

# ISSUE TRACKING TOOL

A PROJECT REPORT

*Submitted by*

**NITHYA.L. (71201104028)**

**NITHYA.R. (71201104029)**

*in partial fulfillment for the award of the degree  
of*

**BACHELOR OF ENGINEERING**

*in*

**COMPUTER SCIENCE AND ENGINEERING**

**KUMARAGURU COLLEGE OF TECHNOLOGY  
COIMBATORE – 641006.**

**ANNA UNIVERSITY : CHENNAI 600 025**

**APRIL 2005**

**ANNA UNIVERSITY : CHENNAI 600 025**

**BONAFIDE CERTIFICATE**

Certified that this project report "ISSUE TRACKING TOOL" is the bonafide work of NITHYA.L. (71201104028) and NITHYA.R. (71201104029) who carried out the project work under my supervision.

  
SIGNATURE

**Dr . S . Thangasamy**

**HEAD OF THE DEPARTMENT**

  
SIGNATURE

**Ms . P. Devaki**

**SUPERVISOR**

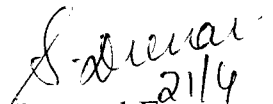
**ASSISTANT PROFESSOR**


**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**KUMARAGURU COLLEGE OF TECHNOLOGY**

**COIMBATORE – 641006.**

Submitted for the Viva-voce Examination held on 21-4-2005

  
21/4  
Internal Examiner

  
External Examiner

# ACKNOWLEDGEMENT

## ACKNOWLEDGEMENT

An endeavor over a long period can be successful only with the advice and support of many well wishers. We take this opportunity to express our gratitude and appreciation to all of them.

We express our immense gratitude to **Dr. K. Arumugam** , Correspondent and **Dr. K. K. Padmanabhan** , Principal ,Kumaraguru College of Technology , Coimbatore for having given us the golden opportunity to study in this prestigious institution .

We are extremely grateful to **Dr .S .Thangasamy** , Head of the Department of Computer Science & Engineering , Kumaraguru College of Technology ,for his constant encouragement and for the facilities made available during the course.

We wish to convey our heartfelt thanks and respect to our Internal faculty guide,Assistant Professor, **Ms . P. Devaki B.E.,M.S.**, Department of Computer Science and Engineering ,Kumaraguru College of Technology , for her meticulous guidance and encouragement to pursue new goals, ideas and being supportive throughout the tenure of our project.

We take pleasant privilege in expressing our heartfelt thanks to all the staff members and Lab technicians of the Computer Science Department , for all the help and support extended by them.

**ABSTRACT**

## **ABSTRACT**

The objective of this project is to design and implement an Issue Tracking Tool (ITT) which is used to track the bugs, feature requests and issues found in software applications during implementation and maintenance.

The main purpose of this tool is to simplify the status monitoring process in each and every project. ITT can also be used as a helpdesk customer support, trouble ticketing, or email management system to collect and manage customer feedbacks, incidents, requests, and issues. ITT provides a cost-effective enterprise-grade solution to increase team work efficiency.

The system is developed using JAVA technology with MS – Access as the backend and Tomcat web server. The project is the result of our exploitation of the multitude facets of JSP and Java Mail Manager.

# CONTENTS

## TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ACKNOWLEDGEMENT	iii
	ABSTRACT	iv
	LIST OF TABLES	v
	LIST OF FIGURES	vi
	LIST OF SYMBOLS	vii
1.	INTRODUCTION	
	1.1 PROJECT OVERVIEW	1
	1.2 EXISTING SYSTEM	3
	1.3 LIMITATIONS OF THE EXISTING SYSTEM	3
2.	PROJECT DESCRIPTION AND METHODOLOGIES	
	2.1 PROPOSED SYSTEM'S ADVANTAGES	4
	2.2 PROPOSED LINE OF ATTACK	6
	2.3 PROPOSED METHODOLOGY	8
	2.4 SYSTEM ANALYSIS	9
	2.4.1 SYSTEM DESIGN	9
	2.4.2 SYSTEM FUNCTIONALITY	11
	2.5 DATA FLOW DIAGRAM	16
	2.5.1 CONTEXT DIAGRAM	16



2.5.2 DATA FLOW DIAGRAM	17
2.6 PROGRAMMING ENVIRONMENT	18
2.6.1 HARDWARE REQUIREMENTS	18
2.6.2 SOFTWARE REQUIREMENTS	18
2.6.3 SOFTWARE FEATURES	19
2.6.3.1 FEATURES OF JSP	19
2.6.3.2 FEATURES OF WEB SERVER	21
2.6.3.3 FEATURES OF HTML	22
2.6.3.4 FEATURES OF MS- ACCESS	22
2.7 DETAILED DESIGN	23
2.7.1 DATABASE DESIGN	23
2.7.2 INPUT DESIGN	27
2.7.3 OUTPUT DESIGN	27
3. CONCLUSION AND FUTURE ENHANCEMENTS	
3.1 CONCLUSION	28
3.2 FUTURE ENHANCEMENTS	29
APPENDIX	30
A.1 SAMPLE CODE	30
A.2 RESULTS	53
REFERENCES	63

## LIST OF TABLES

TABLE NO.	NAME OF THE TABLE	PAGE NO.
1.1	ADMIN TABLE	24
1.2	USER TABLE	24
1.3	ISSUE TABLE	25
1.4	FILTER TABLE	26

## LIST OF FIGURES

FIGURE NO.	NAME OF THE FIGURE	PAGE NO.
2.1	OVERVIEW OF THE SYSTEM	2
2.2	INCREMENTAL MODEL	7
2.3	STATUS : OPEN	12
2.4	STATUS : INPROGRESS	12
2.5	STATUS : RESOLVED	13
2.6	STATUS : REOPENED	14
2.7	STATUS : VERIFY	14
2.8	STATUS : CLOSED	15
2.9	CONTEXT DIAGRAM	16
2.10	DATA FLOW DIAGRAM	17
2.11	SYSTEM ARCHITECTURE	19
2.12	ARCHITECTURE OF JSP	20
2.13	PAGE PROCESSING IN JSP	20

## LIST OF SYMBOLS

<b>ITT</b>	ISSUE TRACKING TOOL
<b>ISS_ID</b>	The unique ID assigned to the issue being created.
<b>ISS_TYPE</b>	Type of issue (Bug, new feature, improvement request)
<b>PRO</b>	The title of the project to which the issue is raised.
<b>PRIOR</b>	The priority assigned ( Blocker , critical, major, minor, trivial).
<b>COMP</b>	The components included in the project(events/listener).
<b>VERS</b>	The version of the project selected.
<b>ASS_TO</b>	The user id of the bug taker to which the issue is raised.
<b>DESC</b>	Brief description about the issue created.
<b>ENV</b>	The environment in which the project is worked on .
<b>FIL_ID</b>	The unique ID assigned to the combination of search saved as a filter.

