	qqqqqqqqqqqqqqqqqqqqq	Tested & Verified	UI Issue	Critical	harishmitha2507+john@gmail.c	harishmitha2507+ramya@g	04-06-2025
BUG-133	Search no result	Assigned	UI Issue	High	harishmitha2507+thanigai@gma	Unassigned	04-06-2025
BUG-132	Title must be at least 15 char	Fix In Progress	Incorrect colors, fonts, or icons (Not as expected)	High	harishmitha2507+thanigai@gma	Unassigned	04-06-2025
BUG-131	Title must be at least 15 char	Tester Assigned	Button or link is not working (Clicking does nothing)	High	harishmitha2507@gmail.com	harishmitha2507+gina@gm	04-06-2025
BUG-130	Title must be at least 15 char	Tested & Verified	Button or link is not working (Clicking does nothing)	Critical	harishmitha2507@gmail.com	harishmitha2507+ramya@g	04-06-2025
BUG-129	Login button not working	Tested & Verified	Button or link is not working (Clicking does nothing)	High	harishmitha2507@gmail.com	harishmitha2507+ramya@g	04-05-2025
BUG-128	Title must be at least 15 char	Assigned	Incorrect colors, fonts, or icons (Not as expected)	Medium	harishmitha2507@gmail.com	harishmitha2507+preethi@	04-04-2025
BUG-122	pjjbknscznjcbzdcbd	Tested & Verified	UI Issue	High	harishmitha2507@gmail.com	harishmitha2507+ramya@g	04-02-2025
BUG-121	qqqqqqqqqqpppppppppppppp	Tester Assigned	UI Issue	Critical	harishmitha2507@gmail.com	harishmitha2507+ramya@g	04-01-2025
BUG-120	bbbbbbbbbbbbbbbbbbbbbbbbb	Assigned	Visual issue (Misaligned elements, overlapping, text cut)	Medium	harishmitha2507@gmail.com	Unassigned	04-01-2025
BUG-118	sadasdasdasdasdasda	Assigned	UI Issue	Critical	harishmitha2507+joeg@gmail.c	Unassigned	04-01-2025
BUG-117	sdfsdsdvxdvxfvfvxfv	Closed	UI Issue	High	harishmitha2507@gmail.com	harishmitha2507+ramya@g	04-01-2025

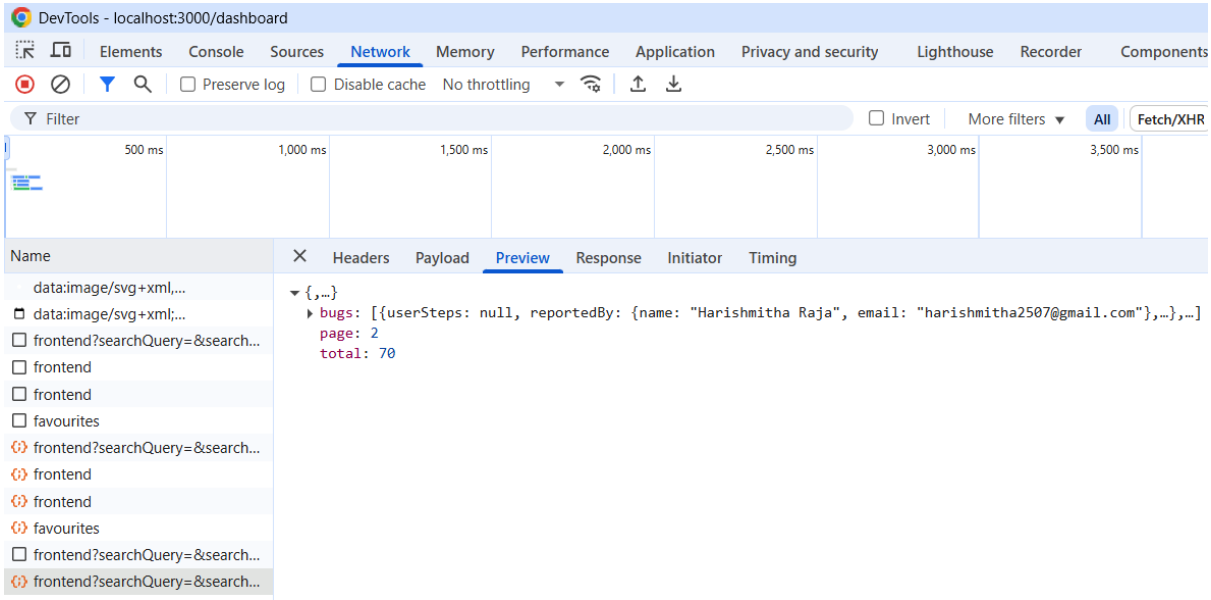


Bug Report Export

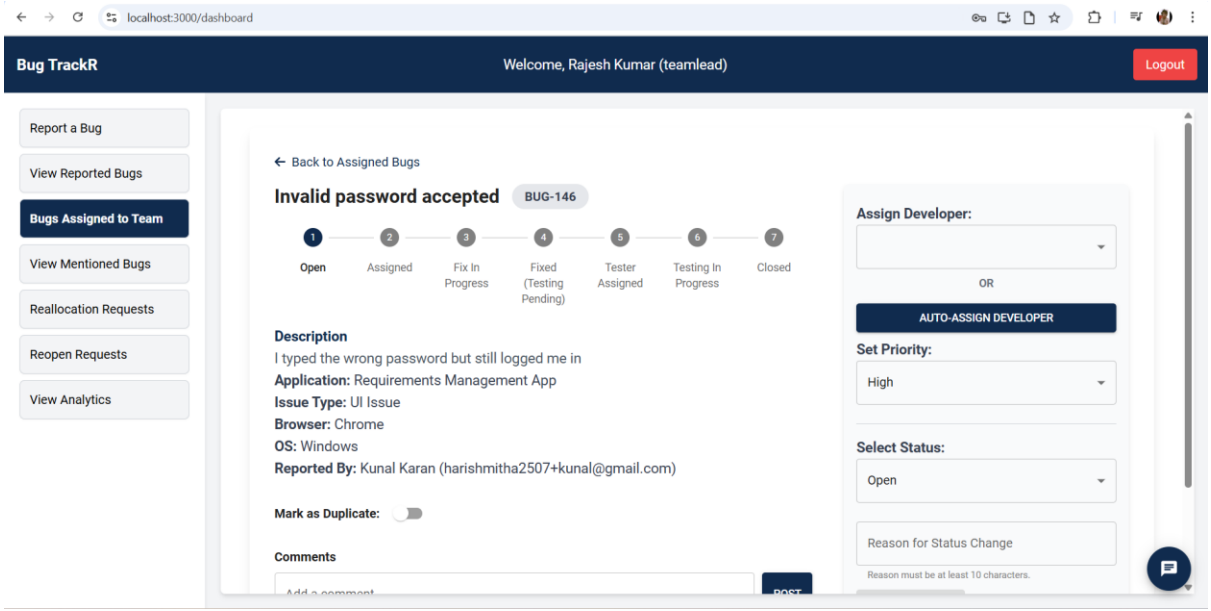
Exported on: 2025-05-10 10:39

Bug ID	Title	Status	Priority	Reported Date
BUG-119	aaaaaaaaaaaaaaaaaaaaaaaaa	Open	Not set	1/4/2025
BUG-120	bbbbbbbbbbbbbbbbbbbbbbbbb	Assigned	Medium	1/4/2025
BUG-128	Title must be at least 15 char	Assigned	Medium	4/4/2025
BUG-129	Login button not working	Tested & Verified	High	5/4/2025
BUG-130	Title must be at least 15 char	Ready For Closure	Critical	6/4/2025
BUG-131	Title must be at least 15 char	Tester Assigned	High	6/4/2025
BUG-132	Title must be at least 15 char	Fix In Progress	High	6/4/2025
BUG-142	Empty Form submitted	Closed	Critical	29/4/2025
BUG-148	aaaaaaaaaaaaaaaaaaaaaaaaa	Assigned	Medium	4/5/2025
BUG-155	Search gives empty data	Open	Medium	9/5/2025

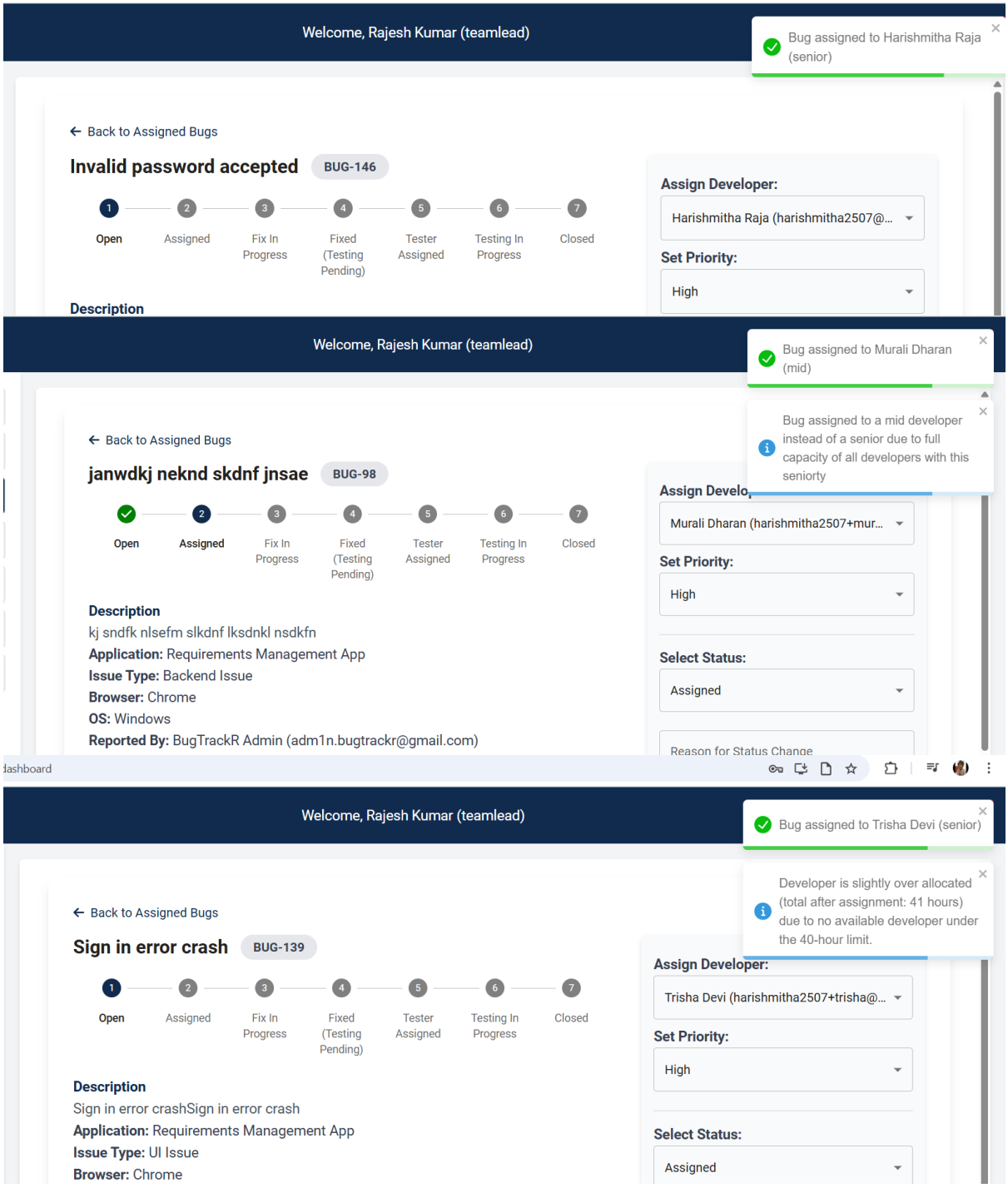
APPENDIX O – Batch Loading (Results from backend)

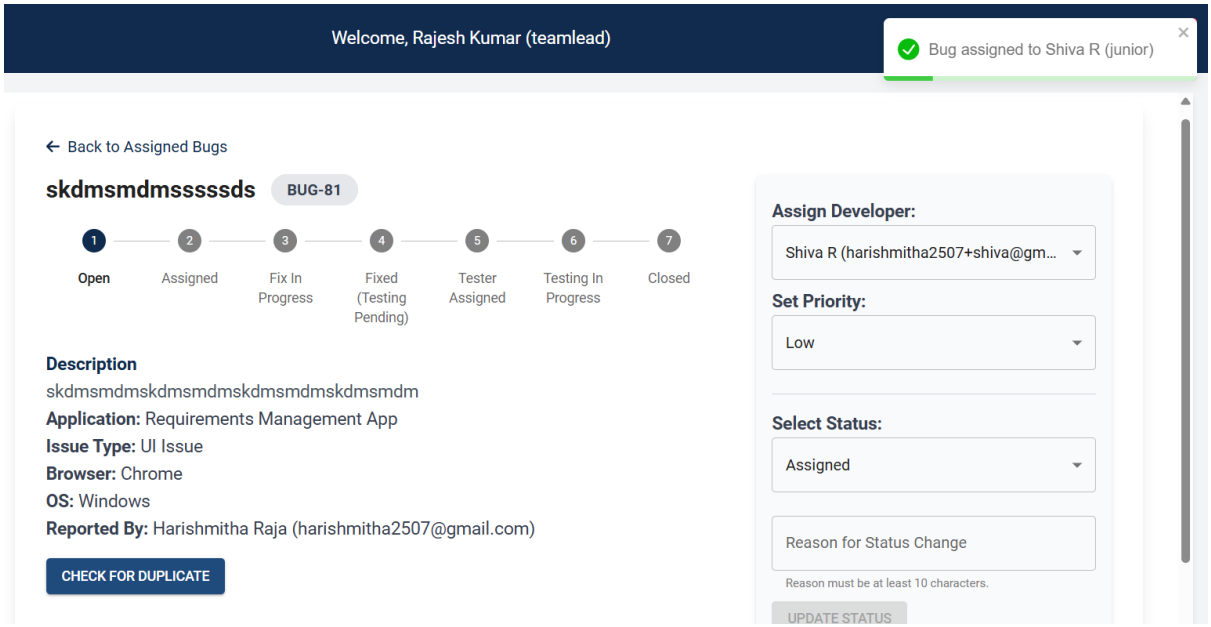


APPENDIX P – Team Lead Team Assigned Bug Details(similarly for admin)

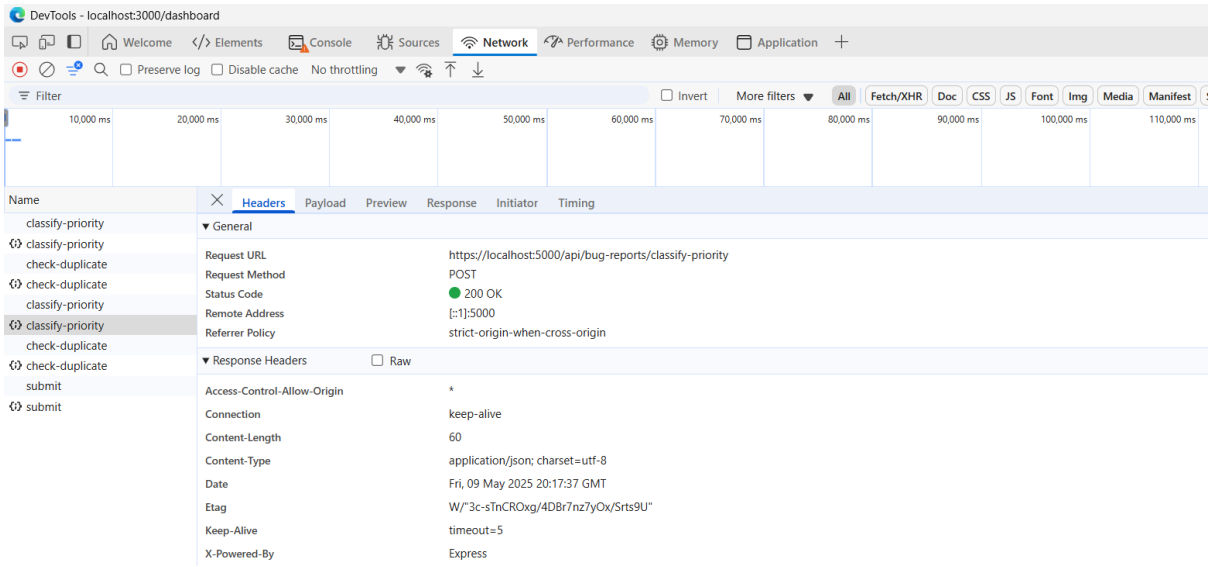


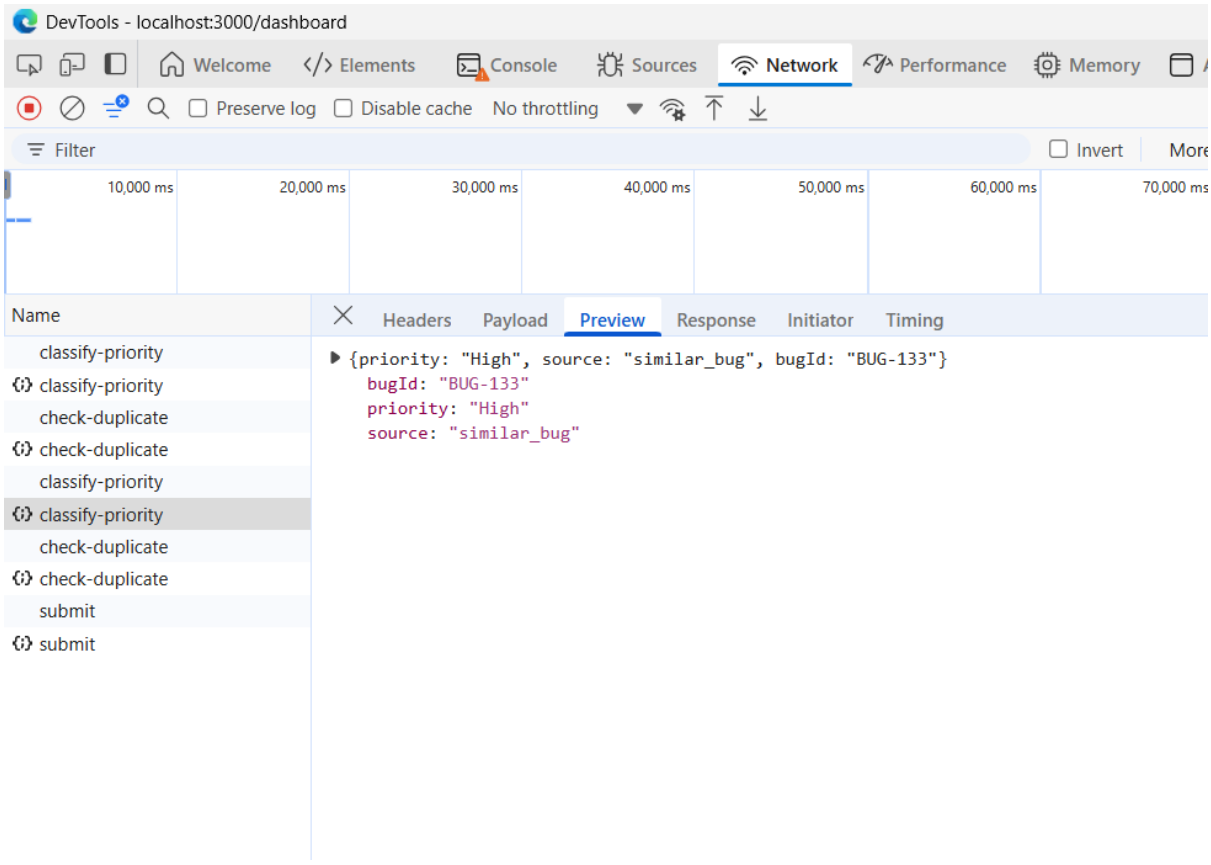
APPENDIX Q – Team Lead Auto Assign Bugs (similarly for admin)

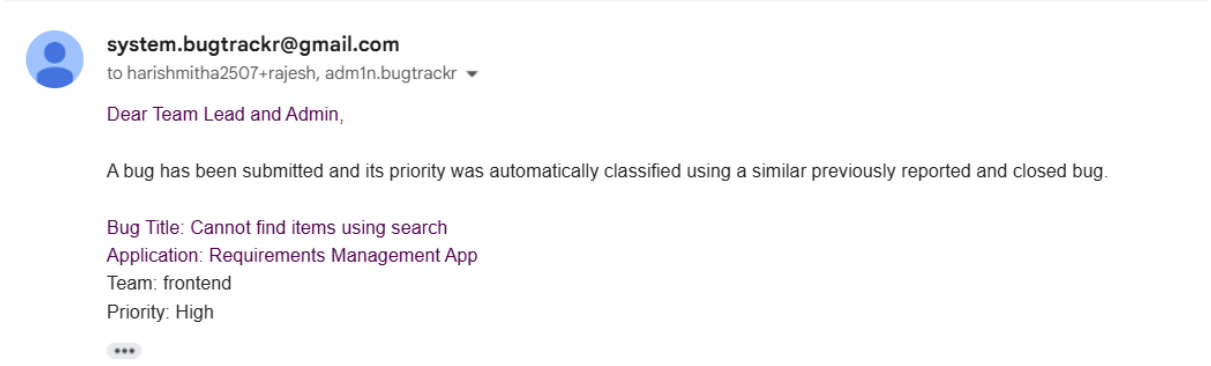




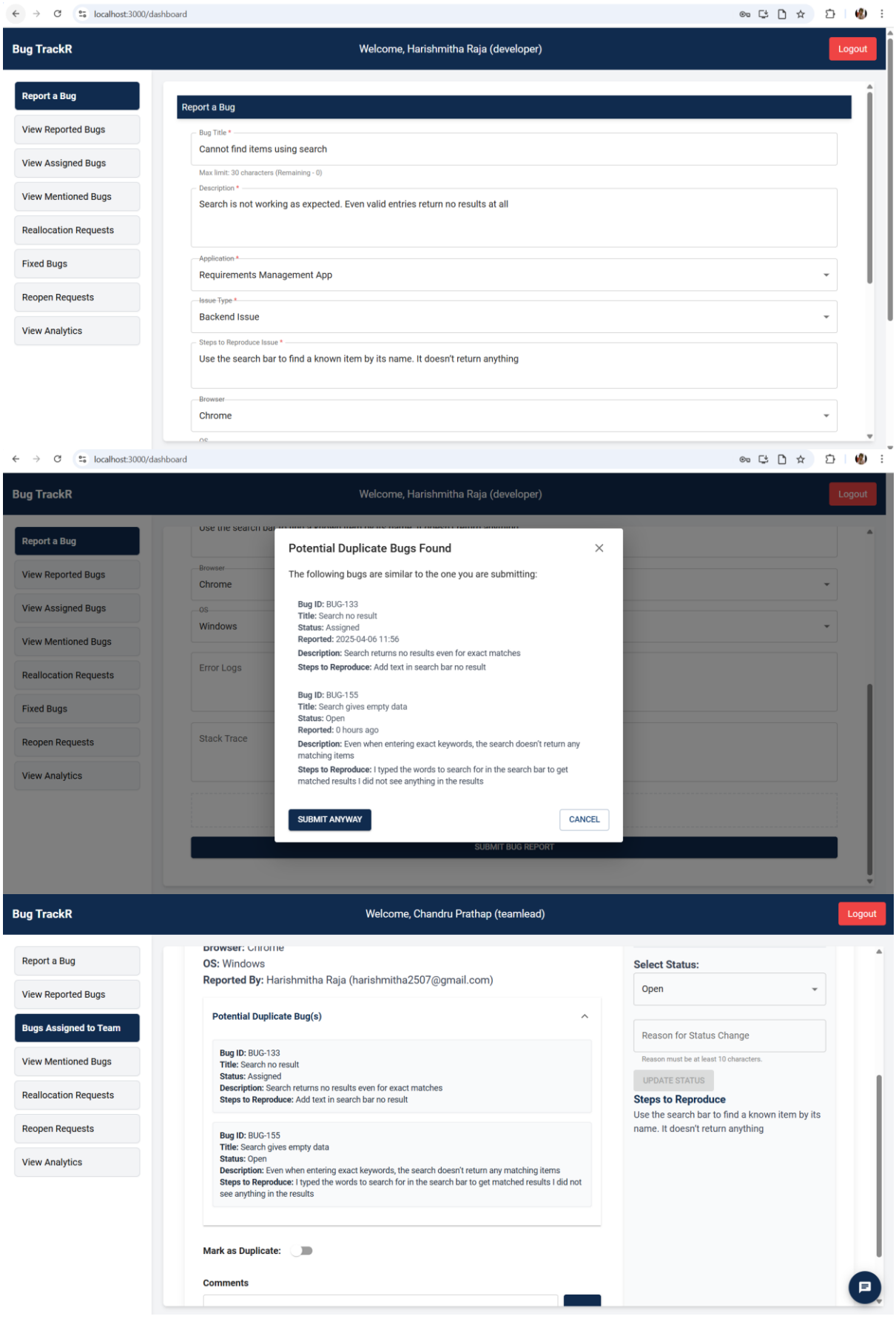
APPENDIX R – Priority Classification using similar bugs during submission




