“Online Bug Tracking System”

A Dissertation submitted in partial fulfillment of the requirements for the award of degree of

MASTERS OF COMPUTER APPLICATIONS

Of

BENGALURU CITY UNIVERSITY

Submitted By

NAGESH V B (P18FY23S126011)

Under the Guidance of

Prof. Sreelakshmy P P

SAMBHRAM ACADEMY OF MANAGMENT STUDIES M S PALYA, VIDYARANYAPURA POST BENGALURU – 560097

May- 2025

“Online Bug Tracking System”

A Dissertation submitted in partial fulfillment of the requirements for the award of degree of

MASTERS OF COMPUTER APPLICATIONS

Of

BENGALURU CITY UNIVERSITY

For the Academic year – Nov 2025

Submitted By

NAGESH V B (P18FY23S126011)

Under the Guidance of

Mrs. Sreelakshmy P P

Asst. Professor

Department of Computer Applications

SAMBHRAM ACADEMY OF MANAGMENT STUDIES
M S PALYA, VIDYARANYAPURA POST
BENGALURU – 560097

SAMBHRAM ACADEMY OF MANAGMENT STUDIES
M S PALYA, VIDYARANYAPURA POST
BENGALURU – 560097

Department of Computer Applications

CERTIFICATE

This is certified that **Nagesh V B** bearing the Reg no **P18FY23S126011** has satisfactorily completed the Third Semester MCA mini project titled "**Online Bug Tracking System**" towards partial fulfillment of the requirements for the completion of 3rd semester of the Degree in "Master of Computer Applications", awarded by BENGALURU CITY UNIVERSITY, during the Academic Year 2023-2025.

USN No: P18FY23S126011

Mrs. Sreelakshmy P P
Asst. Professor, MCA

Mrs. Sunitha K
Assoc. Professor, HOD, MCA

EXAMINERS:

1.

2.

ACKNOWLEDGEMENT

I have taken efforts in this project it would have not been possible without the help and support of many individuals and organizations. I would like to extend my sincere thanks to all of them.

I take this opportunity to thank our honorable principal **Dr. R Prakash** for permitting us to undertake this project as the part of our curriculum.

I express sincere thanks to **Prof. Sunitha K, HOD of MCA** department for their immense support and guidance throughout the project.

I am highly indebted to our project internal guide, **Assistant Professor Sreelakshmy P P** Department of MCA, Sambhram Academy of Management Studies, Bengaluru, who helped me in channeling my effort in the right direction. She has guided me to correctly carry out phases of the project. I would like to express my gratitude to all the lectures of MCA department for their valuable guidance, suggestions, and encouragement throughout this work.

Last but not the least, I thank almighty, my beloved parents and friends for their constant encouragement without which this assignment would have not been possible.

Thank you,

Nagesh V B
[P18FY23S126011]

ABSTRACT

Software bug fixes account for more than 45% of software companies' expenses. Bug triage is a necessary stage in the bug-fixing process that tries to accurately assign a developer to a new bug. Automatic bug triage is carried out using text classification techniques, which reduces the time required for manual labour. We tackle the issue of data reducing for bug remediation in this study, that is, how to lower the scoring system and raise the standards of defect data. In order to concurrently reduce data scale on the error dimension and the word depth, we combine picking features with occurrence select. We take attributes from past bug data sets and create a predictive model for the first bug to determine which order for instance selection and feature selection.

INDEX

Summary

The **Online Bug Tracking System** is a web-based software application developed to efficiently manage and monitor bugs or defects encountered during the software development lifecycle. The system provides a centralized platform where developers, testers, and project managers can log issues, assign them to the appropriate personnel, update their status, and track progress in real-time.

This application enhances team collaboration, improves transparency, and ensures accountability by providing features like user authentication, role-based access control, bug classification, priority tagging, status updates, and historical tracking of each reported issue. Additionally, it helps reduce manual efforts and communication overhead by automating the bug reporting and resolution workflow.

By implementing this system, software development teams can significantly reduce debugging time, improve product quality, and accelerate the delivery of reliable software products.

CHAPTER 1
INTRODUCTION

1.1 PROJECT DESCRIPTION

The goal of this project, dubbed the Bug Tracking System project, is to give online assistance to software developers who encounter bugs or other software-related issues. The goal of this project is to track bugs. Hence its name. The capabilities of this project allow information about the project, developers, and testers to betracked. Any flaws discovered during software development can be monitored using a bug-tracking system. Although there is bug data provided, no actual bugs are found. The Bug Tracking System is the place to go if you want to find out about or report bugs. To make sure the project satisfies the client's requirements, engineers work on it. During testing, the tester will look for bugs. A bug ID and other relevant information are noted by the tester for that create software. One of the most important parts of problem-fixing is the evaluation interpret, which assigns the right engineer to the right bug. Computerized bug triage is accomplished by analysing text computer programs, saving time compared to human issue reporting. The focus of this study will be the reduction of information, which is the practice of reducing the amount of bug reporting while maintaining its level of accuracy with the goal to make issues triage easier. The word dimensions and the bug size may be decreased at the same time when choice of features with choosing an instance are integrated. For examination, the Bug Tracker System is essential. Testing and development personnel are assigned projects by this attribute. The management of projects is managed by bug following, An error, errors, such as failure, or flaw in a software is sometimes referred to as a bug. Few faults are brought on by engines; the majority are the result of human blunders and mistakes regarding the underlying code or design of systems. An online tool called the Bug Tracking System was created to assist engineers alongside quality testing personnel in monitoring software defects that have been reported while working on projects. A repository of data that keeps track of information and reported bugs is a key element of the tracking of bugs. A database report may include multiple pieces of data, often known as facts. The software engineers are given projects by the project administrator. Projects are developed by the developer in accordance with client specifications. The produced programs are assigned to the testers by the project lead. Beneficial for future reference. The problem's location, kind, priority, name, and ID are all included in the bug information. This entire process keeps going until the software's issues are all fixed.

As soon has a bug is found, the report is submitted to the author and the project leader. This guarantees that no mistake will remain uncorrected due to insufficient communication. It guarantees that information about a bug is available to everyone who has knowledge about it as soon as it is reported. During the testing stage, the Bug Tracker System is essential. However, it allows the project management to delegate projects to the developer and tester. The Bug Tracking System offers unique settings for each user, maintaining them independently.

1.2 COMPANY PROFILE

Puducherry-based JP INFOTECH was founded in 2009 and has a strong history of working with students on educational projects. It specializes in solving current IEEE paper and programming, and it continues to focus all of its efforts on fulfilling transcending outstanding performance in the creation and ongoing support of computer software ventures and goods in a variety of fields.
In the current technologically highly competitive climate, student pursuing a degree in IT want to be sure that the business they are receiving guidance from will suit their career requirements. Learners can be confident in our well-trained team of reliable information systems specialists, who research, design, develop, improve customize, implement, maintain, and support a variety of statistical technology-related topics. We only make technology that is easily used for learners so that we can understanding what they want and improving their standards of life in the workplace. We only work in the areas of system insertion, individualization, optimizing search engines, system modelling, and developing software. Our project strategy consists of methods for starting a job, creating specifications, assigning clear tasks to the team, creating an unpredictable schedule, updating leaders on the status of the work, and finding solutions. The indispensable factors, which give the competitive advantages over others in the market, may be slated as:

- Performance
- Pioneering efforts
- Client satisfaction
- Innovative concepts
- Constant Evaluations
- Improvisation
- Cost Effectiveness

ABOUT THE PEOPLE:

We've got a clear vision collectively a team although we also realize it. The team, according to statistics, has over 40,000 hours of experience in providing real-time solutions in the following areas developing Android smartphone applications connections web designing secure technology mobile devices technology the cloud computational imaging and carrying out interaction with OMNET++, Simulation client-server relationships applications in Java(J2EE\J2ME\EJB), ANDROID,DOTNET(ASP.NET,VB.NET,C#.NET)MATLAB,NS2,SIMULINK,EMBEDDED, POWERELECTRONICSVB & VC++)Oracle and operating system concepts with LINUX.

<u>OUR VISION:</u>

“Impossible as Possible” this is our vision; we work according to our vision

CHAPTER 2

LITERATURE SURVEY

1) Finding bugs in web applications using dynamic test generation and explicit-state model checking

AUTHORS: J. Dolby, F. Tip, D. Dig, A. Paradkar, M. D. Ernst, S. Artzi, and A. Kie_zun

Web applications are rendered less usable by frequent problems such as incorrect dynamically produced websites and web script crashes. The dynamically created webpages that are common on the Internet today are too complex for the webpage validation technologies available today. We introduce a method for creating dynamic tests in the field of dynamic Web applications. The method makes use of explicit-state model checking in addition to combining tangible and symbolic executions. In order to provide bug reports that are concise and helpful in identifying and resolving the underlying issues, the approach autonomously builds tests, executes the tests while collecting logical limitations on components, and eliminates the conditions on the data that lead to failing assertions.

2) Reducing the effort of bug report triage: Recommenders for development-oriented decisions

AUTHORS: J. Anvik and G. C. Murphy

The error report library is an essential interactive hub for many software development teams. Reports supplied to the trunk must be prioritized regardless of they are several ways in which their utilization might enhance the procedure for creating software. A trigger assesses the significance of a report. After that, reports with meaning are arranged for incorporation into the project's development procedure.This article describes a method used by machine learning to generate recommendations that help with various judgments targeted at optimizing the design process, hence aiding triggers in their task. The advisors generated with this method are precise; for example, we have built recommendations with accuracy that ranges from 70% and 98% over five open source projects when deciding which developer should be given a report.

2.0 PROBLEM STATEMENT

More than forty-five percent of the resources used by software companies are dedicated to managing and fixing defects in their products. Bug triage is a critical phase in the bug-fixing process that entails accurately assigning a new bug report to the best suitable developer. In order to guarantee that defects are resolved quickly and effectively, this step is crucial. But triaging bugs is a labour-intensive procedure, and manual triage can be time-consuming and impracticable in large projects due to the sheer volume of issue reports. Text categorization techniques have been used to automate the bug triage process in an effort to reduce this burden. Nevertheless, the volume and inconsistent quality of the problem data frequently make computerized bug triage less effective. As a result, the demand for effective

2.1 EXISTING SYSTEM

- Sandusky Tal create a bug report network to look at the dependencies between problem reports

  In addition to analyzing the connections between reported problems, Hong et al. created a development social media page based on Mozilla initiative bug data to look at worker collaboration. This social media site for developers is useful for learning about the development
- Quantal determine who is responsible for prioritizing in opensource problem libraries by matching bug categories to researchers. Maintaining software chores can be aided and programmers can be distinguished by their developer priority system. Zimmermann. create questionnaires that are distributed to users and developers of three open-source projects in order to examine the quality of bug data. They define what constitutes a good bug report and train a classifier to determine whether a bug report's quality needs to be raised based on the examination
- By increasing the expense of handling defects, duplicate bug reports reduce the quality of bug data. Wang et al. create a natural language processing method by comparing the execution details in order to identify duplicate bug reports.

DISADVANTAGES OF EXISTING SYSTEM:

The vast and intricate data found in software repositories is not entirely appropriate for conventional software analysis. In conventional software development, a human trigger, or an expert developer, manually assigns a new bug's priority. Manual bug triage is time-consuming

and inaccurate because of the sheer volume of issues encountered every day and the lack of experience
with all of them.

Proposed System

In this work, we tackle the issue of data reduction for bug triage, i.e., how to decrease the bug data
The goal of data reduction for bug triage is to eliminate redundant or uninformative bug reports and
In our study, we concurrently minimize the bug dimension and the word dimension by combining the known methods of feature selection and instance selection. The reduced bug data offers comparable information over the original bug data while having fewer bug reports and fewer words. Two criteria are used to assess the decreased bug data: The measurement.

ADVANTAGES OF PROPOSED SYSTEM:

The findings of the experiment indicate that while using the instance selection technique on the data set can lower the number of issue reports, it may also affect the accuracy of bug triage. The accuracy can be raised and the number of words in the bug data can be decreased by using the
Concurrently, merging the two methods can improve precision while decreasing the amount of words
Our prediction algorithm can forecast the reduction order with an accuracy of 71.8 percent based
We introduce the data reduction problem for bug triage. This project seeks to improve the bug triage data set in two ways: first, by concurrently lowering the scales of the bug and word dimensions and second.

2.2 FEASIBILITY STUDY

In the last stage, the project's financial feasibility is examined, and an enterprise offer with a fundamental project outline and a few cost estimates is presented. During this evaluation procedure, the predicted its viability needs to be examined. This is to ensure that the recommended solution does not burden the firm. For a feasibility evaluation to be conducted, a basic understanding of the requirements of the method is required.

The following three factors are crucial to the practicality analysis:

- ECONOMICAL FEASIBILITY

- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

2.2.1 ECONOMICAL FEASIBILITY

The study's goal is to assess the strategy's possible financial impact on the business. The company can only invest so much money in system research and development. The expenses must be justified. Since most of the programs used were open source, the developed solution could also be deployed within the budgetary constraints. The customized products were the only things that had to be purchased.

2.2.2 TECHNICAL FEASIBILITY

This study aims to assess the technical requirements, or technical appropriateness, of the system. The development of any system must not overburden the available technological capabilities. As a result, there will be strong demand for the technological resources that become available. The client will therefore have to adhere to stringent restrictions. The system that is developed must be simple in nature, requiring little to no alterations to be put into place.

2.2.3 SOCIAL FEASIBILITY

One of the objectives of research is to assess how acceptable the system is to users. This entails instructing the user on how to operate the equipment efficiently. The framework The user needs to accept it as a necessity rather than a threat. The methods employed to acquaint and instruct the user about the system will ascertain the degree of openness displayed by the clients. As the final user of the system, he needs to have more faith in himself before he can provide some insightful criticism,

constructive criticism is welcomed since he is the system's ultimate user.