# INTRODUCTION

# CHAPTER 1

## INTRODUCTION

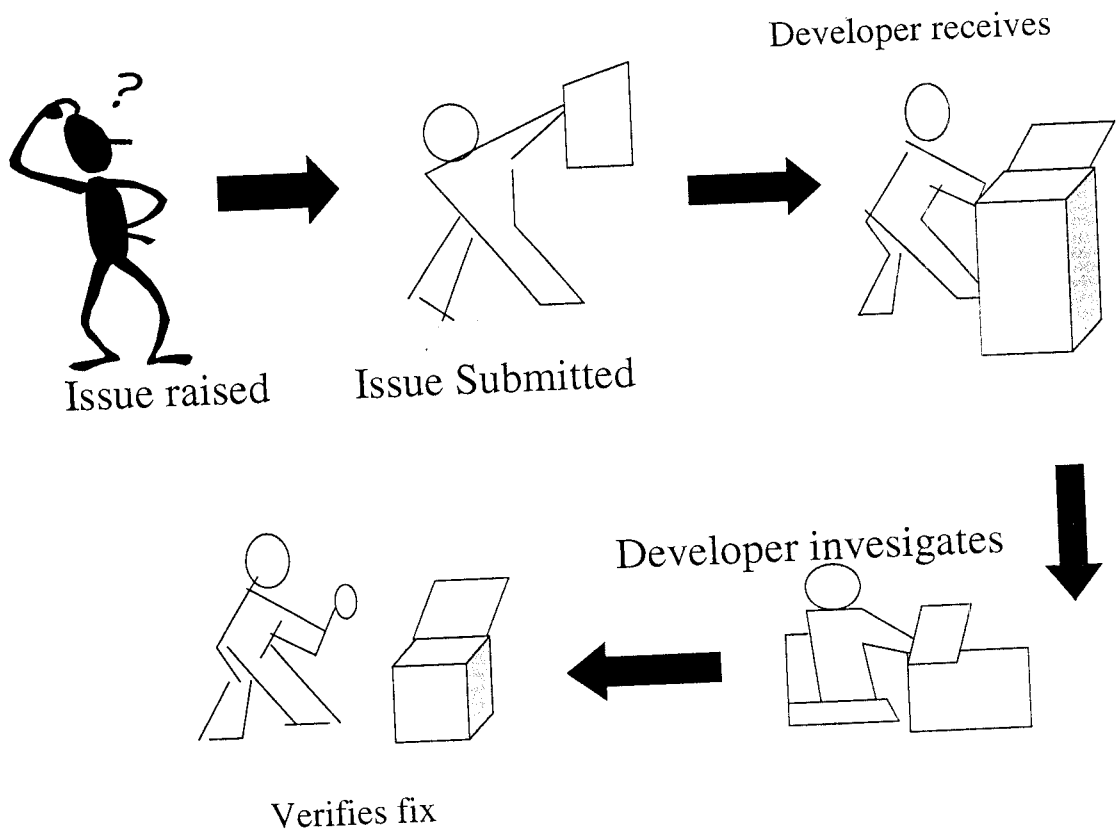
### 1.1 PROJECT OVERVIEW

Issue Tracking Tool (ITT) is a web based application used in a distributed team environment. It is used to manage product development by focusing effort on the tasks required to efficiently bring the project to completion. Developers, testers and managers use it to record and track the progress of defects (bugs), problems and new features. Each bug report progresses according to a defined workflow that can be fully customized by the organization.

When a defect is found, the developer records the bug and raises it as an issue and posts the same to a specific person or the administrator . After it is fixed, the developer will then check if the bug has really been fixed, and close it. Project managers can then see which bugs have been fixed, which are outstanding and how long it is taking to fix defects and understand the state of the development process.

The issues are detected and tracked according to the status and priority fixed to it. The issues can be viewed , browsed , filtered , searched according to certain criteria. E-mail notifications are sent accordingly to the bug taker and bug receiver.

The working of the system is as follows :



**Fig 2.1 Overview of the System**

## **1.2 EXISTING SYSTEM**

Tracking bugs is an important part of developing any software product. The key to software quality is being able to manage problems that occur early on in the development cycle. This can only be done with a proper software design, a dedicated software testing facility with exhaustive test plans that describe how to test every part of the application, and a bug tracking product.

The existing system tracks the bugs /errors developed in the project by a central authority , may be an administrator. He records all the bugs raised in the project and assigns it to a particular person in the testing team. All the bug tracking is controlled solely by the administrator.

## **1.3 LIMITATIONS OF THE EXISTING SYSTEM**

- It is used for bug tracking alone.
- Time taken for bug detecting is high.
- Redundancy of multiple bug reports are generated.
- Automatic email triggering is not supported.
- Working with multiple projects is not supported.
- Doesn't support concurrent users.



# PROJECT DESCRIPTION AND METHODOLOGIES

## CHAPTER 2

### PROJECT DESCRIPTION AND METHODOLOGIES

#### 2.1 PROPOSED SYSTEM'S ADVANTAGES

- **Controlled management of defects from discovery to resolution .**

Frequently, companies find that projects get out of hand - bugs are being raised from different sources to different team members and nobody is really sure how many bugs there are, which ones have been fixed and whether the nightmare will ever end. So this tool enables a controlled management of issues.

- **Reassigner**

Bugs can be passed on to other users, and will automatically be reassigned if they are marked as potentially fixed so the original poster can check the fix.

- **Browser**

See at a glance the whole project's outstanding bugs.

- **Allows Multiple Projects**

Each user can be a member of one or more project. The tool can handle any number of projects simultaneously.

➤ **Project Milestones**

Time is set for each issue and according to expiry time it is allotted to another person.

➤ **Bug History**

Keeps a complete history of who created the bug report and what has happened to it since.

➤ **Project History**

Keeps a complete history of changes made throughout the project

➤ **Redundancy is reduced.**

Issues are recorded once they are created and when the same issue is raised in the future, duplicate issues are discarded and proposed solutions are given at that instant.

➤ **Highly configurable email notification schemes**

It allows creating and updating records through mail messages and accordingly the bug receiver either closes the issue or reopens the issue to another member and notifies it through e-mail.