DevTools - localhost:3000/dashboard
Welcome
Elements
Console
Sources
Network
Performance
Memory
Preserve log
Disable cache
No throttling
Filter
Invert
10,000 ms
20,000 ms
30,000 ms
40,000 ms
50,000 ms
60,000 ms
70,000 ms
Name
Headers
Payload
Preview
Response
Initiator
Timing
classify-priority
classify-priority
check-duplicate
check-duplicate
classify-priority
classify-priority
check-duplicate
check-duplicate
submit
submit
{priority: "High", source: "similar_bug", bugId: "BUG-133"}
bugId: "BUG-133"
priority: "High"
source: "similar_bug"


system.bugtrackr@gmail.com
to harishmitha2507+rajesh, adm1n.bugtrackr
Dear Team Lead and Admin,
A bug has been submitted and its priority was automatically classified using a similar previously reported and closed bug.
Bug Title: Cannot find items using search
Application: Requirements Management App
Team: frontend
Priority: High

APPENDIX S – Duplicate Detection



Bug ID: BUG-133
Title: Search no result
Status: Assigned
Description: Search returns no results even for exact matches
Steps to Reproduce: Add text in search bar no result

Bug ID: BUG-155
Title: Search gives empty data
Status: Open
Description: Even when entering exact keywords, the search doesn't return any matching items
Steps to Reproduce: I typed the words to search for in the search bar to get matched results I did not see anything in the results

Mark as Duplicate:

Original Bug ID *
BUG-133

Explanation *
This bug is already being worked on

MARK AS DUPLICATE

Mark as Duplicate:

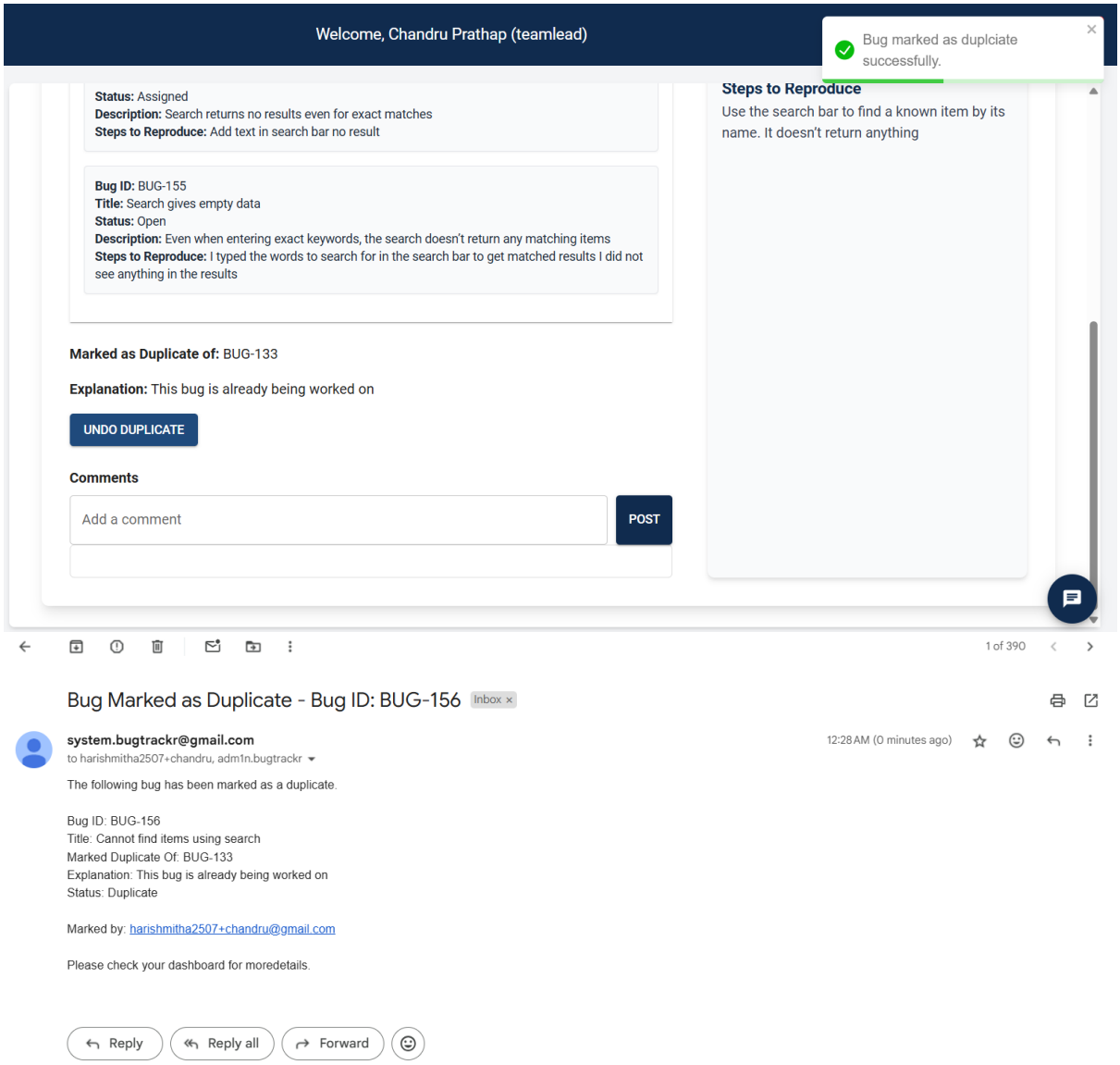
Original Bug ID *
B

Original bug ud must be in the format 'BUG-number' (eg: BUG-1)

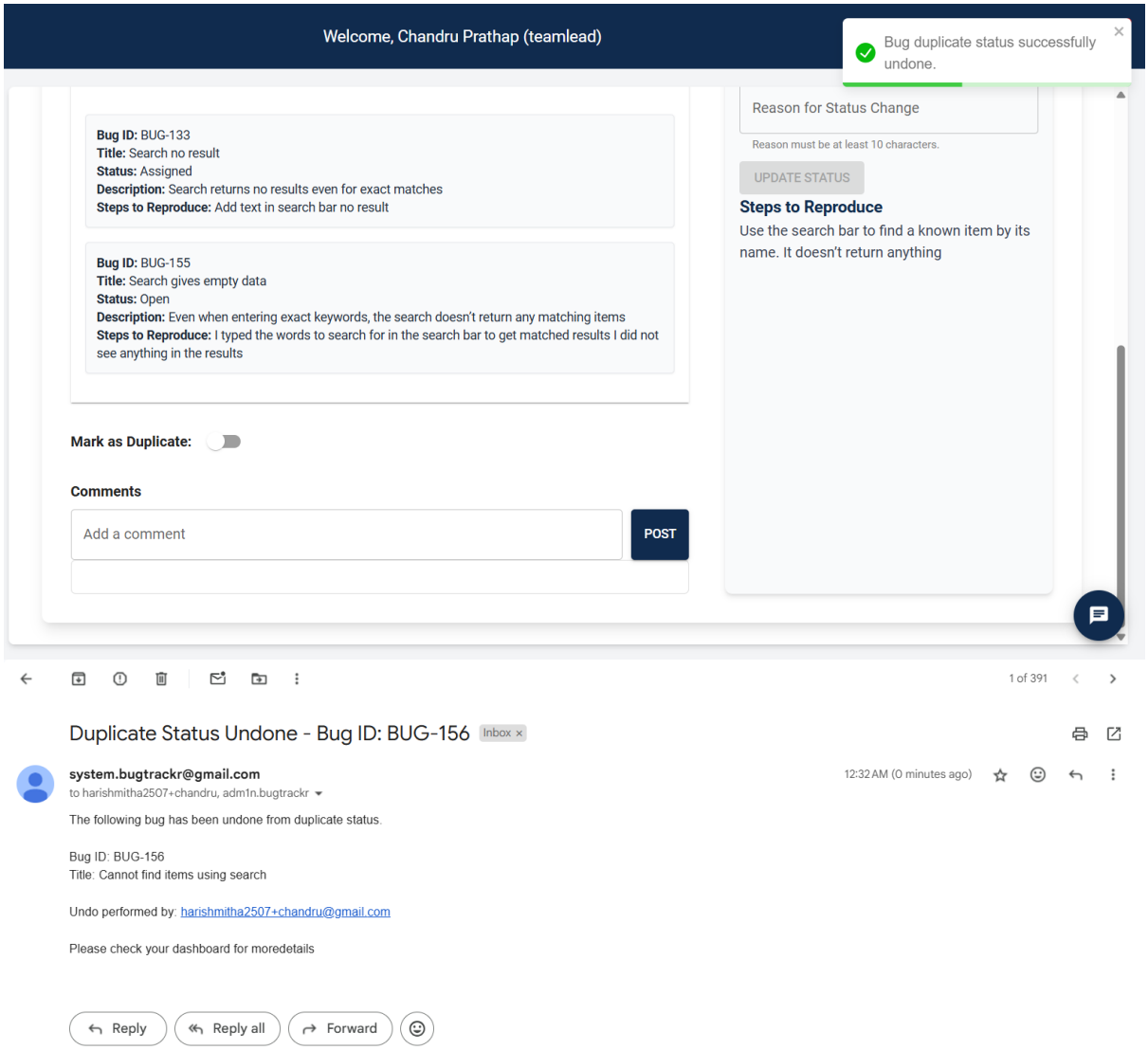
Explanation *
T

Explanation must be at least 10 characters.

MARK AS DUPLICATE

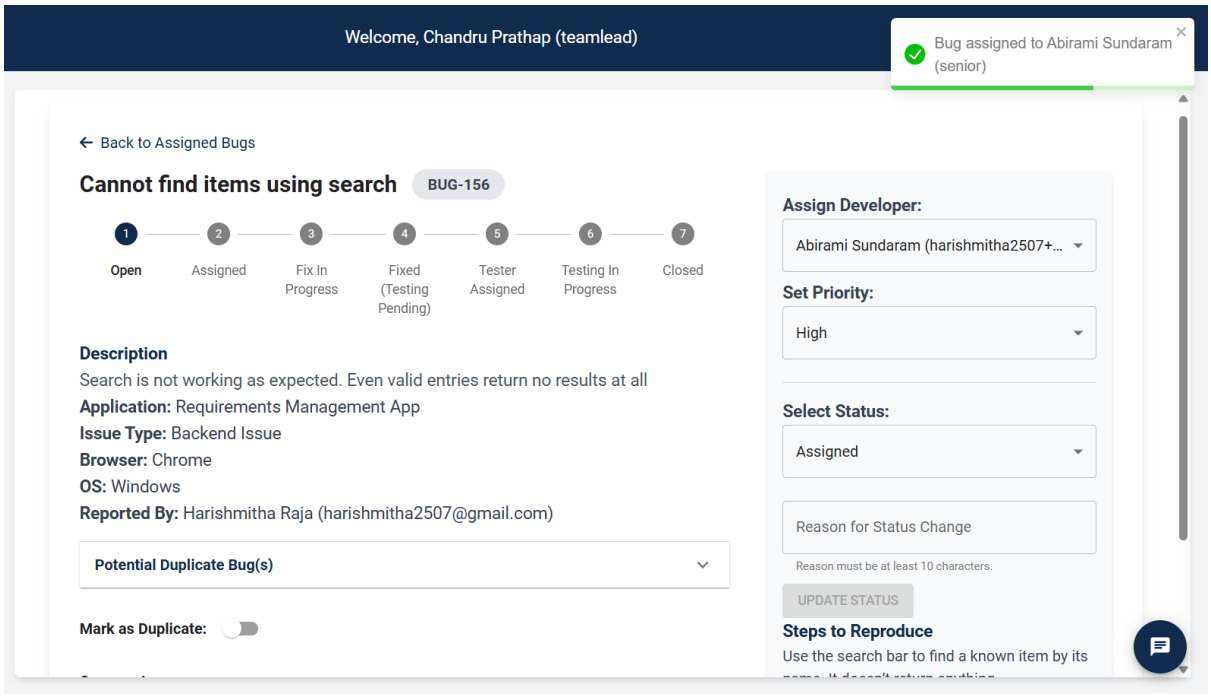

Welcome, Chandru Prathap (teamlead)
Bug marked as duplciate successfully.
Status: Assigned
Description: Search returns no results even for exact matches
Steps to Reproduce: Add text in search bar no result
Bug ID: BUG-155
Title: Search gives empty data
Status: Open
Description: Even when entering exact keywords, the search doesn't return any matching items
Steps to Reproduce: I typed the words to search for in the search bar to get matched results I did not see anything in the results
Steps to Reproduce
Use the search bar to find a known item by its name. It doesn't return anything
Marked as Duplicate of: BUG-133
Explanation: This bug is already being worked on
UNDO DUPLICATE
Comments
Add a comment
POST

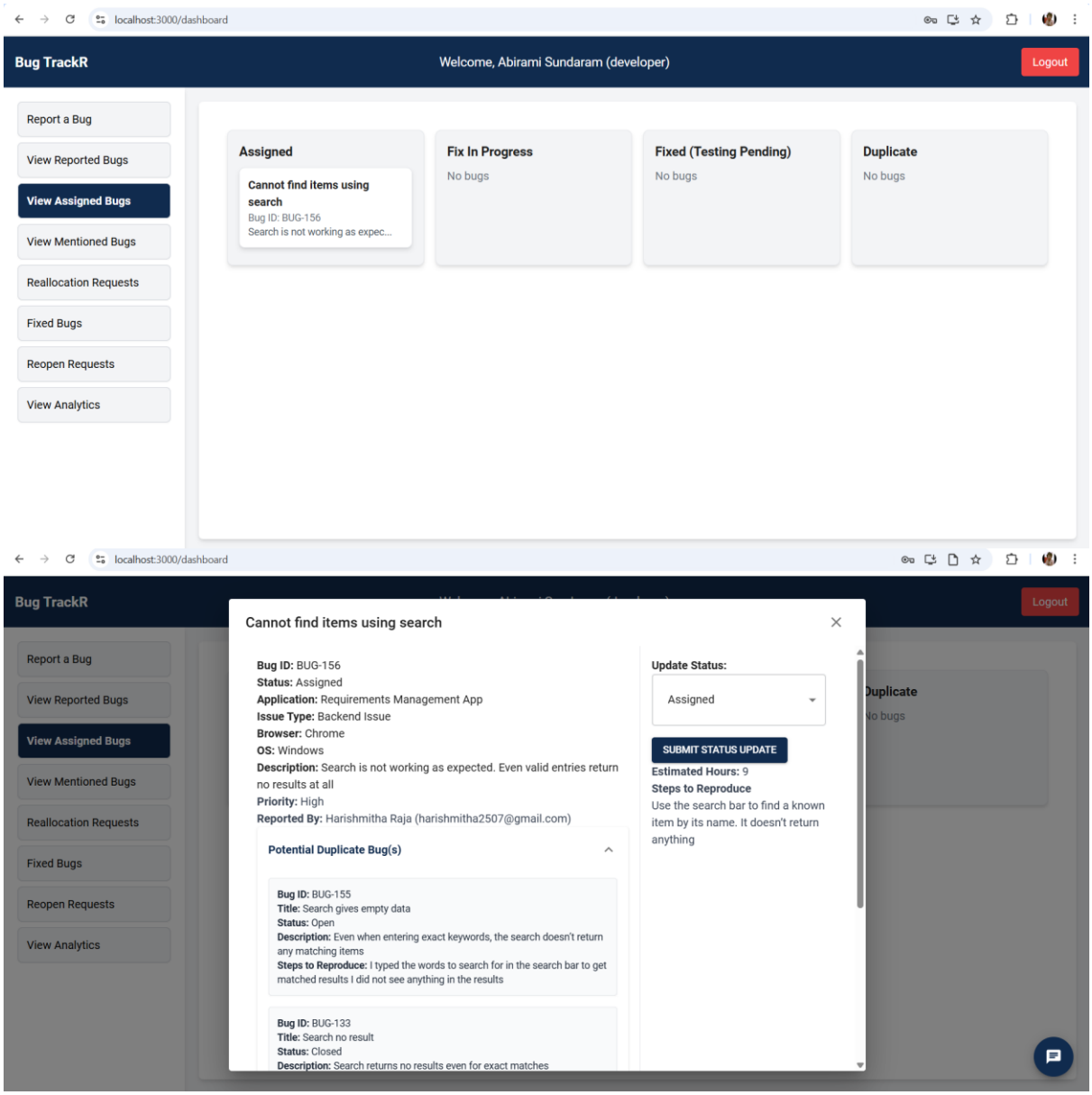
1 of 390
Bug Marked as Duplicate - Bug ID: BUG-156
Inbox ×
system.bugtrackr@gmail.com
to harishmitha2507+chandru, adm1n.bugtrackr
12:28 AM (0 minutes ago)
The following bug has been marked as a duplicate.
Bug ID: BUG-156
Title: Cannot find items using search
Marked Duplicate Of: BUG-133
Explanation: This bug is already being worked on
Status: Duplicate
Marked by: harishmitha2507+chandru@gmail.com
Please check your dashboard for moredetails.
Reply
Reply all
Forward


Welcome, Chandru Prathap (teamlead)
Bug duplicate status successfully undone.
Bug ID: BUG-133
Title: Search no result
Status: Assigned
Description: Search returns no results even for exact matches
Steps to Reproduce: Add text in search bar no result
Bug ID: BUG-155
Title: Search gives empty data
Status: Open
Description: Even when entering exact keywords, the search doesn't return any matching items
Steps to Reproduce: I typed the words to search for in the search bar to get matched results I did not see anything in the results
Mark as Duplicate:
Comments
Add a comment
POST
Reason for Status Change
Reason must be at least 10 characters.
UPDATE STATUS
Steps to Reproduce
Use the search bar to find a known item by its name. It doesn't return anything

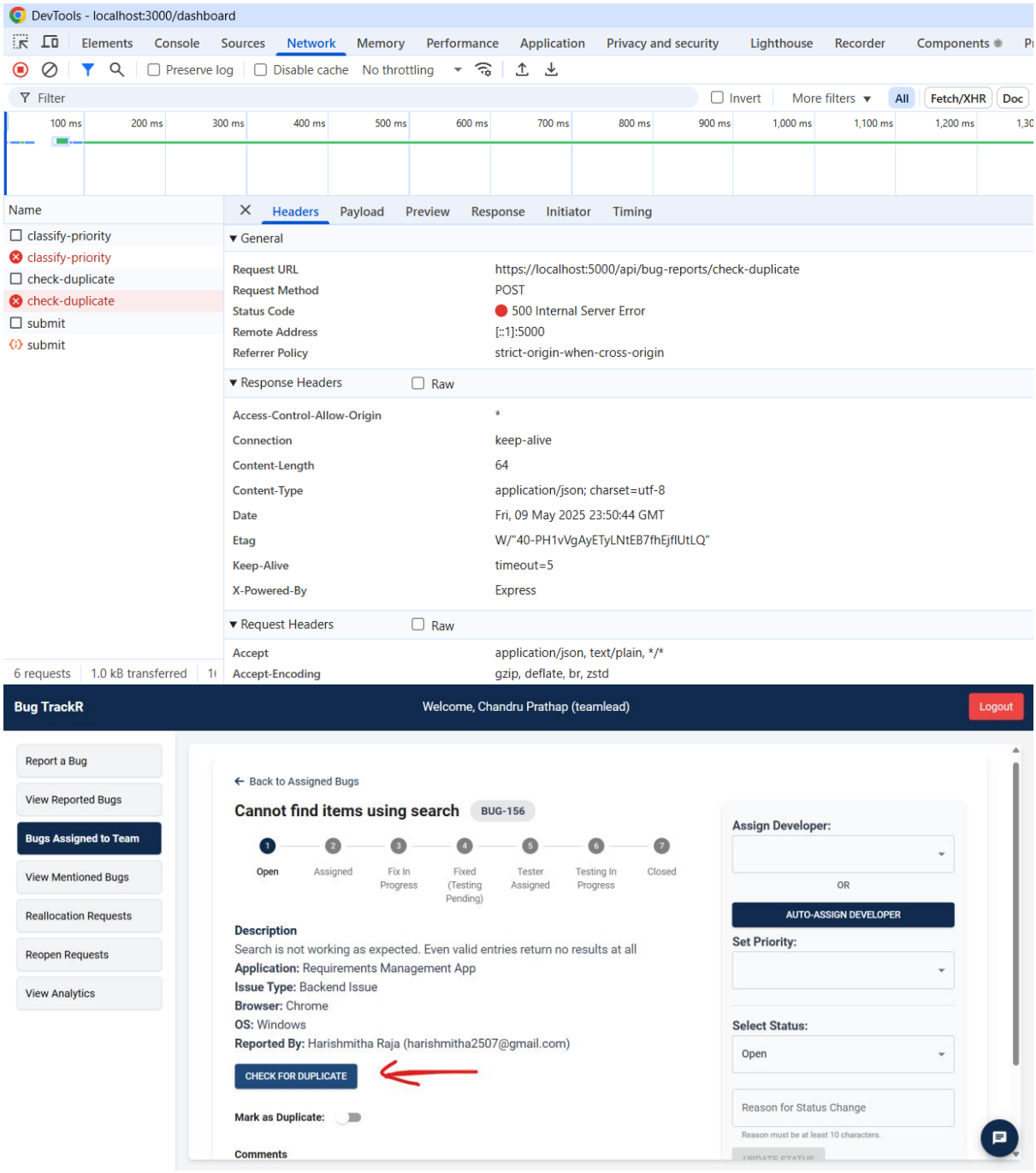
1 of 391
Duplicate Status Undone - Bug ID: BUG-156
Inbox ×
system.bugtrackr@gmail.com
to harishmitha2507+chandru, adm1n.bugtrackr
12:32 AM (0 minutes ago)
The following bug has been undone from duplicate status.
Bug ID: BUG-156
Title: Cannot find items using search
Undo performed by: harishmitha2507+chandru@gmail.com
Please check your dashboard for moredetails
Reply
Reply all
Forward

