



e-ISSN: 2319-8753 | p-ISSN: 2347-6710

IJRSET

International Journal of Innovative Research in
SCIENCE | ENGINEERING | TECHNOLOGY

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

Volume 13, Issue 8, August 2024

ISSN INTERNATIONAL
STANDARD
SERIAL
NUMBER
INDIA

Impact Factor: 8.524

 9940 572 462

 6381 907 438

 ijirset@gmail.com

 www.ijirset.com

Bug Tracking System

M.AmithRaj¹, Mrs.P.Shenbagam²

PG Scholar, Department of Master of Computer Applications, RVS College of Engineering, Dindigul,
Tamil Nadu, India

Head & Assistant Professor, Department of Computer Applications, RVS College of Engineering, Dindigul,
Tamil Nadu, India

ABSTRACT: Bug-Tracking mechanism is employed only in some of the large software development houses. Most of the others never bothered with bug tracking at all, and instead simply relied on shared lists and email to monitor the status of defects. This procedure is error-prone and tends to cause those bugs judged least significant by developers to be dropped or ignored. Bug-Tracking System is an ideal solution to track the bugs of a product, solution or an application. Bug Tracking System allows individual or groups of developers to keep track of outstanding bugs in their product effectively. This can also be called as Defect Tracking System. The Bug Tracking System can dramatically increase the productivity and accountability of individual employees by providing a documented work flow and positive feedback for good performance.

KEYWORDS: Bug- Tracking mechanism, ErrorHandling, IssueTracking, Defect Tracking System, Software Development

I. INTRODUCTION

Bug-Tracking mechanism is employed only in some of the large software development houses. Most of the others never bothered with bug tracking at all, and instead simply relied on shared lists and email to monitor the status of defects. This procedure is error-prone and tends to cause those bugs judged least significant by developers to be dropped or ignored.

Bug-Tracking System is an ideal solution to track the bugs of a product, solution or an application. Bug Tracking System allows individual or groups of developers to keep track of outstanding bugs in their product effectively. This can also be called as Defect Tracking System.

The Bug Tracking System can dramatically increase the productivity and accountability of individual employees by providing a documented work flow and positive feedback for good performance.

II. RESEARCH METHODOLOGY

1. Literature Review:

- Study existing bug tracking systems and their features. Analyze the benefits and limitations of current systems.
- Identify best practices in bug tracking and defect management.

2. Surveys and Interviews:

- Conduct surveys among software development teams to understand their current bug tracking practices.
- Interview developers, project managers, and QA teams to gather insights into their needs and challenges.

3. Case Studies:

- Select a few software development companies that use bug tracking systems.
- Study their implementation, usage, and benefits.
- Document the challenges they faced and how they overcame them.

4. Requirements Gathering:

- Based on the research, gather requirements for the ideal bug tracking system.

- Identify the features, functionalities, and user interface expectations.

5. System Design:

- Design the architecture and database schema for the bug tracking system.
- Create a detailed system design document.

6. Prototyping and Testing:

- Develop a prototype of the bug tracking system.
- Test the system with a small group of users.
- Gather feedback and iterate on the design.

7. Implementation and Deployment:

- Develop the full-fledged bug tracking system.
- Deploy the system in a real-world software development environment.

8. Evaluation and Validation:

- Collect data on the system's performance, adoption, and user satisfaction.
- Evaluate the system's impact on productivity, accountability, and defect management.
- Validate the research hypotheses and identify areas for future improvement.

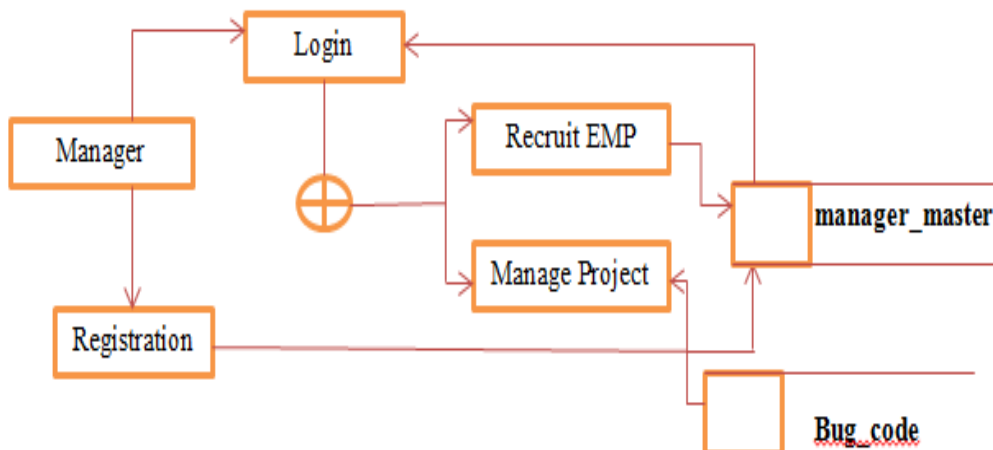
This research methodology provides a structured approach to understanding the needs and challenges of bug tracking in software development, designing an effective system, and evaluating its impact.

III. DATA FLOW DIAGRAM

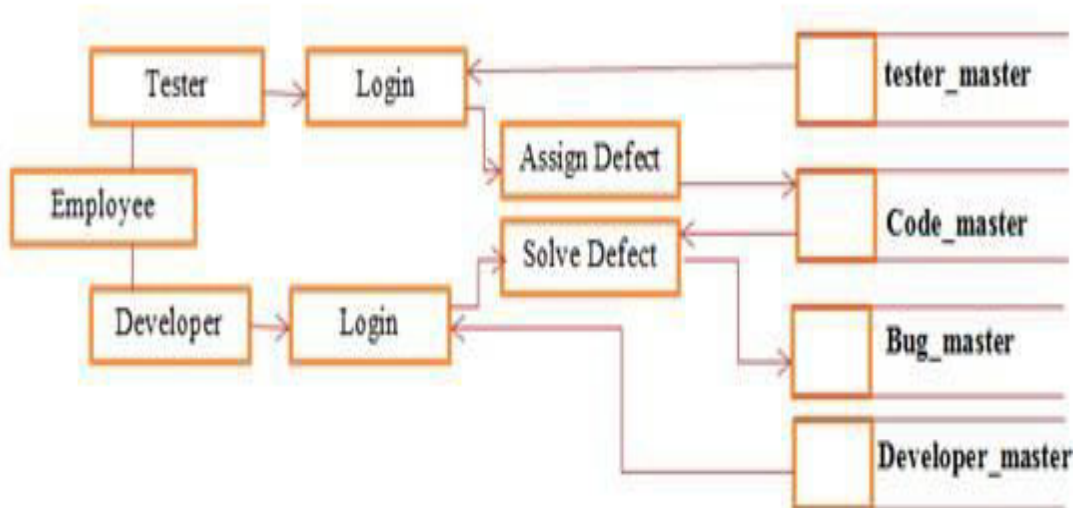
In an Information system, the flow of the data around the system is graphically represented by the data flow diagram. The DFD takes an input-process-output view of a system. That is, data objects flow into the software, are transformed by processing elements, and resultant data objects flow out of the software. Data objects are represented by labeled arrows, and transformations are represented by circles (also called bubbles).

A graphical tool used to describe and analyze the moment of data through a system manual or automated including the process, stores of the data and delays in the system. DFDs are the model of the proposed system. They clearly show the requirements on which the new system should be built.