➤ **Analysis and categorization of bugs found .**

Issues are given status and priority to fix their degree of complexity. Priorities such as blocker, critical, major, minor, trivial are used to categorize the bugs/issues.

➤ **Better customer service**

Because there is an instantly available database collecting the detailed history of bugs developed in any project, the information to serve the customers better is always available .

## **2.2 PROPOSED LINE OF ATTACK**

A process model for developing any project is chosen based on its nature and application, the methods and tools to be used, and the controls and deliverables that are required. The proposed line of attack chosen for the “Issue Tracking Tool” is **Management of defects from discovery to resolution** and this objective is achieved using one of the software engineering models called “INCREMENTAL MODEL”.

The Incremental model combines elements of the linear sequential model (or) Waterfall model with the iterative philosophy of prototyping.

The Incremental model delivers software in small but usable pieces, called “increments”. In general, each increment builds on those that have already been delivered. Early increments are stripped down versions of the final product, but they do provide capability that serves the user and also provide a platform for evaluation by the user.

The architecture of the Incremental model is shown below:

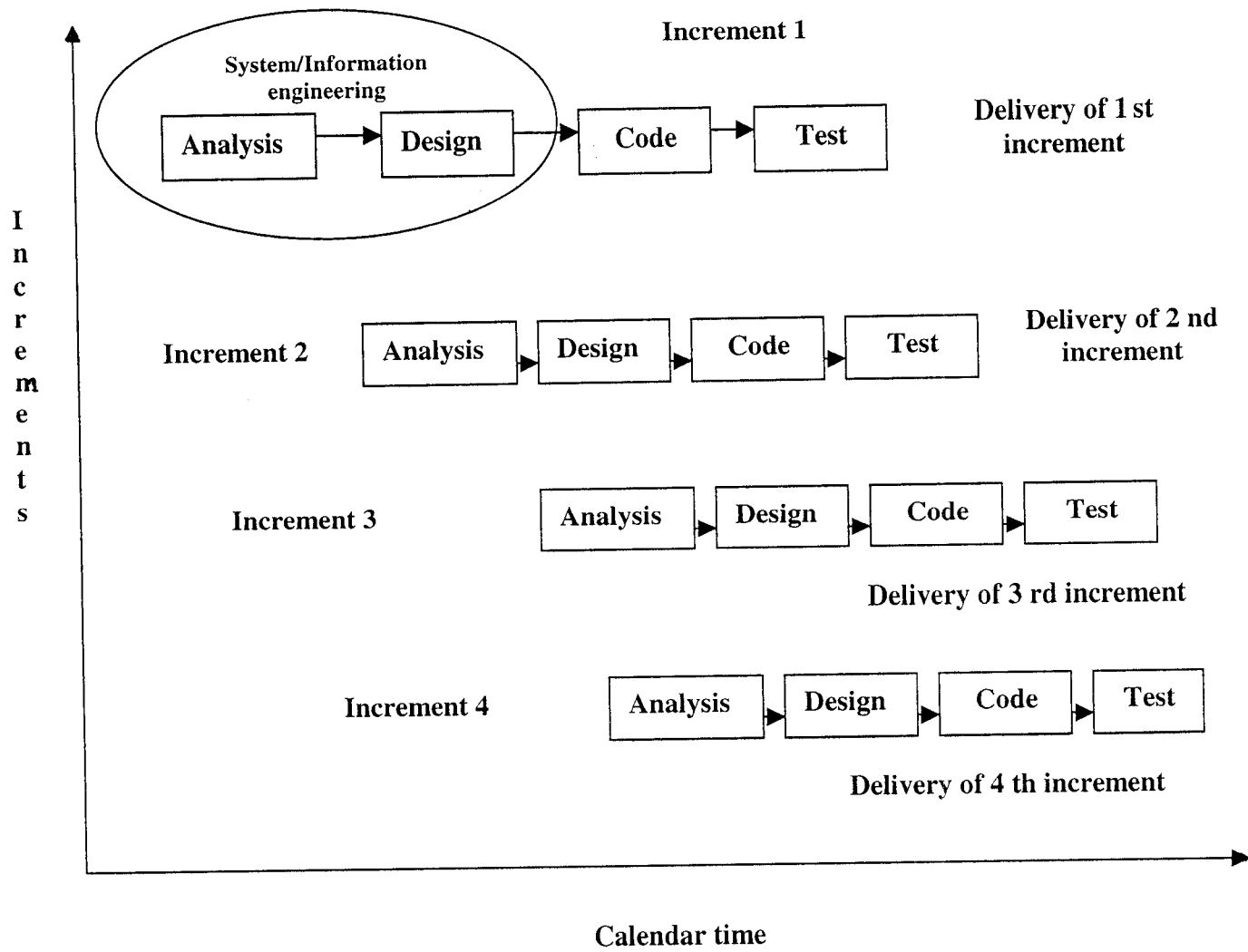


Fig. 2.2 Incremental Model

## 2.3 PROPOSED METHODOLOGY

When an Incremental model is used , the first increment is called “CORE PRODUCT”. Here, the basic requirements are addressed , but many supplementary features remain undelivered. After evaluation, a plan is developed for the next increment. The plan addresses the modification of the core product to better meet the needs of the customer and the delivery of additional features and functionality. This process is repeated following the delivery of each increment , until the complete product is produced.

The core product produced for this tool, consists of a database with two tables namely issue table and admin table. The basic forms were designed creating a convenient GUI.The JDBC connectivity was also established as the result of first increment.

The second increment incorporates the addition of two more tables which were used for the future enhancement of the project.The filtering module, searching module were added here which was accomplished by the creation of Filter table and User table.This increment was tested and found successful by the updation of the database.

The third increment was the inclusion of the highly configurable email scheme which notifies the bug taker and bug receiver about the current status of the project.

The final increment produces the intuitive and user friendly interface and tests the overall working of the tool.

## **2.4 SYSTEM ANALYSIS**

Analysis involves a detailed study of the current system leading to specifications of a new system. In this system, the tool maintains a database of all the issues so far produced, the details about the bug reporter who raised the issue and the bug receiver who is allotted to solve it and the optimal solutions proposed for it.

### **2.4.1 SYSTEM DESIGN**

The project involves six modules namely,

- Create Module
- View Module
- Browse Module
- Searching & Sorting Module
- Filtering Module
- Mailing Module

#### **CREATE MODULE:**

This module records the details of the issue created for both existing as well as new projects. This module involves

monitoring of the duplicate issues and allocates an unique Issue ID for the issue created .

#### **VIEW MODULE:**

Given an Issue ID, this module displays the details of the issue along with their current status .