2.3 TOOLS AND TECHNOLOGIES USED

Java Technology

Java technology is both a programming language and a platform.

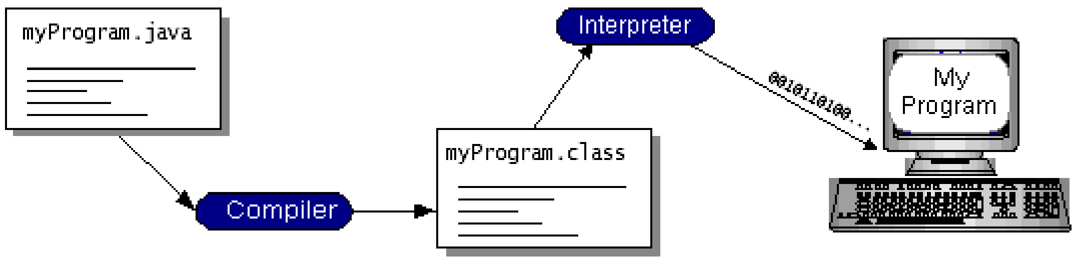
The Java Programming Language

All of the following buzzwords can be used to describe the high-level programming language

Java:

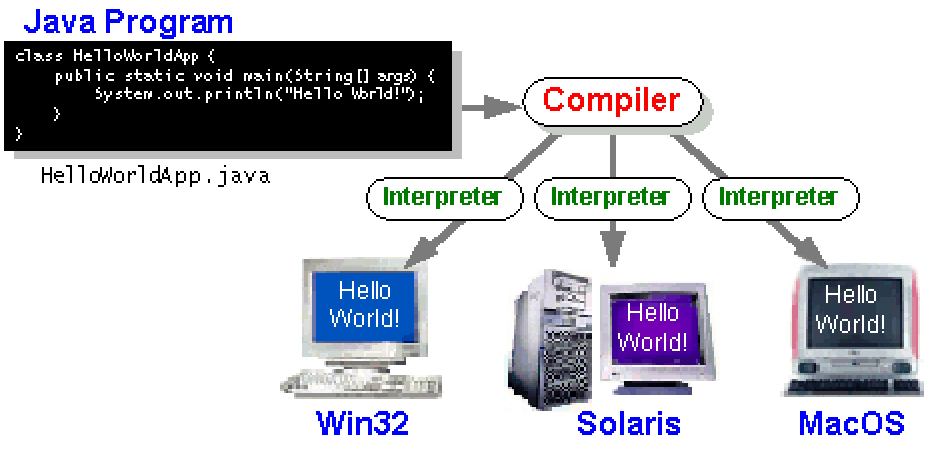
- Simple
- Architecture neutral
- Object oriented
- Portable
- Distributed
- High performance
- Interpreted
- Multithreaded
- Robust
- Dynamic
- Secure

A computer's compiler parses and executes each Java byte-wise command. Instead of build, which happens only once, interpret happens each time the application is run. To run a single application on the machine you use, you must either assemble its or comprehend it using one of themajority of programming dialects. The reality that the code can be both translated and translated makes the programming language used by Java unique. The first step in using the compilation tool is translating an executable into Sun byte rules, which are platform-neutral symbols that the Android platform's interpretation can understand. Every one Java byte programming command is processed and executed by the compiler on the system being used. While interpreting happened whenever the program is run, assembly only happens sometimes. The way this operates is shown in the accompanying figure.



Java bytes codes can be thought of as a Java Virtual Machine’s (Java VM) software command. All Java interpreters, be they Web browsers capable of executing applets and instruments for developing, are just implementations of the Java Virtual Machine. It is feasible to "write a single time, operate wherever" with the aid of Java bytecode sequences. Any computer with a