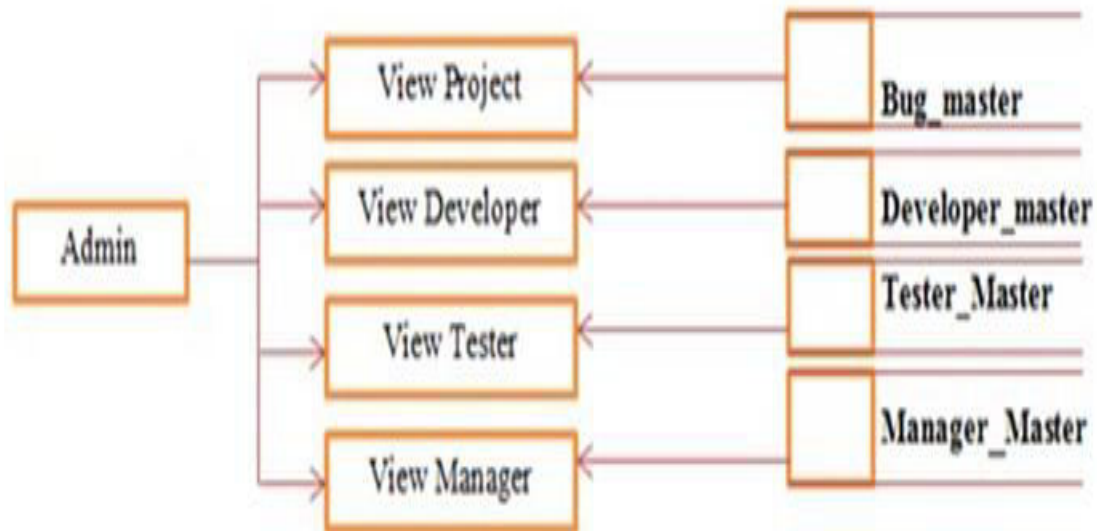
Manager:



Employee:



Admin:

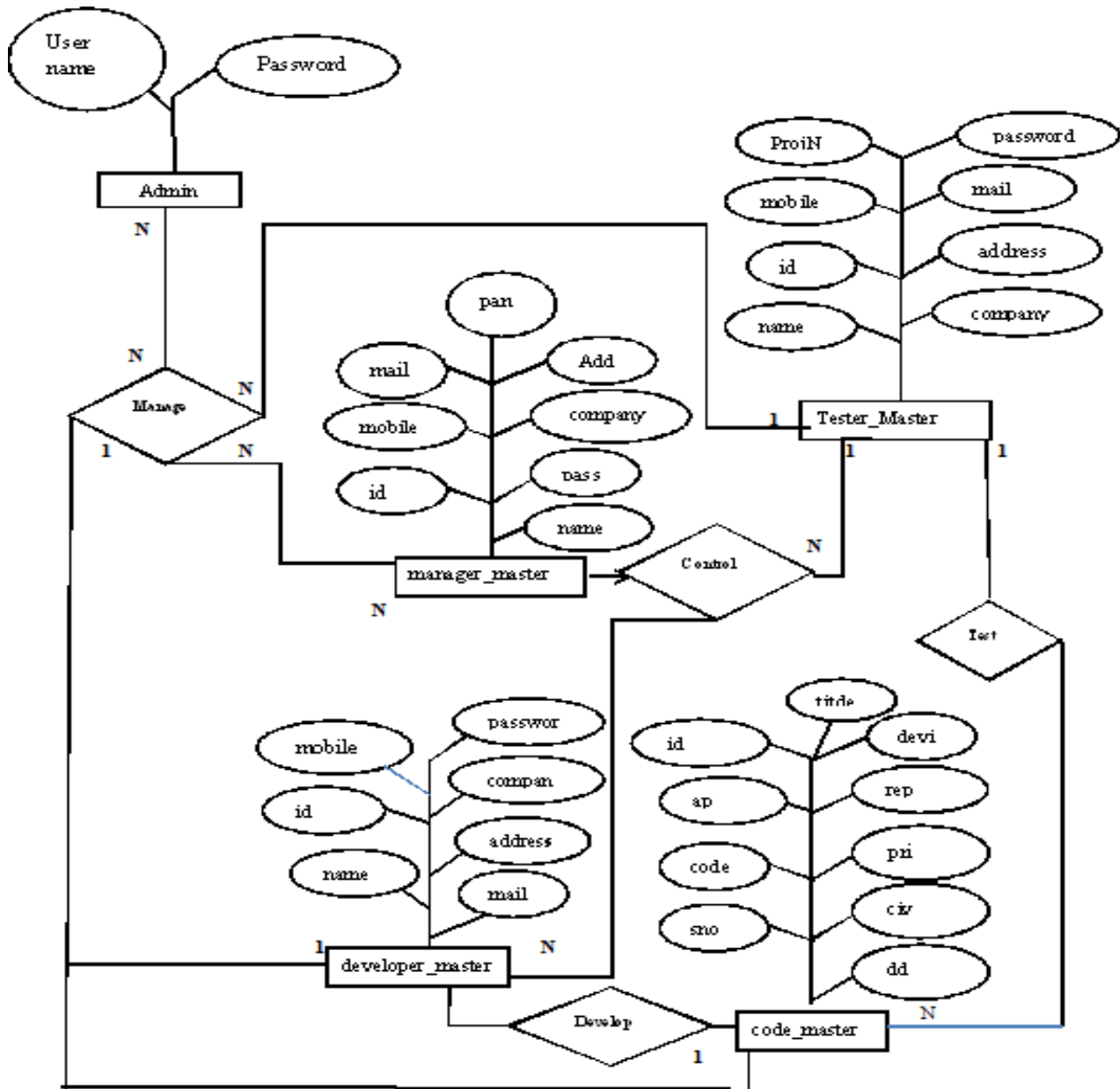


IV. ER- DIAGRAM

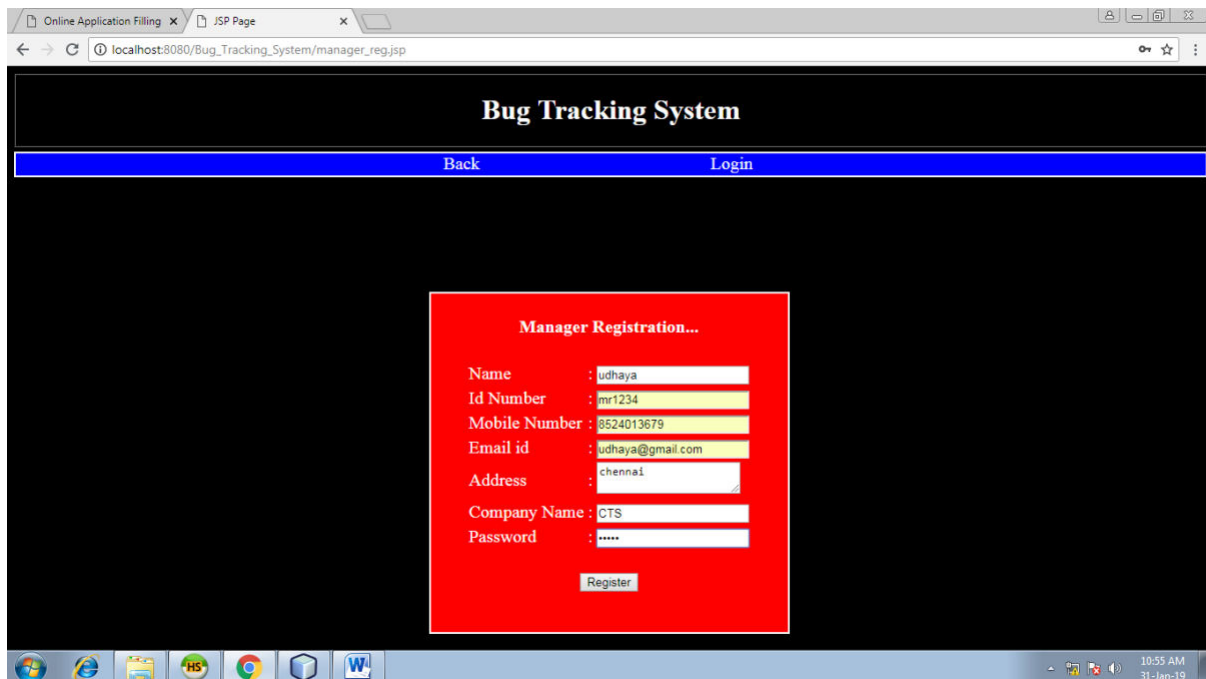
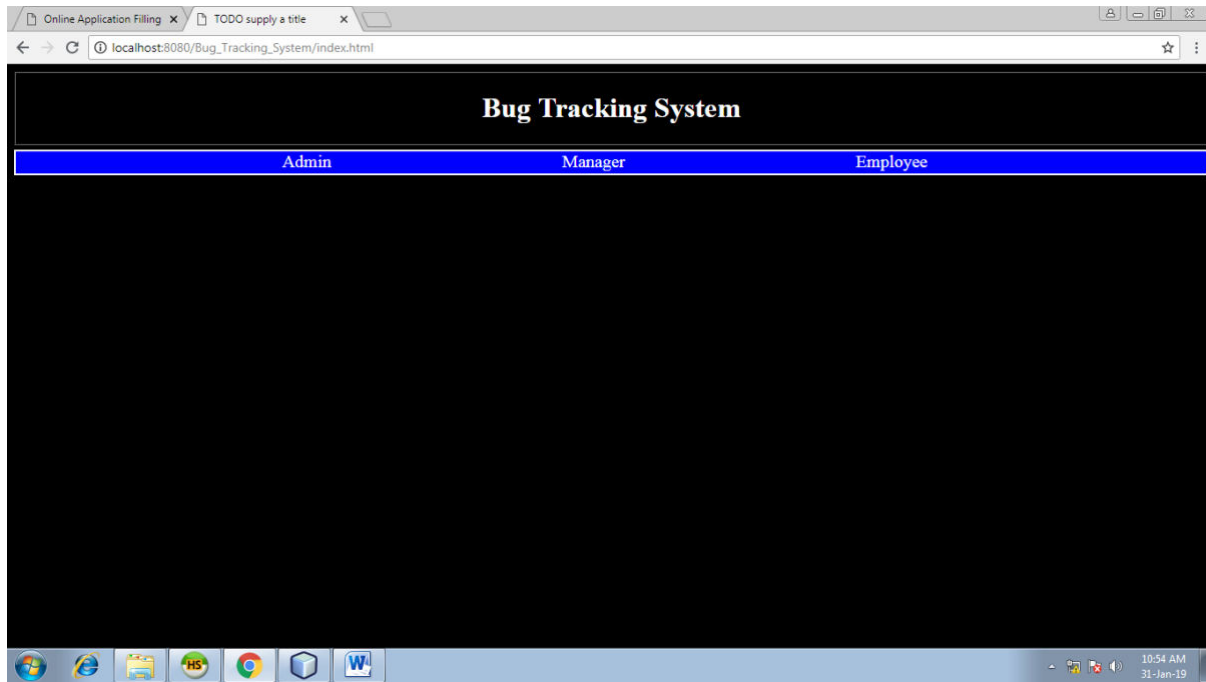
The ER or (Entity Relational Model) is a high-level conceptual data model diagram. Entity-Relation model is based on the notion of real-world entities and the relationship between them.