#### **BROWSE MODULE:**

Given a project name, this module displays all the details of the issues raised in this project.

#### **SEARCHING & SORTING MODULE:**

This module is responsible for automatically updating the status of the issues under process. The issues can be searched according to their status specified.

#### **FILTERING MODULE:**

A specific set of criteria can be entered and the matching record(s) can be displayed. It can be performed in basic as well as advanced manner. For each filter, an unique Filter ID is assigned and thus redundancy is avoided.

#### **MAILING MODULE:**

This module deals with the mailing activities between the persons of each module in the project.



## 2.4.2 SYSTEM FUNCTIONALITY

The system progresses through four phases after the issue is submitted and the each issue is assigned a status and priority level which determines its degree of complexity. The phases are : Issue submission phase, Investigation phase, Resolution phase, Verification phase.

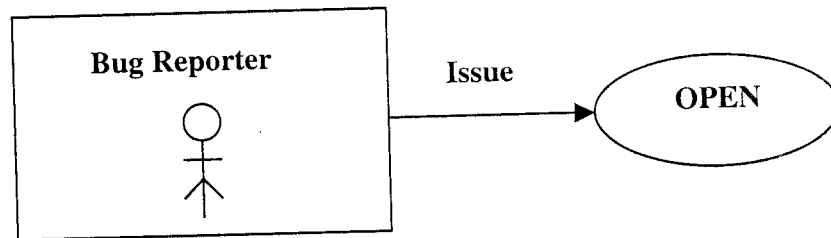
Each issue has a status, which indicates the stage of the issue. The different statuses through which the issue progresses are : Open, In progress , Resolved, Reopened, Closed . After each stage, the resolutions are identified.

An issue also has a priority level which indicates its importance and based on it, the tasks are assigned to the bug takers. The categorization of the issues depends on the priority levels namely blocker, critical, major, minor and trivial.

### **ISSUE SUBMISSION PHASE:**

The Issue Submission phase comprises of recording the issues and initializing the status to "open". The issues can be raised for both existing projects and new projects. The duplication and redundancy are avoided here.

## Status : Open



**Fig 2.3 Status : Open**

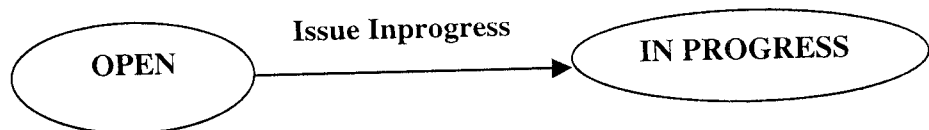
### Brief Description

The issue once submitted enters the “open” state and is given a unique issue id. The description and a brief summary of the project is also recorded in the database.

### INVESTIGATION PHASE:

During the Investigation phase , the issue enters the status “open” and after it is assigned to a bug taker its status is changed to “in progress”.

## Status : In Progress



**Fig 2.4 Status : In Progress**

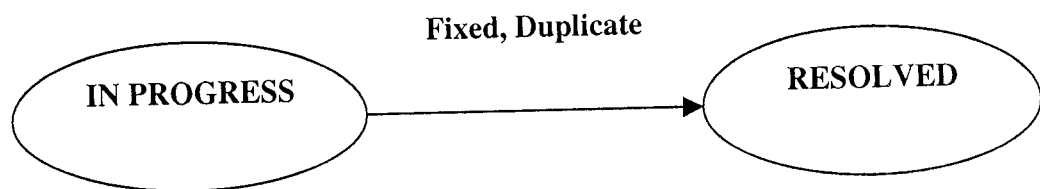
### Brief Description

The issue which is being actively worked on at the moment by the assignee enters the “in progress” state. When the email is sent successfully to the bug taker the status changes to “in progress” state.

### **RESOLUTION PHASE:**

During the Resolution phase , the issues are either resolved or found duplicate and accordingly the solutions are proposed. If the solution is found correct the status changes to “resolved” otherwise the issue is marked duplicate and the status is changed to “reopen”.

### **Status : Resolved**

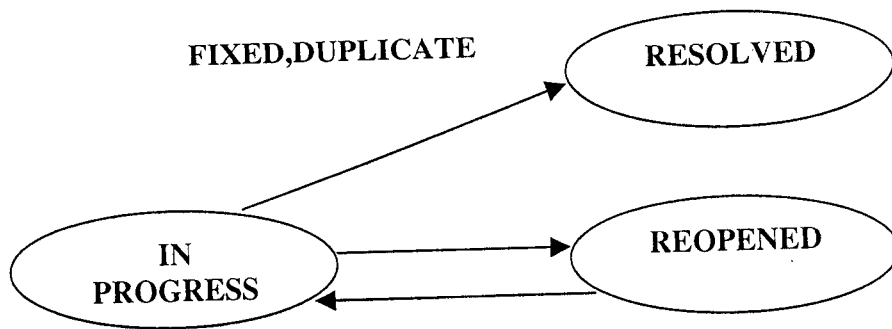


**Fig 2.5 Status : Resolved**

### Brief Description

A resolution has been taken, and it is awaiting verification by Bug reporter.

**Status : Reopened**



**Fig 2.6 Status : Reopened**

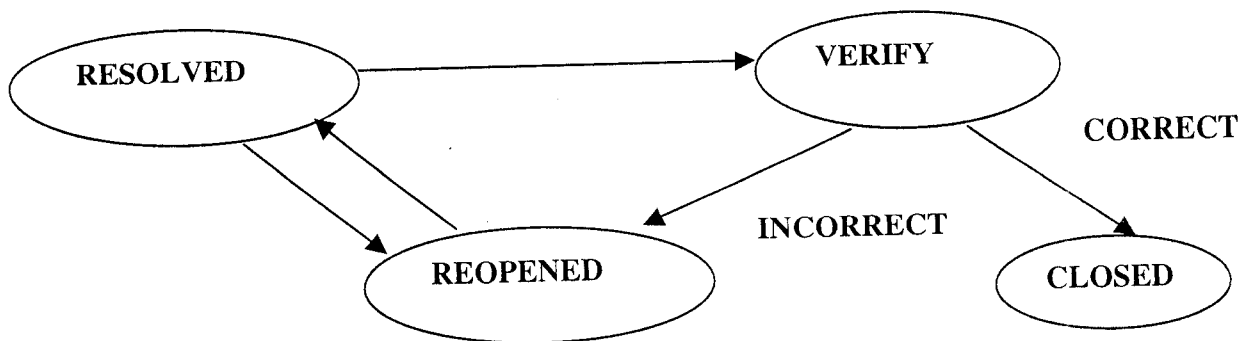
Brief Description

This issue was once resolved, but the resolution was deemed incorrect. From here issues are either marked In Progress, Resolved .