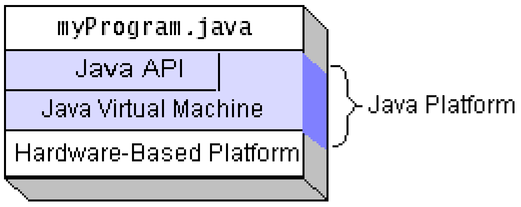
Java compiling person can be managed to compile your application into byte instructions. After that, the byte instructions can be executed on any Java virtual machine. This implies that identical Java software may operate on a Windows 2000 system, a Solaris place of employment, or an iMac as lengthy as it limits a Java virtual system (VM).



```
class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

The Java Platform

In this context, a platform refers to the mechanical architecture or programming that a software runs on. As we've already discussed, some of the most prevalent operating systems include Window 2000, Linux, Solaris, and MacOS. Most designs can be described as a combination of an atmosphere for operation and equipment. Differentiating itself from most other platforms, the Java platform is a
The Java Software Programming Interface (Java API) and the Java virtual machine (Java VM) are the two components that make up the platform that runs Android. You've already been shown the
The Java API merely makes a wide range of pre-made applications—such as computer user interface (GUI) widgets—available. Bundles, which are collections of related classes with concepts, are how the Java API is arranged. The next part is called What Java Technology Can Do. illustrates the features available in some of the different Java application programming interface packages.
The following figure displays a Java program that is particular to a given platform. As displayed in the screenshot below, the operating system and Java's API protect the software from any devices.



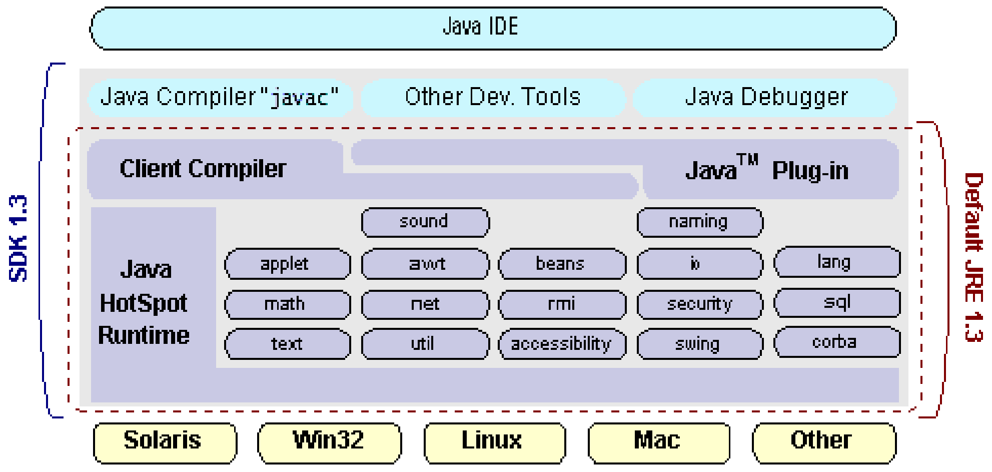
Native code is code that, upon compilation, operates on a particular equipment environment. Java platforms may operate a little slower over native code because it is a vendor-independent context. Just-in-case bit-code compilers, intelligent compilers, and properly calibrated processors can all achieve speed that approaches native code with jeopardizing universality.

What Can Java Technology Do?

Apps and apps are the double most popular program kinds crafted with a language like Java. You are undoubtedly comfortable with applets if you have ever perused the Web. An application is a software

Still, there's more to the Java language than merely creating adorable, amusing Web applets to development framework. A piece of software that operates independently atop the platform known as Java is called an application. On a network, a unique type of application known as a server provides client support and service. Web servers, proxy servers, mail servers, and other services are illustrations of platform gives you the following features:

- The fundamentals: Objects, text, threads, and values, input and the result, data constructions method houses, date along with time, along with so on.
- Apps: The collection of meetings utilized by widgets.
- Socializing: URLs, TCP (Transfer Control Protocol), UDP (User It Each protocol) outlets, and IP (Internet Protocol) tackles.
- Globalization: Aids in creating courses that can be consequently translated into different tongues as well as locales.
- Safety: Both modest and excellent, including computer signatures, public alongside discrete key executives, gain access control, and certificates.
- Software components: Plug into existing aspect architectures with Java Beans TM.
- Object serialization: Enables lightweight programming.



Will Java Technology Change My Life?

- If you learn for coding in Java, we can't guarantee you fame, wealth, or even a job. It still needs less work than other tongues and will probably improve your projects. We think a few
- Get going right away: Despite being a strong based on object- words, Java is simple to learn, in particular for coders who are already conversant with C or C++.
- Reduce the amount of code written: Application metrics (class counts, method counts, etc.) studies indicate that a Java program can be up to four lower than a program in C++ of a comparable kind.
- Improve the code: Good coding procedures are encouraged by the use of Java language, while garbage pickup keeps memory leaks at bay.

ODBC

the corporation Open Database Connectivity (ODBC) is a standard programming connection for database platforms practitioners and implementation creators. Prior to ODBC becoming the accepted norm for desktop apps that worked alongside tables, coders had to use private tongues for every database you sought to have access to. Today, ODBC, or making their database system preference nearly insignificant to a coding viewpoints, which is appropriate because app makers have more significant issues to worry about beyond the syntax required to port their software from a single server to another when their requirements suddenly alteration. Programmatically speaking, the appealing feature of ODBC is that software can be created to connect with any kind of data, no matter the SQL vendor, using the same set of function calls. When the app communicates with the Microsoft SQL Server nor Oracle doesn't affect its original code. These are the only two we bring up as examples. The drivers for ODBC are accessible for a number of well-known database systems. It is possible to convert simple text files including Excel documents into historical information. The OS determines which a small amount ODBC drivers are required to communicate with the data sources (such as the Oracle or SQL Server interface) using the Registry data supplied by the ODBC Manager. The driver packages for ODBC are loaded in an honest way.

JDBC

In an endeavour to define an isolated network standardization API for Java; Sun Labs established Java Network Connection, or JDBC. A uniform connection to a bounds of RDBMSs is gave by the flexible SQL record access technology offered by JDBC. "Plug-in" dB connecting components, also known as drivers, are used to create this uniform access. If a system manufacturer intends to have Jd assistance he or she would deliver the code for each device the the store and Swing run on.
To gain a larger recognition of JDBC, are Sun modelled JDBC’s structure on MySQL. As you found earlier in these chapter, ODBC has extensive acceptance on a multitude of devices. Based JDBC on ODBC may permit companies to deliver JDBC adapters to market much easier than inventing
The JDBC was unveiled on February 1996. It was made available for scrutiny by the public for ninety days, ending on June 8, 1996. Soon later, the final JDBC v1.0 specification was published as a result
The next section will offer students with enough info on JDBC to comprehend who it is and the best way to utilize it efficiently. This is by no means an exhaustive introduction to JDBC. That may take up a whole book. While this section provides an overview of JDBC's purpose and effective usage, it does not encompass a comprehensive examination of JDBC, which spans entire books. The objectives that JDBC was given are significant. why specific functions and classes act in the ways that they do. The following are the eight JDBC design objectives:

SQL Level API: JDBC aims to provide a SQL language interface suitable for both application programmers and higher-level tools, striking a balance between abstraction and usability.

SQL Conformance: JDBC supports passing any SQL query statement to underlying database drivers, accommodating the varied syntax across different database vendors.

Implementation a top Common Database Interfaces: JDBC is built to Layered upon standard database interfaces, enabling compatibility with existing ODBC drivers through a software interfaces.

Consistency with Java System: JDBC maintains alignment with the broader Java system, ensuring its interface is consistent with established Java design principles.

Simplicity: JDBC simplifies its design to ensure clarity and ease of use, avoiding duplicate functionalities to minimize confusion for developers.

Strong Static Typing: Utilizing strong, static typing enhances error checking during compilation, reducing runtime errors and enhancing overall code reliability.

Keeping Common Cases Simple: JDBC prioritizes ease of use for common SQL operations like SELECT, INSERT, DELETE, and UPDATE, while still supporting more complex SQL statements when needed.

2.4 System Requirement

➢ Hardware Requirements

• System	: Pentium IV 2.4 GHz.
• Hard Disk	: 40 GB.
• Floppy Drive	: 1.44 Mb.
• Monitor	: 15 VGA Colour.
• Mouse	: Logitech.
• Ram	: 512 Mb.

➢ SOFTWARE REQUIREMENTS:

• Operating system	: Windows XP.
• Coding Language	: J2EE
• Data Base	: MYSQ

CHAPTER 3
SOFTWARE REQUIREMENTS SPECIFICATION

3.1 USERS

3.1.1 User Functions