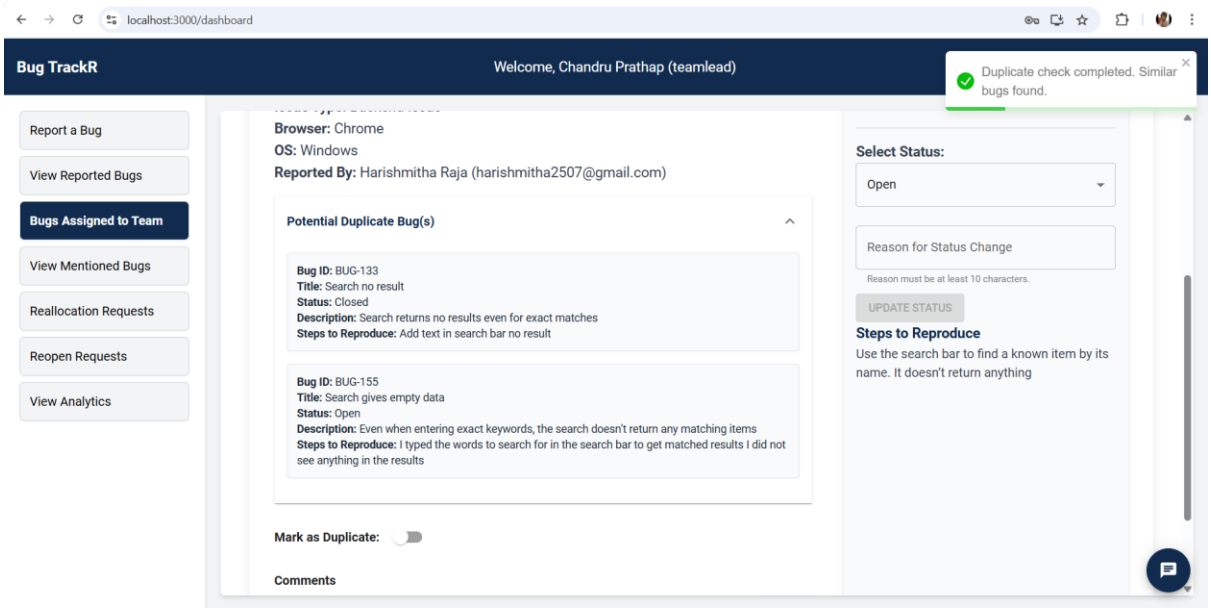
APPENDIX t – Assigned Developer View of Duplicates

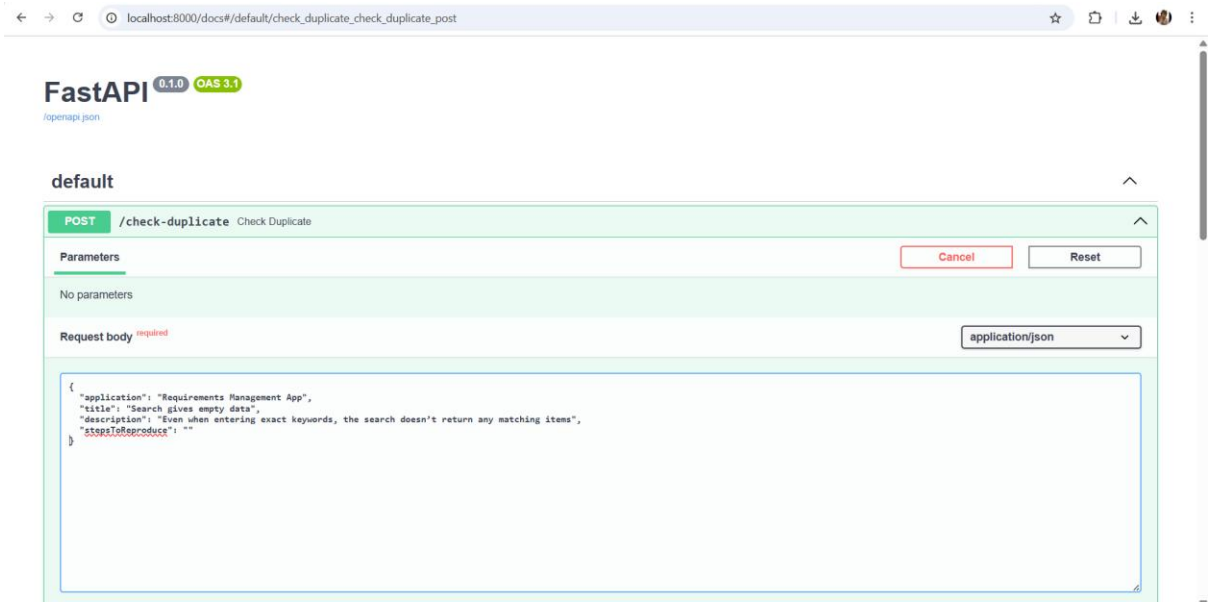


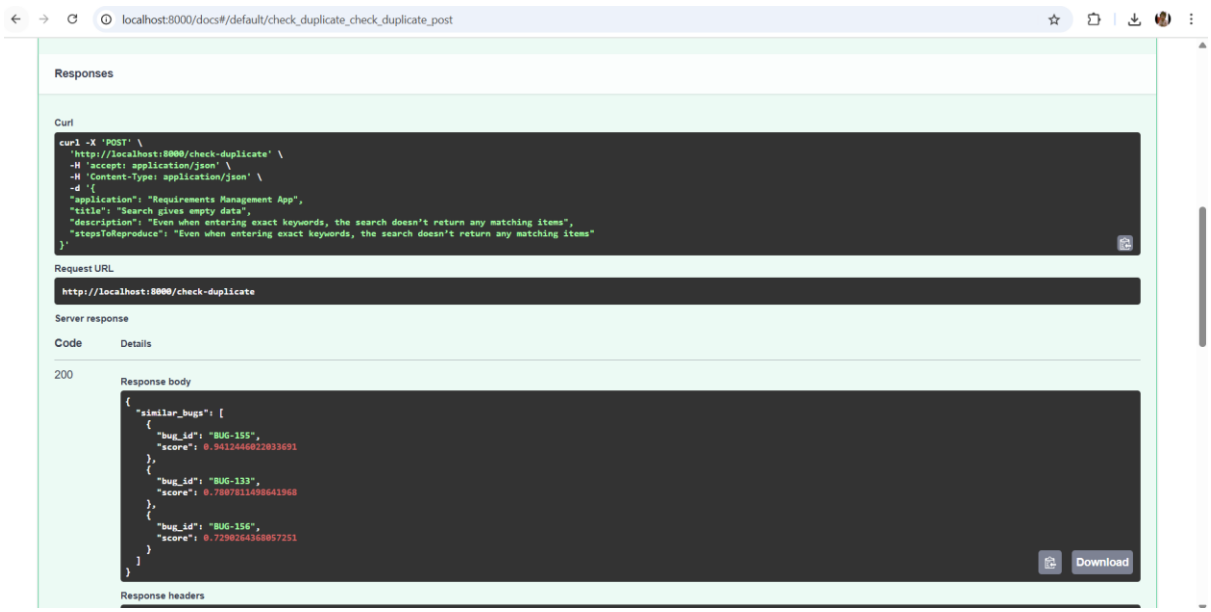


APPENDIX U – Duplicate Detection Service Down (duplicate check trigger) + Fast API check

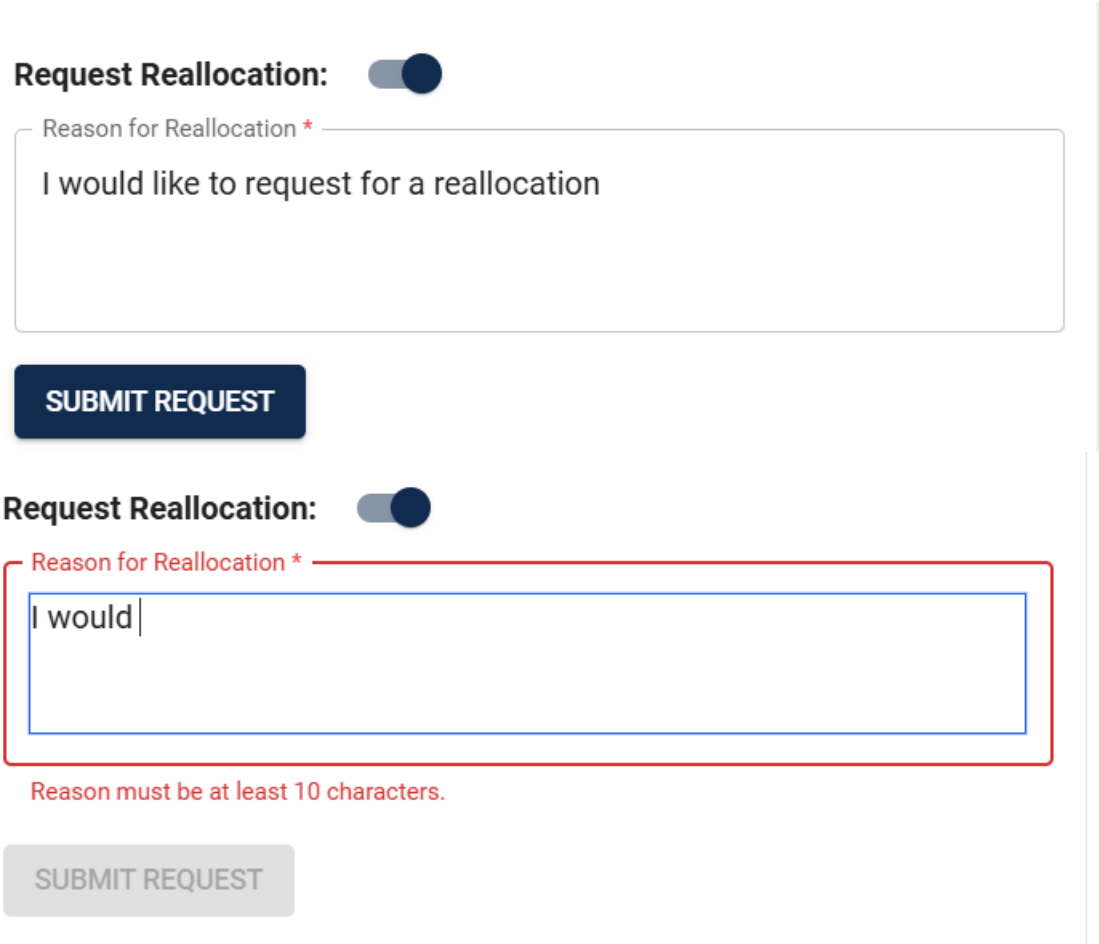

DevTools - localhost:3000/dashboard
Elements
Console
Sources
Network
Memory
Performance
Application
Privacy and security
Lighthouse
Recorder
Components
Preserve log
Disable cache
No throttling
Filter
Invert
More filters
All
Fetch/XHR
Doc
100 ms
200 ms
300 ms
400 ms
500 ms
600 ms
700 ms
800 ms
900 ms
1,000 ms
1,100 ms
1,200 ms
Name
classify-priority
classify-priority
check-duplicate
check-duplicate
submit
submit
Headers
Payload
Preview
Response
Initiator
Timing
General
Request URL
https://localhost:5000/api/bug-reports/check-duplicate
Request Method
POST
Status Code
500 Internal Server Error
Remote Address
[::1]:5000
Referrer Policy
strict-origin-when-cross-origin
Response Headers
Raw
Access-Control-Allow-Origin
*
Connection
keep-alive
Content-Length
64
Content-Type
application/json; charset=utf-8
Date
Fri, 09 May 2025 23:50:44 GMT
Etag
W/"40-PH1vVgAyETyLNtEB7fhEjfIUtLQ"
Keep-Alive
timeout=5
X-Powered-By
Express
Request Headers
Raw
Accept
application/json, text/plain, */*
Accept-Encoding
gzip, deflate, br, zstd
6 requests
1.0 kB transferred
Bug TrackR
Welcome, Chandru Prathap (teamlead)
Logout
Report a Bug
View Reported Bugs
Bugs Assigned to Team
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
Back to Assigned Bugs
Cannot find items using search
BUG-156
Open
Assigned
Fix In Progress
Fixed (Testing Pending)
Tester Assigned
Testing In Progress
Closed
Description
Search is not working as expected. Even valid entries return no results at all
Application: Requirements Management App
Issue Type: Backend Issue
Browser: Chrome
OS: Windows
Reported By: Harishmitha Raja (harishmitha2507@gmail.com)
CHECK FOR DUPLICATE
Mark as Duplicate:
Comments
Assign Developer:
OR
AUTO-ASSIGN DEVELOPER
Set Priority:
Select Status:
Open
Reason for Status Change
Reason must be at least 10 characters.

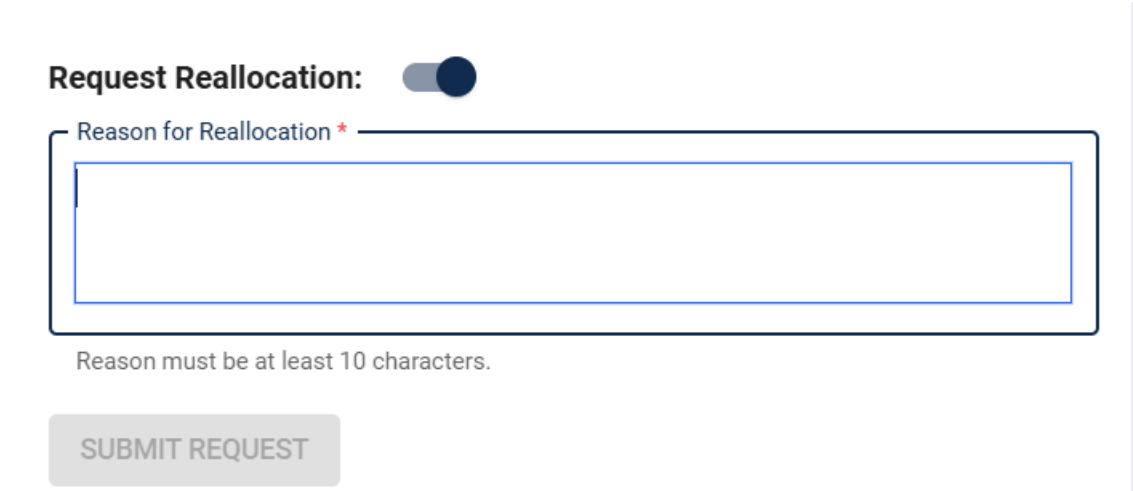

localhost:3000/dashboard
Bug TrackR
Welcome, Chandru Prathap (teamlead)
Duplicate check completed. Similar bugs found.
Report a Bug
View Reported Bugs
Bugs Assigned to Team
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
Browser: Chrome
OS: Windows
Reported By: Harishmitha Raja (harishmitha2507@gmail.com)
Potential Duplicate Bug(s)
Bug ID: BUG-133
Title: Search no result
Status: Closed
Description: Search returns no results even for exact matches
Steps to Reproduce: Add text in search bar no result
Bug ID: BUG-155
Title: Search gives empty data
Status: Open
Description: Even when entering exact keywords, the search doesn't return any matching items
Steps to Reproduce: I typed the words to search for in the search bar to get matched results I did not see anything in the results
Mark as Duplicate:
Comments
Select Status:
Open
Reason for Status Change
Reason must be at least 10 characters.
UPDATE STATUS
Steps to Reproduce
Use the search bar to find a known item by its name. It doesn't return anything

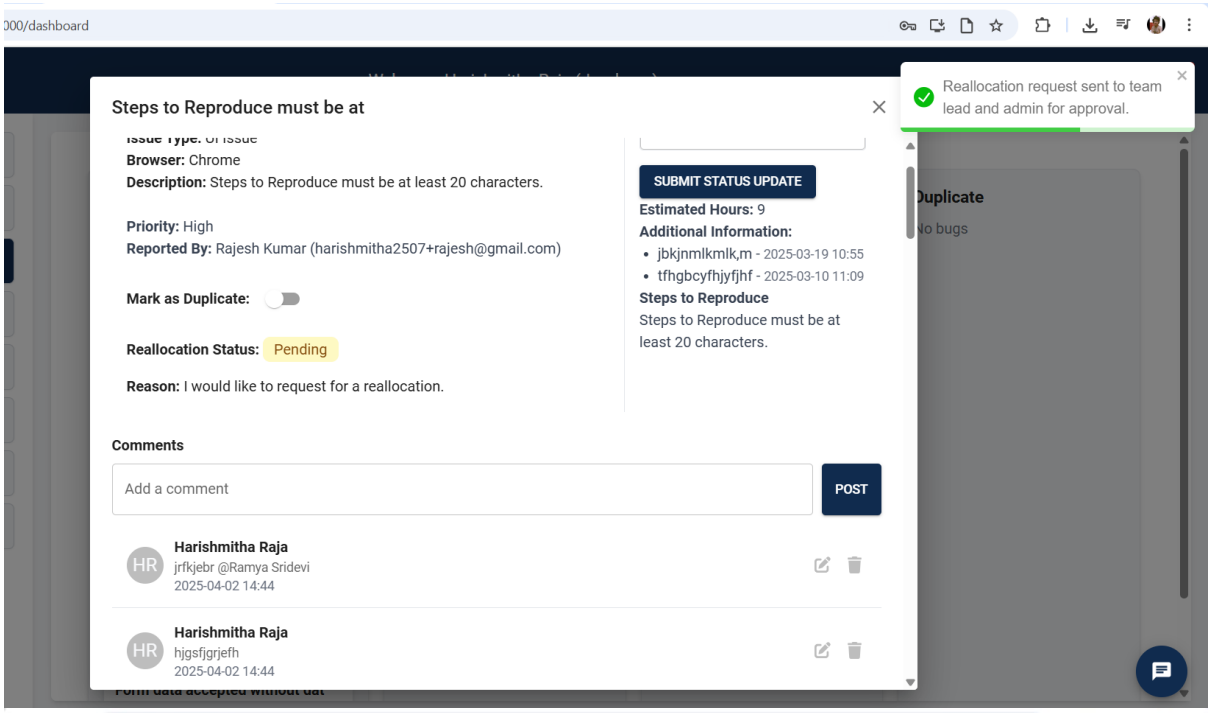

localhost:8000/docs#/default/check_duplicate_check_duplicate_post
FastAPI
0.1.0
OAS 3.1
/openapi.json
default
POST
/check-duplicate
Check Duplicate
Parameters
Cancel
Reset
No parameters
Request body
required
application/json
{
"application": "Requirements Management App",
"title": "Search gives empty data",
"description": "Even when entering exact keywords, the search doesn't return any matching items",
"stepsToReproduce": ""
}