**VERIFICATION PHASE:**

During the verification phase, the proposed solution is verified to be correct or incorrect by the bug raiser.

**Status : Verify**



**Fig 2.7 Status : Verify**

BriefDescription

If solution is optimal and best fit then the status changes to the closed state. If it is incorrect then the status changes to Reopened state and the cycle repeats.

Status : Closed

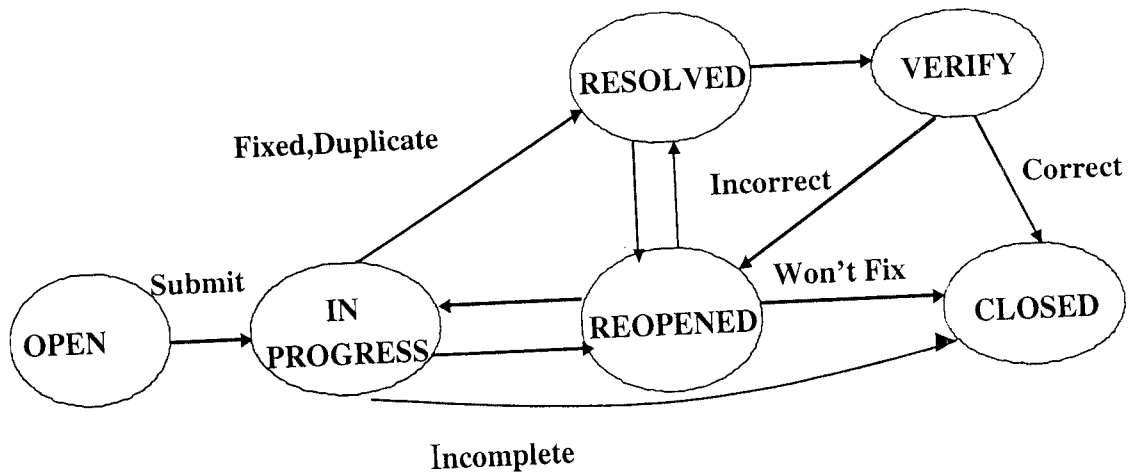


Fig 2.8 Status : Closed

Brief Description

The issue is considered dead, the resolution is correct. Any zombie issues which choose to walk the earth again must do so by becoming **reopened**.

## 2.5 DATA FLOW DIAGRAM

Graphical representation of a system's data and how the processes transform the data is known as Data Flow Diagram(DFD). The DFD is also known as Data Flow Graph or Bubble Chart. The data flow oriented design is an architectural design method that allows a convenient transition from the analysis model to a design to a description of the program structure.

Data Flow Diagrams are the most commonly used way of documenting the process of current and required systems. A DFD shows the movement of the data through series transformation.

### 2.5.1 CONTEXT DIAGRAM

The Context Diagram is the Level 0 DFD.