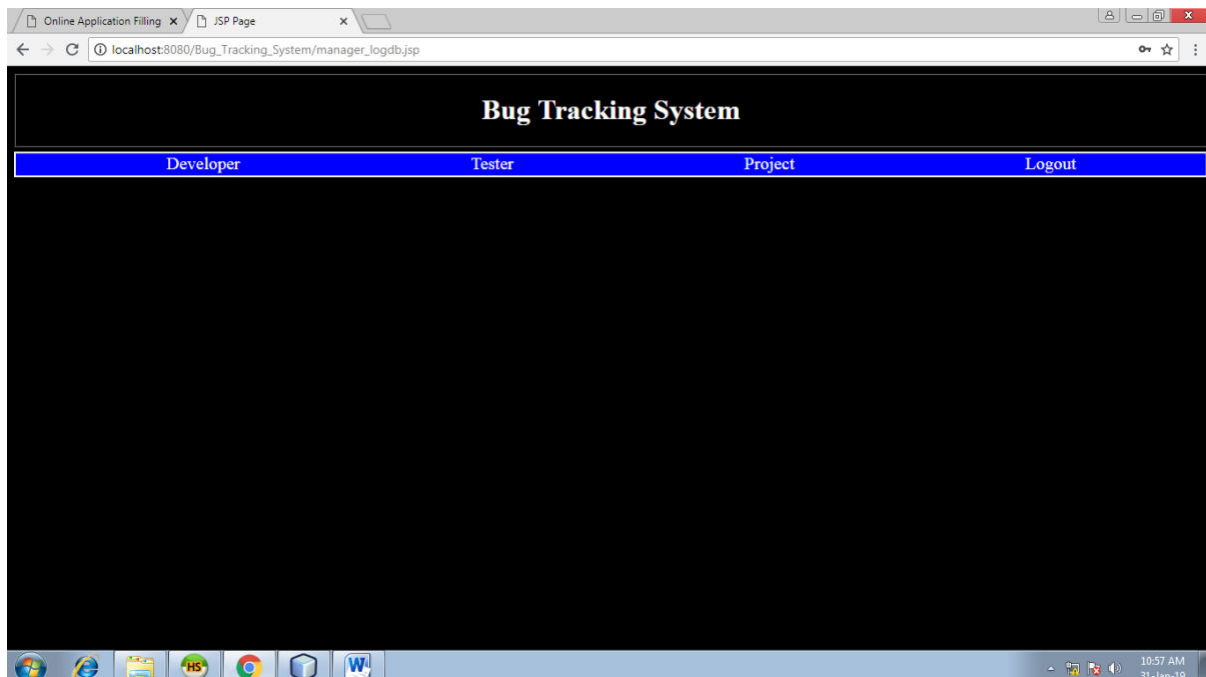
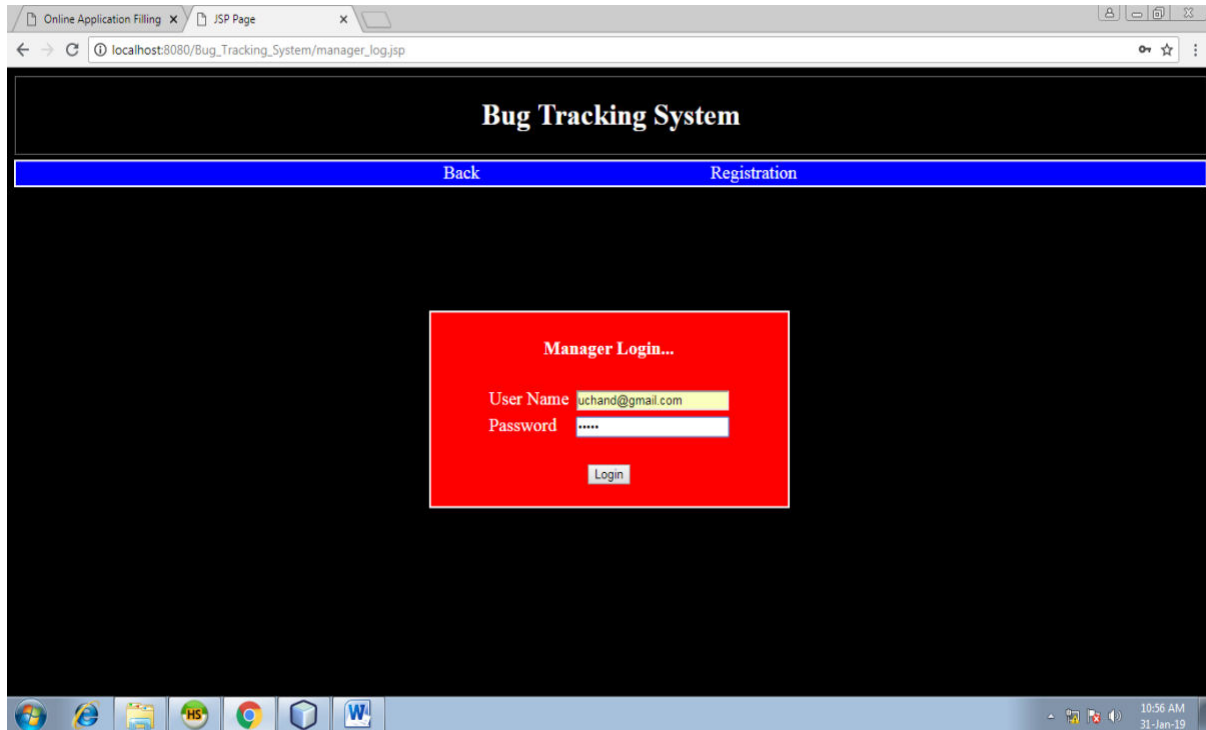
ER modeling helps you to analyze data requirements systematically to produce a well-designed database. So, it is considered a best practice to complete ER modeling before implementing your database.