The project aims to design a system for data protection during transmission and reception. The system will be simple and effective, allowing users with basic storage knowledge to utilize it. The project introduces the concept of device authority for decrypting files and implements a method for revoking device access. It involves creating a system to transmit encrypted files and decrypt them using specific keys, alongside searching for files and sending the corresponding keys.

3.1.2 Admin Functions

Based on the survey, an admin login page is required. Before logging in, the owner can register on the site. The admin page will allow users to log in using a username and password and upload files based on user requirements. This plan is essential for collecting and organizing information.

3.2 FUNCTIONAL REQUIREMENTS

The fundamental tasks that the system must complete are outlined in the functional requirements. The three primary types of these criteria are: cloud details, admin details, receiver details, and their functionalities.

- Clustering Server
- Customer
- Marketing

1. **Clustering Server**:

- With the right data, we can identify high homogeneity within groups and high uniqueness among individuals.

- A two-layer clustering model based on client attributes, contributions, and segment analysis to accurately categorize client data.
- Businesses can track and differentiate client usage, using the clustering model for business analysis to detect changes in customer behaviour and adjust product processes to retain top customers.

2. **Customer**:

- This research can manage quality homogeneity across large and small customer bases, aiding in client relationship management (CRM) and marketing standards.
- The methodology helps businesses prepare for long-term CRM and retain loyal customers, with temporary advertisers using it for targeted product or service promotion.
- Business applications include targeted advertising, customization services, CRM enhancement, and analysis of client behavior and preferences.

3. **Marketing**:

- Creating appropriate segments allows the company to target customers and develop CRM and marketing strategies.
- Cross-examination of data maintains customer groupings and supports marketing strategies.
- Providing diverse, rich client data through pre-analysis strengthens the target customer base and eases the burden on marketing personnel.
- Utilizing data mining to identify potential customers, extend advertising reach, and improve precision in targeted advertising.

3.3 NON-FUNCTIONAL REQUIREMENTS

3.3.1 Performance Requirements

- Performance requirements relate to the system's response time to requests.
- The system should support user requirements and meet end-user needs.
- Login details will be verified within seconds across user, cloud, and admin information.
- The product should ensure security through a device that holds part of the password, with the other part stored on the system.

3.3.2 Safety Requirements

- The product must meet safety standards, allowing the system to revoke access to devices that are lost or stolen. The product will disable such devices and assign new passwords.

3.3.3 Security Requirements

- The system should include security features and password authentication for devices. Exception handling mechanisms should prevent errors.

3.3.4 Software Quality Attributes

Availability:

- The program should not hang, will open quickly, and access data promptly, ensuring user satisfaction.

Reliability:

- The system should identify and reject erroneous inputs, displaying error messages when necessary, and should not fail. It should operate in an acceptable manner even when interfacing with the operating system.

Usability:

- The system should transmit well with other clients and distribute data efficiently.

Maintainability:

- The structure would be created for long-term use and be well-maintained, with secure device password management.

Portability:

- The structure should run on any web browser and platform with little or no modifications.

CHAPTER 4

METHODOLOGY

1. Data Collection and Preprocessing

1.1 Data Collection

Source Identification: Identify and collect bug reports from various repositories such as GitHub, Bugzilla, and JIRA.

Data Extraction: Extract relevant fields from bug reports, including report ID, title, description, reporter, assignee, creation date, and status.

1.2 Data Cleaning

Duplicate Removal: Remove duplicate bug reports to ensure the dataset contains unique instances.

Missing Values Handling: Address missing values by either filling them using imputation techniques or discarding incomplete reports.

Noise Reduction: Filter out irrelevant or noisy bug reports that do not contribute meaningful information for bug triage.

2. Data Reduction Techniques

2.1 Instance Selection

Technique Selection: Choose an instance selection technique, such as clustering-based selection, to cut the figure of bug reports.

Application: Apply the chosen technique to the dataset to retain a subset of bug reports that are representative of the original dataset.

Evaluation: Evaluate the reduced dataset based on the scale of the data and the impact on bug triage accuracy.

2.2 Feature Selection

Technique Selection: Select a feature selection technique, such as Chi-Square, Information Gain, or Principal Component Analysis (PCA), to shrink the integer of words/features in the bug reports.

Application: Apply the selected technique to remove non-informative and redundant words from the bug reports.

Evaluation: Assess the reduced dataset created on the dazed of retained text and the effect on bug triage accuracy.

3. Predictive Model for Reduction Order

3.1 Attribute Extraction

Historical Data Analysis: Extract qualities from historic bug data sets, such as the number of bug reports, average length of reports, frequency of words, and distribution of bug severity levels.

Feature Engineering: Engineer extra features that may persuade the effectiveness of instance and nose choice techniques.

3.2 Model Training

Binary Classifier: Train a binary classifier (e.g., logistic regression, decision tree, or random forest) using the extracted attributes to predict the optimal order of employing example collection and characteristic selection.

Training Data Preparation: Prepare a labelled training dataset by manually determining the effective reduction order for a subset of historical bug data sets.

3.3 Model Evaluation

Evaluation Metrics: Use evaluation metrics such as accuracy, precision, recall, and F1-score to assess the performance of the predictive model.

Cross-Validation: Perform cross-validation to ensure the robustness and generalizability of the model.

4. Combined Approach Implementation

4.1 Sequential Application

Order Prediction: Use the trained binary encoder to forecast which attributes to select first for fresh bug records and in what sequence to apply case selection.

Instance Selection: Apply instance selection first if predicted, followed by feature selection.

Feature Selection: Apply feature selection first if predicted, followed by instance selection.

4.2 Evaluation of Combined Approach

Scale Reduction: Measure the reduction in the integer of bug intelligence and words in the reduced dataset.

Accuracy Assessment: Evaluate the truthfulness of error triage using the reduced dataset compared to the original dataset.

5. Experimental Setup

5.1 Dataset Description

Provide a detailed type of the datasets used, including the source, size, and characteristics of the bug reports.

5.2 Experimental Procedure

Outline the steps followed in the experiments, including data collection, preprocessing, application of reduction techniques, model training, and evaluation.

5.3 Performance Metrics

Define the metrics used to estimate the efficiency of the data reduction techniques and the predictive model, such as reduction rate, triage accuracy, and computational efficiency.

6. Results and Discussion

6.1 Reduction Results

Present the results of applying instance selection and feature selection, including the number of bug reports and words reduced.

6.2 Triage Accuracy

Discuss the impact of data reduction on the accuracy of bug triage, highlighting any improvements or trade-offs observed.

6.3 Predictive Model Performance

Report the performance of the predictive model in determining the optimal reduction order, including accuracy and other relevant metrics.

6.4 Comparative Analysis

Compare the results of the combined approach with the individual application of instance and feature selection techniques.

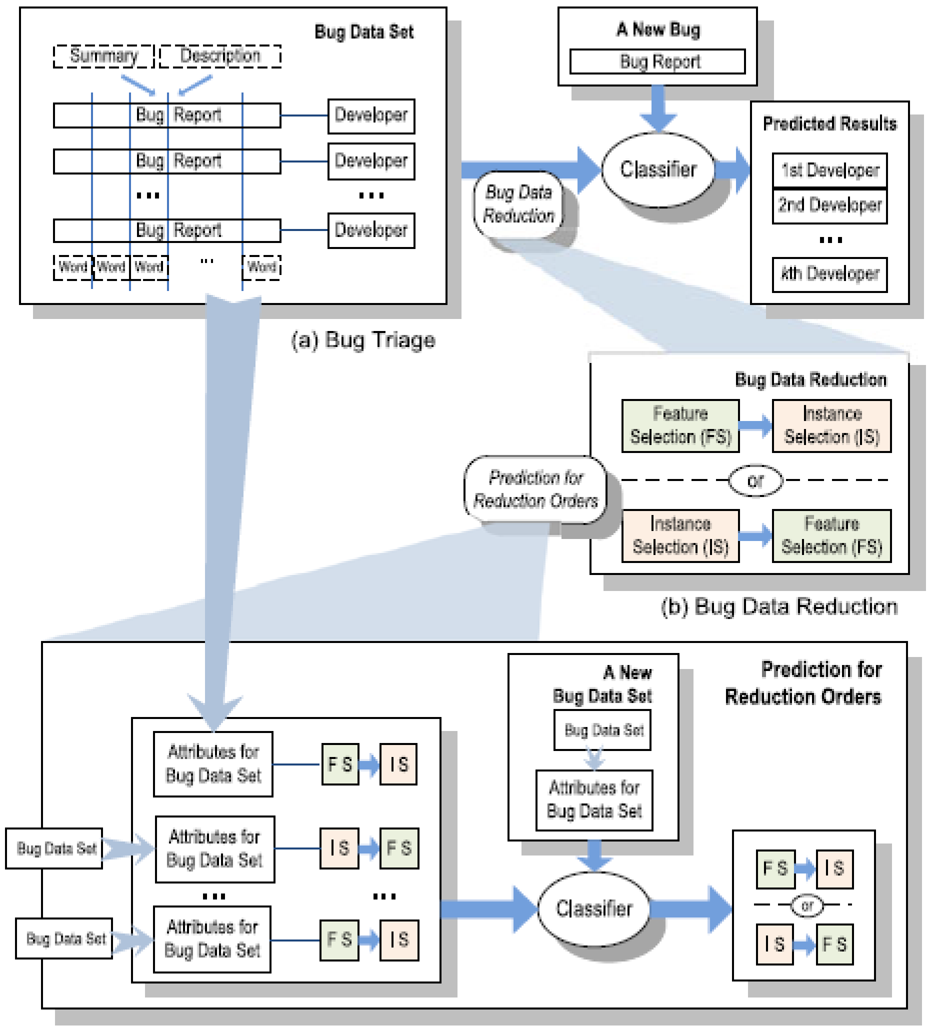
6.5 Insights and Implications

Provide insights into the findings and discuss the implications for improving bug triage processes in software development.

CHAPTER 5

SYSTEM DESIGN

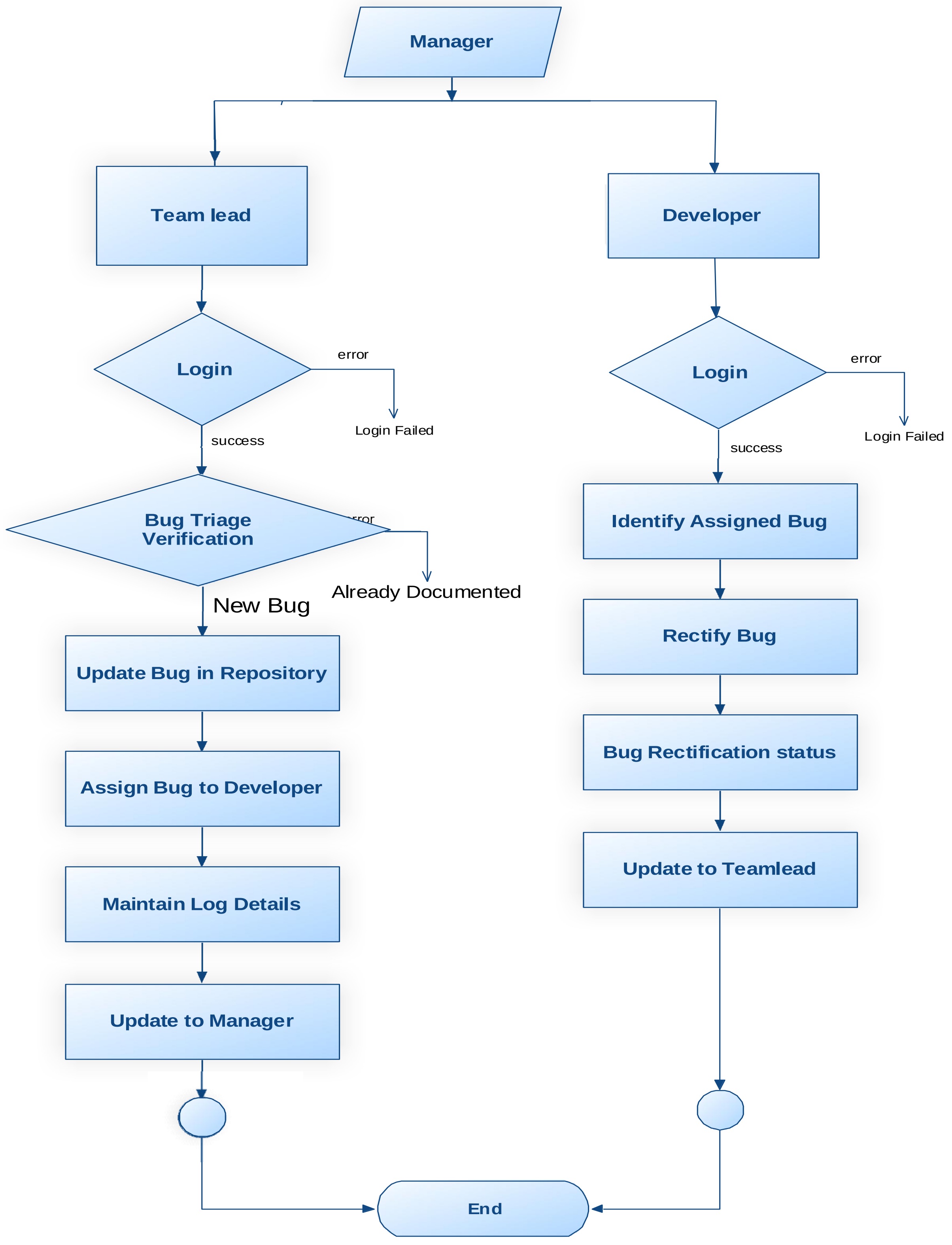
4.1 SYSTEM ARCHITECTURE:



4.2 Data Flow Diagram

- Another name for It's a bull market chart, that of the DFD. It's an straightforward graphical formalism that may be used to illustrate how data enters a system.