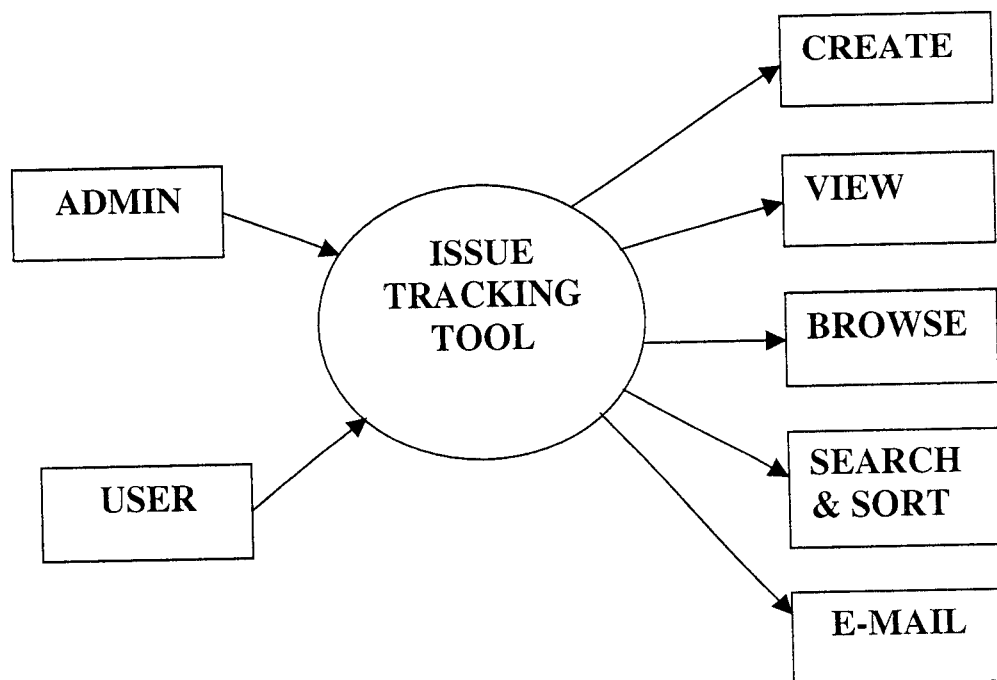


Fig 2.9 Context Diagram

## 2.5.2 DATA FLOW DIAGRAM

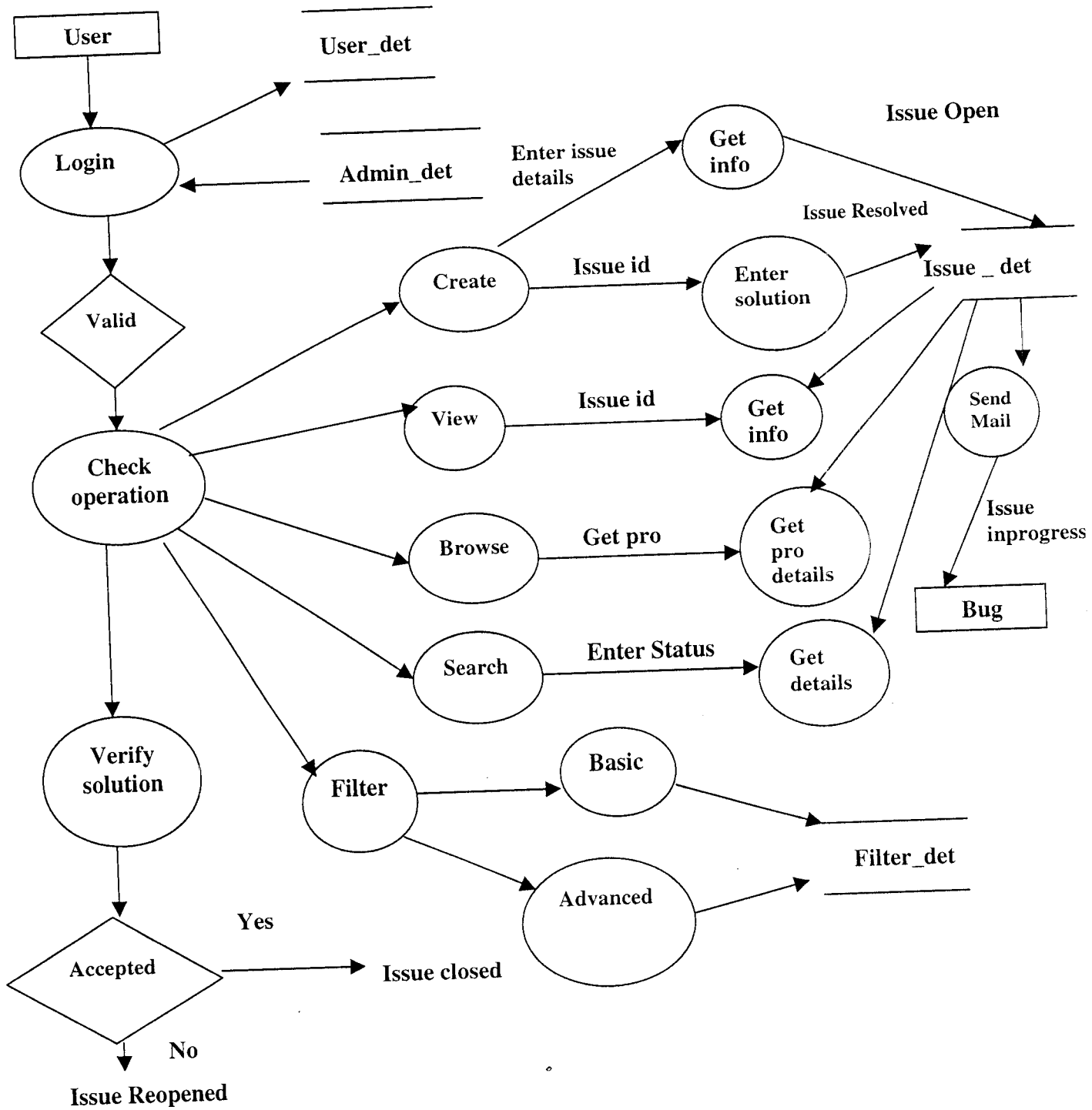


Fig 2.10 Data Flow Diagram

## **2.6 PROGRAMMING ENVIRONMENT**

The necessary specifications for the hardware, software, people and data resources and the information products that will satisfy the functional requirements of the proposed system can be determined. The design will serve as a blueprint for the system and helps detect problems before these errors or problems are built into the final system.

### **2.6.1 HARDWARE REQUIREMENTS**

Processor	:Pentium – III
Processor's Speed	: 800 Mhz
Memory	: 128 RAM
Hard Disk	: 40 GB
Monitor	: 15"
Keyboard	: 101 keys
Mouse	: Scrolling

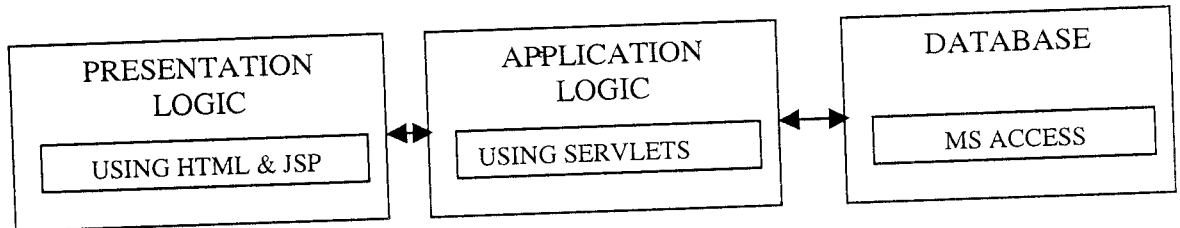
### **2.6.2 SOFTWARE REQUIREMENTS**

Operating System	: Windows 98 /Windows XP / Linux
Web Server	: Tomcat 4.1.0
Web Browser	: Internet Explorer
Client Tools	: HTML, JavaScript
Server Tools	: JSP, Servlets
DBMS	: MS - ACCESS



### 2.6.3 SOFTWARE FEATURES

The tool is built using JSP technology in JAVA with MS Access as the backend. The Tomcat webserver processes the JAVA code and displays the results in the Internet Explorer web browser.



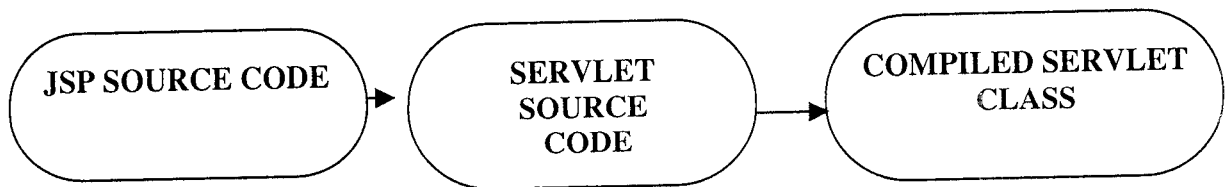
**Fig 2.11 System Architecture**

#### 2.6.3.1 FEATURES OF JSP

An interactive user interface is developed using Java Server Pages. JSP is used to provide a declarative, presentation-centric method of developing servlets. A JSP page is just like any other HTML file. It contains HTML formatting tags and can include client-side JAVA script, Java applets and Java code.

An important feature of JSP is the ability to connect to a database. JSP pages can be used to make information stored in a database available to the users by accessing the pages. Java server pages can also allow users to manipulate the data in the database such as addition, deletion, updation, modification.

