**BUG TRACKING SYSTEM USING JAVA**

MINOR PROJECT REPORT

By

**SRISHTI SHARMA(RA2211003030111)**

**PRAKHAR VERMA(RA2211003030122)**

**KANISHK SAH(RA2211003030124)**

**VANSHIKA SANKHOLIA(RA2211003030131)**

**WRITOBRATA CHATTERJEE(RA2211003030132) ARCHIT JAIN(RA2211003030138)**

Under the guidance of

**MS. SHRUTHY GOVINDAN**

*In partial fulfilment for the Course*

of

**21CSC203P – ADVANCED PROGRAMMING PRACTICE** in COMPUTER SCIENCE AND ENGINEERING



**FACULTY OF ENGINEERING AND TECHNOLOGY**

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**DELHI-NCR CAMPUS**

**NOVEMBER-2023**

**BONAFIDECERTIFICATE**

Certified that this minor project report for the course **21CSC203P ADVANCED PROGRAMMING PRACTICE** entitled in "**BUG TRACKING SYSTEM USING**

**JAVA**" is the bonafide work of **SRISHTI SHARMA(RA2211003030111),PRAKHAR VERMA(RA2211003030122),KANISHK SAH(RA2211003030124),VANSHIKA**

**SANKHOLIA(RA2211003030131),WRITOBRATACHATTERJEE(RA2211003030 132), ARCHIT JAIN(RA2211003030138)** who carried out the work under my

supervision.

**SIGNATURE**

Ms. Shruthy Govindan

**Assistant Professor**

**Computer Science and Engineering**

SRM Institute of Science and Technology

Delhi-Ncr Campus

**ABSTRACT**

The Java Bug Tracking System is a comprehensive software

application designed to facilitate efficient and systematic management of software development and quality assurance processes. This project aims to address the critical need for a structured and user-friendly bug tracking system to streamline the identification, reporting, monitoring, and resolution of software defects.The system allows administrators, developers, and testers to log in with role-based access, ensuring data security and control.Users can submit detailed bug reports, including description, severity, steps to reproduce, and attached files. This information streamlines the bug identification process.Bugs are assigned unique identifiers and can be tracked from creation to resolution. Users can categorize, prioritize, and assign bugs to appropriate team members.Real-time Notifications: Automatic notifications are sent to relevant stakeholders when bugs are assigned or resolved, ensuring transparency and timely updates.

**ACKNOWLEDGEMENT**

We express our profound gratitude to our **Dean (College of Engineering and Technology) Dr. R.P. Mahapatra** for bringing out novelty in all executions.

We are highly thankful to our my Course project Faculty **MS. SHRUTHY GOVINDAN , ASSISTANT PROFESSOR , CSE,** for his/her assistance, timely suggestion and guidance throughout the duration of this course project.

We extend my gratitude to our **HoD Dr. Avneesh Vashistha, HOD(ASSOCIATE PROFESSOR) , CSE** and my Departmental colleagues for their Support.

Finally, we thank our parents and friends near and dear ones who directly and indirectly contributed to the successful completion of our project. Above all, I thank the almighty for showering his blessings on me to complete my Course project.

**TABLE OF CONTENTS**

**1. CHAPTER 2. CONTENTS 3. PAGE**

**NO NO**

1. **INTRODUCTION 6**
   1. Motivation
   2. Objective
   3. Problem Statement
   4. Challenges
2. **LITERATURE 8**

**SURVEY**

1. **REQUIREMENT 9**

**ANALYSIS**

1. **ARCHITECTURE & 11**

**DESIGN**

1. **IMPLEMENTATION 12**
2. **EXPERIMENT 17**

**RESULTS &**

**ANALYSIS**

1. **CONCLUSION 18**
2. **REFERENCES 19**

# 1. INTRODUCTION

Software development is a complex and iterative process, often involving multiple developers, testers, and stakeholders working on different aspects of a project. During this process, identifying, managing, and resolving software defects or "bugs" is a critical aspect of ensuring the quality and reliability of the final product. A Bug Tracking System is an essential tool that helps software development teams streamline this process, ensuring that bugs are discovered, documented, assigned, and resolved efficiently.