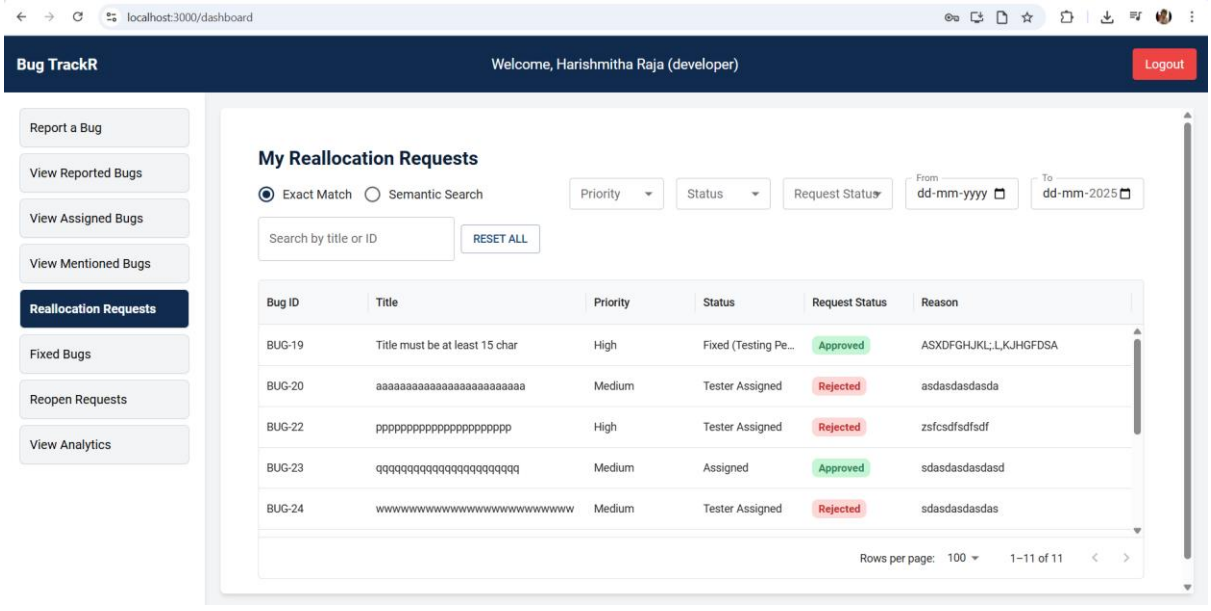


APPENDIX V – Developer/Tester request reallocation

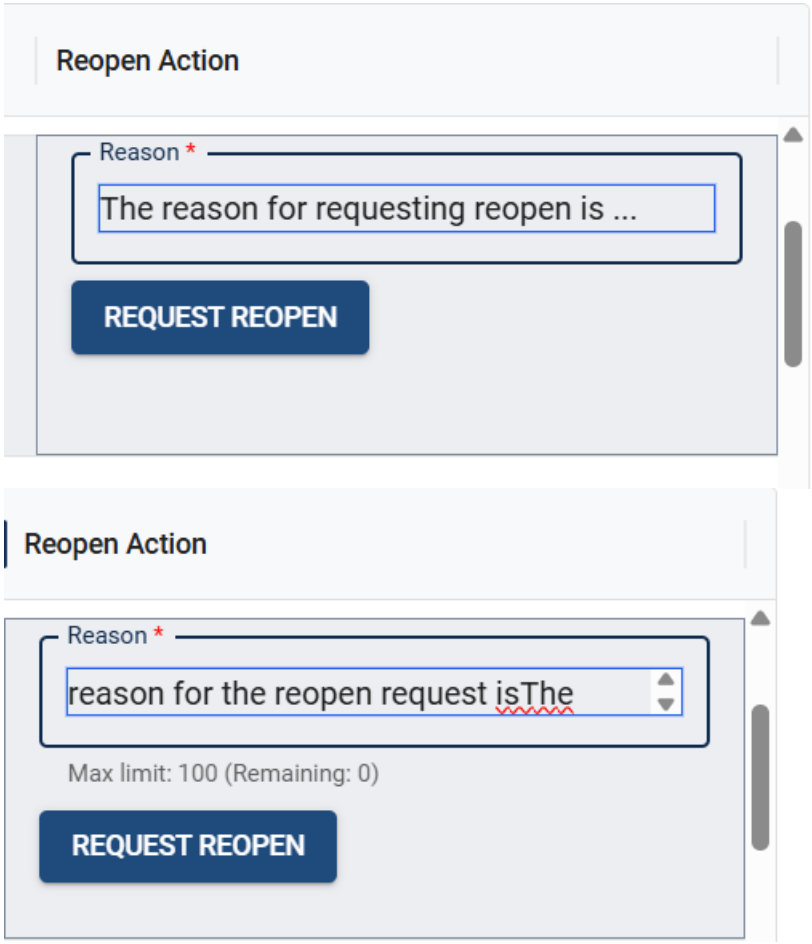


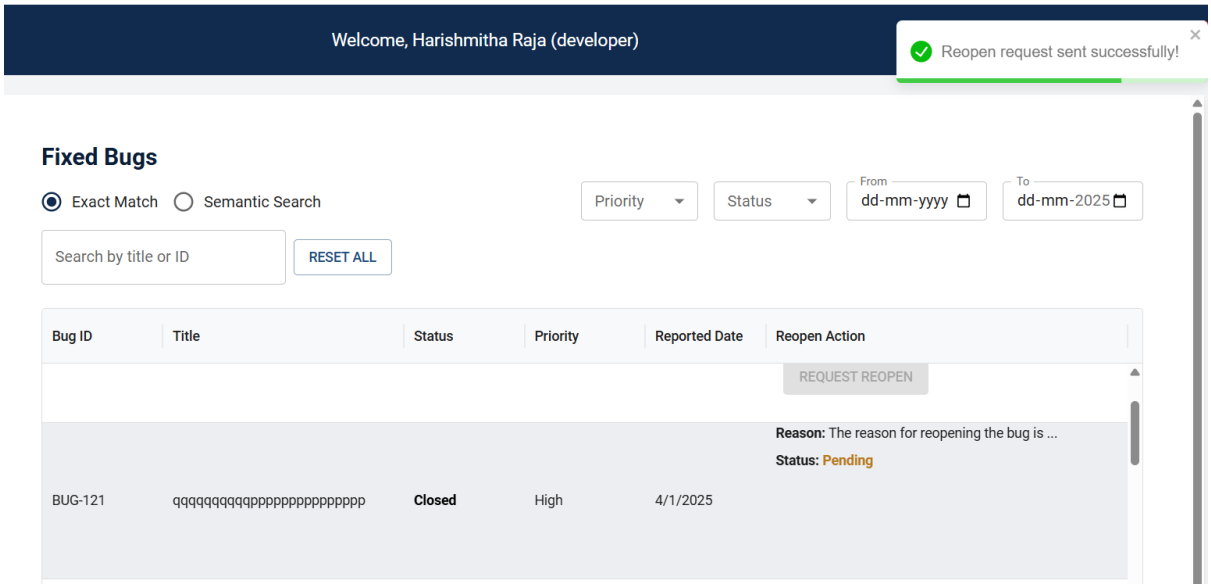

Request Reallocation:
Reason for Reallocation *
Reason must be at least 10 characters.
SUBMIT REQUEST

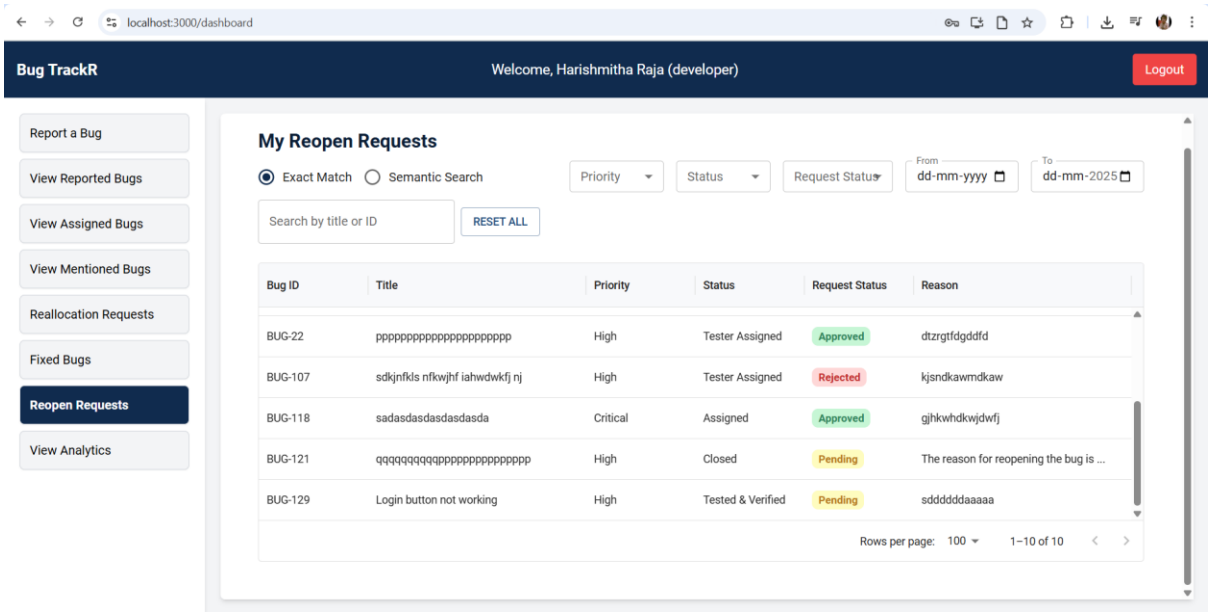

Steps to Reproduce must be at
Browser: Chrome
Description: Steps to Reproduce must be at least 20 characters.
Priority: High
Reported By: Rajesh Kumar (harishmitha2507+rajesh@gmail.com)
Mark as Duplicate:
Reallocation Status: Pending
Reason: I would like to request for a reallocation.
SUBMIT STATUS UPDATE
Estimated Hours: 9
Additional Information:
jbkjnmlkmlk,m - 2025-03-19 10:55
tfhgbcyfhjyfjhf - 2025-03-10 11:09
Steps to Reproduce
Steps to Reproduce must be at least 20 characters.
Comments
Add a comment
POST
Harishmitha Raja
jrfkjebr @Ramya Sridevi
2025-04-02 14:44
Harishmitha Raja
hjgsfjgrjefh
2025-04-02 14:44
Reallocation request sent to team lead and admin for approval.
Duplicate
No bugs


localhost:3000/dashboard
Bug TrackR
Welcome, Harishmitha Raja (developer)
Logout
Report a Bug
View Reported Bugs
View Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Fixed Bugs
Reopen Requests
View Analytics
My Reallocation Requests
Exact Match
Semantic Search
Priority
Status
Request Status
From
dd-mm-yyyy
To
dd-mm-2025
Search by title or ID
RESET ALL
Bug ID
Title
Priority
Status
Request Status
Reason
BUG-19
Title must be at least 15 char
High
Fixed (Testing Pe...
Approved
ASXDFGHJKL;.L,KJHGFDSA
BUG-20
aaaaaaaaaaaaaaaaaaaaaaaa
Medium
Tester Assigned
Rejected
asdasdasdasda
BUG-22
ppppppppppppppppppppp
High
Tester Assigned
Rejected
zsfcsdfsdfsdf
BUG-23
qqqqqqqqqqqqqqqqqqqqqq
Medium
Assigned
Approved
sdasdasdasdasd
BUG-24
wwwwwwwwwwwwwwwwwwwwww
Medium
Tester Assigned
Rejected
sdasdasdasdas
Rows per page: 100
1–11 of 11

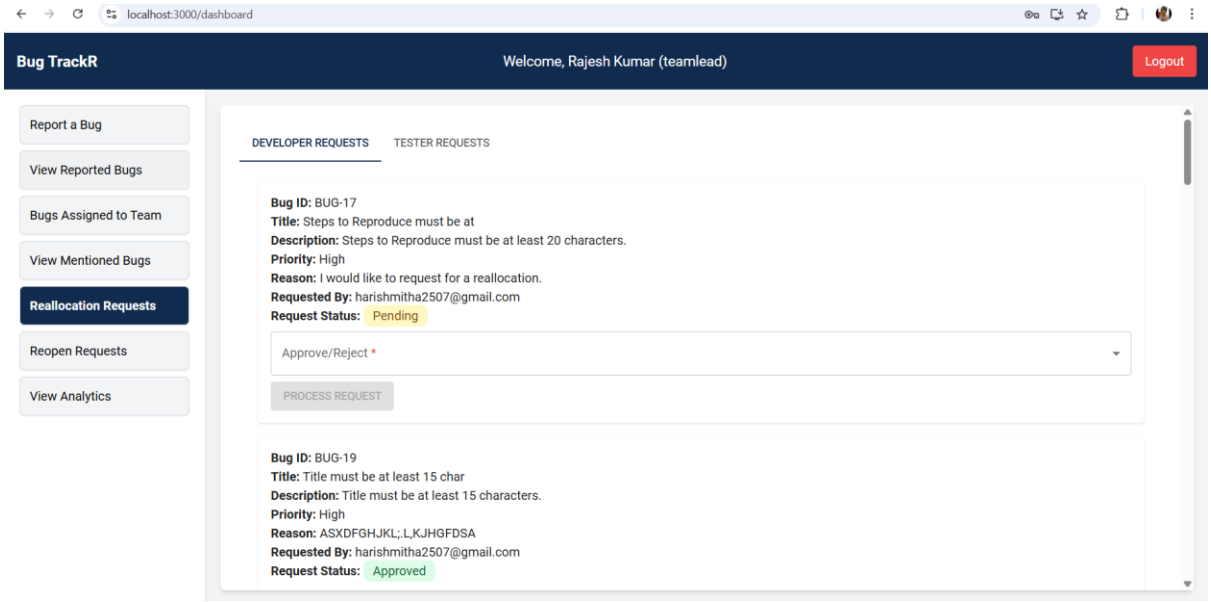
APPENDIX W – Developer/Tester reopen request + Fixed bugs view

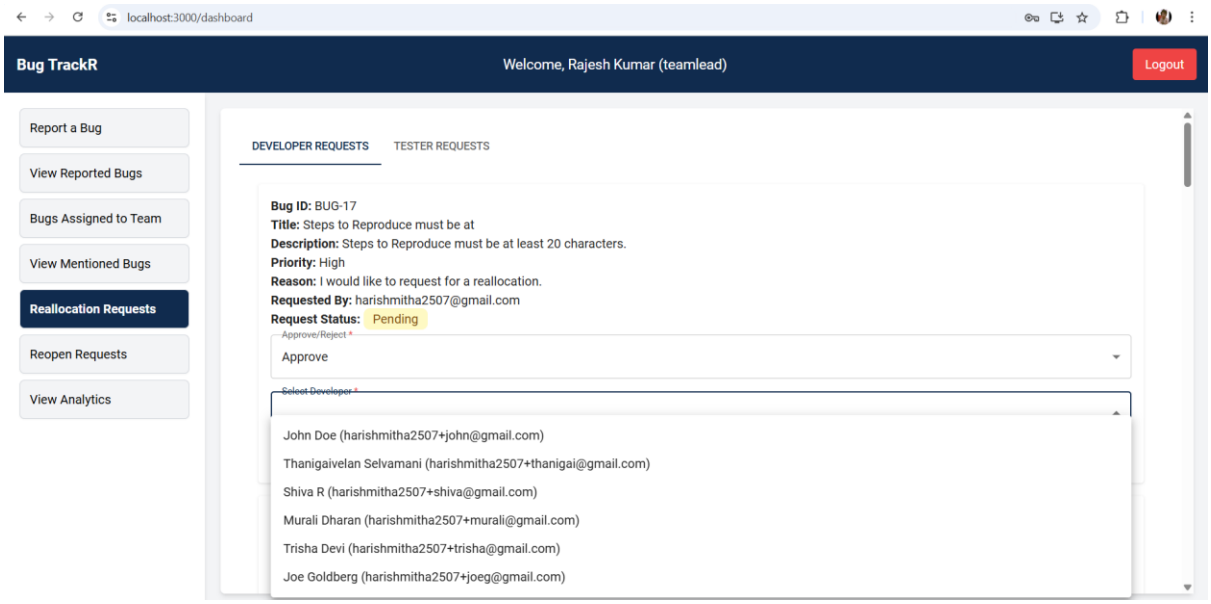


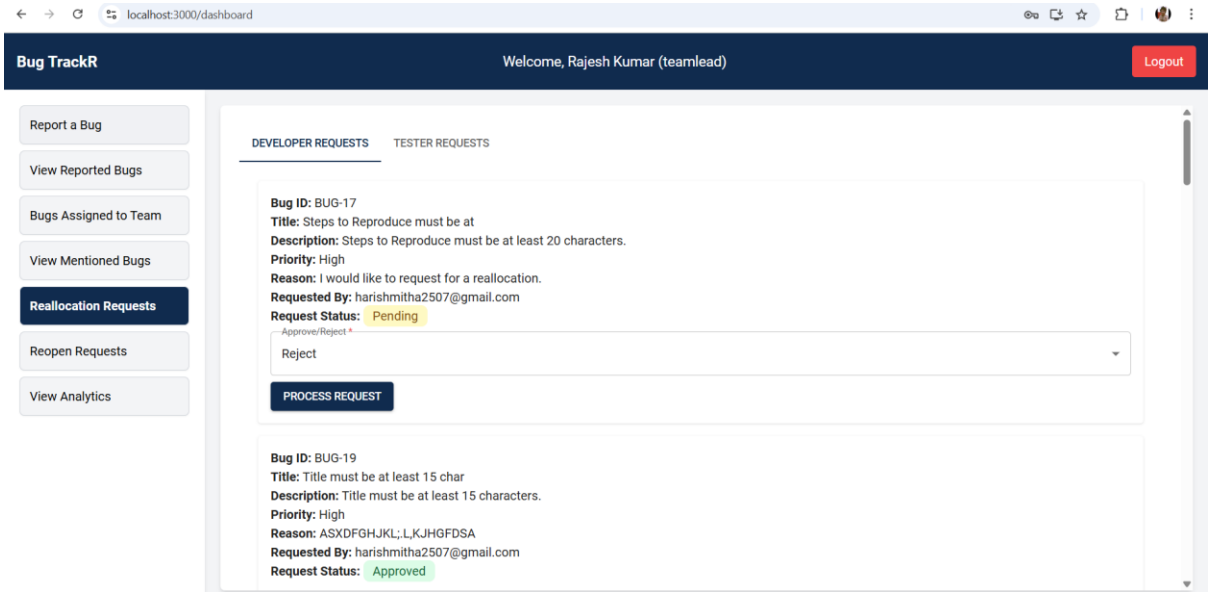


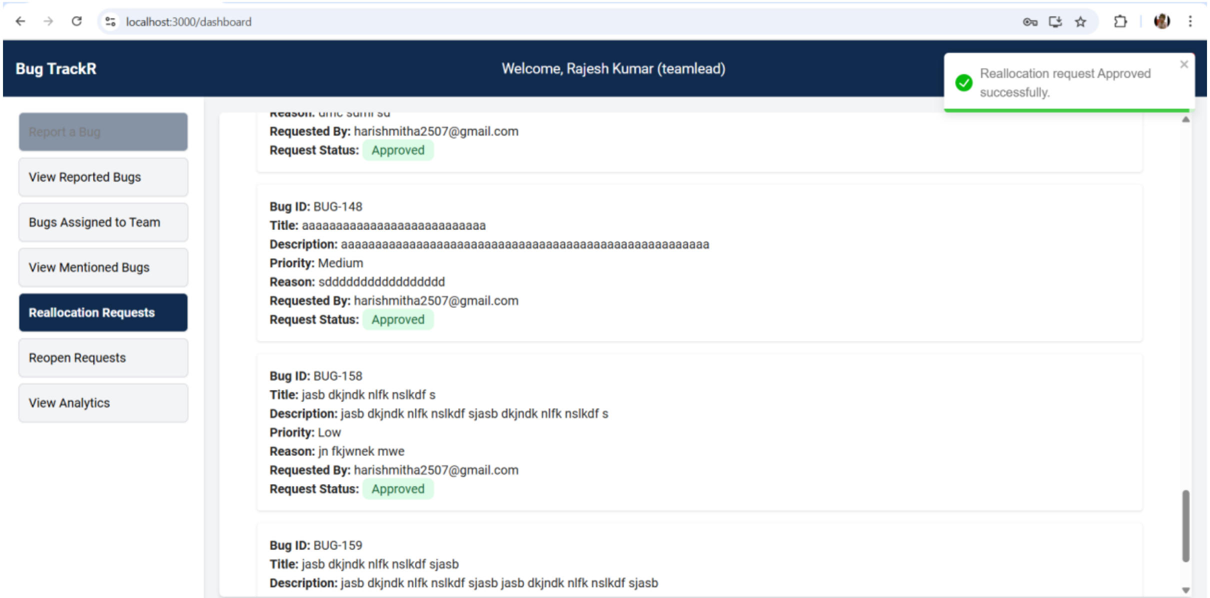


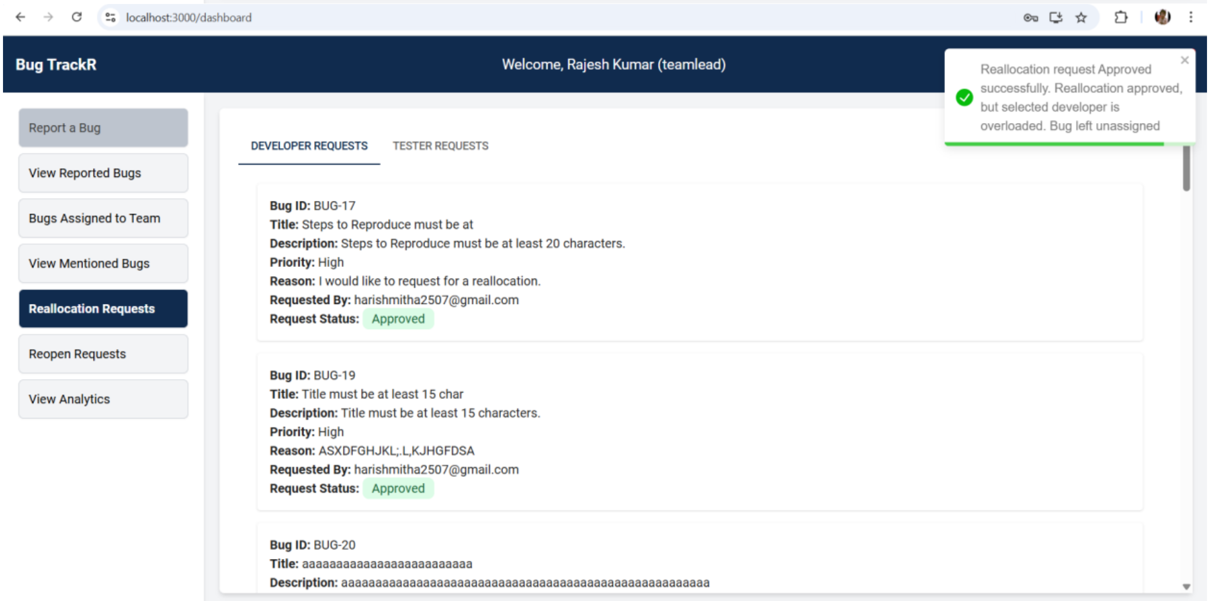
APPENDIX X – Team Lead/Admin review request reallocation