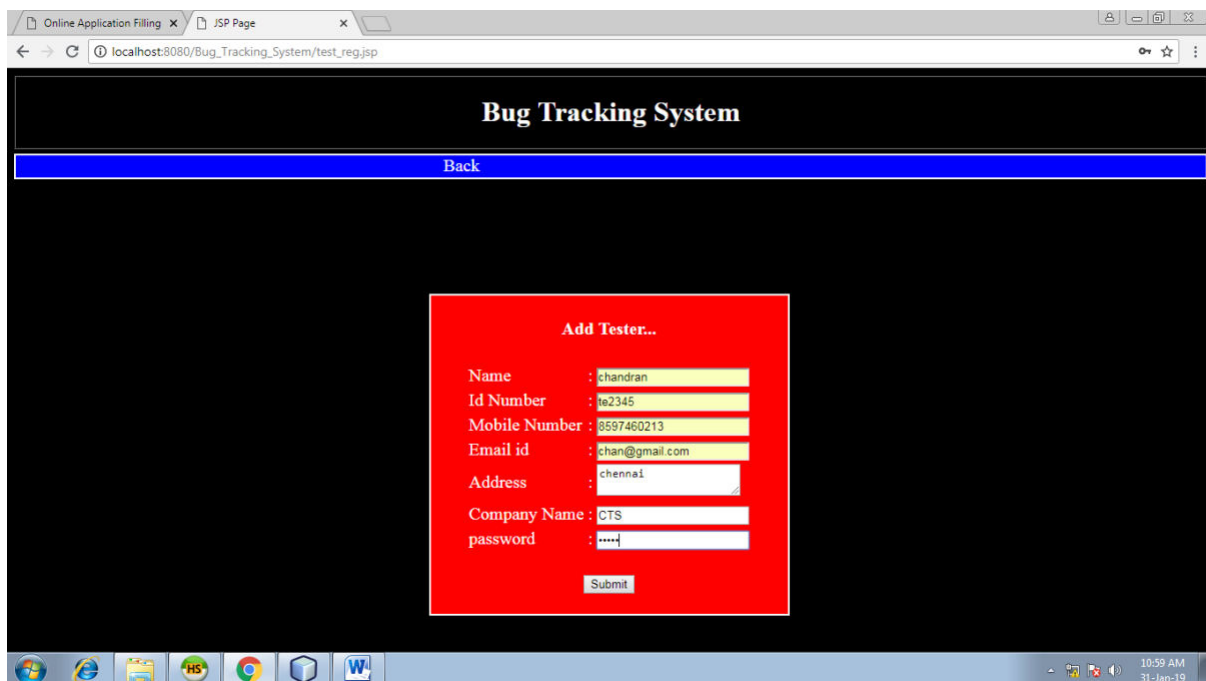
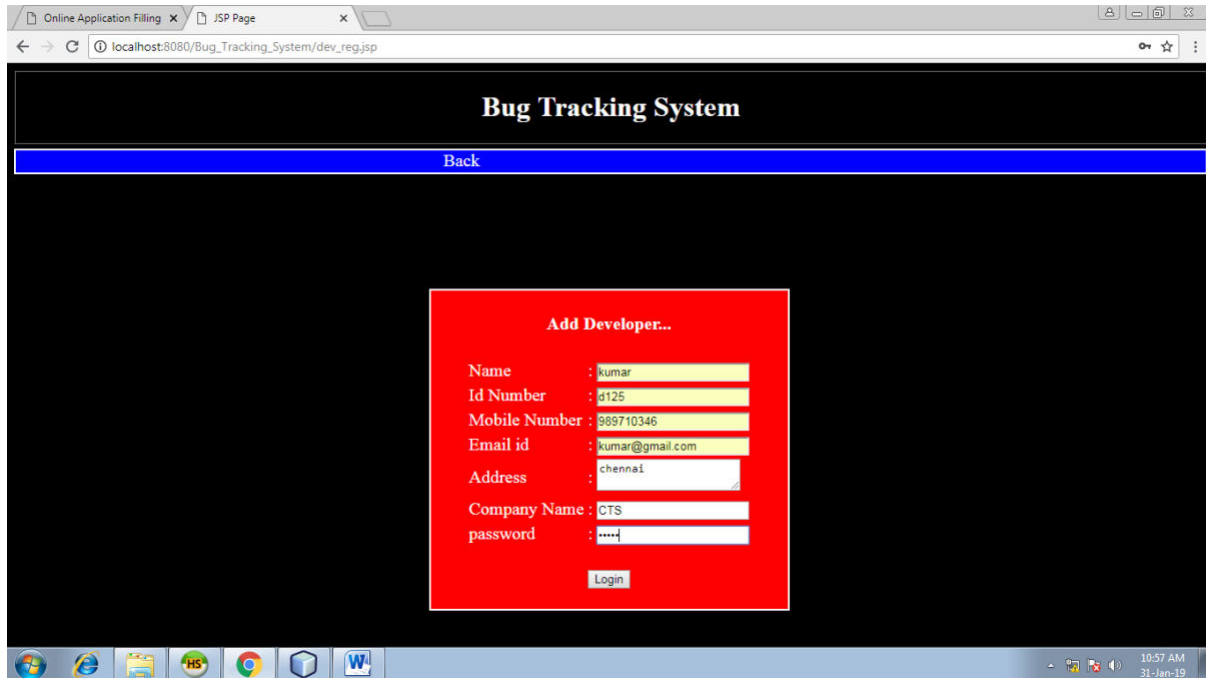
Er Diagram