The Bug Tracking System using Java is a sophisticated software application designed to address the challenges associated with bug tracking in the context of Java-based software development projects. Java, being a versatile and platform-independent programming language, is an excellent choice for building such a system because it can run on a variety of operating systems and can be easily integrated into existing development environments.

1.1 MOTIVATION

The motivation for implementing a Bug Tracking System in software development is driven by several key factors and the desire to address specific challenges in the development process. These motivating factors include:

Ensuring the quality of software is paramount. Software defects or bugs can compromise the functionality, stability, and user experience of an application. A Bug Tracking System helps identify and rectify these issues promptly, ensuring a high-quality end product.

Efficient Issue Management: In a complex software development project, it's challenging to keep track of all the issues, changes, and enhancements. A centralized Bug Tracking System streamlines issue management, making it easier to identify, assign, and resolve problems, thereby improving project efficiency.

1.2 OBJECTIVE

Efficient Bug Management: The primary objective of this Bug Tracking System is to streamline the bug management process. It enables developers and testers to report, track, and prioritize bugs effectively, reducing the chances of critical issues being overlooked.

Enhanced Collaboration: Collaboration among team members is crucial in software development. This system promotes better communication among developers, testers, and project managers, allowing for quick resolution of issues.

Transparency: With the Bug Tracking System, all stakeholders have access to real-time bug information. This transparency helps project managers make informed decisions and keeps team members accountable for resolving assigned issues.

Data-Driven Decision-Making: The system provides analytics and reporting features, allowing project managers to gain insights into bug trends, resource allocation, and development process improvement.

1.3 PROBLEM STATEMENT

Problem Statement for Bug Tracking System in Java:

In modern software development, the efficient management of bugs and issues is a critical aspect of ensuring the quality and reliability of software products. A Bug Tracking System is a fundamental tool that aids in identifying, documenting, prioritizing, and resolving these issues. However, there are several challenges and issues associated with the current bug tracking systems, especially those developed using Java. This problem statement outlines the key problems and challenges that need to be addressed:

# 2. LITERATURE SURVEY

A literature survey on bug tracking systems using Java reveals a wealth of research and practical applications in the field of software engineering and project management. These studies explore various aspects of bug tracking systems, including their design, implementation, features, and the impact on software development. Here are some notable findings from the literature:

1. **Design and Architecture of Bug Tracking Systems**:
   * "A Flexible Bug Tracking System" by Riedl, et al., explores the design of a flexible bug tracking system that uses Java to support customizable workflows and extensibility.
   * "Design and Implementation of a Bug Tracking System" by Rajkumar and

Prashanth investigates the design and implementation of a bug tracking system in Java, highlighting the importance of database design and user interface considerations.

1. **Usability and User Experience**:
   * "Improving the Usability of Bug Tracking Systems" by Zimmerman and Keith focuses on usability issues in bug tracking systems and how they can be improved for better user experience.
   * "An Empirical Study on the Usability of Bug Tracking Systems" by Sillitti, et al., discusses the results of an empirical study evaluating the usability of various bug tracking systems, with Java-based systems among those analyzed.
2. **Integration with Development Tools**:
   * "Enhancing Bug Tracking Systems with Integration" by Azuma, et al., delves into the integration of bug tracking systems with version control systems, continuous integration tools, and project management software.
3. **Scalability and Performance**:
   * "Scalability Challenges in Bug Tracking Systems" by Alhuwail and Alzouez addresses the challenges related to scalability in bug tracking systems, suggesting methods to enhance their performance in Java-based systems.
4. **Machine Learning and Predictive Analysis**:
   * "Using Machine Learning for Bug Tracking and Prediction" by Ayari, et al., explores the use of machine learning techniques in Java-based bug tracking systems to predict the severity and resolution time of reported issues.