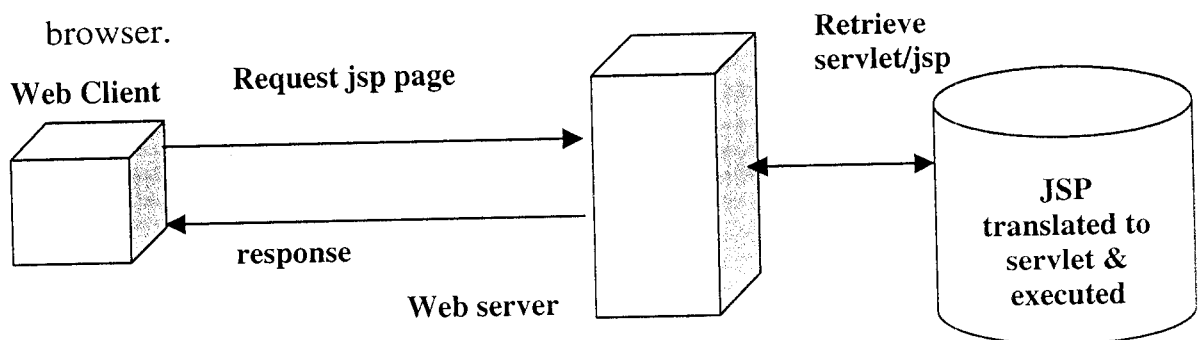
Java Server pages is based on Servlet technology , which allows web developers to use the JAVA code to create dynamic web pages.It uses a JSP engine that is the part of the web server , so the processing of JSP code takes place on the server.The web server converts the JSP code into a servlet code, compiles it internally and creates class files.



**Fig 2.12 Architecture of JSP**

### JSP PAGE PROCESSING:

When the user request JSP page, the JSP engine processes the page and then sends the result as HTML code to the user's web browser .This allows the JSP pages to be viewed by every web browser.



**Fig 2.13 Page Processing in JSP**

## ADVANTAGES OF JSP:

- **Vs. Active Server Pages(ASP):** ASP is a similar technology from Microsoft. The advantages of JSP are two-fold. First, the dynamic part is written in Java , not Visual Basic or other MS- specific language, so it is more powerful and easier to use. Second, it is portable to other operating systems and Microsoft web-servers.
- **Vs. Pure Servlets:** JSP does not give you anything that you couldn't in principle do with the servlet. It separates the look from the content.
- **Vs. Server-Side Includes (SSI):** SSI is a widely supported technology for including external defined pieces into a static web page. JSP is better because it lets to use servlets instead of a separate program to generate the dynamic part.
- **Vs. Java Script:** Java Script can generate HTML dynamically on the client. Java Scripts can't access server-side resources like databases, catalogs, pricing information and the like.
- **Vs. Static HTML:** JSP is so easy and convenient for including dynamic data.

### 2.6.3.2 FEATURES OF WEB SERVER:

The web server used here is “Tomcat” web server which interprets and processes Java Server Pages code. Tomcat is fully- functional web server that is used to create and test JSP pages. It is designed to be a fast and efficient implementation of the Servlet Specification.

### **2.6.3.3 FEATURES OF HTML**

HyperText Markup Language is used to create interactive webpages. HTML is used to create attractive and mind catching web sites. HTML contains powerful features:

- ◆ Advanced layout control.
- ◆ Banners.
- ◆ Client-side handling of hot spots in images.
- ◆ Customized lists.
- ◆ Dynamic documents with client-push/server- pull.
- ◆ Mathematical equations.
- ◆ Style sheets.
- ◆ Tables.
- ◆ Tables within forms.

### **2.6.3.4 FEATURES OF MS-ACCESS**

Microsoft Access is a relational database management system (DBMS). At the most basic level, a DBMS is a program that facilitates the storage and retrieval of structured information on a computer's hard drive.

The Access package contains the following elements:

- A relational database system that supports two industry standard query languages.
- A full-featured procedural programming language.

- A rapid application development environment.
- A sprinkling of object-oriented extensions.
- Various wizards and builders to make development easier.

An access database file contains several different types of database objects:

- Saved queries for organizing data.
- Forms for interacting with data on screen.
- Reports for printing results.
- Macros and Visual Basic programs for extending the functionality of database applications.

## **2.7 DETAILED DESIGN**

In the detailed design, computer oriented work begins in earnest. At this stage, the design of the system becomes more structured. It comprises of database design, input design and output design.

### **2.7.1 DATABASE DESIGN**

A database is a collection of information or data stored in a organized way. Data are raw facts and figures in isolation. The database is made up of one or more tables containing zero or more fields .The interconnection of record and field in table is called item. A relational database is a single database spread across multiple tables.

**TABLE 1.1**

NAME : ADMIN TABLE

DESCRIPTION: The Admin table is used for granting access rights for authorized users.

FIELD NAME	KEY	TYPE	SIZE
USER_ID	PRIMARY KEY	TEXT	15
PASS_WD		TEXT	10
EMAIL-ID		TEXT	30

**TABLE 1.2**

NAME : USER TABLE

DESCRIPTION : The User table consists of the id of the user currently logged on.

FIELD NAME	KEY	TYPE	SIZE
USER_ID	PRIMARY KEY	TEXT	15

**TABLE 1.3**

NAME : ISSUE TABLE

DESCRIPTION: The issue maintains the details of the issue submitted and the table is as follows:

FIELD NAME	KEY	TYPE	SIZE
ISSUE_ID	PRIMARY KEY	NUMBER	10
ISSUE_TYPE		TEXT	10
USER-ID		TEXT	15
EMAIL-ID		TEXT	30
PRO		TEXT	15
PRIOR		TEXT	15
STATUS		TEXT	15
COMPONENTS		TEXT	15
VERSIONS		TEXT	15
ASS_TO		TEXT	15
SUMMARY		TEXT	40
ENV		TEXT	15
DESC		TEXT	40
PERIOD		NUMBER	3
SOLUTION		TEXT	40

**TABLE 1.4**

NAME : FILTER TABLE

DESCRIPTION : Any combination of complex searches can be made and the issues are saved in the filter table and it avoids duplicate issues to be created in future.

FIELD NAME	KEY	TYPE	SIZE
FIL_ID	PRIMARY KEY	NUMBER	15
PRO		TEXT	15
ISS_TYPE		TEXT	15
PRIOR		TEXT	15
STATUS		TEXT	15
COMPONENTS		TEXT	15
VERSIONS		TEXT	15
ASS_TO		TEXT	15
SUMMARY		TEXT	40



### **2.7.2 INPUT DESIGN**

The tool is for all kind of users. People who have little or no knowledge about this tool can also use this software as it is very user-friendly. The forms designed are more intuitive and interactive for entering the input requirements.

The inputs are got from bug raisers, processed and corresponding output actions are delivered to them through e-mail notifications.

Some sample inputs obtained in this software are :  
Specifying the project name, priority , type of the issue, name and e-mail Id of the person whom the bug is assigned along with a short description of the issue raised.

The sample input forms are shown in Appendix A.2.