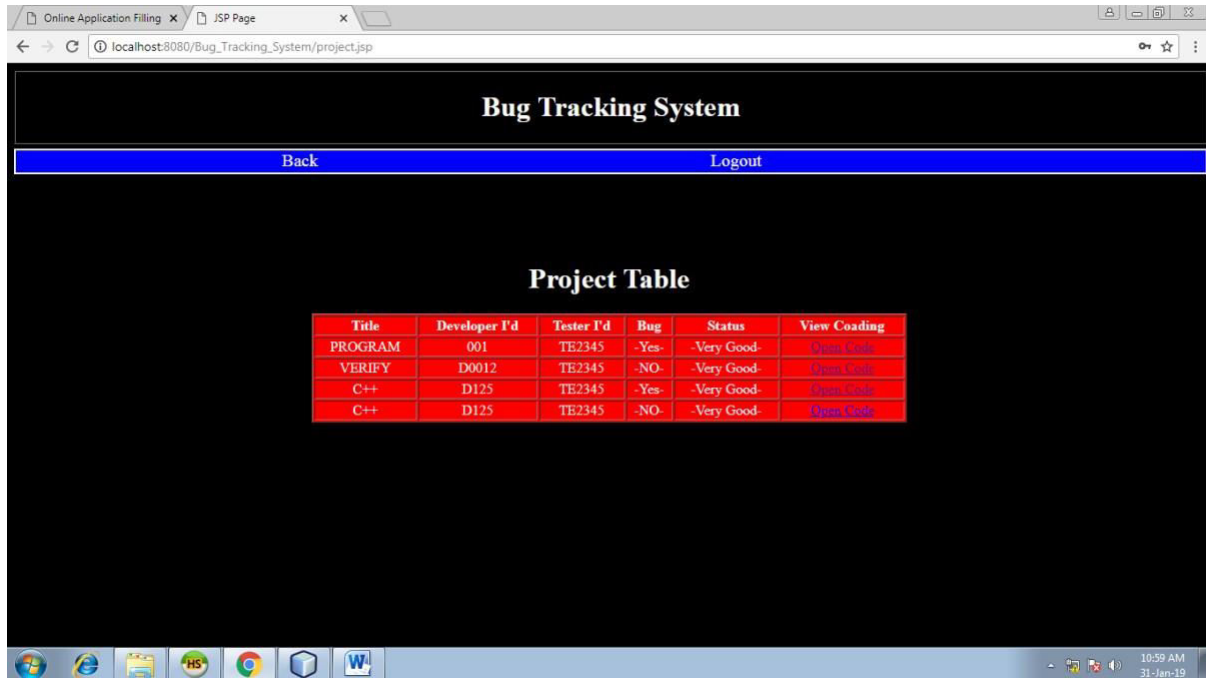


V. RESULT SCREENSHOT



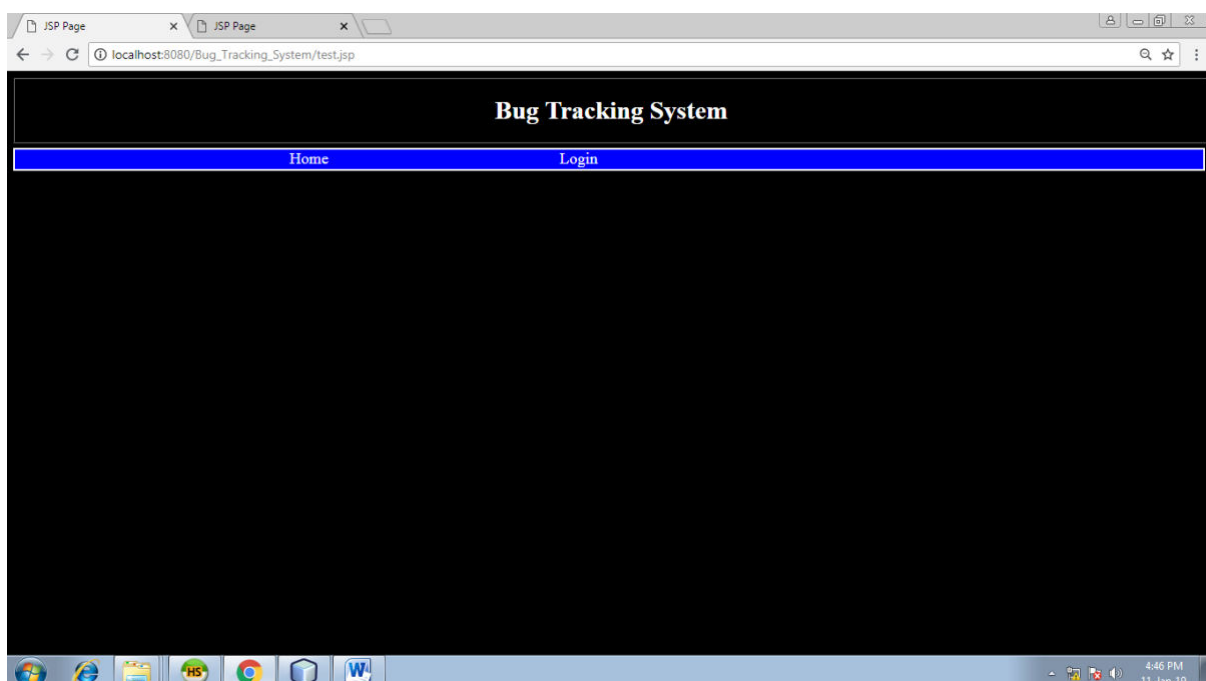




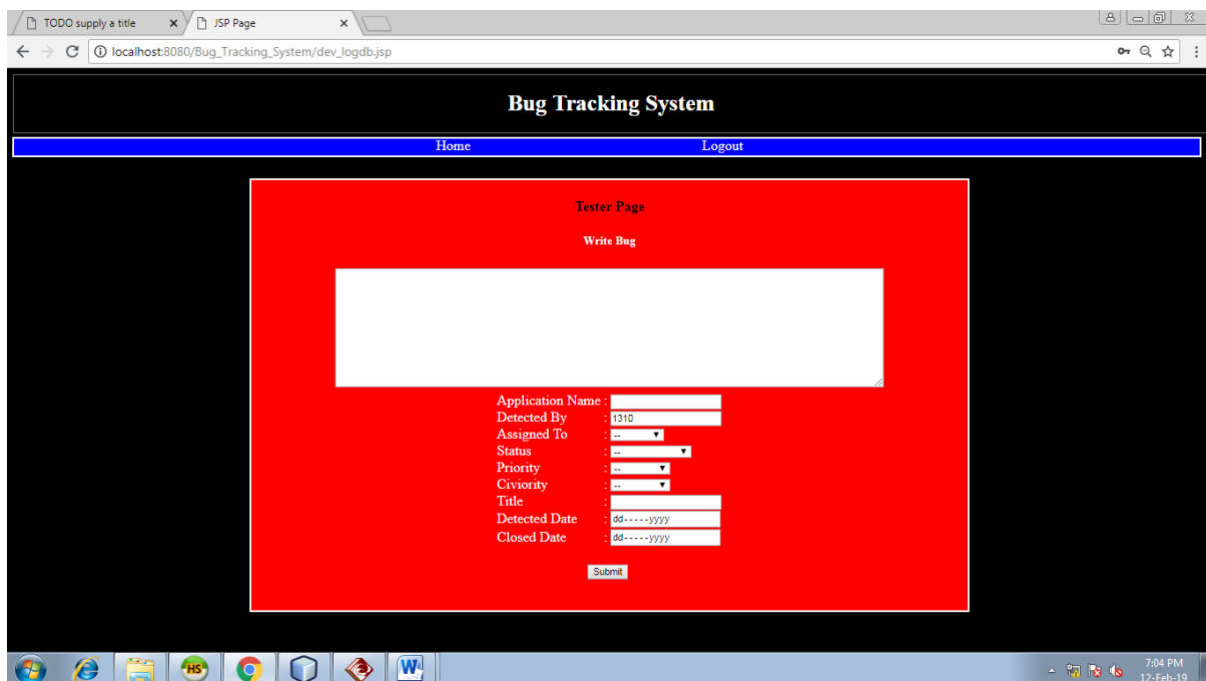
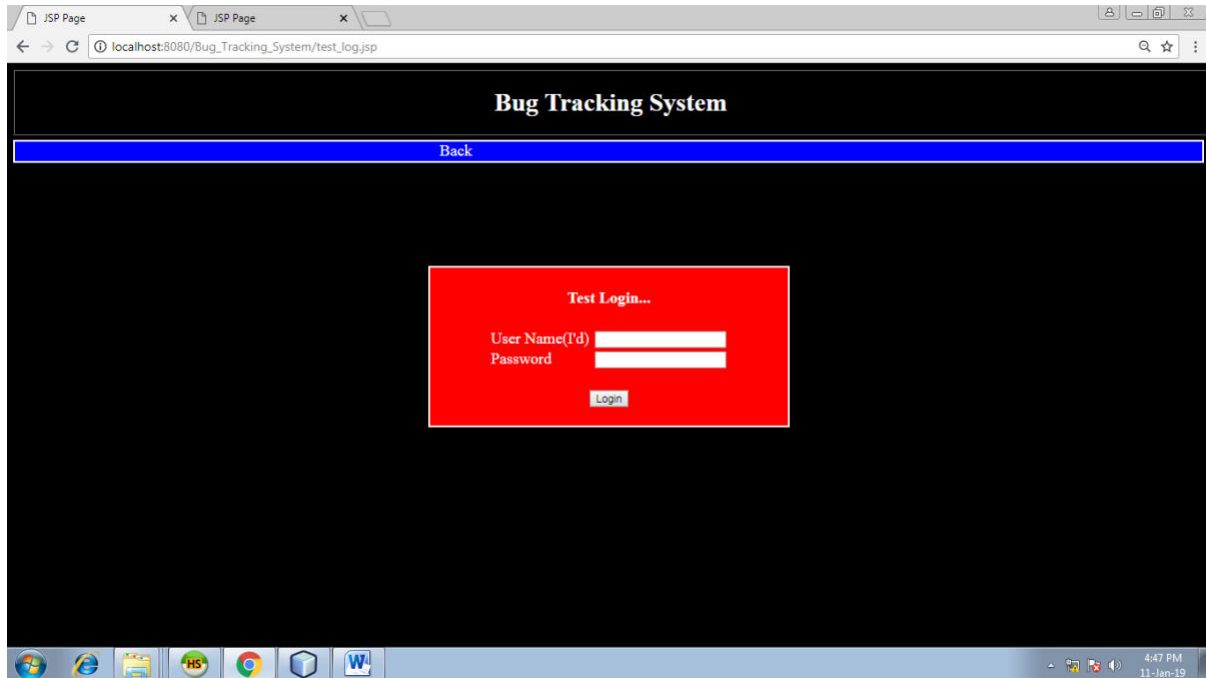