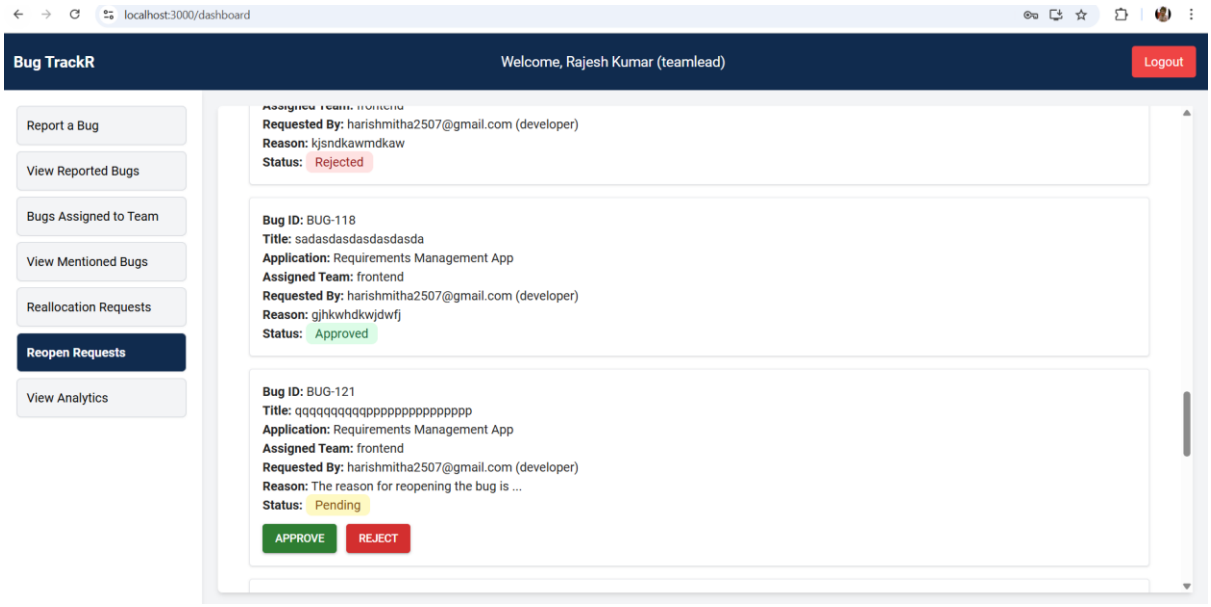

localhost:3000/dashboard
Bug TrackR
Welcome, Rajesh Kumar (teamlead)
Logout
Report a Bug
View Reported Bugs
Bugs Assigned to Team
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
DEVELOPER REQUESTS
TESTER REQUESTS
Bug ID: BUG-17
Title: Steps to Reproduce must be at
Description: Steps to Reproduce must be at least 20 characters.
Priority: High
Reason: I would like to request for a reallocation.
Requested By: harishmitha2507@gmail.com
Request Status: Pending
Approve/Reject *
Approve
Select Developer *
John Doe (harishmitha2507+john@gmail.com)
Thanigaivelan Selvamani (harishmitha2507+thanigai@gmail.com)
Shiva R (harishmitha2507+shiva@gmail.com)
Murali Dharan (harishmitha2507+murali@gmail.com)
Trisha Devi (harishmitha2507+trisha@gmail.com)
Joe Goldberg (harishmitha2507+joeg@gmail.com)

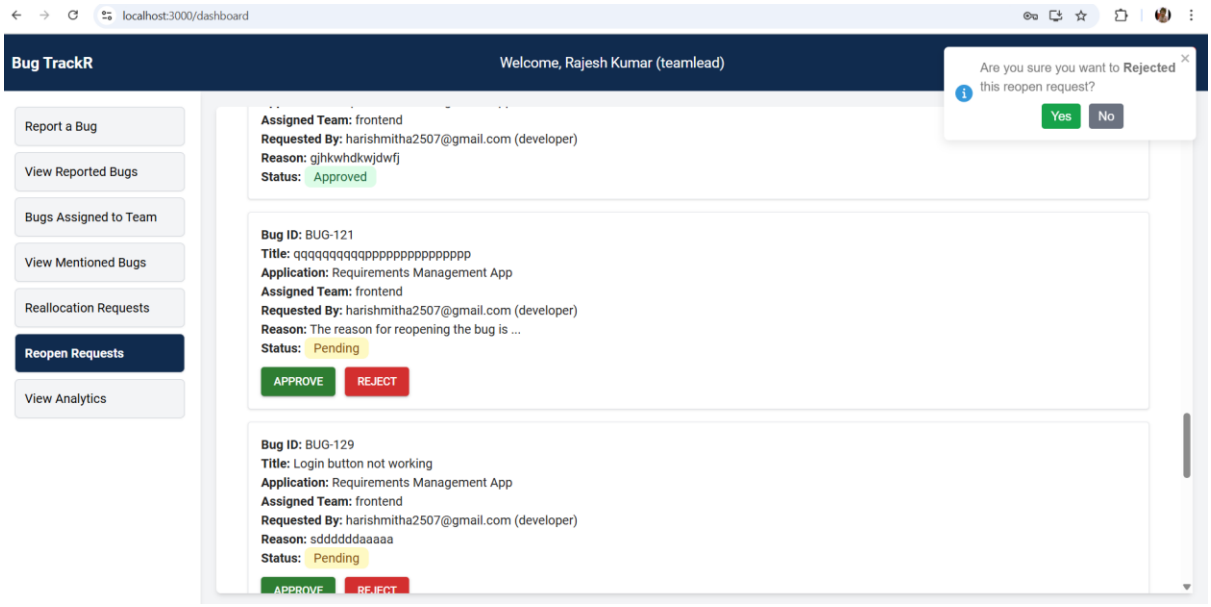

localhost:3000/dashboard
Bug TrackR
Welcome, Rajesh Kumar (teamlead)
Logout
Report a Bug
View Reported Bugs
Bugs Assigned to Team
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
DEVELOPER REQUESTS
TESTER REQUESTS
Bug ID: BUG-17
Title: Steps to Reproduce must be at
Description: Steps to Reproduce must be at least 20 characters.
Priority: High
Reason: I would like to request for a reallocation.
Requested By: harishmitha2507@gmail.com
Request Status: Pending
Approve/Reject *
Reject
PROCESS REQUEST
Bug ID: BUG-19
Title: Title must be at least 15 char
Description: Title must be at least 15 characters.
Priority: High
Reason: ASXDFGHJKL;.L,KJHGFDSA
Requested By: harishmitha2507@gmail.com
Request Status: Approved

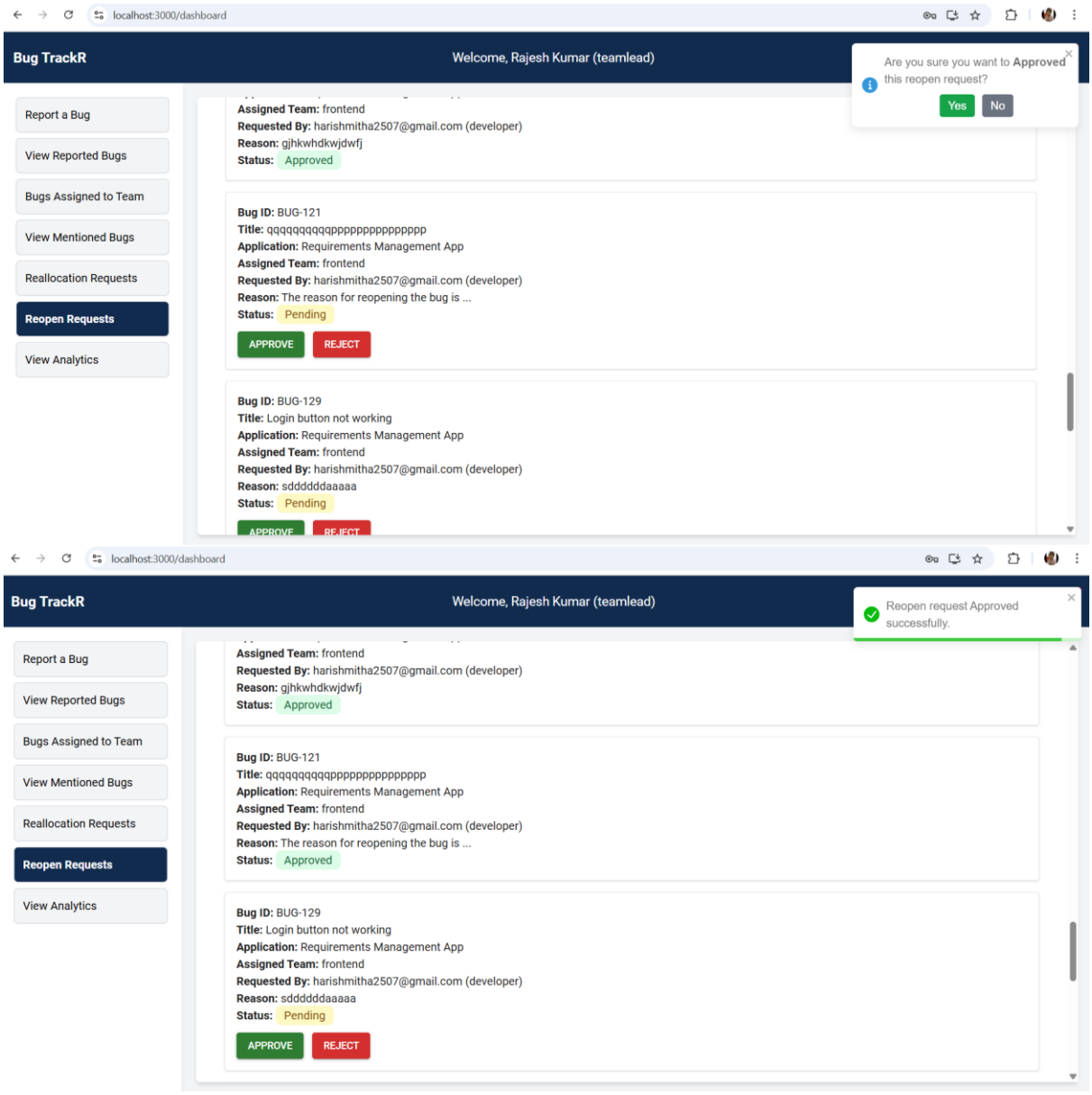

localhost:3000/dashboard
Bug TrackR
Welcome, Rajesh Kumar (teamlead)
Reallocation request Approved successfully.
Report a Bug
View Reported Bugs
Bugs Assigned to Team
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
Requested By: harishmitha2507@gmail.com
Request Status: Approved
Bug ID: BUG-148
Title: aaaaaaaaaaaaaaaaaaaaaaaaaa
Description: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
Priority: Medium
Reason: sdddddddddddddddd
Requested By: harishmitha2507@gmail.com
Request Status: Approved
Bug ID: BUG-158
Title: jasb dkjndk nlfk nslkdf s
Description: jasb dkjndk nlfk nslkdf sjasb dkjndk nlfk nslkdf s
Priority: Low
Reason: jn fkjwnek mwe
Requested By: harishmitha2507@gmail.com
Request Status: Approved
Bug ID: BUG-159
Title: jasb dkjndk nlfk nslkdf sjasb
Description: jasb dkjndk nlfk nslkdf sjasb jasb dkjndk nlfk nslkdf sjasb


localhost:3000/dashboard
Bug TrackR
Welcome, Rajesh Kumar (teamlead)
Reallocation request Approved successfully. Reallocation approved, but selected developer is overloaded. Bug left unassigned
Report a Bug
View Reported Bugs
Bugs Assigned to Team
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
DEVELOPER REQUESTS
TESTER REQUESTS
Bug ID: BUG-17
Title: Steps to Reproduce must be at
Description: Steps to Reproduce must be at least 20 characters.
Priority: High
Reason: I would like to request for a reallocation.
Requested By: harishmitha2507@gmail.com
Request Status: Approved
Bug ID: BUG-19
Title: Title must be at least 15 char
Description: Title must be at least 15 characters.
Priority: High
Reason: ASXDFGHJKL;.L,KJHGFDSA
Requested By: harishmitha2507@gmail.com
Request Status: Approved
Bug ID: BUG-20
Title: aaaaaaaaaaaaaaaaaaaaaaaa
Description: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa

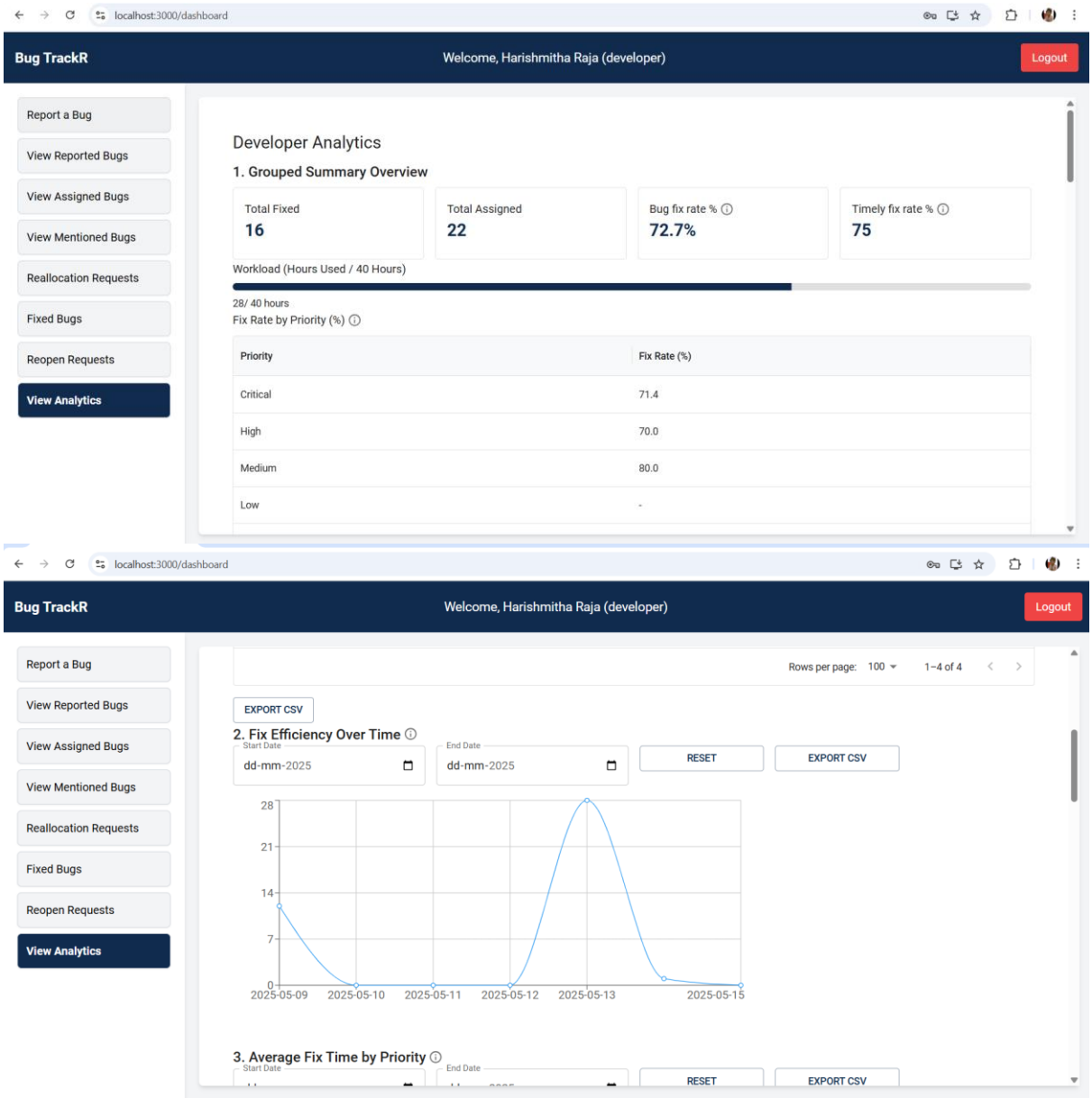
APPENDIX Y – Team Lead/Admin review reopen request

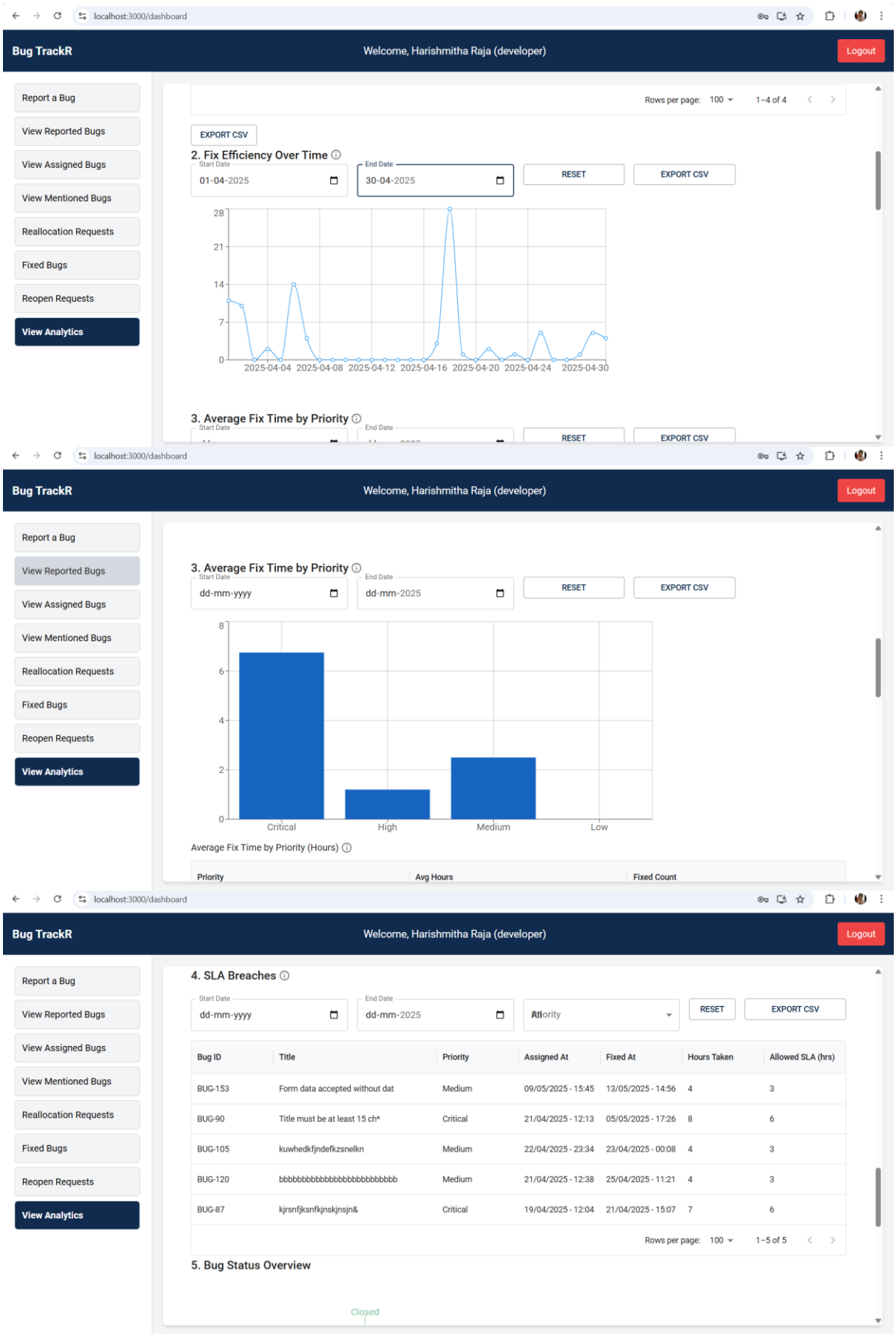


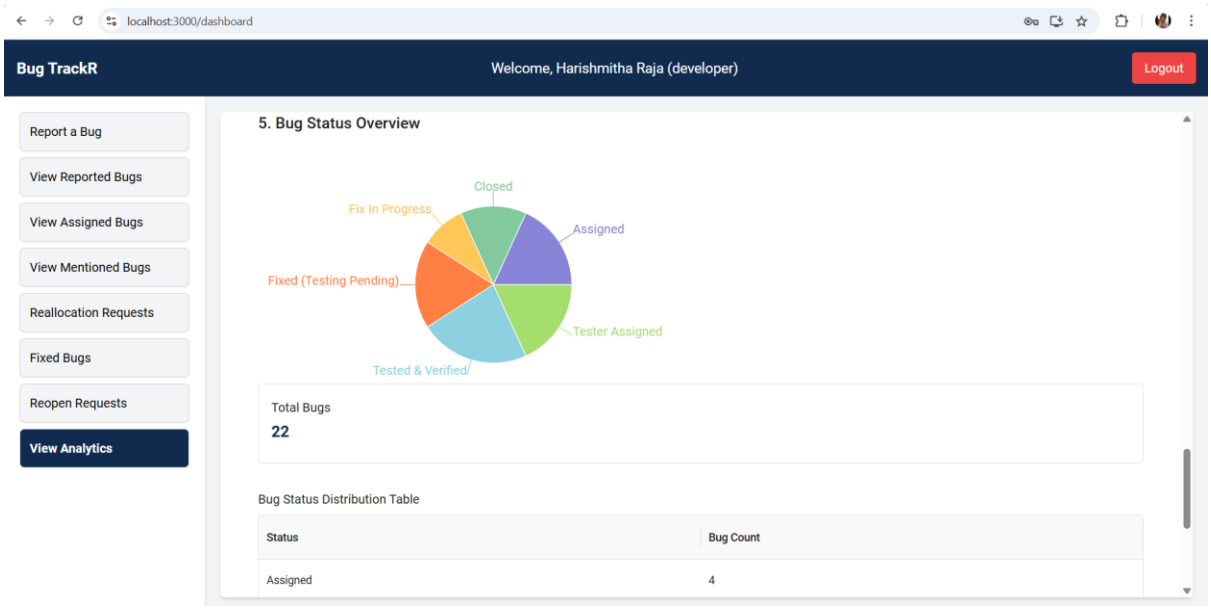


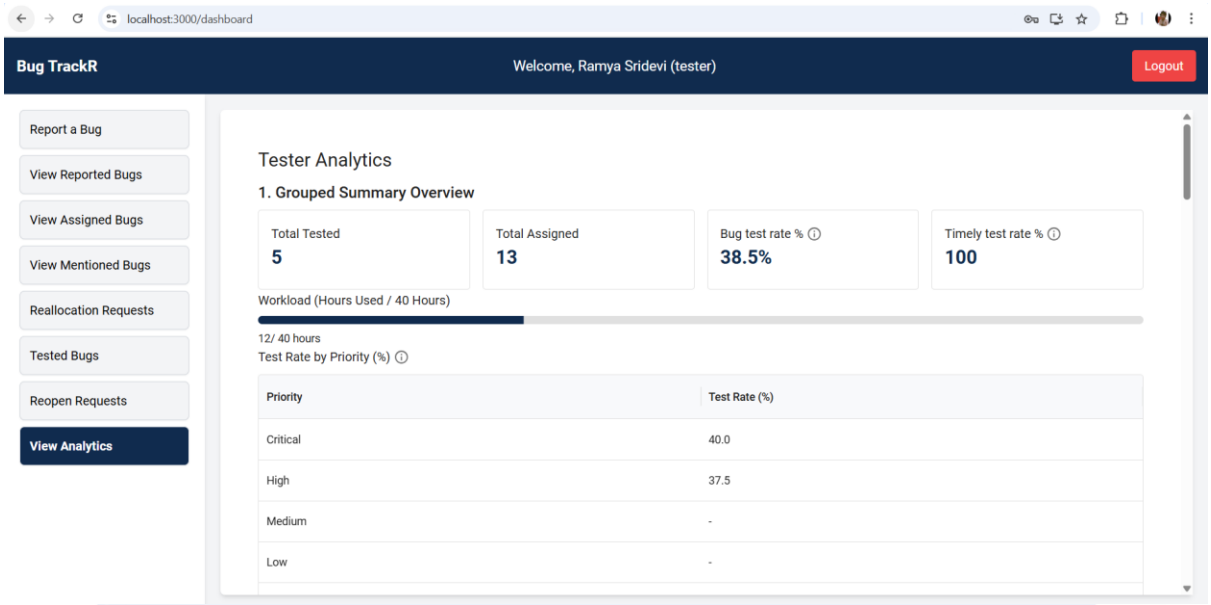

localhost:3000/dashboard
Bug TrackR
Welcome, Rajesh Kumar (teamlead)
Are you sure you want to Approved this reopen request?
Yes
No
Report a Bug
View Reported Bugs
Bugs Assigned to Team
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
Assigned Team: frontend
Requested By: harishmitha2507@gmail.com (developer)
Reason: gjhkwhdkwjdwfj
Status: Approved
Bug ID: BUG-121
Title: qqqqqqqqqqpppppppppppppp
Application: Requirements Management App
Assigned Team: frontend
Requested By: harishmitha2507@gmail.com (developer)
Reason: The reason for reopening the bug is ...
Status: Pending
APPROVE
REJECT
Bug ID: BUG-129
Title: Login button not working
Application: Requirements Management App
Assigned Team: frontend
Requested By: harishmitha2507@gmail.com (developer)
Reason: sddddddaaaaa
Status: Pending
APPROVE
REJECT
localhost:3000/dashboard
Bug TrackR
Welcome, Rajesh Kumar (teamlead)
Reopen request Approved successfully.
Report a Bug
View Reported Bugs
Bugs Assigned to Team
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
Assigned Team: frontend
Requested By: harishmitha2507@gmail.com (developer)
Reason: gjhkwhdkwjdwfj
Status: Approved
Bug ID: BUG-121
Title: qqqqqqqqqqpppppppppppppp
Application: Requirements Management App
Assigned Team: frontend
Requested By: harishmitha2507@gmail.com (developer)
Reason: The reason for reopening the bug is ...
Status: Approved
Bug ID: BUG-129
Title: Login button not working
Application: Requirements Management App
Assigned Team: frontend
Requested By: harishmitha2507@gmail.com (developer)
Reason: sddddddaaaaa
Status: Pending
APPROVE
REJECT