# 3.REQUIREMENTS

**2.1 Requirement Analysis**

Requirements for a Bug Tracking System in Java:

1. **User Authentication and Access Control**:
   * + The system should provide user authentication mechanisms to ensure that only authorized users can access and perform actions within the system.
     + Different user roles should be defined, such as administrators, developers, testers, and project managers, each with specific access permissions.
2. **Bug Reporting**:
   * + Users should be able to report bugs with detailed information, including a title, description, steps to reproduce, expected and actual results, and attachments (e.g., screenshots or logs).
     + Users should be able to categorize bugs based on severity, priority, and product/component.
3. **Bug Tracking**:
   * + Each bug should have a unique identifier for easy reference.
     + Bugs should be assignable to specific team members or individuals responsible for resolution.
     + Status tracking, such as "open," "in progress," "resolved," and "closed," should be available to monitor bug lifecycle.
4. **Workflow Customization**:
   * + The system should allow project managers to define and customize workflows to match the organization's bug resolution processes.
     + Workflow stages may include "new," "assigned," "in progress," "testing," and

"resolved."

**2.2 Hardware Requirement**

The hardware requirements for a Bug Tracking System in Java can vary based on the scale of the system, the number of users, and the expected workload. However, for a typical small to medium-sized bug tracking system, the following hardware requirements should suffice:

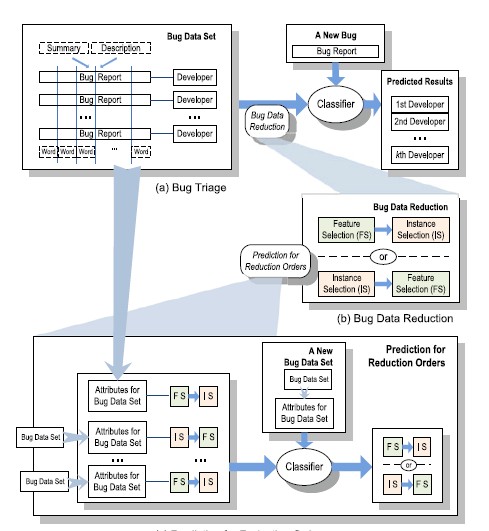
1. **Server**:
   * **CPU**: A multi-core processor (e.g., quad-core or higher) with a clock speed of at least 2.5 GHz is recommended.
   * **RAM**: 8 GB or more to ensure smooth performance, especially when handling a significant number of concurrent users and bug reports.
   * **Storage**: A solid-state drive (SSD) or a combination of SSD and traditional hard disk drives (HDD) with sufficient storage capacity (e.g., 256 GB or more). SSDs significantly improve system responsiveness.
2. **Network**:
   * A stable and high-speed internet connection is necessary for remote access and integration with other development tools.
3. **Redundancy**:
   * For high availability and fault tolerance, consider redundant hardware components, such as redundant power supplies, storage, and network connections.
4. **Backup System**:
   * Implement a reliable backup system with sufficient storage capacity to back up bug data and ensure data recovery in case of hardware failures or data corruption.
5. **Security Measures**:
   * Ensure the server hardware is located in a physically secure environment with restricted access to prevent unauthorized tampering.

# 4.ARCHITECTURE AND DESIGN

● **BUG TRACKING SYSTEM ARCHITECTURE**

The architecture consists of two major data sets:

* Bug Triade
* Bug Data Reduction.



# 5.IMPLEMENTATION

**● PROGRAM FOR BUG TRACKING SYSTEM :**

**package com.company; public class Main { public static void main(String[] args) { new Bug();**

**}**

**}**

**package com.company; import javax.swing.\*;**

**import javax.swing.table.DefaultTableModel; import java.awt.\* import java.sql.\*; import java.util.Vector; public class Bug { private JTextField idData; private JTable table1; private JButton ADDRECORDButton; private JButton UPDATERECORDButton; private JPanel bugPanel; private JTextArea description; private JComboBox product; private JComboBox environment; private JComboBox type; private JComboBox status; JFrame bugF = new JFrame(); public Bug(){ bugF.setContentPane(bugPanel); bugF.pack(); bugF.setLocationRelativeTo(null); bugF.setVisible(true);**