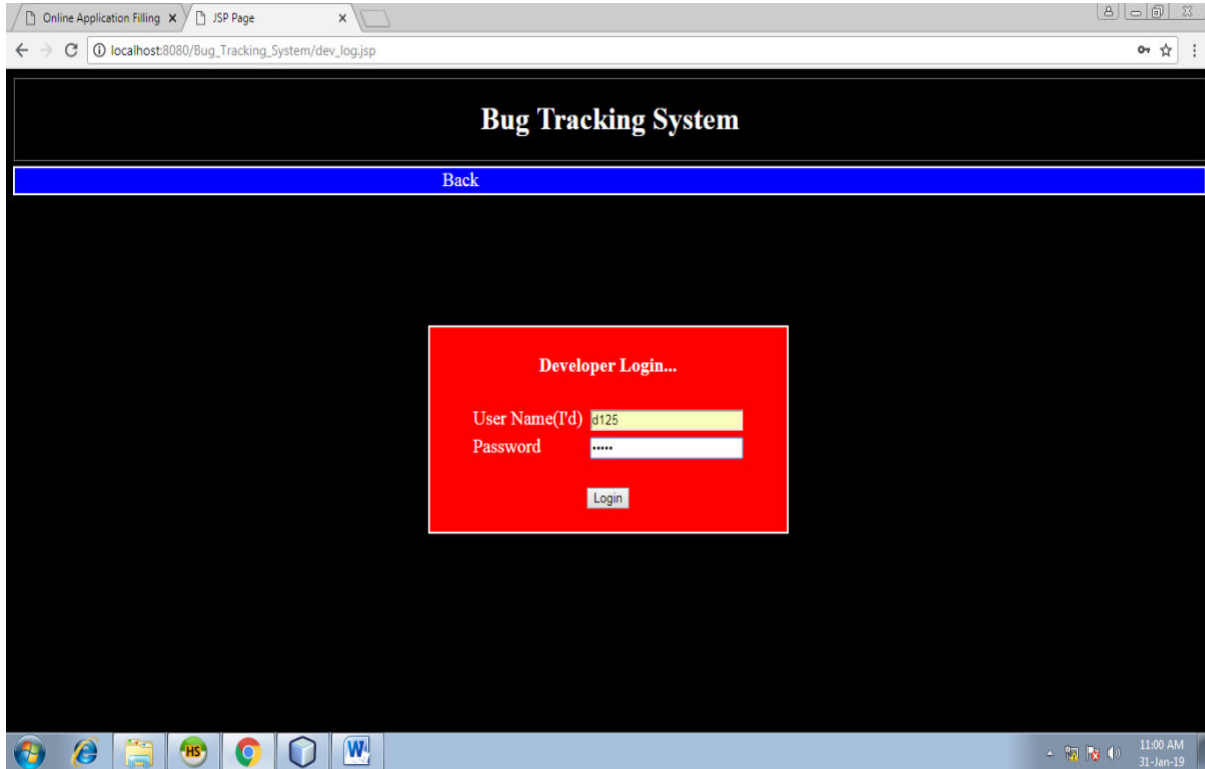
The screenshot shows a web browser window displaying the 'Bug Tracking System' project.jsp page. The page has a black background with a blue navigation bar at the top containing 'Back' and 'Logout' links. Below the navigation bar, the title 'Bug Tracking System' is centered. Underneath, the title 'Project Table' is centered above a table with a red background. The table contains the following data:

Title	Developer I'd	Tester I'd	Bug	Status	View Coading
PROGRAM	001	TE2345	-Yes-	-Very Good-	View Page
VERIFY	D0012	TE2345	-NO-	-Very Good-	View Page
C++	D125	TE2345	-Yes-	-Very Good-	View Page
C++	D125	TE2345	-NO-	-Very Good-	View Page



The screenshot shows a web browser window displaying the 'Bug Tracking System' test.jsp page. The page has a black background with a blue navigation bar at the top containing 'Home' and 'Login' links. Below the navigation bar, the title 'Bug Tracking System' is centered.

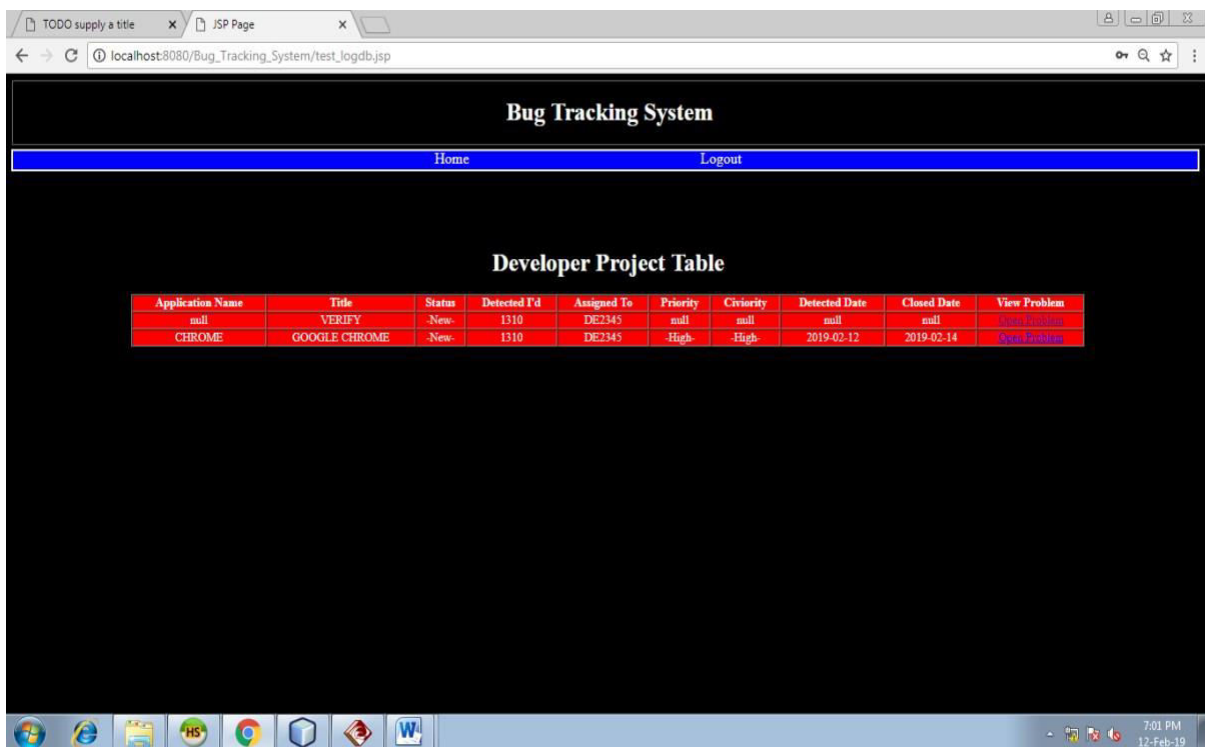




The screenshot shows a web browser window with the URL `localhost:8080/Bug_Tracking_System/dev_log.jsp`. The page title is "Bug Tracking System". A blue navigation bar contains a "Back" button. The main content area features a red "Developer Login..." form with the following fields:

- User Name(I'd):
- Password:
- Login button

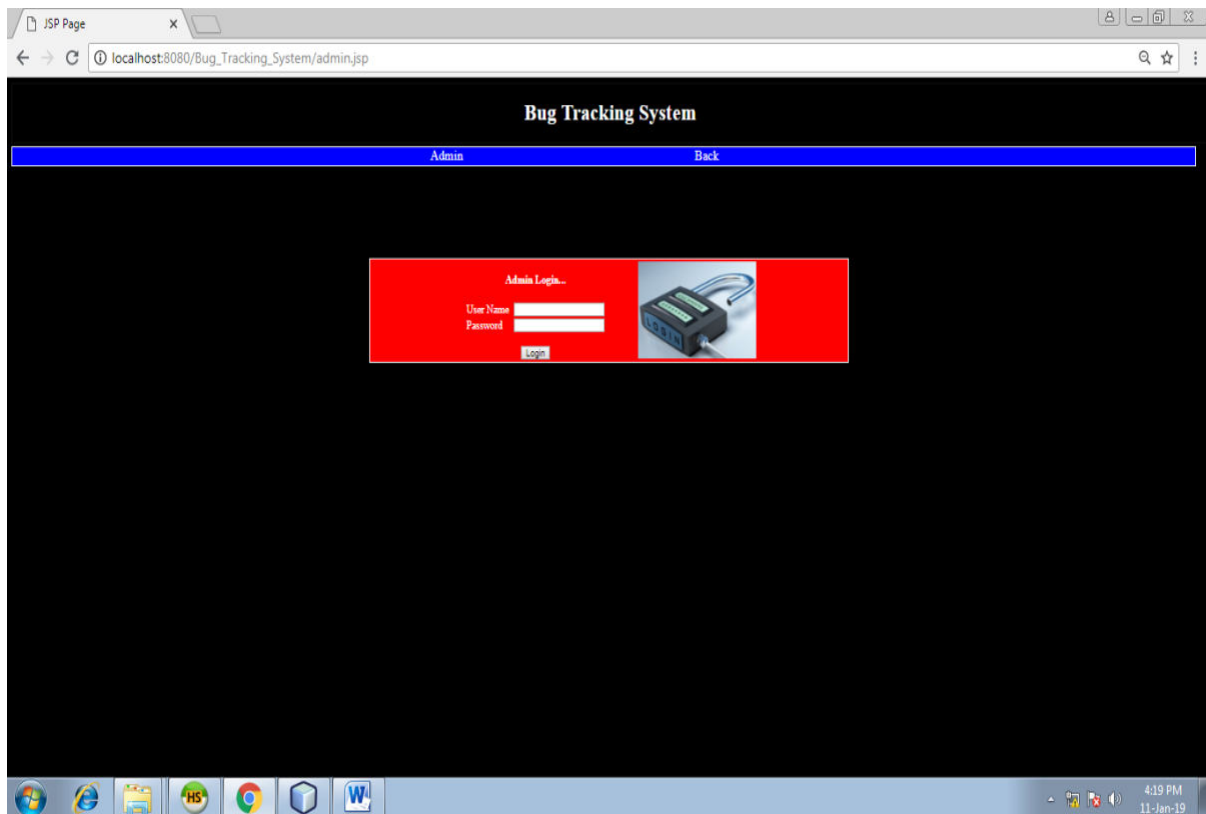
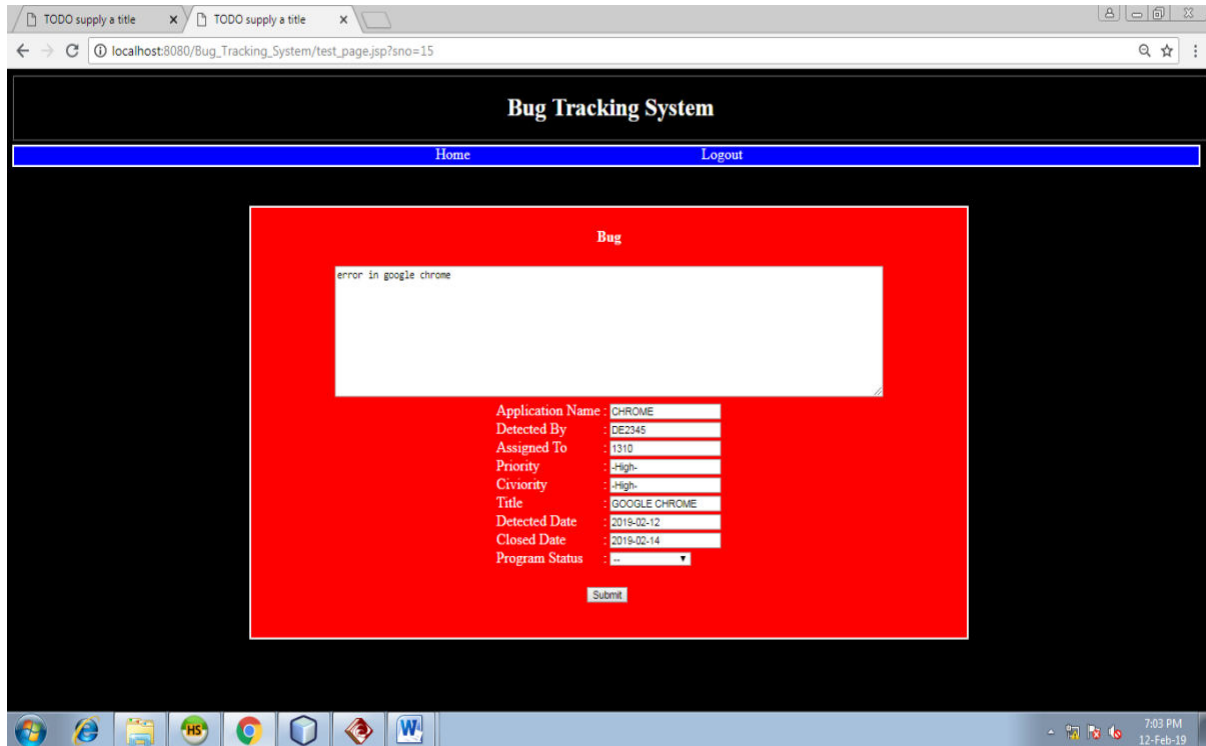
The Windows taskbar at the bottom shows the time as 11:00 AM on 31-Jan-19.