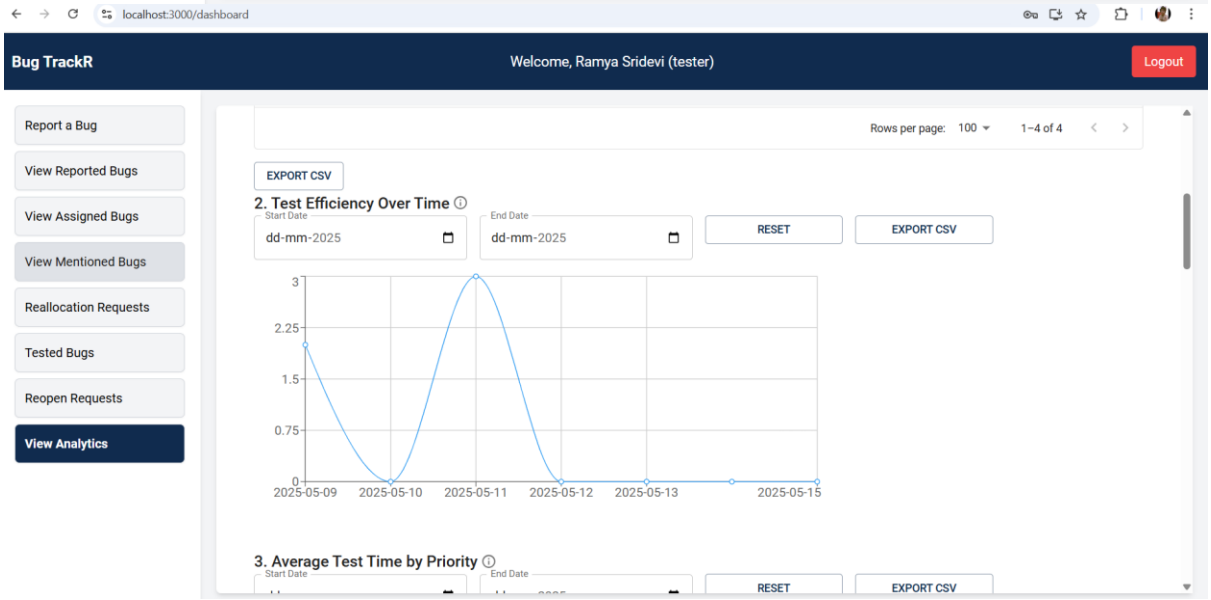
APPENDIX Z - Tech users analytics

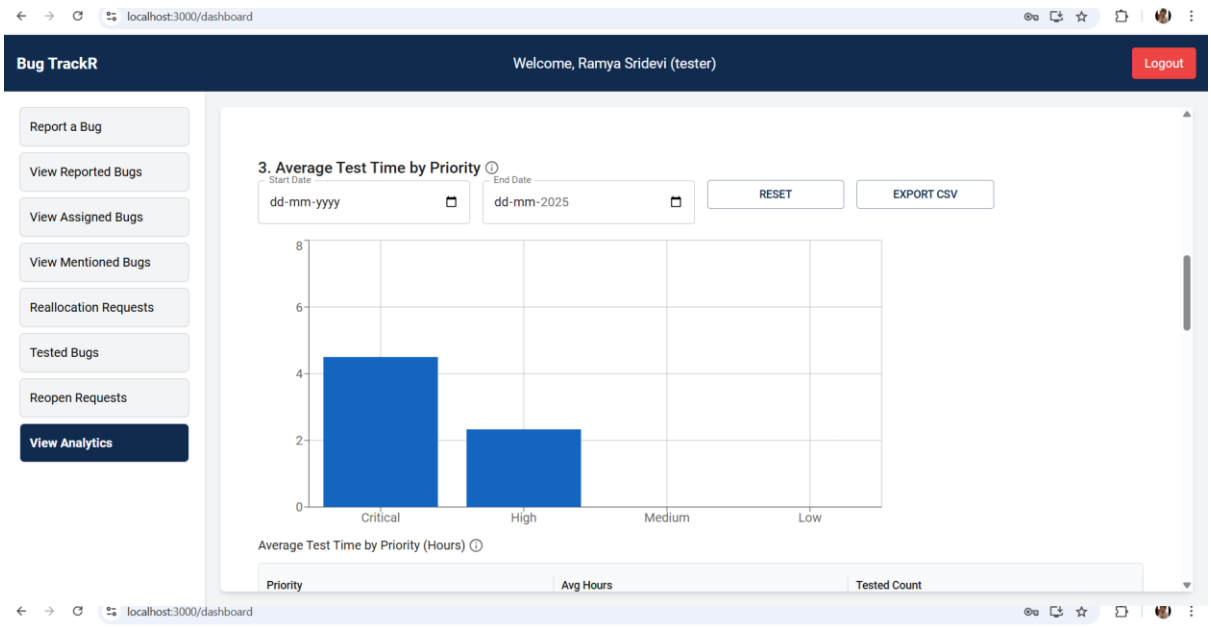


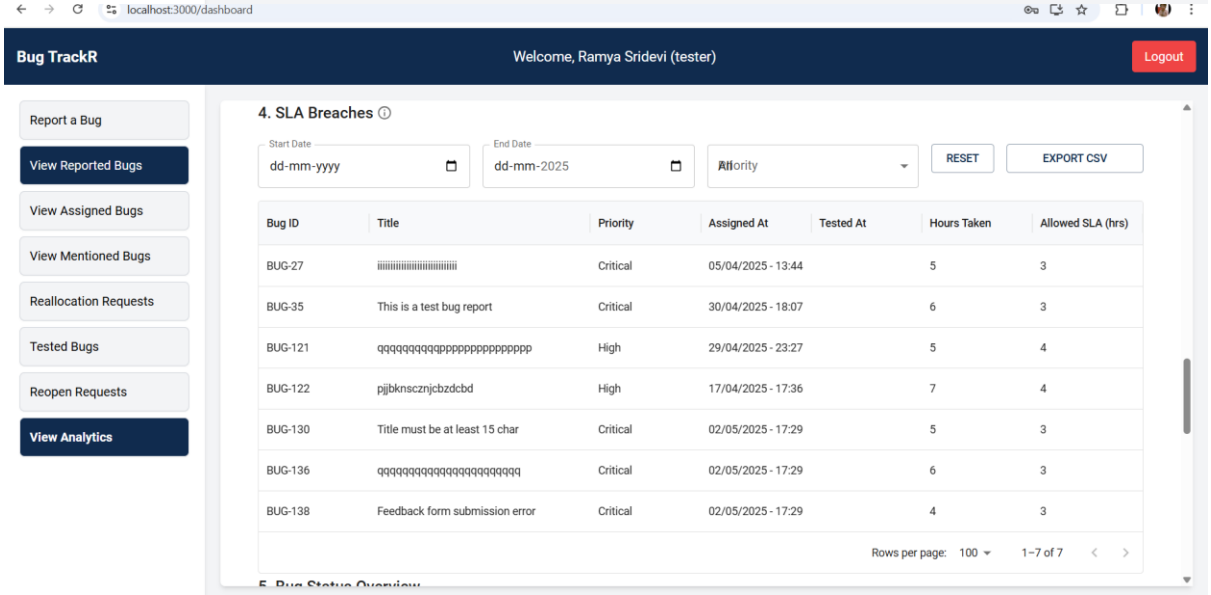

localhost:3000/dashboard
Bug TrackR
Welcome, Harishmitha Raja (developer)
Logout
Report a Bug
View Reported Bugs
View Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Fixed Bugs
Reopen Requests
View Analytics
Rows per page: 100
1–4 of 4
EXPORT CSV
2. Fix Efficiency Over Time
Start Date
01-04-2025
End Date
30-04-2025
RESET
EXPORT CSV
28
21
14
7
0
2025-04-04
2025-04-08
2025-04-12
2025-04-16
2025-04-20
2025-04-24
2025-04-30
3. Average Fix Time by Priority
dd-mm-yyyy
dd-mm-2025
8
6
4
2
0
Critical
High
Medium
Low
Average Fix Time by Priority (Hours)
Priority
Avg Hours
Fixed Count
4. SLA Breaches
All
Bug ID
Title
Priority
Assigned At
Fixed At
Hours Taken
Allowed SLA (hrs)
BUG-153
Form data accepted without dat
Medium
09/05/2025 - 15:45
13/05/2025 - 14:56
4
3
BUG-90
Title must be at least 15 ch*
Critical
21/04/2025 - 12:13
05/05/2025 - 17:26
8
6
BUG-105
kuwhedkfjndefkzsnelkn
Medium
22/04/2025 - 23:34
23/04/2025 - 00:08
4
3
BUG-120
bbbbbbbbbbbbbbbbbbbbbbbb
Medium
21/04/2025 - 12:38
25/04/2025 - 11:21
4
3
BUG-87
kjrsnfjksnfkjnskjnsjn&
Critical
19/04/2025 - 12:04
21/04/2025 - 15:07
7
6
Rows per page: 100
1–5 of 5
5. Bug Status Overview
Closed

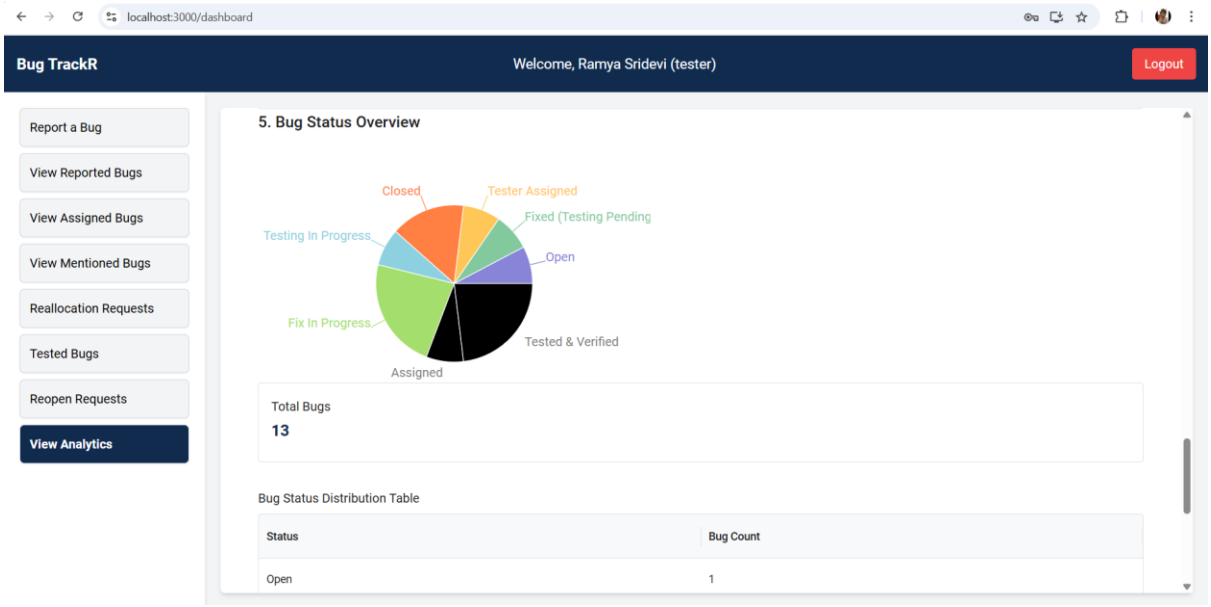

localhost:3000/dashboard
Bug TrackR
Welcome, Harishmitha Raja (developer)
Logout
Report a Bug
View Reported Bugs
View Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Fixed Bugs
Reopen Requests
View Analytics
5. Bug Status Overview
Closed
Fix In Progress
Assigned
Fixed (Testing Pending)
Tester Assigned
Tested & Verified
Total Bugs
22
Bug Status Distribution Table
Status
Bug Count
Assigned
4

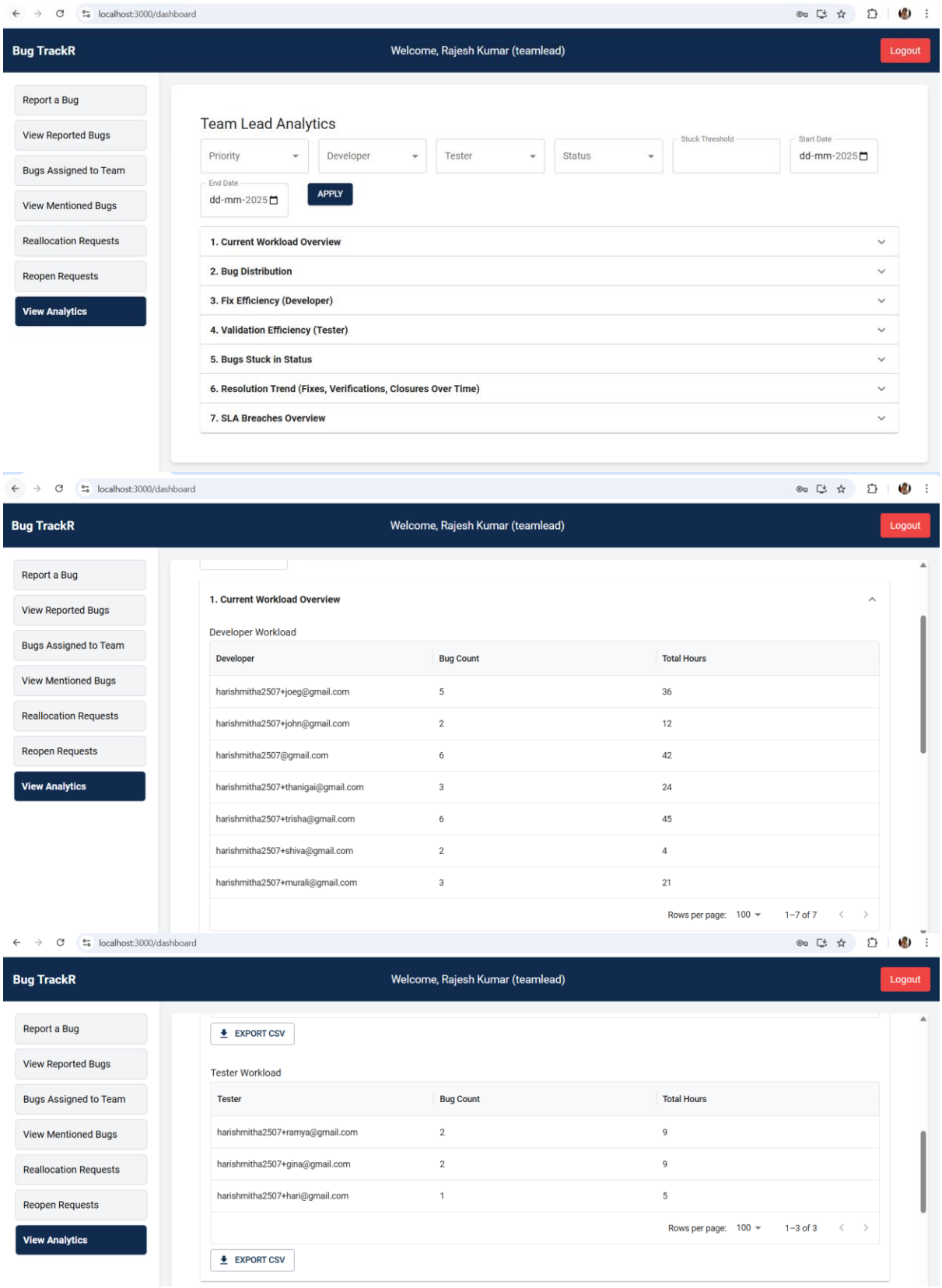

localhost:3000/dashboard
Bug TrackR
Welcome, Ramya Sridevi (tester)
Logout
Report a Bug
View Reported Bugs
View Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Tested Bugs
Reopen Requests
View Analytics
Tester Analytics
1. Grouped Summary Overview
Total Tested
5
Total Assigned
13
Bug test rate %
38.5%
Timely test rate %
100
Workload (Hours Used / 40 Hours)
12/ 40 hours
Test Rate by Priority (%)
Priority
Test Rate (%)
Critical
40.0
High
37.5
Medium
-
Low
-

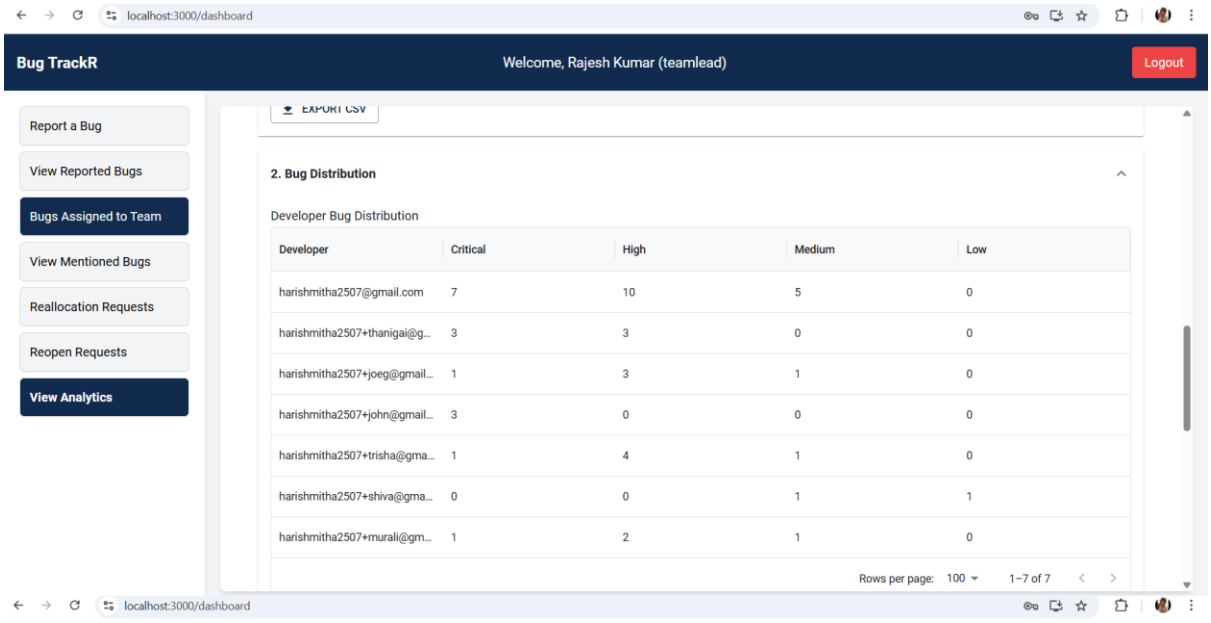

localhost:3000/dashboard
Bug TrackR
Welcome, Ramya Sridevi (tester)
Logout
Report a Bug
View Reported Bugs
View Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Tested Bugs
Reopen Requests
View Analytics
Rows per page: 100
1–4 of 4
EXPORT CSV
2. Test Efficiency Over Time
Start Date
dd-mm-2025
End Date
dd-mm-2025
RESET
EXPORT CSV
3
2.25
1.5
0.75
0
2025-05-09
2025-05-10
2025-05-11
2025-05-12
2025-05-13
2025-05-15
3. Average Test Time by Priority
Start Date
End Date
RESET
EXPORT CSV


localhost:3000/dashboard
Bug TrackR
Welcome, Ramya Sridevi (tester)
Logout
Report a Bug
View Reported Bugs
View Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Tested Bugs
Reopen Requests
View Analytics
3. Average Test Time by Priority
Start Date
dd-mm-yyyy
End Date
dd-mm-2025
RESET
EXPORT CSV
8
6
4
2
0
Critical
High
Medium
Low
Average Test Time by Priority (Hours)
Priority
Avg Hours
Tested Count