**tableData();**

**ADDRECORDButton.addActionListener(new ActionListener() {**

**@Override**

**public void actionPerformed(ActionEvent e) {**

**if(idData.getText().equals("")||description.getText().equals("")){**

**JOptionPane.showMessageDialog(null,"Please Fill All Fields to add Bug."); }else{**

**//**

**try {**

**String sql = "insert into**

**bug"+"(Bug\_ID,Product,Environment,Type,Description,Status)"+"values**

**(?,?,?,?,?,?)";**

**Class.forName("com.mysql.cj.jdbc.Driver");**

**Connection connection =**

**DriverManager.getConnection("jdbc:mysql://localhost:3306/intern","root","root"); PreparedStatement statement = connection.prepareStatement(sql); statement.setInt(1,Integer.parseInt(idData.getText())); statement.setString(2, ""+product.getSelectedItem()); statement.setString(3, ""+environment.getSelectedItem()); statement.setString(4,""+type.getSelectedItem()); statement.setString(5,description.getText()); statement.setString(6,""+status.getSelectedItem()); statement.executeUpdate();**

**JOptionPane.showMessageDialog(null,"ITEM ADDED SUCCESSFULLY");**

**idData.setText(""); description.setText("");**

**}catch (Exception ex){**

**JOptionPane.showMessageDialog(null,ex.getMessage());**

**} tableData();**

**} }**

**});**

**UPDATERECORDButton.addActionListener(new ActionListener() {**

**@Override**

**public void actionPerformed(ActionEvent e) { try{**

**String sql = "UPDATE bug " +**

**"SET Product = '"+ product.getSelectedItem()+"',Environment='"+ environment.getSelectedItem()+**

**"',Description='"+description.getText()+"',Status='"+status.getSelectedItem()+"'" +**

**" WHERE Bug\_ID="+Integer.parseInt(idData.getText());**

**Class.forName("com.mysql.cj.jdbc.Driver");**

**Connection connection =**

**DriverManager.getConnection("jdbc:mysql://localhost:3306/intern","root","root"); PreparedStatement statement = connection.prepareStatement(sql); statement.executeUpdate();**

**JOptionPane.showMessageDialog(null,"Updated successfully");**

**}catch (Exception e2){**

**System.out.println(e2);**

**} tableData();**

**}**

**});**

**table1.addMouseListener(new MouseAdapter() {**

**@Override**

**public void mouseClicked(MouseEvent e) {**

**DefaultTableModel dm = (DefaultTableModel)table1.getModel(); int selectedRow = table1.getSelectedRow(); idData.setText(dm.getValueAt(selectedRow,0).toString()); description.setText(dm.getValueAt(selectedRow,4).toString());**

**}**

**}); }**

**public void tableData() { try{**

**String a= "Select\* from bug";**

**Class.forName("com.mysql.cj.jdbc.Driver");**

**Connection connection =**

**DriverManager.getConnection("jdbc:mysql://localhost:3306/intern","root","root");**

**Statement statement = connection.createStatement();**

**ResultSet rs = statement.executeQuery(a);**

**// table1.setModel(new DefaultTableModel(null, new String[]{"ID", "ITEM NAME",**

**"QUANTITY", "PRICE"}));**

**table1.setModel(buildTableModel(rs));**

**}catch (Exception ex1){**

**JOptionPane.showMessageDialog(null,ex1.getMessage());**

**}**

**}**

**public static DefaultTableModel buildTableModel(ResultSet rs) throws SQLException {**

**ResultSetMetaData metaData = rs.getMetaData();**

**// names of columns**

**Vector<String> columnNames = new Vector<String>(); int columnCount = metaData.getColumnCount(); for (int column = 1; column <= columnCount; column++) { columnNames.add(metaData.getColumnName(column));**