- One of One of its most important tools for planning is the data flow flowchart. (DFD). It's employed to simulate the parts within the mechanism.
- DFD illustrates the flow of information through the system and the various changes that alter it.

- Another name for DFD is a bubble chart Each level of complexity can be utilised to portray the DFD-using system. DFD is separated among phases that correspond to escalating functional detail and information flow.



UML DIAGRAMS:

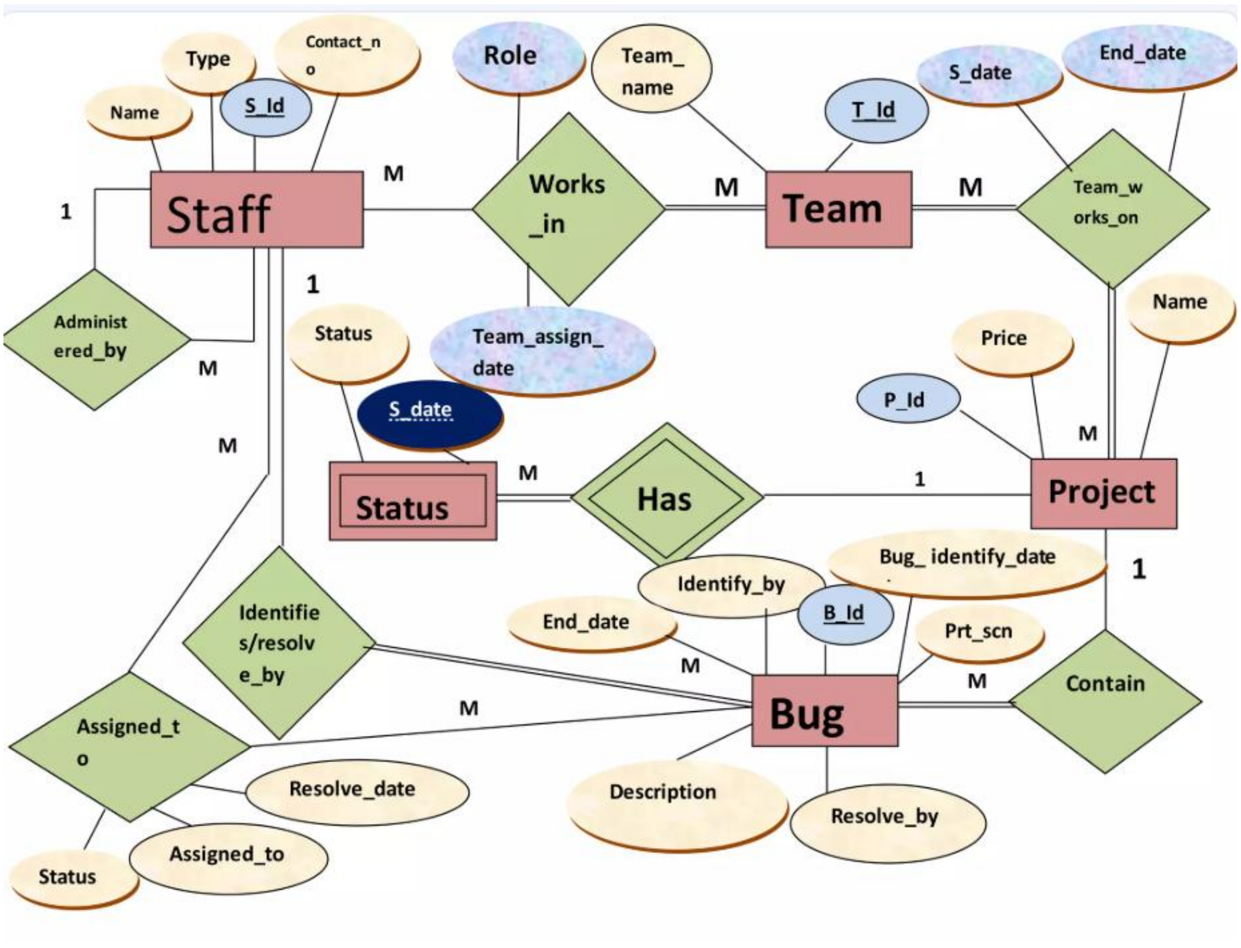
Unified modelling language is referred to as UML. A universal modelling language with standards, UML is used n the field of software engineering that is object focused. The Object Management Group, which oversees and developed the standard. The intention is for UML to prove it as a standard language for designing object-oriented software. UML now consists of a meta-model and a code as its two main parts. In the future, UML might also be coupled or added to in the format of a technique or method. Unified Modelling Language (UM) is a widely used language in non-software networks, business modelling, and discussing, visualizing, creating, and documenting there facts of software systems. The UML is an integration of best modelling procedures that have been operational in recreating huge, complicated systems. The UML is an important part of the method of developing software and the creation of objects-oriented systems. It uses mainly graphical notations to convey the software's layout.

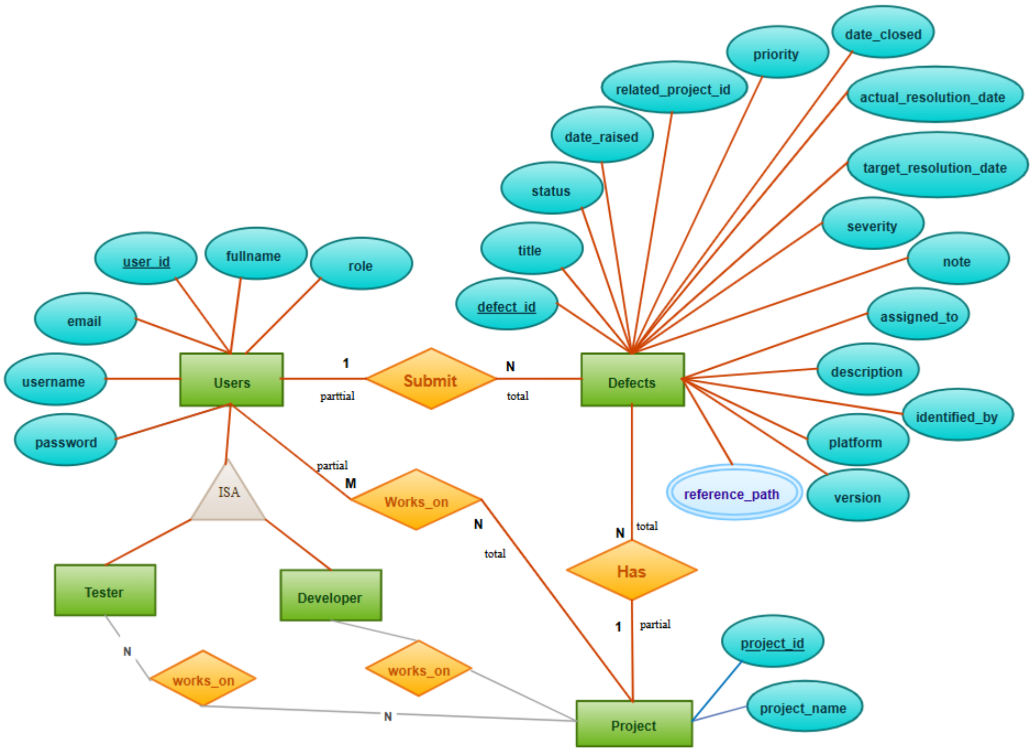
GOALS:

These are the UML's key basic aims:

- Supply users an expressive visual drawing language that is ready to use so they can construct and import meaningful models.
- Offer tools for specialize and extendibility to widen the key concepts.
- Not depend on a certain development methodology or programming language.
- Offer a legal system for knowing the model language.
- Foster business expansion of OO tools.
- Support the use of higher level design concepts like parts frameworks, patterns, and collaborations.
- Employ top tactics.

ER DIAGRAM:


Type
Contact_n o
Name
S_Id
Role
Team_ name
S_date
End_date
T_Id
Staff
Works _in
Team
Team_w orks_on
1
M
M
M
1
Administ ered_by
M
Status
Team_assign_ date
Name
Price
P_Id
S_date
M
M
Status
Has
1
Project
Identify_by
Bug_identify_date
1
End_date
B_Id
Prt_scn
Identifie s/resolv e_by
M
M
M
Contain
Assigned_t o
Bug
Resolve_date
Description
Resolve_by
Assigned_to
Status


priority
date_closed
related_project_id
actual_resolution_date
date_raised
target_resolution_date
status
severity
title
note
user_id
fullname
role
defect_id
assigned_to
email
Users
1
Submit
N
Defects
description
username
partial
total
identified_by
password
partial
M
platform
ISA
Works_on
reference_path
version
N
total
N
total
Has
Tester
Developer
1
partial
project_id
N
works_on
works_on
project_name
N
Project

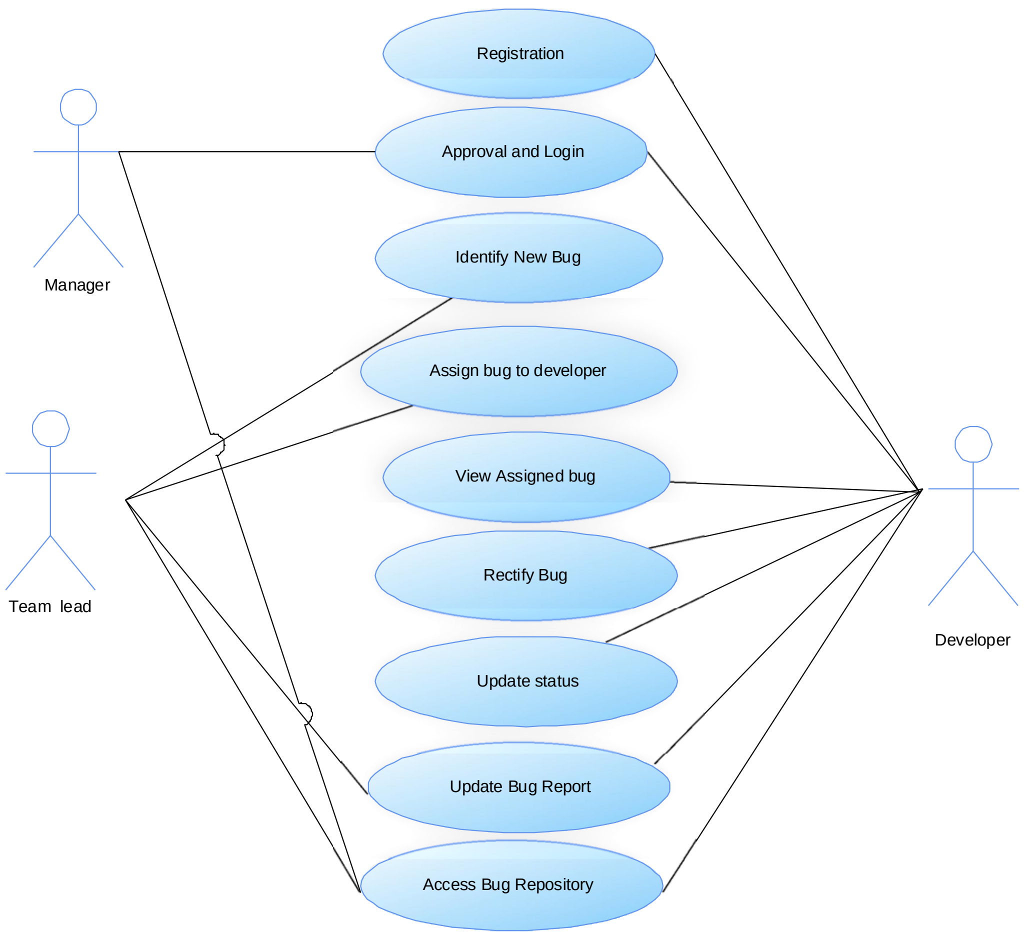
CHAPTER 6

DETAILS DESIGN

The **Detailed Design** phase provides a blueprint for building the system. It includes the design of the system architecture, database schema, module-level designs, and data flow, ensuring that all components are logically and efficiently structured.

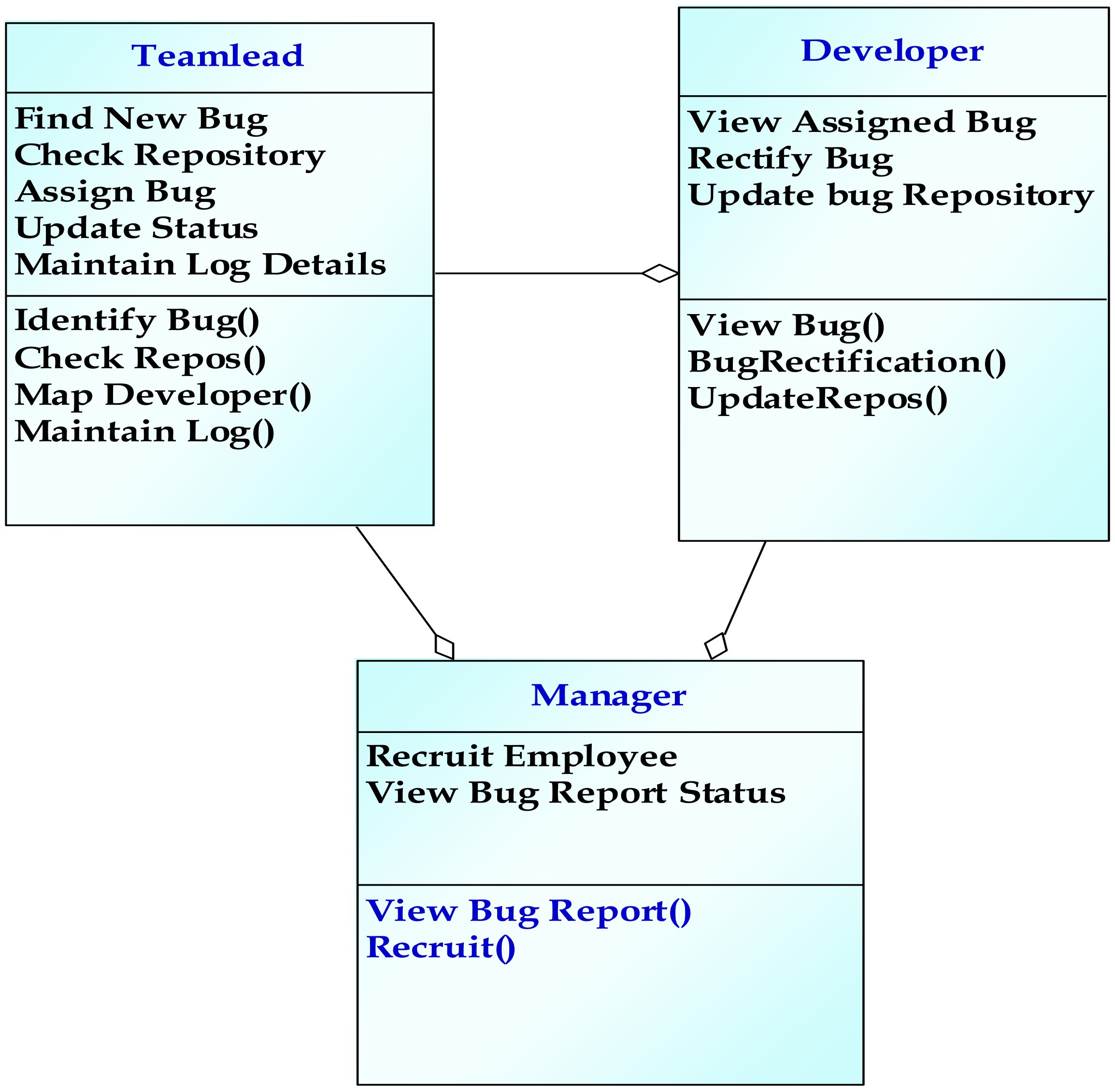
5.1 USE CASE:

A use case diagram visualizing the interactions between actors (Food Producer, Supplier Distributor, Retailer) and the blockchain system would illustrate these processes succinctly, showing how each actor relates with the blockchain to upload, verify, access, and validate crucial documents within the food supply chain. By implementing blockchain technology in this manner, the food industry can potentially transform its operations, making them more efficient, secure, and consumer-friendly, thereby addressing contemporary challenges in food safety and supply chain management.



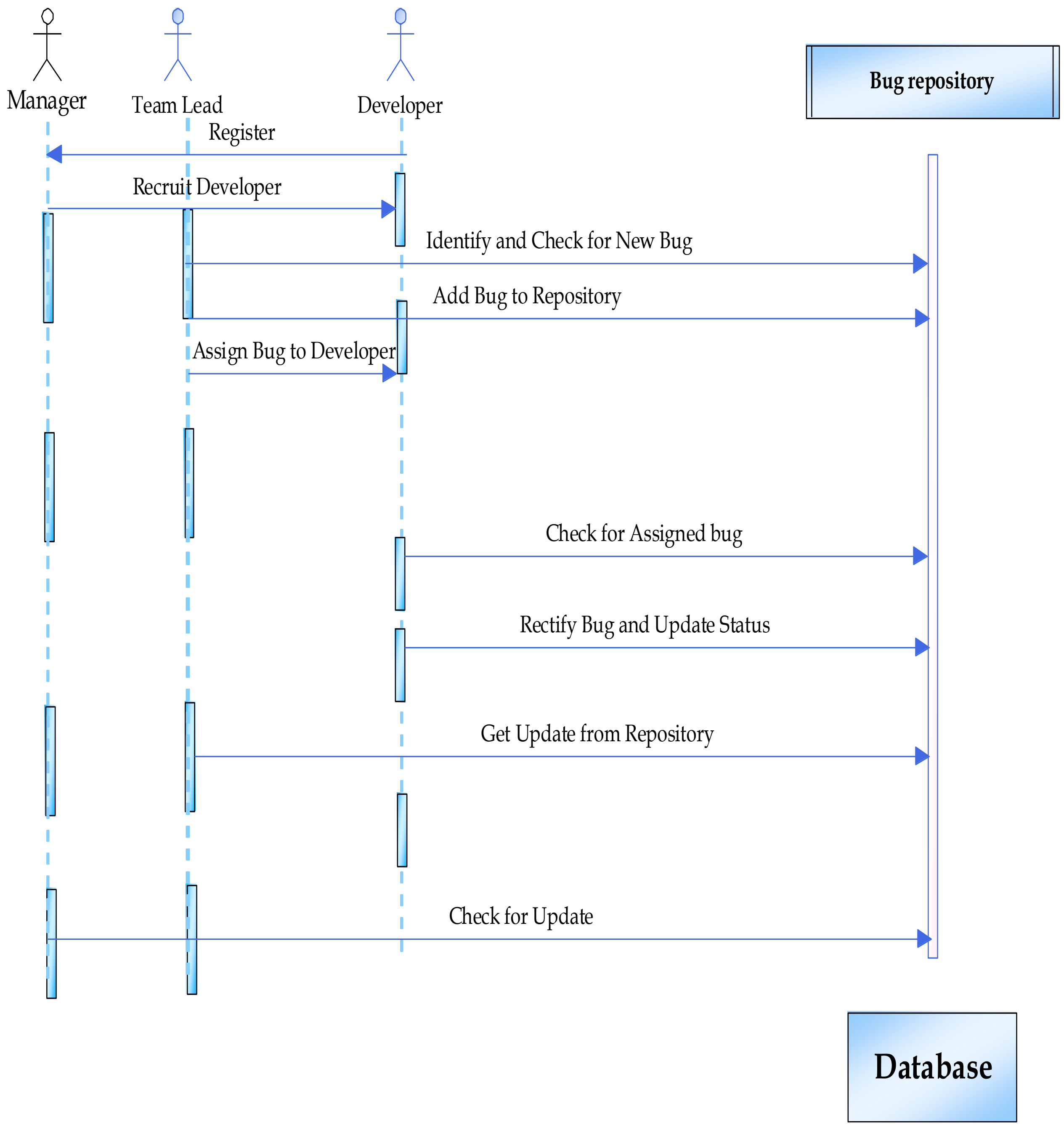
5.2 CLASS DIAGRAM:

Similar to a static structural diagram in the Unified Modelling Language (UML), a class diagram in software development shows the categories, traits, functions (or methods), and relationships between classes of the system. It shows which class the data is in.



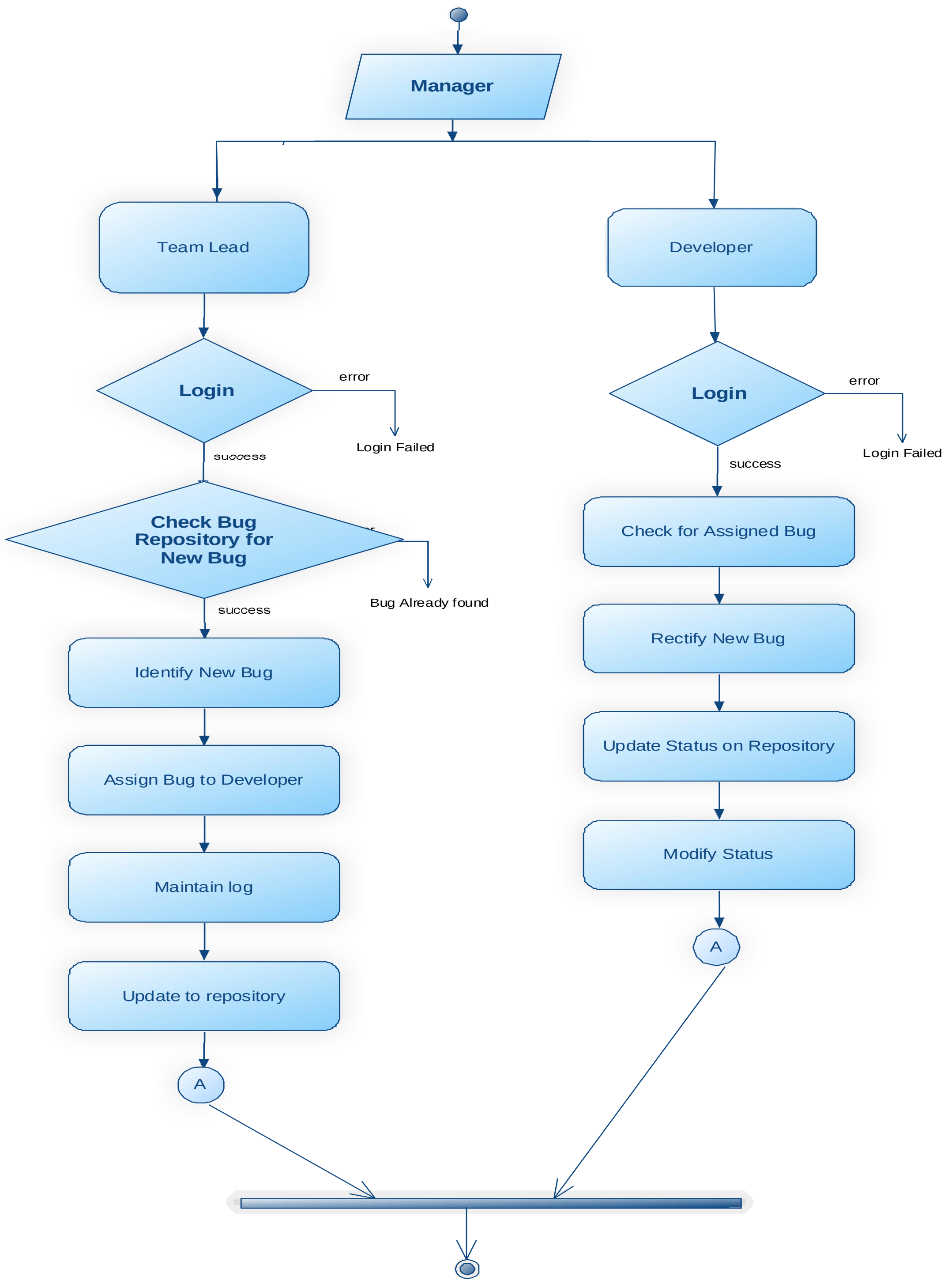
5.3 Sequence Diagram:

Sequence diagrams are one type of diagram called interaction that are included in the Coordinated Organising Language, which is or UML. They show connections and order in which procedures operate in conjunction with one another. This is a signal sequencing diagram. build. Drawings of sequences were represented by event charts, event circumstances, and other labels for timing graphs.



5.4 ACTIVITY DIAGRAM:

Visual activity diagrams are used to illustrate processes for consecutive tasks and actions that allow for choice, iteration, and concurrency. Consolidated Modelling Language activity diagrams are a familiar tool for explaining the sequential administrative and operational procedures of system components. A schematic of an activity.



CHAPTER 7
IMPLEMENTATION

6.1 Modules:

1) Manager

2) Team Lead

3) Developer

Manager:

1. **Dashboard Overview**:

- **Functionality**: Provides an overview of project statuses, including active bugs, resolved bugs, and overall progress.
- **Features**: Summary graphs/charts, project health indicators, and key performance metrics.

2. **Project Management**:

- **Functionality**: Allows the manager to view and manage projects, assign priorities, allocate resources, and set deadlines.
- **Features**: Project timelines, resource allocation tools, project dependencies, and milestone tracking.

3. **Team Management**:

- **Functionality**: Manages team members, their roles, permissions, and workload distribution.
- **Features**: Team member profiles, task assignments, performance evaluations, and skill assessment tools.

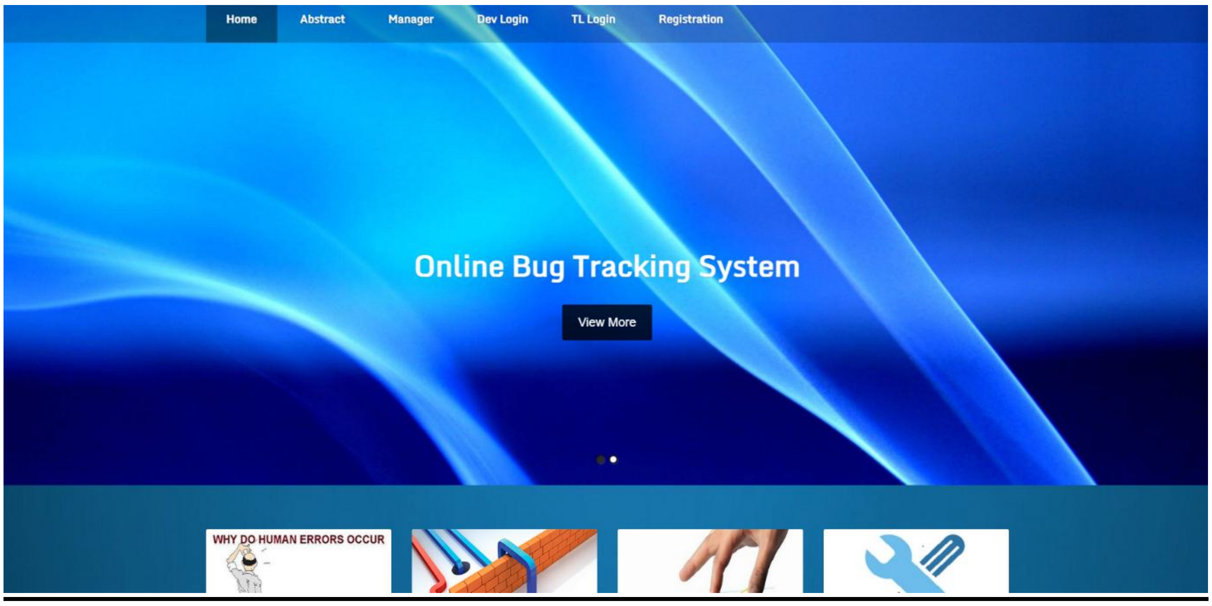
Team Lead:

Monitors and manages bugs assigned to the team, updates bug statuses, and ensures timely resolution. Bug assignment queues, priority setting, bug triaging, and escalation management. Assigns tasks to developers, tracks task progress, and manages dependencies. Task boards or lists, task assignments with deadlines, progress tracking, and task linking. Facilitates communication within the team, resolves blockers, and encourages knowledge sharing. Team chat channels, threaded discussions on bugs/tasks, collaborative editing of bug reports, and shared notes. Monitors team performance metrics, identifies bottlenecks, and implements process improvements. Performance dashboards, burn-down charts, sprint retrospectives, and productivity analytics. Ensures adherence to quality standards, performs code reviews, and validates bug fixes. Code review tools, testing frameworks integration, bug verification processes, and test case management.

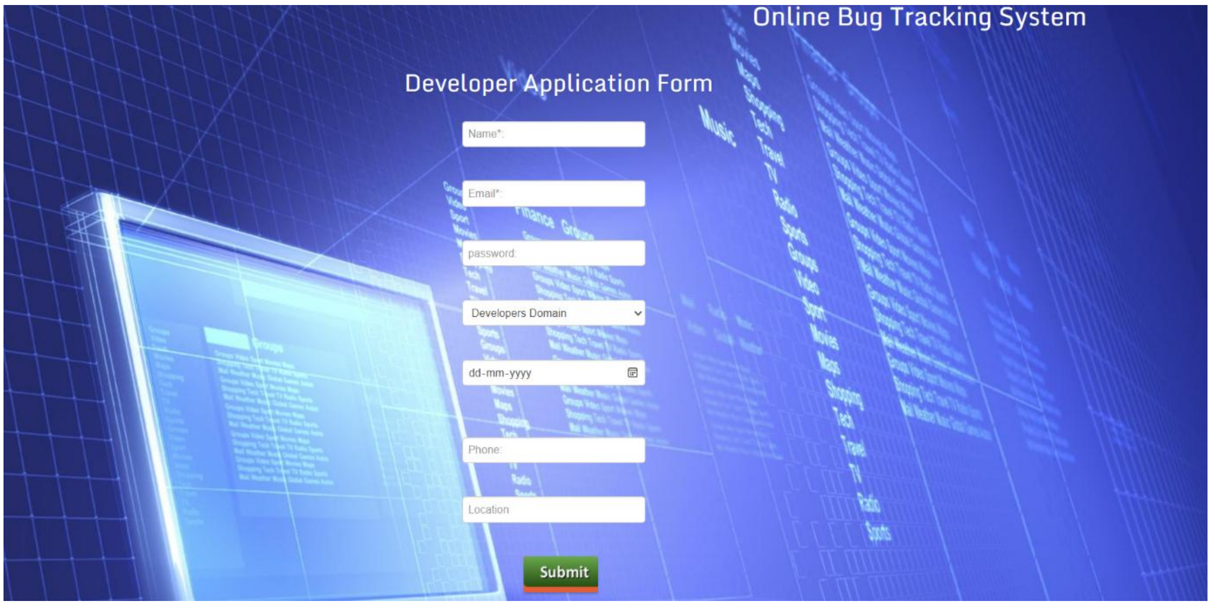
Developer:

Accesses detailed bug reports, understands issue context, and receives bug assignments. Bug attachments/screenshots. Investigates bugs, implements fixes, and updates bug statuses. Code change tracking, version control integration, bug fix comments, and automated testing feedback. Manages assigned tasks, updates task status, and communicates progress. Task boards or lists, task details, estimated vs. actual time tracking, and task dependencies. Collaborates with team members, seeks help when stuck, and participates in discussions. Team chat integration, comments on bugs/tasks, code review discussions, and notification settings. Tests bug fixes, validates solutions against bug reports, and ensures quality assurance. Test case execution, bug verification reports, regression testing tools, and feedback loops with QA team.

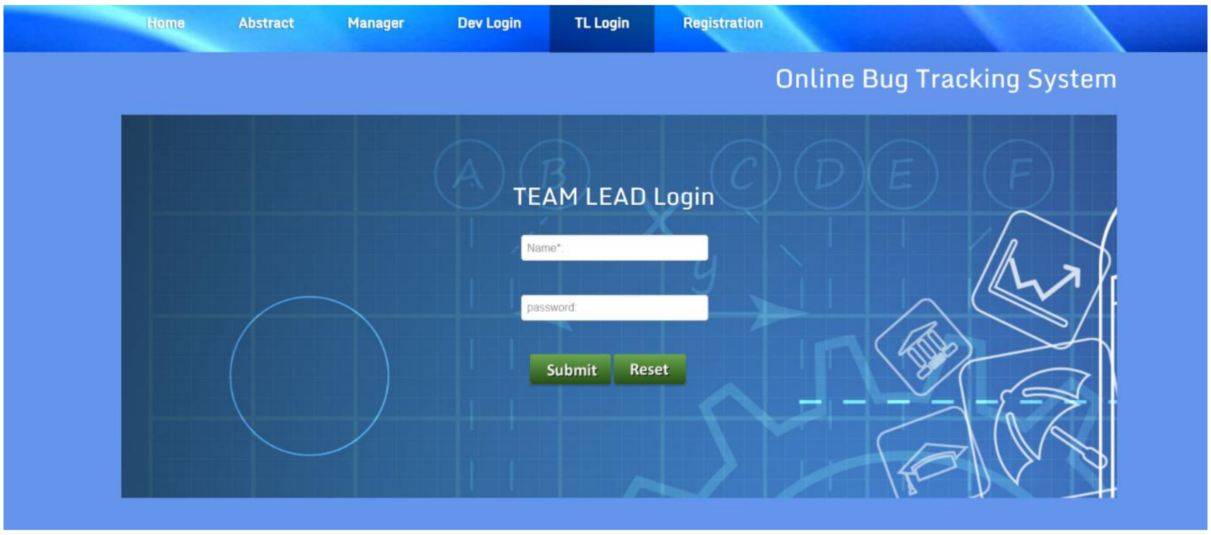
6.2 SCREEN SHOTS:



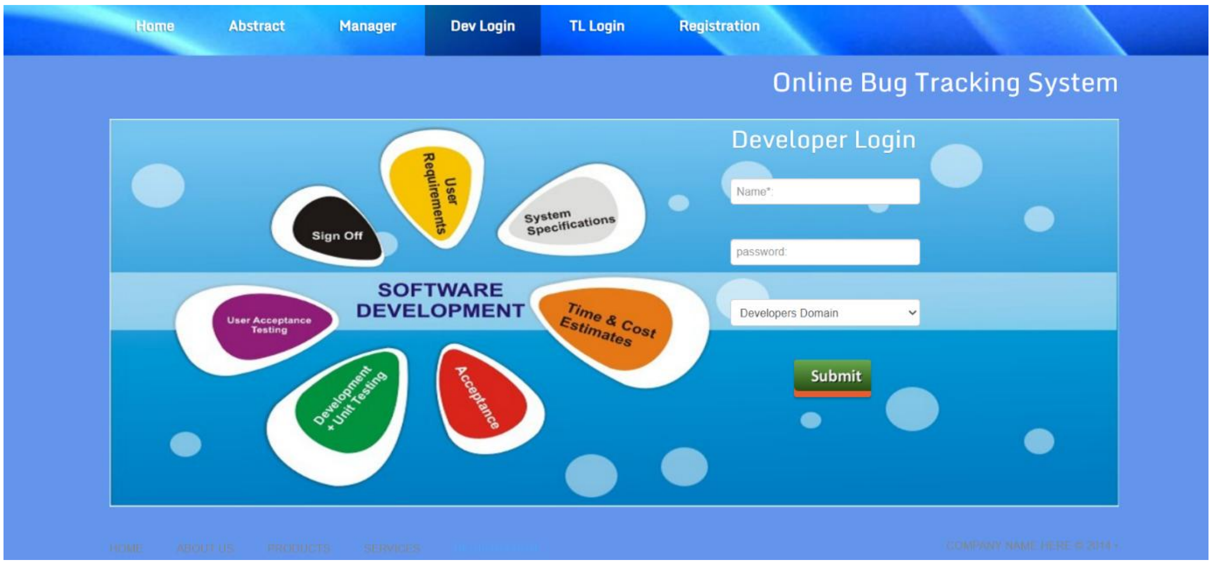
6.1.1. Home Page



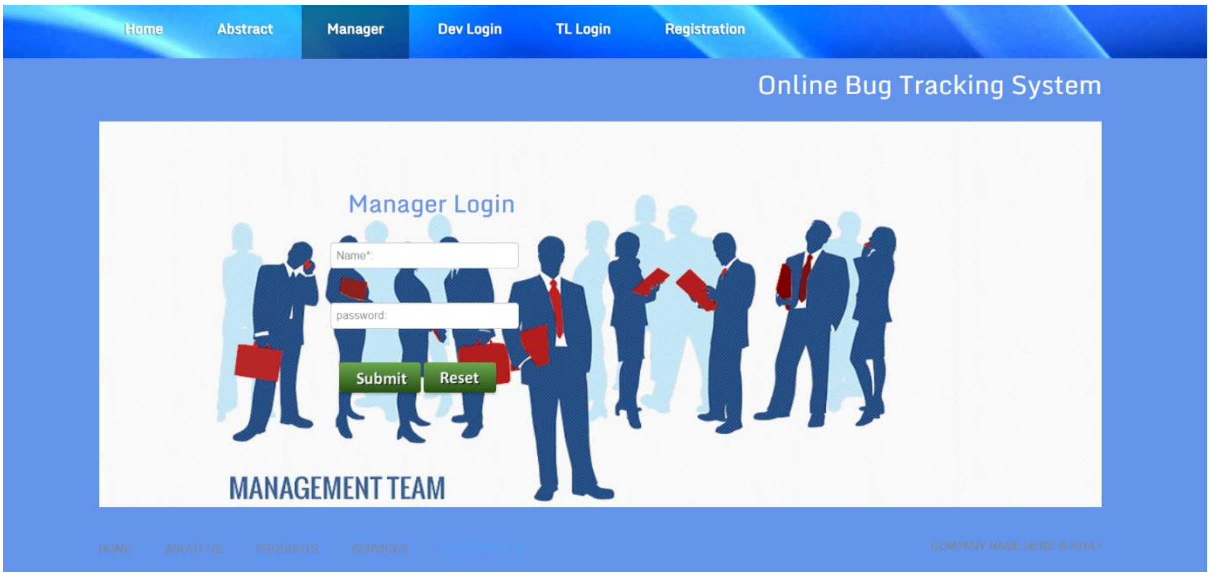
6.1.2.Registration Page



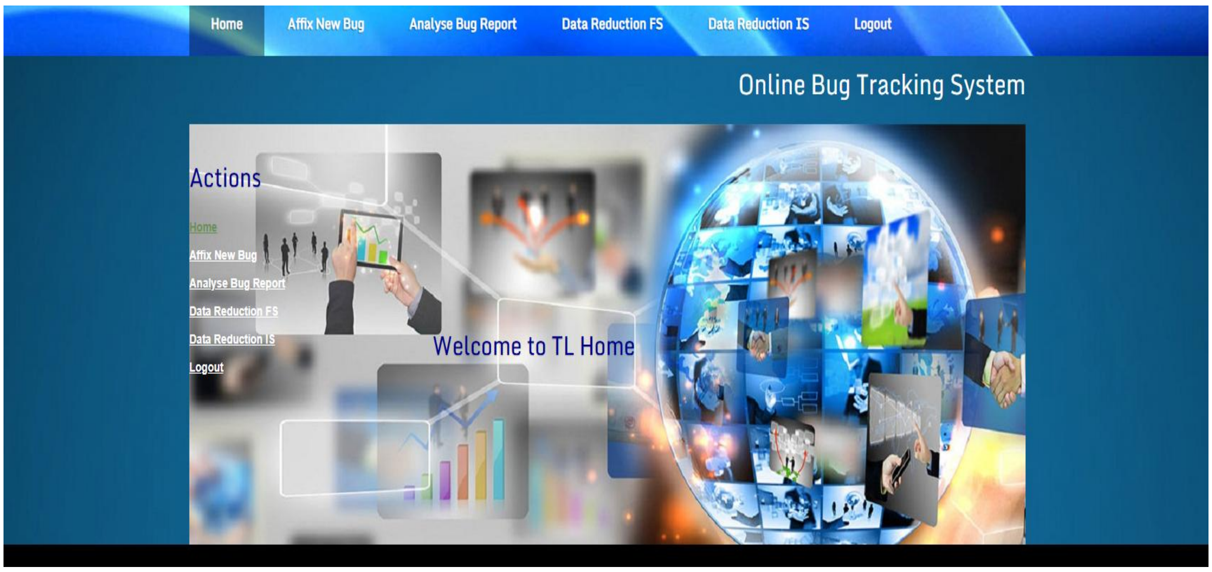
6.1.3.Team Lead Login page



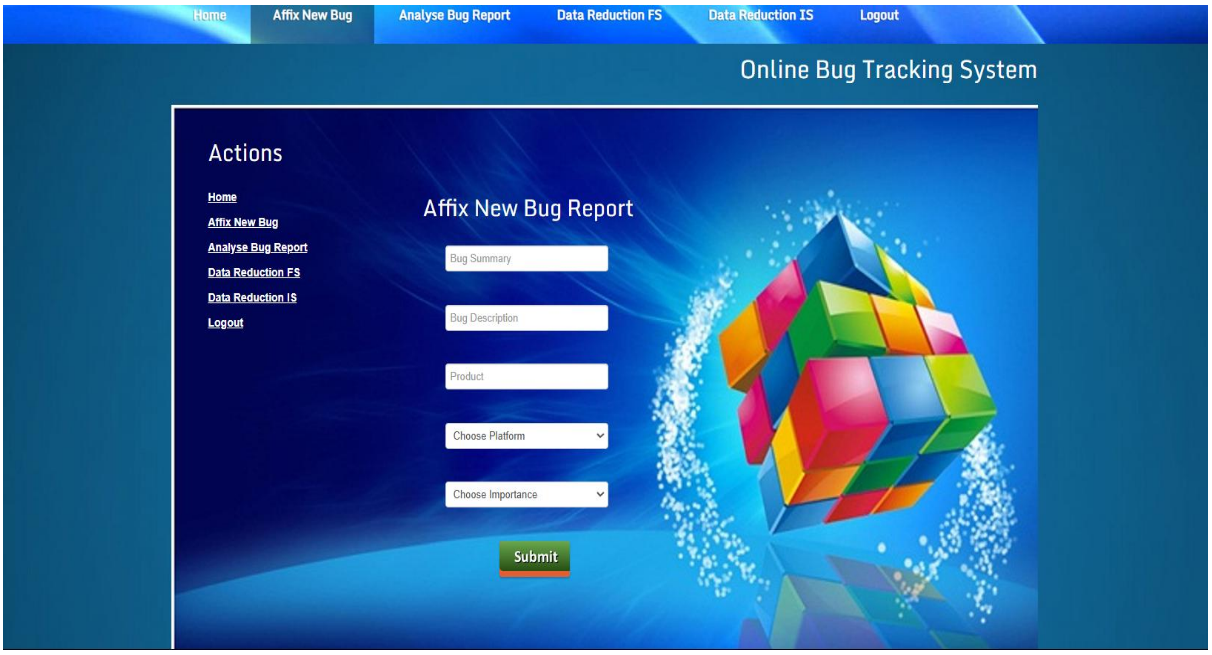
6.1.4.Developer Login Page



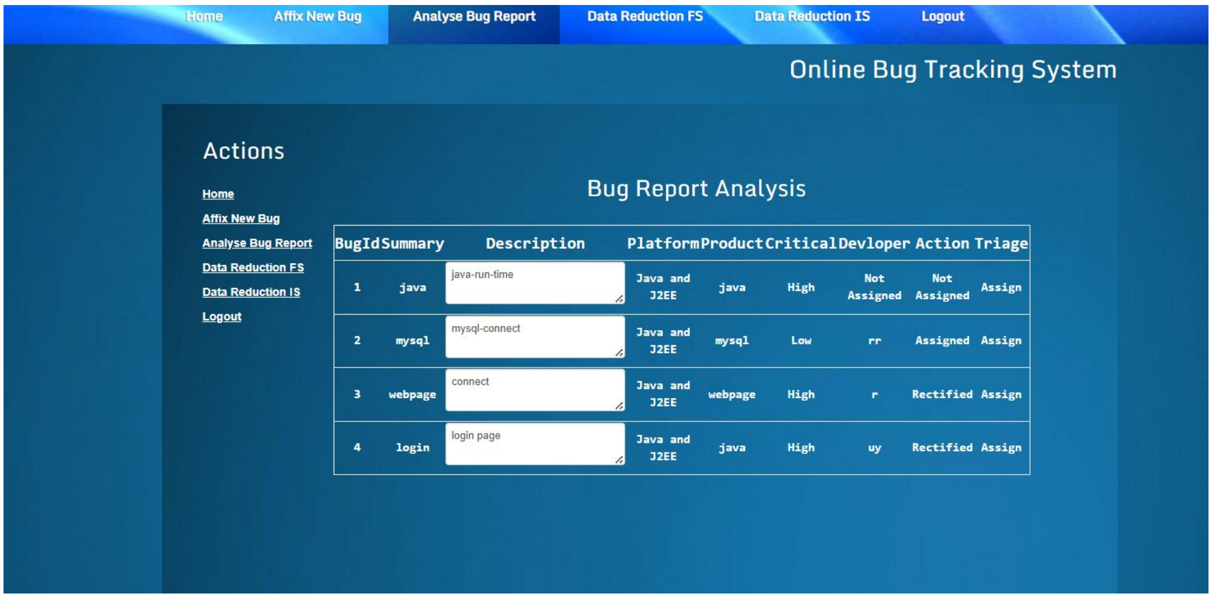
6.1.5.Manager Login Page.



6.1.6.Team Lead Home Page



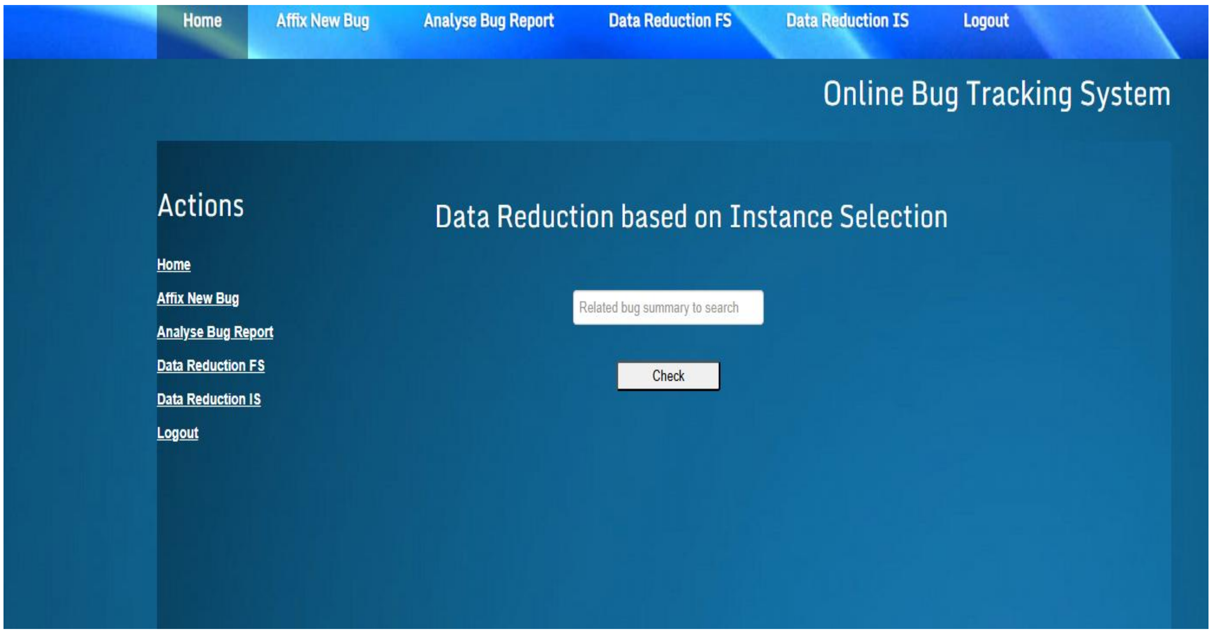
6.1.7.Fixing New bug report Page.



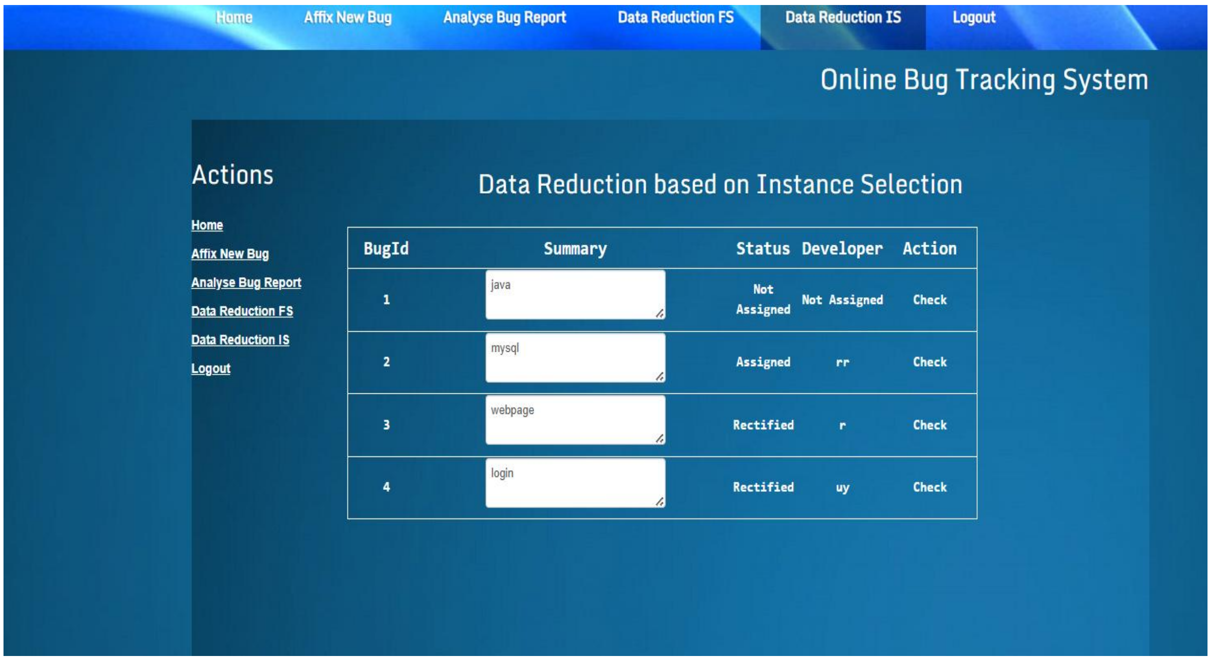
Bug Report Analysis

BugId	Summary	Description	Platform	Product	Critical	Devloper	Action	Triage
1	java	java-run-time	Java and J2EE	java	High	Not Assigned	Not Assigned	Assign
2	mysql	mysql-connect	Java and J2EE	mysql	Low	rr	Assigned	Assign
3	webpage	connect	Java and J2EE	webpage	High	r	Rectified	Assign
4	login	login page	Java and J2EE	java	High	uy	Rectified	Assign

6.1.8.Bug Report Analysis Page.



6.1.9. Bug summary search page.



BugId	Summary	Status	Developer	Action
1	java	Not Assigned	Not Assigned	Check
2	mysql	Assigned	rr	Check
3	webpage	Rectified	r	Check
4	login	Rectified	uy	Check

6.1.10.Data Reduction based on Instance selection page

CHAPTER 8

CODING

7.1 Bug Report:

```
package action;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.Statement;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
@author java1
public class bugreport extends HttpServlet {
Processes requests for both HTTP <code>GET</code> and <code>POST</code>
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            String sum = request.getParameter("sum");
            System.out.println("nmae:" + sum);
            String des = request.getParameter("des");
            String product = request.getParameter("product");
            System.out.println("date" + product);
            String platform = request.getParameter("platform");
            String imp = request.getParameter("imp");
            Connection con = Dbcon.getCon();
            Statement st = con.createStatement();
            String insertQuery = "insert into bug(summary, description, product, platform, importance)
values('" + sum + "','" + des + "','" + product + "','" + platform + "','" + imp + "')";
            int i = st.executeUpdate(insertQuery);
            if (i != 0) {
                response.sendRedirect("thome.jsp?msg=Registration_success");
            } else {
                response.sendRedirect("affix.jsp?msg=Register Error");
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to
edit the code."
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }
    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }
     * Returns a short description of the servlet.
     * @return a String containing servlet description
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>
}
```

7.2 Update dev:

```
package action;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse
import java.sql.Statement;
import java.sql.ResultSet
 * @author java1
public class updatedev extends HttpServlet {
 Processes requests for both HTTP <code>GET</code> and <code>POST</code>
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            String id = request.getParameter("id");
            String dev = request.getParameter("dev");
            String sum = request.getParameter("sum");
            String sql = "update bug set dev='"+dev+"' , status ='Assigned' where id = '"+id+"'";
            String sql1 = "insert into hist (id, dev, status, summary) values
('"+id+"','"+dev+"','Assigned','"+sum+"')";
            Connection con = Dbcon.getCon();
            Statement st = con.createStatement();
            int i = st.executeUpdate(sql);
            Statement st1 = con.createStatement();
            int i1 = st1.executeUpdate(sql1);
            if(i!=0){
                response.sendRedirect("buganalyse.jsp?Devloper_Assigned");
            }else{
                response.sendRedirect("buganalyse.jsp?Pls_Check");
            }
        }