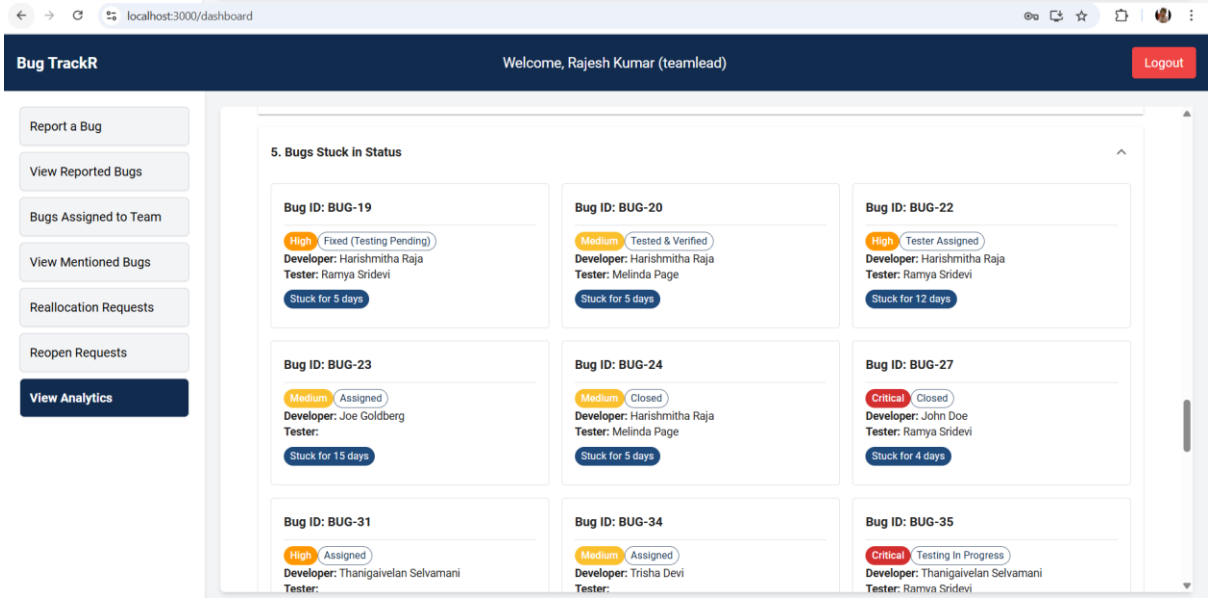

localhost:3000/dashboard
Bug TrackR
Welcome, Ramya Sridevi (tester)
Logout
Report a Bug
View Reported Bugs
View Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Tested Bugs
Reopen Requests
View Analytics
4. SLA Breaches
Start Date
dd-mm-yyyy
End Date
dd-mm-2025
RESET
EXPORT CSV
Bug ID
Title
Priority
Assigned At
Tested At
Hours Taken
Allowed SLA (hrs)
BUG-27
Critical
05/04/2025 - 13:44
5
3
BUG-35
This is a test bug report
Critical
30/04/2025 - 18:07
6
3
BUG-121
High
29/04/2025 - 23:27
5
4
BUG-122
pjjbknscznjcbzdcbd
High
17/04/2025 - 17:36
7
4
BUG-130
Title must be at least 15 char
Critical
02/05/2025 - 17:29
5
3
BUG-136
Critical
02/05/2025 - 17:29
6
3
BUG-138
Feedback form submission error
Critical
02/05/2025 - 17:29
4
3
Rows per page: 100
1–7 of 7

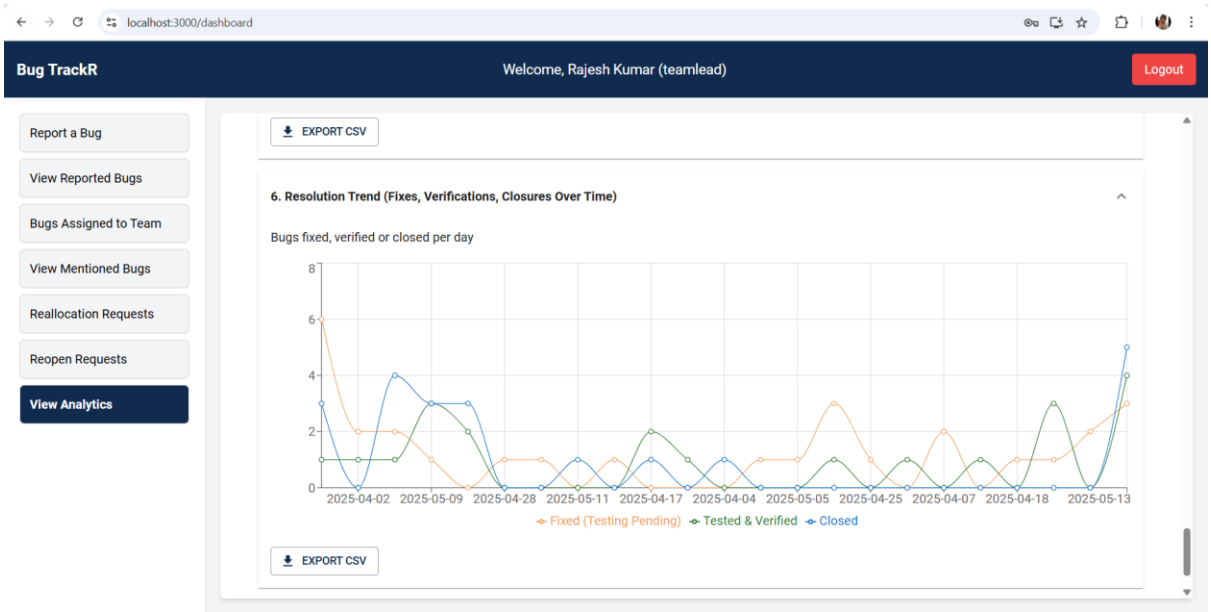

localhost:3000/dashboard
Bug TrackR
Welcome, Ramya Sridevi (tester)
Logout
Report a Bug
View Reported Bugs
View Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Tested Bugs
Reopen Requests
View Analytics
5. Bug Status Overview
Closed
Tester Assigned
Fixed (Testing Pending
Testing In Progress
Open
Fix In Progress
Tested & Verified
Assigned
Total Bugs
13
Bug Status Distribution Table
Status
Bug Count
Open
1

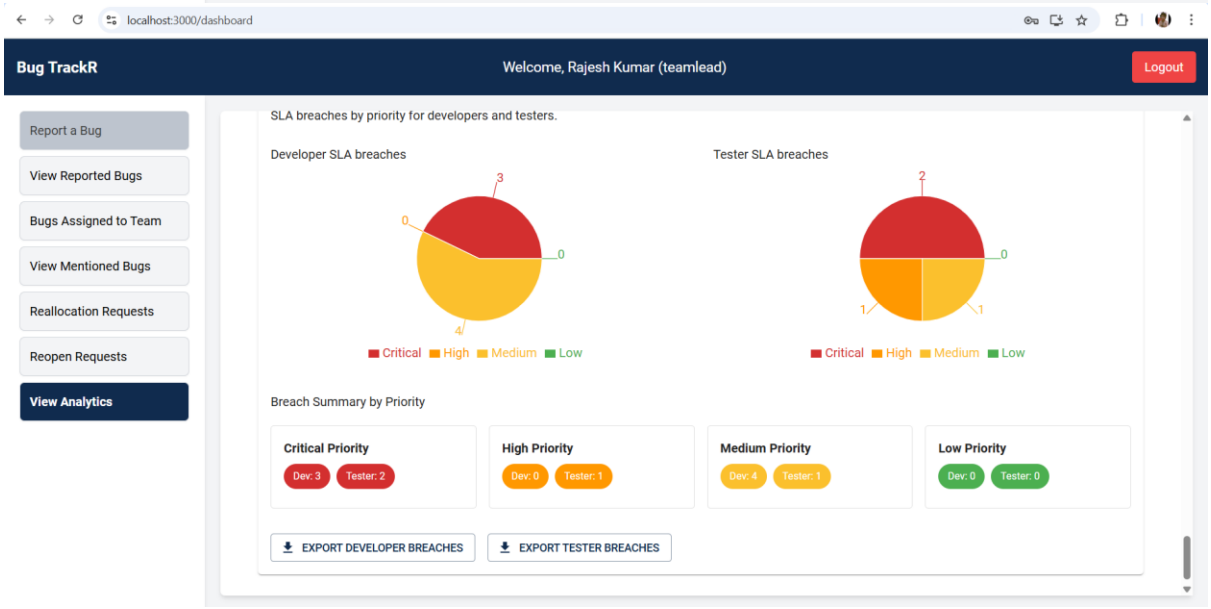

localhost:3000/dashboard
Bug TrackR
Welcome, Rajesh Kumar (teamlead)
Logout
Report a Bug
View Reported Bugs
Bugs Assigned to Team
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
Team Lead Analytics
Priority
Developer
Tester
Status
Stuck Threshold
Start Date
dd-mm-2025
End Date
dd-mm-2025
APPLY
1. Current Workload Overview
2. Bug Distribution
3. Fix Efficiency (Developer)
4. Validation Efficiency (Tester)
5. Bugs Stuck in Status
6. Resolution Trend (Fixes, Verifications, Closures Over Time)
7. SLA Breaches Overview
1. Current Workload Overview
Developer Workload
Developer
Bug Count
Total Hours
harishmitha2507+joeg@gmail.com 5 36
harishmitha2507+john@gmail.com 2 12
harishmitha2507@gmail.com 6 42
harishmitha2507+thanigai@gmail.com 3 24
harishmitha2507+trisha@gmail.com 6 45
harishmitha2507+shiva@gmail.com 2 4
harishmitha2507+murali@gmail.com 3 21
Rows per page: 100
1–7 of 7
EXPORT CSV
Tester Workload
Tester
Bug Count
Total Hours
harishmitha2507+ramya@gmail.com 2 9
harishmitha2507+gina@gmail.com 2 9
harishmitha2507+hari@gmail.com 1 5
Rows per page: 100
1–3 of 3
EXPORT CSV

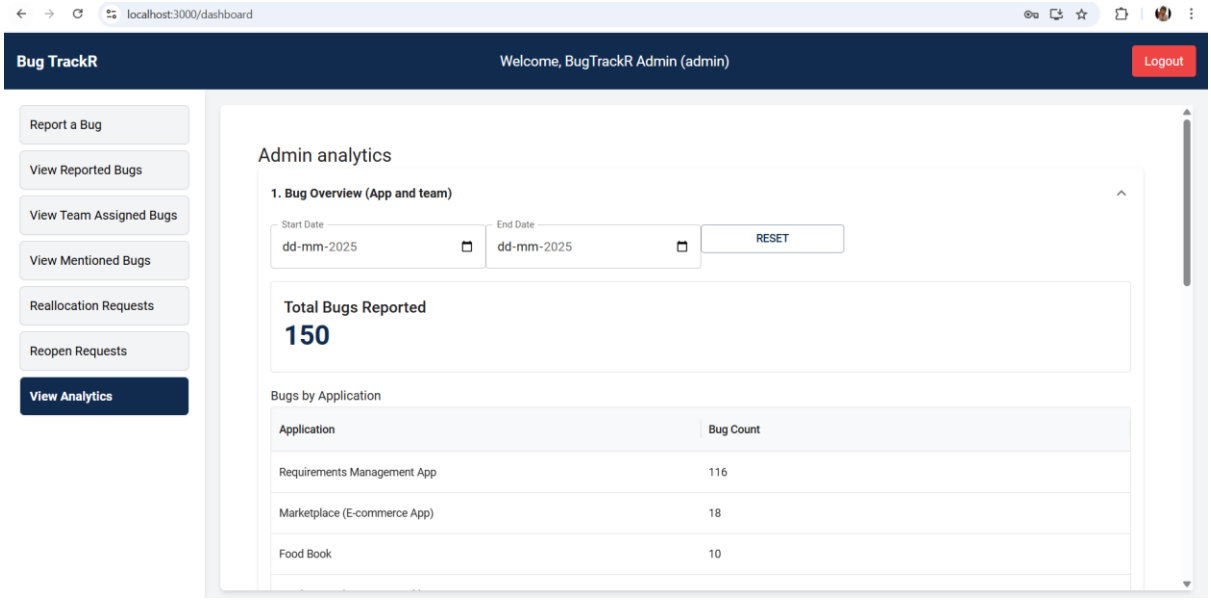

localhost:3000/dashboard
Bug TrackR
Welcome, Rajesh Kumar (teamlead)
Logout
Report a Bug
View Reported Bugs
Bugs Assigned to Team
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
EXPORT CSV
2. Bug Distribution
Developer Bug Distribution
Developer
Critical
High
Medium
Low
harishmitha2507@gmail.com 7 10 5 0
harishmitha2507+thanigai@g... 3 3 0 0
harishmitha2507+joeg@gmail... 1 3 1 0
harishmitha2507+john@gmail... 3 0 0 0
harishmitha2507+trisha@gma... 1 4 1 0
harishmitha2507+shiva@gma... 0 0 1 1
harishmitha2507+murali@gm... 1 2 1 0
Rows per page: 100
1–7 of 7

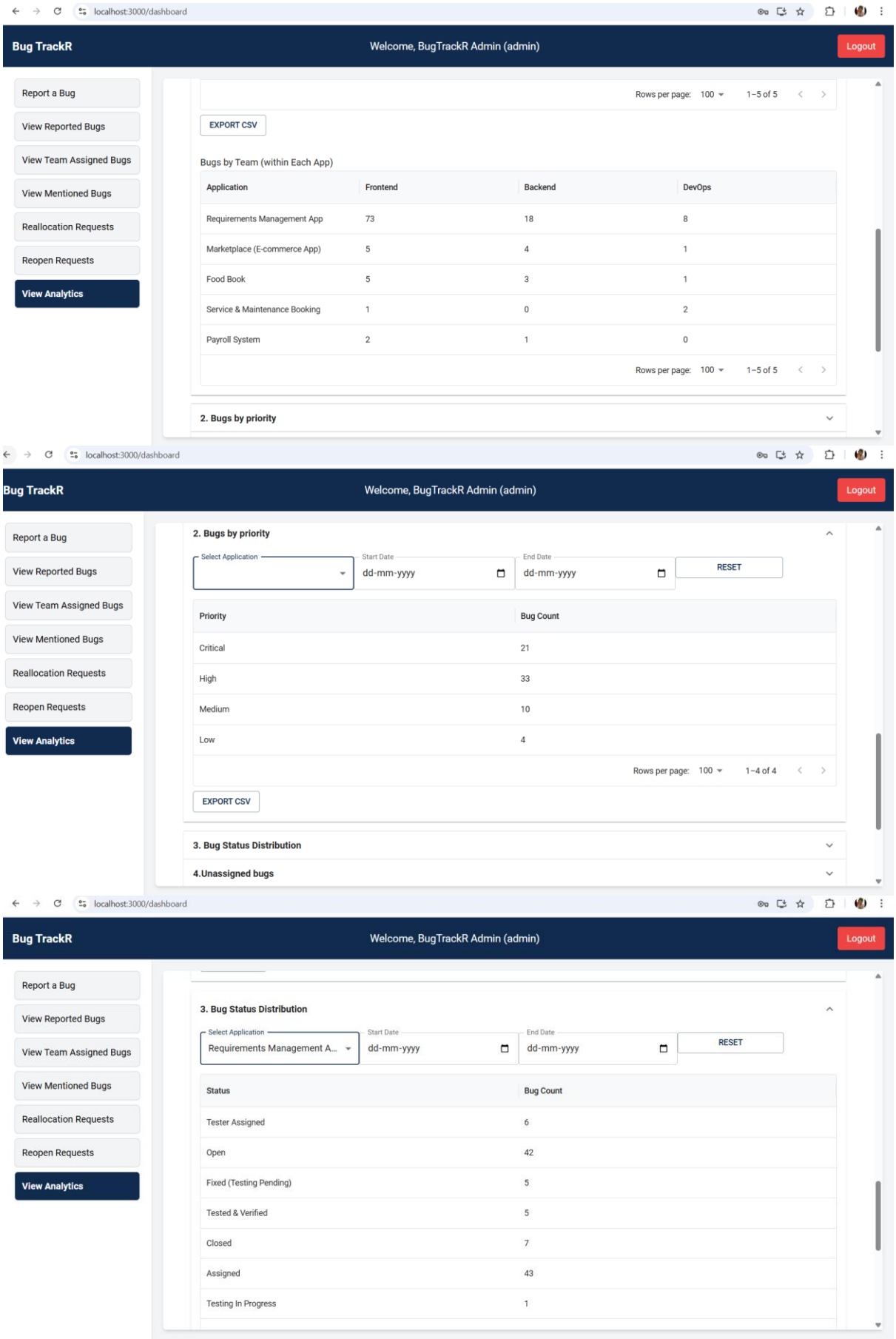

localhost:3000/dashboard
Bug TrackR
Welcome, Rajesh Kumar (teamlead)
Logout
Report a Bug
View Reported Bugs
Bugs Assigned to Team
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
EXPORT CSV
3. Fix Efficiency (Developer)
Average time taken by developer (in hours)
8
6
4
2
0
Harishmitha Raja
Shiva R
John Doe
Trisha Devi
Murali Dharan
Avg Hours
EXPORT CSV
4. Validation Efficiency (Tester)

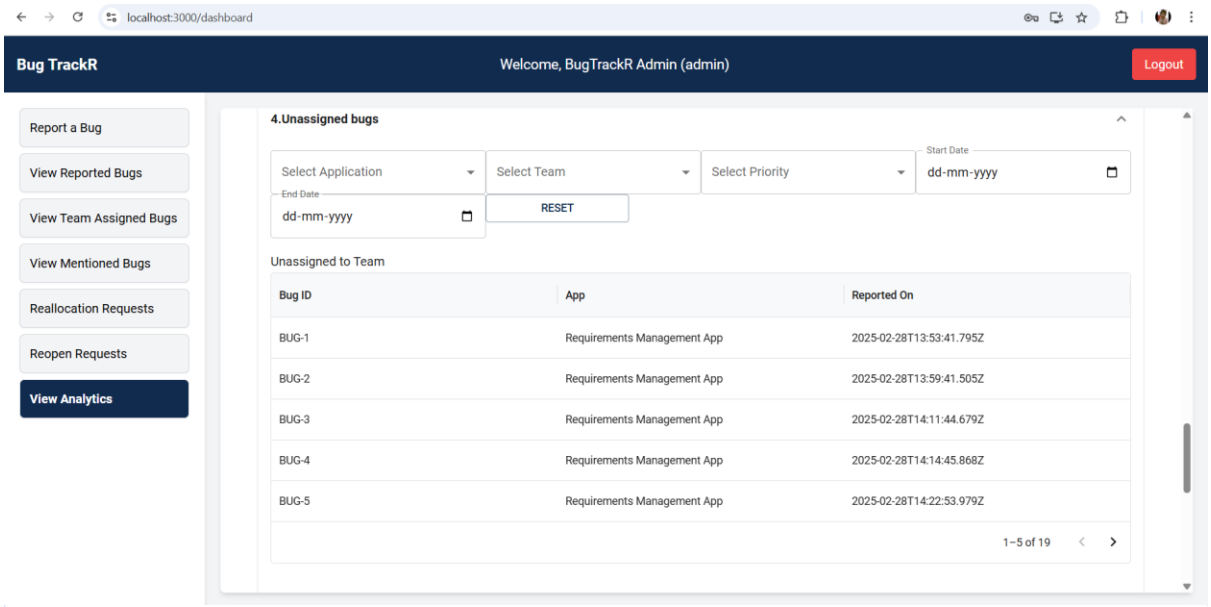

localhost:3000/dashboard
Bug TrackR
Welcome, Rajesh Kumar (teamlead)
Logout
Report a Bug
View Reported Bugs
Bugs Assigned to Team
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
5. Bugs Stuck in Status
Bug ID: BUG-19
High Fixed (Testing Pending)
Developer: Harishmitha Raja
Tester: Ramya Sridevi
Stuck for 5 days
Bug ID: BUG-20
Medium Tested & Verified
Developer: Harishmitha Raja
Tester: Melinda Page
Stuck for 5 days
Bug ID: BUG-22
High Tester Assigned
Developer: Harishmitha Raja
Tester: Ramya Sridevi
Stuck for 12 days
Bug ID: BUG-23
Medium Assigned
Developer: Joe Goldberg
Tester:
Stuck for 15 days
Bug ID: BUG-24
Medium Closed
Developer: Harishmitha Raja
Tester: Melinda Page
Stuck for 5 days
Bug ID: BUG-27
Critical Closed
Developer: John Doe
Tester: Ramya Sridevi
Stuck for 4 days
Bug ID: BUG-31
High Assigned
Developer: Thanigaivelan Selvamani
Tester:
Bug ID: BUG-34
Medium Assigned
Developer: Trisha Devi
Tester:
Bug ID: BUG-35
Critical Testing In Progress
Developer: Thanigaivelan Selvamani
Tester: Ramya Sridevi

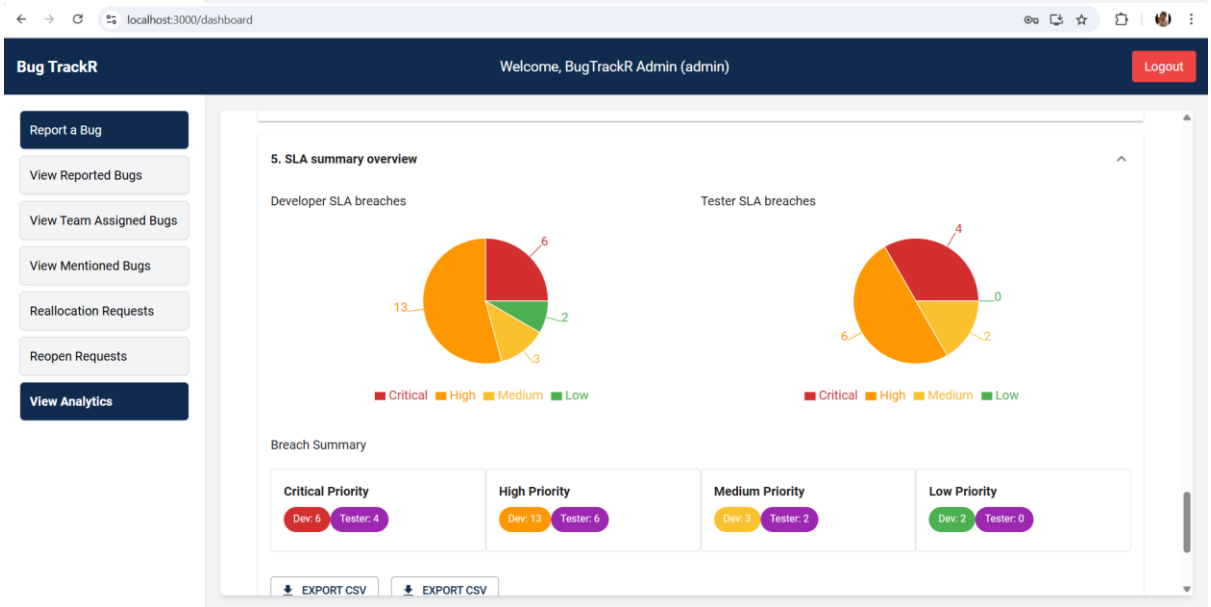

localhost:3000/dashboard
Bug TrackR
Welcome, Rajesh Kumar (teamlead)
Logout
Report a Bug
View Reported Bugs
Bugs Assigned to Team
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
EXPORT CSV
6. Resolution Trend (Fixes, Verifications, Closures Over Time)
Bugs fixed, verified or closed per day
8
6
4
2
0
2025-04-02
2025-05-09
2025-04-28
2025-05-11
2025-04-17
2025-04-04
2025-05-05
2025-04-25
2025-04-07
2025-04-18
2025-05-13
Fixed (Testing Pending)
Tested & Verified
Closed
EXPORT CSV