**}**

**// data of the table**

**Vector<Vector<Object>> data = new Vector<Vector<Object>>(); while (rs.next()) {**

**Vector<Object> vector = new Vector<Object>();**

**for (int columnIndex = 1; columnIndex <= columnCount; columnIndex++) { vector.add(rs.getObject(columnIndex));**

**}**

**data.add(vector);**

**}**

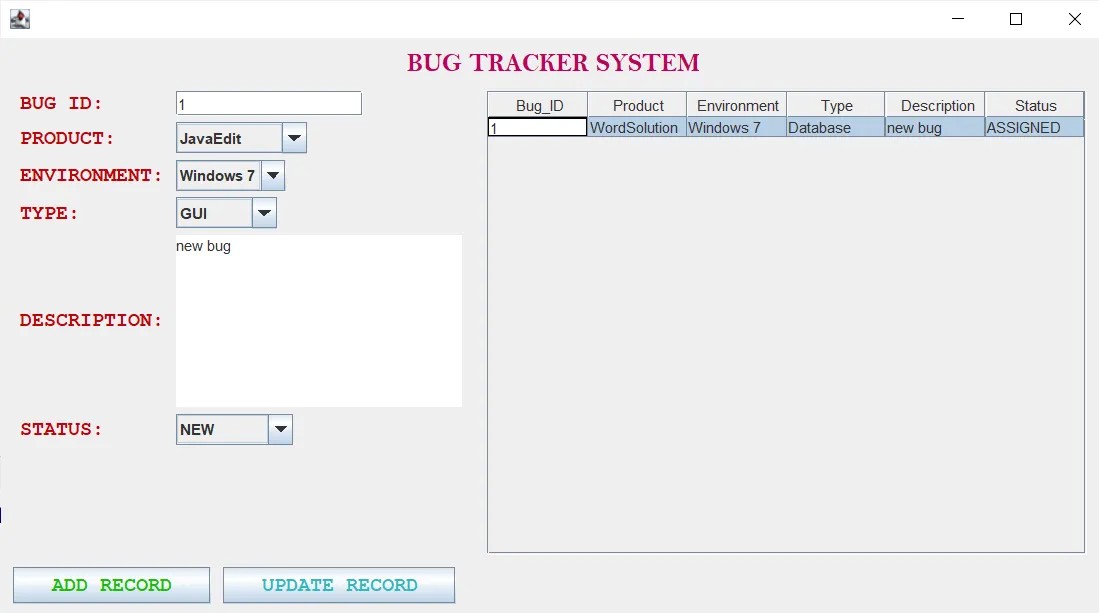
**return new DefaultTableModel(data, columnNames);**

**} }**

## 6. RESULTS AND DISCUSSION

**OUTPUT :**

**● MAIN INTERFACE**



# 7. CONCLUSION

**In conclusion, a Bug Tracking System is an indispensable tool in the realm of software development and quality assurance. It serves as the central nervous system of a development project, facilitating the identification, tracking, and resolution of software defects. Here are the key takeaways and the significance of a Bug Tracking System:**

1. **Quality Assurance:** A Bug Tracking System is paramount for maintaining and improving the quality of software products. It enables teams to identify and address defects, ensuring that the final product is robust and reliable.
2. **Efficiency:** It streamlines the bug management process, making it easier to report, track, prioritize, and resolve issues. This efficiency reduces the chances of critical bugs going unnoticed.
3. **Transparency:** Transparency in software development is essential. A Bug Tracking System provides real-time visibility into the status of reported issues, enhancing communication and accountability among team members and stakeholders.
4. **Customization:** The ability to customize workflows, issue types, and access controls allows organizations to tailor the system to their specific needs, ensuring that it aligns with their development processes.
5. **Integration:** Integration with version control systems, project management tools, and continuous integration systems is crucial for collaboration and data synchronization. It allows different development tools to work seamlessly together.
6. **Data-Driven Decisions:** Bug Tracking Systems provide valuable insights into bug trends, resolution times, and team performance, facilitating data-driven decision-making and resource allocation.
7. **Security:** Robust security measures protect sensitive bug data and ensure that only authorized users can access and modify information, safeguarding the integrity of the system.
8. **User-Friendly Interface:** A user-friendly interface simplifies navigation and reduces the learning curve for new users, making the system accessible to all team members.

**Overall, a well-implemented Bug Tracking System not only enhances the development process but also contributes to the overall success of a software project. It empowers development teams to identify and rectify issues efficiently, maintain quality standards, and ultimately deliver software products that meet or exceed user expectations. As software development continues to evolve, the importance of Bug Tracking Systems remains steadfast in ensuring the reliability and quality of software applications.**

## 8. REFERENCES

1. Bug tracking System (Project Ideas):

<https://myprojectideas.com/bug-tracking-system-in-java-java-project/>.

1. Chat GPT : [https://chat.openai.com](https://chat.openai.com/)
2. Javatpoint : [https://javatpoint.com](https://javatpoint.com/)