        catch(Exception e){
            e.printStackTrace();
    }
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }   @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>
}
```

7.3 Dbcon:

```
package action;
import java.sql.Connection;
import java.sql.DriverManager;
* @author java1
public class Dbcon
    public static Connection getCon() throws ClassNotFoundException {
        Connection con = null;
        try {
            Class.forName("com.mysql.jdbc.Driver");
            con=DriverManager.getConnection("jdbc:mysql://localhost:3306/triage", "root", "root");
            System.out.println("Connection Established"+con);
                    } catch (Exception e) {
            e.printStackTrace();
        }
        return con;
    }
}
```

7.4 Dlogin:

```
package action;
import action.Dbcon;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.Statement;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
* @author java1
public class dlogin extends HttpServlet {
processRequest(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            /* TODO output your page here. You may use following sample code. *
            String name = request.getParameter("name");
            String pass = request.getParameter("pass");
            String dom = request.getParameter("domain")
            HttpSession ses = request.getSession();
            Connection con = Dbcon.getCon();
            Statement st = con.createStatement();
            String sql = "select * from user where name='" + name + "'";
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                if (pass.equals(rs.getString("pass")) && dom.equals(rs.getString("domain"))) {
                    ses.setAttribute("UID", name);
                    response.sendRedirect("dhome.jsp?Login_Success");
                } else {
                    response.sendRedirect("dlogin.jsp?Password_Mismatch");
                }
            }
            else{
                response.sendRedirect("dlogin.jsp?User_not_Exist");
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
}
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }

    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>
}
```

7.5 Update Status:

```
package action;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.Statement;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
public class updatestatus extends HttpServlet {
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try{
            HttpSession ses = request.getSession(true);
            String user = ses.getAttribute("UID").toString();
            /* TODO output your page here. You may use following sample code.*/
            String id = request.getParameter("id");
            String sts = request.getParameter("status");
            String sum = request.getParameter("sum");
            String sql = "update bug set status ='"+sts+"' where id = '"+id+"'";
            String sql1 = "insert into hist (id, dev, status, summary) values
('"+id+"','"+user+"','"+sts+"','"+sum+"')";
            Connection con = Dbcon.getCon();
            Statement st = con.createStatement();
            Statement st1 = con.createStatement();
            int i = st.executeUpdate(sql);
            int i1 = st1.executeUpdate(sql1);
            if(i!=0){
                response.sendRedirect("Bug_Tkt.jsp?Status_Updated");
            }else{
                response.sendRedirect("Bug_Tkt.jsp?Try_Again");
            }
        }
        catch(Exception e){
            e.printStackTrace();
        }
    }
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }
    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>
}
```

7.6 dpalog:

```
package action;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
public class tpalog extends HttpServlet {
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            /* TODO output your page here. You may use following sample code. */
            String name = request.getParameter("name");
            String pass = request.getParameter("pass");

            if(name.equals("team") && pass.equals("team")){
                response.sendRedirect("thome.jsp?Login_Success");
            }else{
                response.sendRedirect("tpalogin.jsp?Try_Again");
            }
        }catch(Exception e){
            e.printStackTrace();
        }
    }
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }
    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>
}
```

7.6 Register Action:

```
package action;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.Statement;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
public class RegisterAction extends HttpServlet {    protected void processRequest(HttpServletRequest
request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            String name = request.getParameter("name");
            System.out.println("nmae:" + name);
            String pass = request.getParameter("pass");
            String Day = request.getParameter("date");
            System.out.println("date" + Day);
            String mail = request.getParameter("email");
            String mobile = request.getParameter("phone");
            String place = request.getParameter("location");
            String domain = request.getParameter("domain");
            Connection con = Dbcon.getCon();
            Statement st = con.createStatement();
            String insertQuery = "insert into user(name, pass, jdt, email, domain, phone, loc) values('" +
name + "','" + pass + "','" + Day + "','" + mail + "','" + domain + "','" + mobile + "','" + place + "')"
            int i = st.executeUpdate(insertQuery);
            if (i != 0) {
                response.sendRedirect("index.jsp?msg=Registration_success");
            } else {
                response.sendRedirect("index.jsp?msg=Register Error");
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to
edit the code.">
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }
    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>
}
```

7.6 Alogin:

```
package action;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            /* TODO output your page here. You may use following sample code. */
            String name = request.getParameter("name");
            String pass = request.getParameter("pass");

            if(name.equals("manager") && pass.equals("manager")){
                response.sendRedirect("ahome.jsp?Login_Success");
            }else{
                response.sendRedirect("alogin.jsp?Try_Again");
            }
        }
        catch(Exception e){
            e.printStackTrace();
        }
    }
    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to
edit the code.">
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>
}
```

CHAPTER 9
SOFTWARE TESTING

That goal of testing needs to locate mistakes. The motive of testing Need to locate every potential A completed product's danger or defect. It provides a way to testing capacity or individual parts, sections, gatherings, and/or final products. Software inspection means the action in ensuring that it meets every requirement in users. demands and needs and doesn't malfunction in a system that is objectionable. Different test in fact several varieties. Each kind of test reacts to the test. certain testing need.

TESTING CLASSES

Unit testing

The practice of Making test scenarios for functional testing guarantees the core logic if it is functioning properly and if inputs from the system yield accurate results. All of the must receive certification.
internal flows and conclusion branches. It's an inquiry of the application's separate software components. Prior to integrating, it is completed following the conclusion to just one unit. Such invasive structure examination is dependent upon an understanding of its framework. Tests in units evaluate a particular application, setup of the system, or business process at the unit level. Examinations of units ensure all of the distinct path in The organisation processes has clearly defined both inputs and outcomes. and that it operates precisely according Those who mentioned specifications.

Evaluation for collaboration

Its purpose for integration tests is to evaluate integrated parts of software to see if they function as a single unit. Testing is based on events and focuses mostly on the fundamental results of fields or screens. Integration tests verify that even though unit testing successfully shown that while each part met its requirements on its own, the sum of the elements is very accurate and dependable. The purpose of test interfaces is to resolve any possible trouble that from the combining of different elements.

Functional test.

Real-world assessments give thorough evidence that they operate be real examined are presented in accordance with the technical and company demands, system paperwork, and user guides.

Focus of testing for functionality is on these areas:

Reputable Data	: Recognised legitimate supplied categories must be approved.
Invalid Input	:Rejecting identified types of incorrect input is necessary.
Functions	:The mentioned functions Have to be used.
Output	:It's essential too exercise the designated classes to obtain outputs.

Platforms/Procedures: You need to call upon the interacting systems or operations.

Operational testing planning and organisation are centred on requirements, important features, additionally distinct test scenarios. Moreover, testing needs to consider data fields, specified procedures, sequential processes, and systematic coverage related to identifying company operations flows. Further examinations is situated.

System Check

Framework assessment proves that all requirements are met by all the gadgets programme on its own a whole. It puts a setup undergo an examination to ensure dependable outcomes. The configuration-oriented implementation test is an illustration within an arrangement test. System assessment emphasizes pre-driven industry connections and interaction points and draws regarding protocols of routines.

White Box Testing

White box quizzing is a type of test for software where the tester is privy to the program's inner functioning, structure, and language—or at the very least, exactly the purpose of it is. It possesses one purpose. It is employed to test regions that have become inaccessible from a level of the black box.

Black Box Testing

A software testing "black box" means doing it lacking of idea Its internal design, functionality, and the module's specific languages under test. such as testing with black boxes, which comprise A lot of additional exam types have to be taken as well form an official source documenting, such a report outlining specifications or prerequisites. This type of testing treats the software being tested as a "black box." It’s impossible in "see" inside. Not considering how it went the programming, the test generates responds to both inputs and outcomes.

Tester Units:

Testing of units is typically carried out in tandem with other code with section inspection stage in the project, while It isn't very It is customary to finish unit evaluation and development in two separate stages lifecycle of a programme.

Test plan and methodology

The functional assessments will be meticulously prepared, and There may be field tests by hand.

Test objectives

- Every field entry needs to function correctly.
- You have to click This particular link that opens the respective web pages.
- There shouldn't be any delays in the entry display, messages, or answers.

Functionalities to be examined

- Make for every single website entries follow the proper structure;
- That no additional entries are permitted;
- And that Every single one of them links point visitors at which to appropriate site

Integration Testing

The steps for incrementally integrating a single base with A maximum of two connected software pieces to identify interface flaws that lead to failures is known as integrated software integration. The purpose purpose aims the resulting assessment is to validate the computer programmes or system components, or even higher up, company-level software products, function together flawlessly.

Test Results

Each examine case that was earlier said was successful. Nothing was flawed.

Acceptance Testing

Appreciation by Users Any project's evaluation phase is crucial, and it involves several customer comments. Additionally, it guarantees this infrastructure meets objective.

Test Results: Each examine cause that was before stated was successful. No flaws were found.

CHAPTER 10

TESTING AND IMPLEMENTATION.

8.1 Testing: Testing is a crucial phase in software development that ensures the system meets the specified requirements and functions correctly. The Online Bug Tracking System was subjected to various levels of testing to validate its performance, reliability, and functionality.

8.1.1 Types of Testing Performed:

a) Unit Testing:

Each module (e.g., Login, Bug Report Submission, Admin Panel) was tested individually to ensure they function as expected.
Tools Used: Manual test cases and assertions in PHP.

b) Integration Testing:

After individual modules were verified, they were integrated and tested together to identify any interface defects.
Example: Integration of the login module with the bug reporting module.

c) System Testing:

The entire system was tested as a whole to ensure that all components work together and meet the business requirements.

d) User Acceptance Testing (UAT):

The application was shared with a sample group of users to test its usability and gather feedback.
Based on the feedback, minor UI and functionality adjustments were made.

e) Security Testing:

Verified that unauthorized users cannot access restricted areas like the admin dashboard.
Ensured user data like passwords were encrypted and protected.

8.1.2 Test Causes Example:

Test Case ID	Description	Input	Expected Result	Actual Result	Status
TC_01	Login with valid credentials	Email & Password	Redirect to dashboard	Success	Pass
TC_02	Submit a new bug report	Bug form data	Bug successfully saved and notified	Success	Pass
TC_03	Access admin panel as user	User session	Access denied message	Access Denied	Pass

8.2 Implementation:

Implementation involves deploying the developed system into the live environment where users can begin using it.

8.2.1 System Requirements:

Frontend: HTML, CSS, JavaScript

Backend: PHP

Database: MySQL

Server: Apache (XAMPP/WAMP/LAMP)

8.2.2 Deployment Steps:

1. **Database Setup:**

   Created the MySQL database schema using SQL scripts.

   Imported initial tables such as `users`, `bugs`, and `projects`.

2. **Code Migration:**

   Uploaded project files to the server directory (e.g., `htdocs` in XAMPP).

   Configured database connection in the PHP files (`config.php`).

3. **Testing in Live Environment:**

   Retested key modules post-deployment to ensure proper functioning.

   Addressed environment-specific issues (e.g., path configurations).

4. **Go-Live:**

   System made available to end-users.

   Monitoring initiated for performance and bug reports.

CHAPTER 11
COST ESTIMATION OF PROJECT

Since this project is a mini project made for the Educational and Learning purpose it does not contain any specific code estimates.

Given, the project is locally hosted and there are no extra charges included for any type of external integration, there is no actual cost.

CHAPTER 12
TIME SCHEDULING OF PROJECT

Phase	**Duration**	**Description**
1. Project Planning	1 week	Define objectives, gather requirements, and plan.
2. Data Collection	1–2 weeks	Acquire and organize chest X-ray datasets.
3. Data Preprocessing	1–2 weeks	Clean, resize, augment, and split the data.
4. Model Development	2–3 weeks	Build or fine-tune the machine learning model.
5. Model Training	1–2 weeks	Train and tune the model with validation.
6. Model Evaluation	1 week	Test the model and analyze performance metrics.
7. Model Optimization	1–2 weeks	Improve model based on evaluation feedback.
8. Deployment	1–2 weeks	Develop user interface and deploy model online.
9. User Training & Docs	1 week	Train users and prepare documentation.
10. Maintenance Planning	Ongoing	Schedule regular updates and monitoring.

CHAPTER 13

CONCLUSION

System maintenance's bug triage phase is costly in terms of labor and time. In order to decrease the size of bug data sets and enhance the Caliber of the data, we mix picking features with instance choice in this study. We gather attributes from each bug data set and train a prediction model on previous data sets to decide which order to apply when choosing instances + feature selection for the latest bug data set. We conduct an empirical study on decreasing information for problem adjudication in bug files of two major open-source tasks Mozilla, Eclipse. Our research offers a method for utilizing data processing methods to create high-quality, reduced bug information necessary for application creation and management.

FUTURE ENHANCEMENTS

In future. The interactive system's prototype will soon be replaced by a full-scale version that can manage a wide limit of information, as in the real world. This Bug Chasing Method may be expanded to accomplish more sophisticated tasks. In addition to this online facility, chat room, SMS alerts, and a separate account for the testing team to compare problem severity, many more changes are planned. Creating a second test account is one option.

CHAPTER 14
REFERENCES

[1] J. Anvik, L. Hiew, and G. C. Murphy, “Who should fix this bug?” in Proc. 28th Int. Conf. Softw. Eng., May 2006, pp. 361–370.

[2] S. Artzi, A. Kie_zun, J. Dolby, F. Tip, D. Dig, A. Paradkar, and M. D. Ernst, “Finding bugs in web applications using dynamic test generation and explicit-state model checking,” IEEE Softw., vol. 36, no. 4, pp. 474–494, Jul./Aug. 2010.

[3] J. Anvik and G. C. Murphy, “Reducing the effort of bug report triage: Recommenders for development-oriented decisions,” ACM Trans. Soft. Eng. Methodol., vol. 20, no. 3, article 10, Aug. 2011.

[4] C. C. Aggarwal and P. Zhao, “Towards graphical models for text processing,” Knowl. Inform. Syst., vol. 36, no. 1, pp. 1–21, 2013.

[5] Bugzilla, (2014). [Online]. Avaialble: http://bugzilla.org/

[6] S. Shivaji, E. J. Whitehead, Jr., R. Akella, and S. Kim, “Reducing features to improve code change based bug prediction,” IEEE Trans. Soft. Eng., vol. 39, no. 4, pp. 552–569, Apr. 2013.

[7] P. S. Bishnu and V. Bhattacherjee, “Software fault prediction using quad tree-based k-means clustering algorithm,” IEEE Trans. Knowl. Data Eng., vol. 24, no. 6, pp. 1146–1150, Jun. 2012.

[8] H. Brighton and C. Mellish, “Advances in instance selection for instance-based learning algorithms,” Data Mining Knowl. Discovery, vol. 6, no. 2, pp. 153–172, Apr. 2002.

[9] Eclipse. (2014). [Online]. Available: http://eclipse.org/

[10] V. Bol_on-Canedo, N. S_anchez-Maro~no, and A. Alonso-Betanzos, “A review of feature selection methods on synthetic data,” Knowl. Inform. Syst., vol. 34, no. 3, pp. 483–519, 2013.

CHAPTER 15
BIBLIOGRAPHY

- Sommerville, Ian. *Software Engineering*, 10th Edition, Pearson Education, 2015.

- Pressman, Roger S. *Software Engineering: A Practitioner's Approach*, 8th Edition, McGraw-Hill Education, 2014.

- Jalote, Pankaj. *An Integrated Approach to Software Engineering*, 3rd Edition, Springer, 2005.

- Gamma, Erich, et al. *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, 1994.

- Grubb, Penny, and Armstrong A. Takang. *Software Maintenance: Concepts and Practice*, 2nd Edition, World Scientific, 2003.

- IEEE. *IEEE Std 830-1998 - IEEE Recommended Practice for Software Requirements Specifications*.

- W3Schools. "PHP Tutorial." https://www.w3schools.com/php/

- MySQL Documentation. "MySQL 8.0 Reference Manual." https://dev.mysql.com/doc/

- Mozilla Developer Network (MDN). "HTML and CSS Documentation." https://developer.mozilla.org/

- GitHub. "Open Source Bug Tracking Systems." https://github.com

- Google Developers. "Bug Tracking and Issue Management Tools." https://developers.google.com/

- JIRA Documentation. Atlassian. https://www.atlassian.com/software/jira