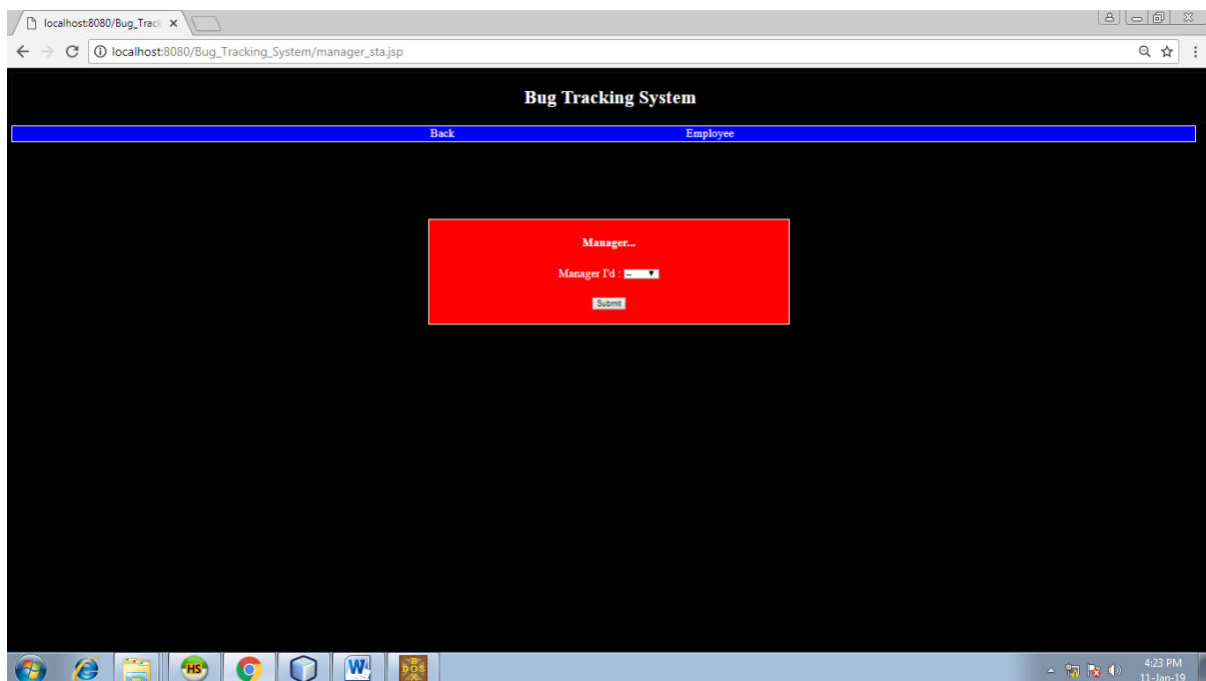
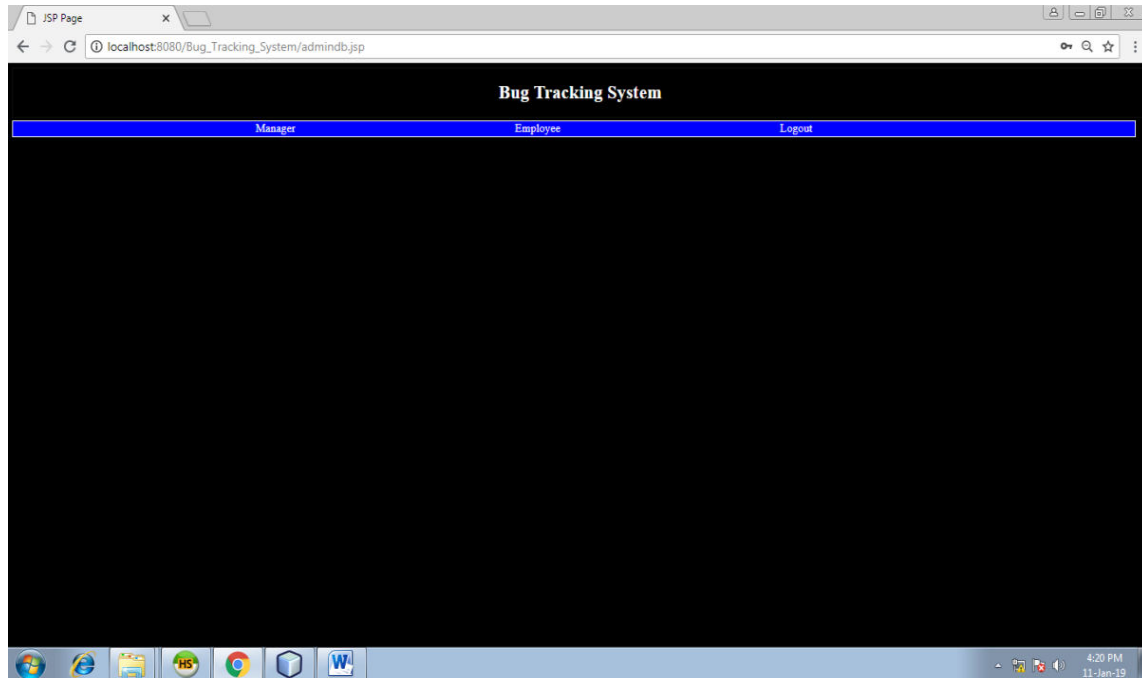


The screenshot shows a web browser window with the URL `localhost:8080/Bug_Tracking_System/test_logdb.jsp`. The page title is "Bug Tracking System". A blue navigation bar contains "Home" and "Logout" buttons. The main content area displays a "Developer Project Table" with the following data:

Application Name	Title	Status	Detected I'd	Assigned To	Priority	Civirity	Detected Date	Closed Date	View Problem
null	VERIFY	New	1310	DE2345	null	null	null	null	View Problem
CHROME	GOOGLE CHROME	New	1310	DE2345	High	High	2019-02-12	2019-02-14	View Problem

The Windows taskbar at the bottom shows the time as 7:01 PM on 12-feb-19.





VI. CONCLUSION

The Bug Tracking System (or Defect Tracking System) serves as a crucial tool in software development environments, addressing significant challenges and enhancing productivity through structured bug management. Here's a summary of its impact and benefits:

1. **Efficient Bug Management:** Traditional methods like shared lists and email are error-prone and often result in overlooked or ignored bugs, especially those deemed less significant. The Bug Tracking System provides a centralized and systematic approach to manage bugs, ensuring all issues are logged, tracked, and addressed

promptly.

2. **Increased Accountability:** By assigning bugs to specific developers and tracking their progress, the system enhances accountability within teams. Developers are responsible for resolving assigned bugs within defined timelines, fostering a culture of ownership and responsibility.
3. **Improved Productivity:** With clear workflows and documented processes, the Bug Tracking System streamlines bug resolution. Developers can efficiently prioritize tasks based on bug severity and status, optimizing their time and efforts.
4. **Enhanced Collaboration:** Teams can collaborate effectively through the Bug Tracking System, sharing information, updates, and solutions related to bugs. This collaborative environment reduces duplicate efforts and ensures everyone is aligned on bug status and resolution strategies.
5. **Quality Assurance:** Continuous monitoring and tracking of bugs contribute to improved software quality. By addressing and resolving defects early in the development cycle, the system helps prevent potential issues from impacting end- users.
6. **Feedback and Performance:** The system provides structured feedback mechanisms, acknowledging developers for resolving bugs promptly and effectively. Positive feedback reinforces good performance and motivates team members to maintain high standards of quality.
7. **Scalability and Adaptability:** Designed to accommodate various team sizes and project complexities, the Bug Tracking System scales with organizational growth. It can adapt to different development methodologies (e.g., Agile, Waterfall) and integrate with other software development tools for seamless workflow management.

In conclusion, implementing a Bug Tracking System not only mitigates the shortcomings of informal bug tracking methods but also enhances overall software development efficiency, accountability, and quality. It represents a critical investment in optimizing development processes and delivering reliable software products to users.

REFERENCES

- 1) J. Aranda and G. Venolia. The secret life of bugs: Going past the errors and omissions in software repositories. In ICSE'09: Proceedings of the 31st International Conference on Software Engineering (to appear), 2009.
- 2) S. Artzi, S. Kim, and M. D. Ernst. Recrash: Making software failures reproducible by preserving object states. In ECOOP'08: Proceedings of the 22nd European Object-Oriented Programming Conference, pages 542–565, 2008.
- 3) N. Bettenburg, S. Just, A. Shorter, C. Weiss, R. Premraj, and T. Zimmermann. What makes a good bug report? In FSE'08: Proceedings of the 16th International Symposium on Foundations of Software Engineering, pages 308–318, November 2008.
- 4) N. Bettenburg, R. Premraj, T. Zimmermann, and S. Kim. Duplicate bug reports considered harmful ... really? In ICSM'08: Proceedings of the 24th IEEE International Conference on Software Maintenance, pages 337–345, 2008.
- 5) S. Brey, J. Sillito, R. Premraj, and T. Zimmermann. Frequently asked questions in bug reports. Technical report, University of Calgary, March 2009.
- 6) P. Fritzson, T. Gyimothy, M. Kamkar, and N. Shahmehri. Generalized algorithmic debugging and testing. In PLDI'91: Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation, pages 317–326, 1991.
- 7) P. Hooimeijer and W. Weimer. Modeling bug report quality. In ASE'07: Proceedings of the 22nd International Conference on Automated Software Engineering, pages 34–43, 2007.
- 8) S. Just, R. Premraj, and T. Zimmermann. Towards the next generation of bug tracking systems. In VL/HCC'08: Proceedings of the 2008 IEEE Symposium on Visual Languages and Human-Centric Computing, pages 82–85, September 2008.
- 9) A. J. Ko and B. A. Myers. Debugging reinvented: asking and answering why and why not questions about program behavior. In ICSE'08: Proceedings of the International Conference on Software Engineering, pages 301–310, 2008.
- 10) B. Liblit, M. Naik, A. X. Zheng, A. Aiken, and M. I. Jordan. Scalable statistical bug isolation. In PLDI'05: Proceedings of the 2005 ACM SIGPLAN Conference on Programming Language Design and Implementation, pages 15–26, 2005.



INTERNATIONAL
STANDARD
SERIAL
NUMBER
INDIA



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

 9940 572 462  6381 907 438  ijirset@gmail.com



www.ijirset.com

Scan to save the contact details