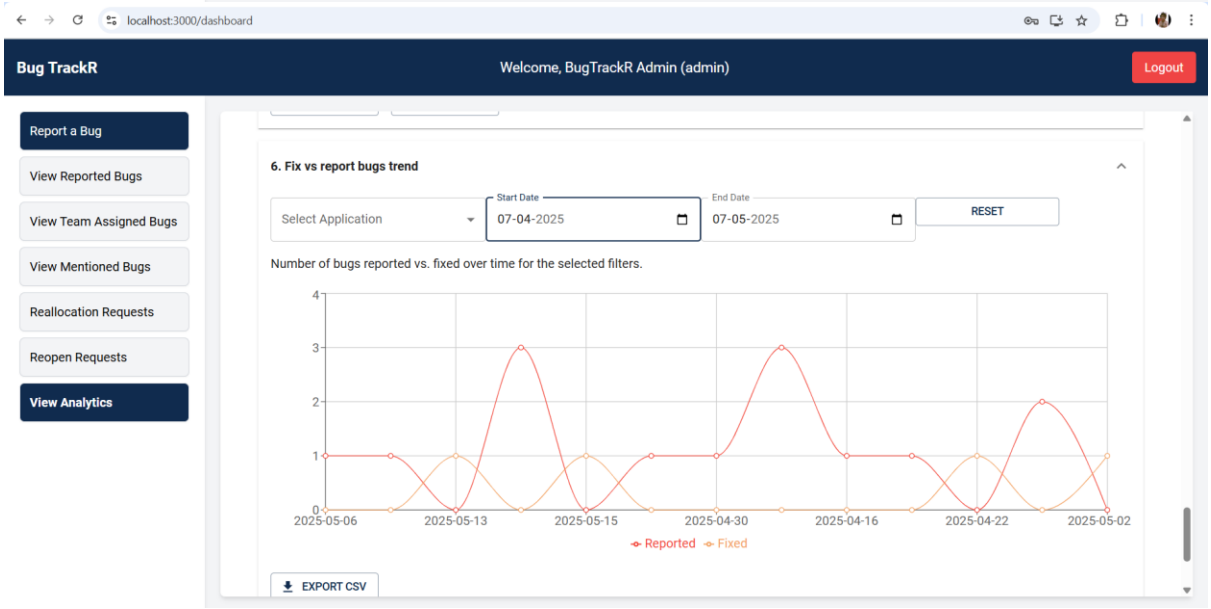

localhost:3000/dashboard
Bug TrackR
Welcome, Rajesh Kumar (teamlead)
Logout
Report a Bug
View Reported Bugs
Bugs Assigned to Team
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
SLA breaches by priority for developers and testers.
Developer SLA breaches
3
0
0
4
Critical
High
Medium
Low
Tester SLA breaches
2
0
1
1
Critical
High
Medium
Low
Breach Summary by Priority
Critical Priority
Dev: 3
Tester: 2
High Priority
Dev: 0
Tester: 1
Medium Priority
Dev: 4
Tester: 1
Low Priority
Dev: 0
Tester: 0
EXPORT DEVELOPER BREACHES
EXPORT TESTER BREACHES

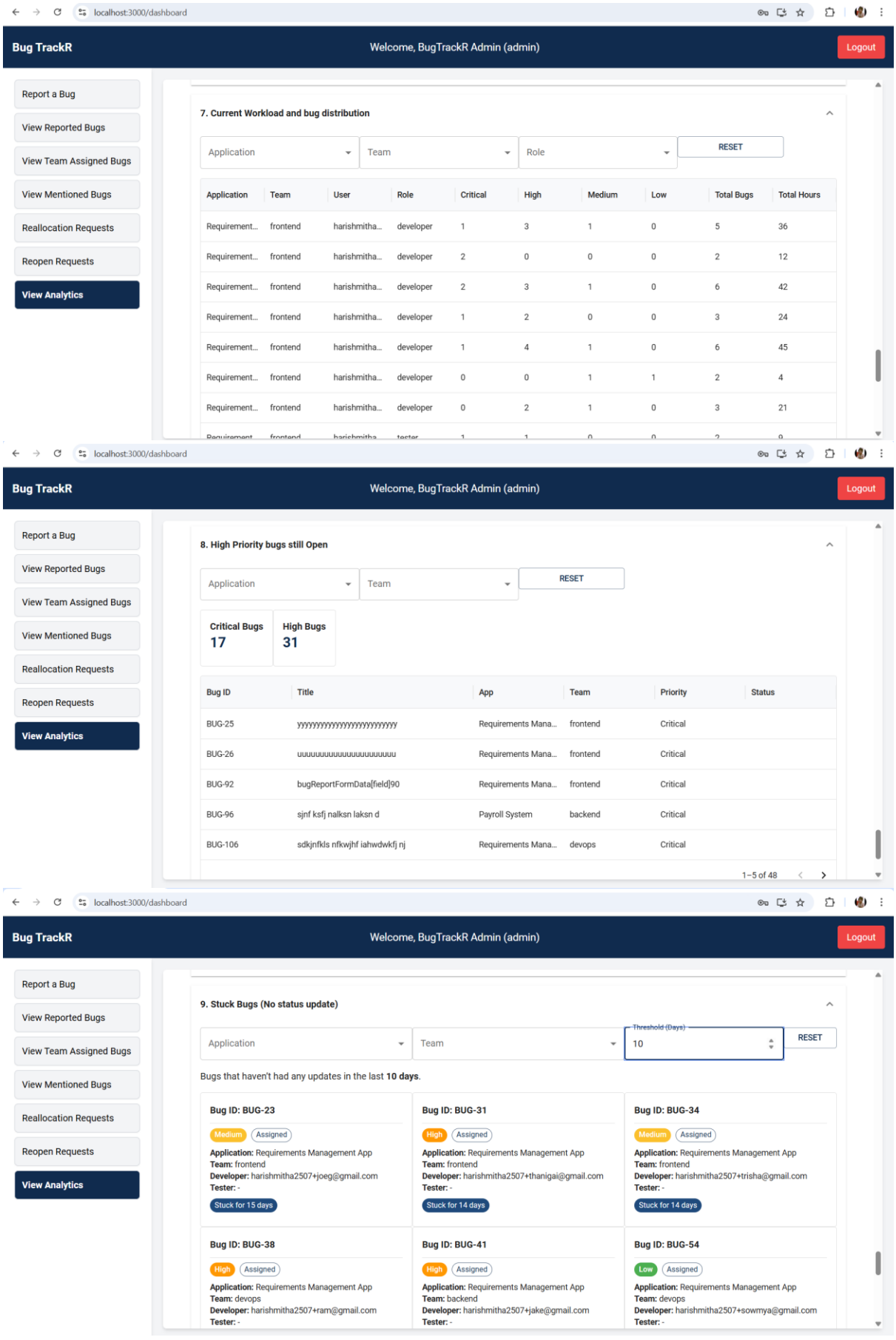

localhost:3000/dashboard
Bug TrackR
Welcome, BugTrackR Admin (admin)
Logout
Report a Bug
View Reported Bugs
View Team Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
Admin analytics
1. Bug Overview (App and team)
Start Date
dd-mm-2025
End Date
dd-mm-2025
RESET
Total Bugs Reported
150
Bugs by Application
Application
Bug Count
Requirements Management App
116
Marketplace (E-commerce App)
18
Food Book
10

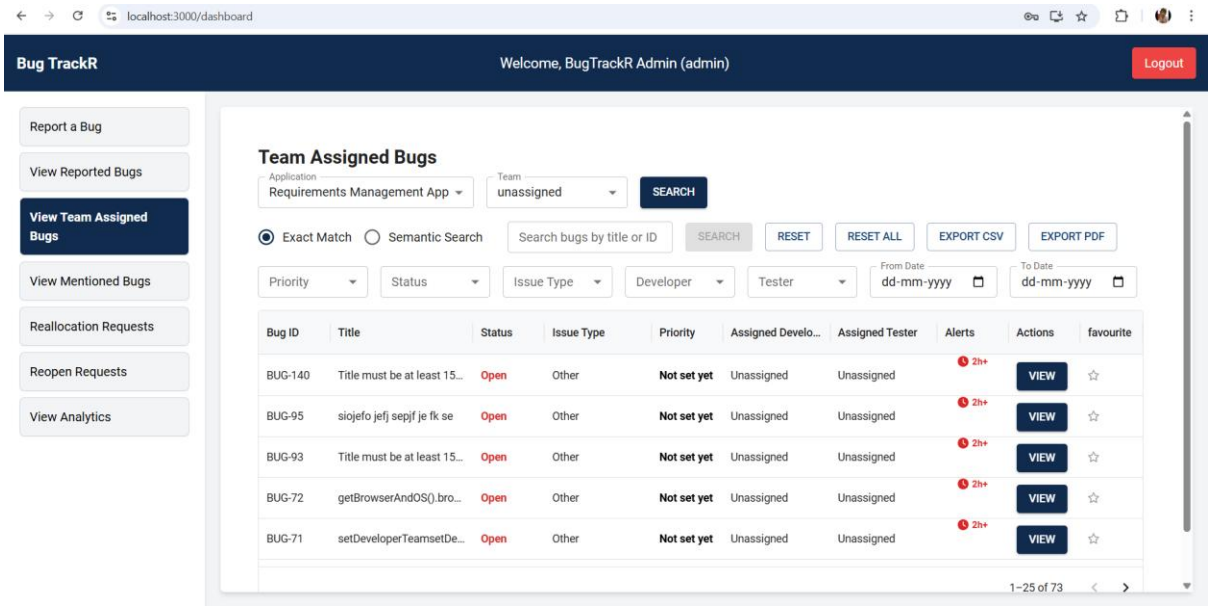

localhost:3000/dashboard
Bug TrackR
Welcome, BugTrackR Admin (admin)
Logout
Report a Bug
View Reported Bugs
View Team Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
Rows per page: 100
1–5 of 5
EXPORT CSV
Bugs by Team (within Each App)
Application
Frontend
Backend
DevOps
Requirements Management App
73
18
8
Marketplace (E-commerce App)
5
4
1
Food Book
5
3
1
Service & Maintenance Booking
1
0
2
Payroll System
2
1
0
Rows per page: 100
1–5 of 5
2. Bugs by priority
localhost:3000/dashboard
Bug TrackR
Welcome, BugTrackR Admin (admin)
Logout
Report a Bug
View Reported Bugs
View Team Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
2. Bugs by priority
Select Application
Start Date
dd-mm-yyyy
End Date
dd-mm-yyyy
RESET
Priority
Bug Count
Critical
21
High
33
Medium
10
Low
4
Rows per page: 100
1–4 of 4
EXPORT CSV
3. Bug Status Distribution
4.Unassigned bugs
localhost:3000/dashboard
Bug TrackR
Welcome, BugTrackR Admin (admin)
Logout
Report a Bug
View Reported Bugs
View Team Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
3. Bug Status Distribution
Select Application
Requirements Management A...
Start Date
dd-mm-yyyy
End Date
dd-mm-yyyy
RESET
Status
Bug Count
Tester Assigned
6
Open
42
Fixed (Testing Pending)
5
Tested & Verified
5
Closed
7
Assigned
43
Testing In Progress
1

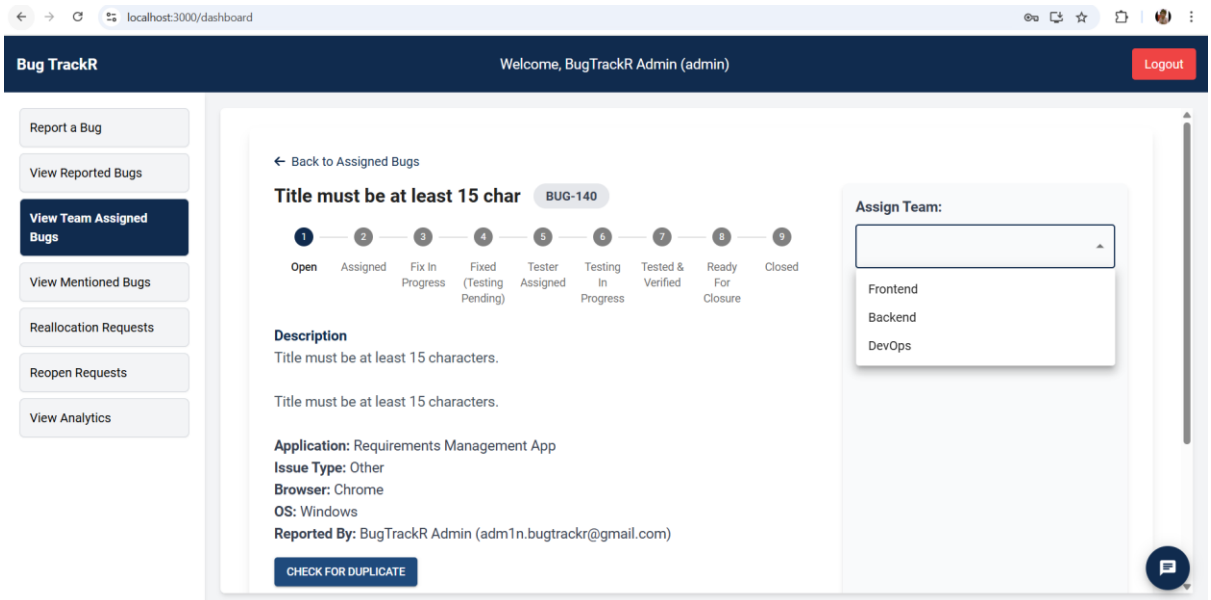

localhost:3000/dashboard
Bug TrackR
Welcome, BugTrackR Admin (admin)
Logout
Report a Bug
View Reported Bugs
View Team Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
4.Unassigned bugs
Select Application
Select Team
Select Priority
Start Date
dd-mm-yyyy
End Date
dd-mm-yyyy
RESET
Unassigned to Team
Bug ID
App
Reported On
BUG-1
Requirements Management App
2025-02-28T13:53:41.795Z
BUG-2
Requirements Management App
2025-02-28T13:59:41.505Z
BUG-3
Requirements Management App
2025-02-28T14:11:44.679Z
BUG-4
Requirements Management App
2025-02-28T14:14:45.868Z
BUG-5
Requirements Management App
2025-02-28T14:22:53.979Z
1–5 of 19

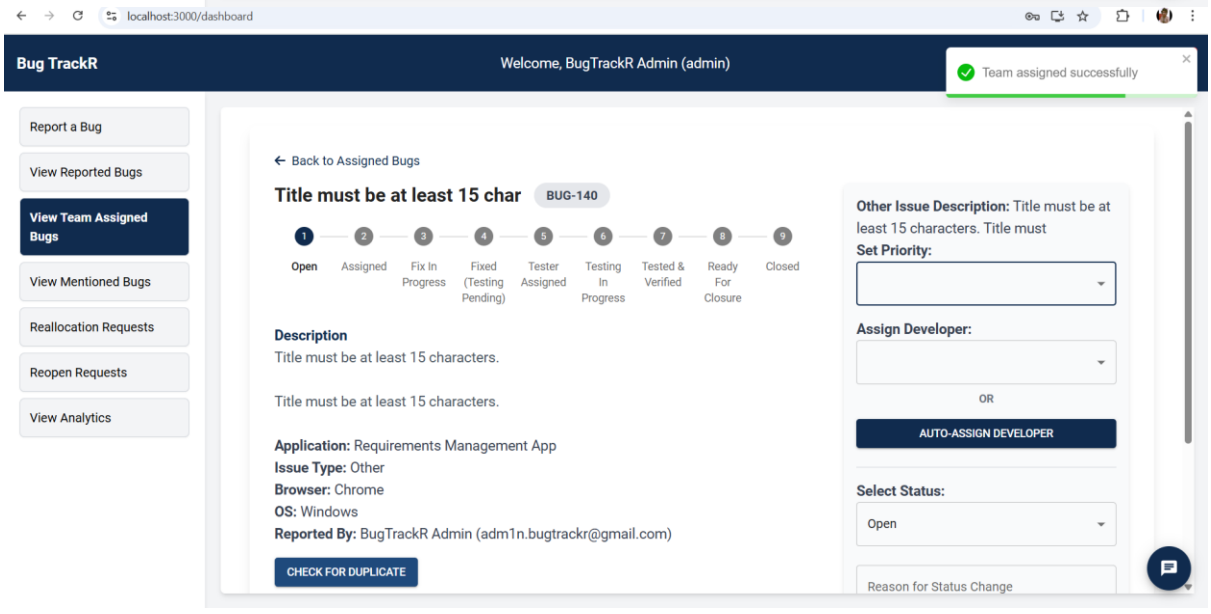

localhost:3000/dashboard
Bug TrackR
Welcome, BugTrackR Admin (admin)
Logout
Report a Bug
View Reported Bugs
View Team Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
5. SLA summary overview
Developer SLA breaches
Tester SLA breaches
Critical
High
Medium
Low
Breach Summary
Critical Priority
Dev: 6
Tester: 4
High Priority
Dev: 13
Tester: 6
Medium Priority
Dev: 3
Tester: 2
Low Priority
Dev: 2
Tester: 0
EXPORT CSV
EXPORT CSV


localhost:3000/dashboard
Bug TrackR
Welcome, BugTrackR Admin (admin)
Logout
Report a Bug
View Reported Bugs
View Team Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
6. Fix vs report bugs trend
Select Application
Start Date
07-04-2025
End Date
07-05-2025
RESET
Number of bugs reported vs. fixed over time for the selected filters.
2025-05-06
2025-05-13
2025-05-15
2025-04-30
2025-04-16
2025-04-22
2025-05-02
Reported
Fixed
EXPORT CSV


localhost:3000/dashboard
Bug TrackR
Welcome, BugTrackR Admin (admin)
Logout
Report a Bug
View Reported Bugs
View Team Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
7. Current Workload and bug distribution
Application
Team
Role
RESET
Application Team User Role Critical High Medium Low Total Bugs Total Hours
Requirement... frontend harishmitha... developer 1 3 1 0 5 36
Requirement... frontend harishmitha... developer 2 0 0 0 2 12
Requirement... frontend harishmitha... developer 2 3 1 0 6 42
Requirement... frontend harishmitha... developer 1 2 0 0 3 24
Requirement... frontend harishmitha... developer 1 4 1 0 6 45
Requirement... frontend harishmitha... developer 0 0 1 1 2 4
Requirement... frontend harishmitha... developer 0 2 1 0 3 21
8. High Priority bugs still Open
Application
Team
RESET
Critical Bugs 17
High Bugs 31
Bug ID Title App Team Priority Status
BUG-25 yyyyyyyyyyyyyyyyyyyyyyyy Requirements Mana... frontend Critical
BUG-26 uuuuuuuuuuuuuuuuuuuuu Requirements Mana... frontend Critical
BUG-92 bugReportFormData[field]90 Requirements Mana... frontend Critical
BUG-96 sjnf ksfj nalksn laksn d Payroll System backend Critical
BUG-106 sdkjnfkls nfkwjhf iahwdwkfj nj Requirements Mana... devops Critical
1–5 of 48
9. Stuck Bugs (No status update)
Application
Team
Threshold (Days)
10
RESET
Bugs that haven't had any updates in the last 10 days.
Bug ID: BUG-23
Medium Assigned
Application: Requirements Management App
Team: frontend
Developer: harishmitha2507+joeg@gmail.com
Tester: -
Stuck for 15 days
Bug ID: BUG-31
High Assigned
Application: Requirements Management App
Team: frontend
Developer: harishmitha2507+thanigai@gmail.com
Tester: -
Stuck for 14 days
Bug ID: BUG-34
Medium Assigned
Application: Requirements Management App
Team: frontend
Developer: harishmitha2507+trisha@gmail.com
Tester: -
Stuck for 14 days
Bug ID: BUG-38
High Assigned
Application: Requirements Management App
Team: devops
Developer: harishmitha2507+ram@gmail.com
Tester: -
Bug ID: BUG-41
High Assigned
Application: Requirements Management App
Team: backend
Developer: harishmitha2507+jake@gmail.com
Tester: -
Bug ID: BUG-54
Low Assigned
Application: Requirements Management App
Team: devops
Developer: harishmitha2507+sowmya@gmail.com
Tester: -


localhost:3000/dashboard
Bug TrackR
Welcome, BugTrackR Admin (admin)
Logout
Report a Bug
View Reported Bugs
View Team Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
Team Assigned Bugs
Application
Requirements Management App
Team
unassigned
SEARCH
Exact Match
Semantic Search
Search bugs by title or ID
SEARCH
RESET
RESET ALL
EXPORT CSV
EXPORT PDF
Priority
Status
Issue Type
Developer
Tester
From Date
dd-mm-yyyy
To Date
dd-mm-yyyy
Bug ID
Title
Status
Issue Type
Priority
Assigned Develo...
Assigned Tester
Alerts
Actions
favourite
BUG-140
Title must be at least 15...
Open
Other
Not set yet
Unassigned
Unassigned
2h+
VIEW
BUG-95
siojefo jefj sepjf je fk se
BUG-93
Title must be at least 15...
BUG-72
getBrowserAndOS().bro...
BUG-71
setDeveloperTeamsetDe...
1–25 of 73


localhost:3000/dashboard
Bug TrackR
Welcome, BugTrackR Admin (admin)
Logout
Report a Bug
View Reported Bugs
View Team Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
Back to Assigned Bugs
Title must be at least 15 char
BUG-140
Open
Assigned
Fix In Progress
Fixed (Testing Pending)
Tester Assigned
Testing In Progress
Tested & Verified
Ready For Closure
Closed
Description
Title must be at least 15 characters.
Title must be at least 15 characters.
Application: Requirements Management App
Issue Type: Other
Browser: Chrome
OS: Windows
Reported By: BugTrackR Admin (adm1n.bugtrackr@gmail.com)
CHECK FOR DUPLICATE
Assign Team:
Frontend
Backend
DevOps


localhost:3000/dashboard
Bug TrackR
Welcome, BugTrackR Admin (admin)
Team assigned successfully
Report a Bug
View Reported Bugs
View Team Assigned Bugs
View Mentioned Bugs
Reallocation Requests
Reopen Requests
View Analytics
Back to Assigned Bugs
Title must be at least 15 char
BUG-140
Open
Assigned
Fix In Progress
Fixed (Testing Pending)
Tester Assigned
Testing In Progress
Tested & Verified
Ready For Closure
Closed
Description
Title must be at least 15 characters.
Title must be at least 15 characters.
Application: Requirements Management App
Issue Type: Other
Browser: Chrome
OS: Windows
Reported By: BugTrackR Admin (adm1n.bugtrackr@gmail.com)
CHECK FOR DUPLICATE
Other Issue Description: Title must be at least 15 characters. Title must
Set Priority:
Assign Developer:
OR
AUTO-ASSIGN DEVELOPER
Select Status:
Open
Reason for Status Change