### **2.7.3 OUTPUT DESIGN**

The outputs obtained by processing the given input specifications gives a clear layout of the details of the issue along with a unique id.

The results have been made effective in such a way that even multiple projects' details can be viewed concurrently. Various types of search namely basic, advanced provide a valuable search criteria for the users.

The sample output forms are shown in Appendix A.2.

**CONCLUSION AND FUTURE  
ENHANCEMENT**

## **CHAPTER 3**

### **CONCLUSION AND FUTURE ENHANCEMENTS**

#### **3.1 CONCLUSION**

Thus our project entitled “ISSUE TRACKING TOOL” has been developed after detailed investigation of the Existing system. It plays an indispensable role in increasing the throughput of any software concern.

The package has been developed so as to reduce the strain of the user. There is no need of special training to use this package, since it is a user-friendly software.

This tool also stores all the records of the issues resolved for future references enabling smooth completion of the project under development.

We thank the ALMIGHTY, our parents, teachers without whom the task of completing this project would have been impossible.

## 3.2 FUTURE ENHANCEMENTS

The system can be further enhanced with the following features:

- Built-in Report Designer and Report Explorer can be created for organizing error reports. The reports can be stored in disk in a variety of formats, including Word, Adobe Acrobat, HTML, Excel, and more, making them easy to email.
- Attaching complex files with images can also be incorporated with the email triggering as one of the future enhancement.

APPENDIX

# APPENDIX

## A.1 SAMPLE CODE

Authen.jsp :

```
<html>
<head>
</head>
<%@ page import="java.sql.*" %>
<%@ page import="java.lang.*" %>
<body>
<!-- code that authenticates the user -->
<%!
int flag=0;
String s=" ";
%>
<%
String name=request.getParameter("user_id");
String pass=request.getParameter("pass_wd");
if (name.equals(s))
response.sendRedirect("login1.html");
if(pass.equals(s))
response.sendRedirect("login3.html");
String query="select * from admin";
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
Connection con=DriverManager.getConnection("jdbc:odbc:project");
Statement
stmt=con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_UPDATABLE);
Statement
stmt1=con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_UPDATABLE);
Statement
stmt2=con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_UPDATABLE);
stmt2.executeUpdate("delete from user");
```

```

stmt1.executeUpdate("insert into user values('"+name+"')");
ResultSet res=stmt.executeQuery(query);
while(res.next())
{
    if(name.equals(res.getString("user_id")))
    {
        if(pass.equals(res.getString("pass_wd")))
        {
            flag = 1;
            break;
        }
        else
            flag = 2;
    }
    else
        flag=3;
}
if (flag == 1)
response.sendRedirect("issue.html");
else if (flag == 2)
response.sendRedirect("login4.html");
else if (flag == 3)
response.sendRedirect("login2.html");
con.close();
%>
</body>
</html>

```

### Create.jsp :

```

<html>
<head>
    <title>CREATE FORM</title>
</head>

<body bgcolor="black" text="white">
<form action="store.jsp" method="post">

<%@ page import="java.sql.*" %>
<%!

```

```

int i=0,x,n,m;
String arr[]=new String[100];
String sp="";
boolean s;
%>

<%
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
Connection con=DriverManager.getConnection("jdbc:odbc:project");
Statement
stmt=con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_UPDATABLE);
ResultSet res= stmt.executeQuery("select pro from issue ");

while(res.next())
{
    arr[i]=res.getString("pro");
    i++;
}
for(m=0;m<i;m++){
    for(n=m+1;n<i;n++){
        if(arr[m].equals(arr[n])){
            arr[n]="";
        }
    }
}
%>
<center>
<h1>Enter the issue details</h1>
<table >
<tr><td>issue type:</td>
<td><select name="iss_type">
<option>
<option>bug
<option>improvement request
<option>new feature
</select></td></tr>
<tr><td>Project:</td>
<td><select name="pro">
<option>
<% for(x=0;x<i;x++)

```



```

{ %>
  <option><%
    s=arr[x].equals(sp);
    if(s) x=x+1;
    if (!s)%> <%=arr[x]%>
  <%
} %></td></tr>
<tr><td>Priority:</td>
<td><select name="prior">
<option>
<option>blocker
<option>critical
<option>major
<option>minor
<option>trivial
</select></td>
</tr>
<tr><td width="30%" >Summary :</td>
<td width="70%"> <textarea name=summary rows="2" cols="50" >
/textarea></td></tr>
<tr><td width="30%">Environment :</td>
<td width="70%"> <textarea name=env rows="2"
cols="50"></textarea></td></tr>
<tr><td width="30%">Description :</td>
<td width="70%"><textarea name=desc rows="6"
cols="50"></textarea></td></tr>
</table>
  <input type=submit name="submit" value="create">
  <input type=submit name="reset" value="reset">
</form>
</body>
</html>

```

### Store.jsp :

```

<html>
<head>
  <title>Store jsp form</title>
</head>

```

```

<%@ page import="java.sql.*" %>
<%@ page import="java.util.*"%>
<body>

<!-- code that stores the data -->

<%!
Random randomValue=new Random();
int iss_id=Math.abs(randomValue.nextInt())%100+1;
String status="open", d;
int period, n,y,flag;
String s1="BLOCKER";
String s2="CRITICAL";
String s3="MAJOR";
String s4="MINOR";
String s5="TRIVIAL";
%>
<%
String iss_type=request.getParameter("iss_type");
String pro=request.getParameter("pro");
String prior=request.getParameter("prior");
String comp=request.getParameter("comp");
String vers=request.getParameter("vers");
String ass_to=request.getParameter("ass_to");
String summary=request.getParameter("summary");
String env=request.getParameter("env");
String desc=request.getParameter("desc");
    iss_id++;
    n=iss_id++;
if(prior.equals(s1))
    period=5;
else if(prior.equals(s2))
    period= 10;
else if(prior.equals(s3))
    period=15;
else if(prior.equals(s4))
    period=20;
else
    period=25;

```

```

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
Connection con=DriverManager.getConnection("jdbc:odbc:project");
Statement
stmt3=con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_UPDATABLE);
ResultSet res2=stmt3.executeQuery("select * from issue");
while(res2.next())
{
    d=res2.getString("iss_id");
    if(iss_type.equals(res2.getString("iss_type")))
    if(pro.equals(res2.getString("pro")))
    if(prior.equals(res2.getString("prior")))
    if(comp.equals(res2.getString("comp")))
    if(vers.equals(res2.getString("vers")))
    if(ass_to.equals(res2.getString("ass_to")))
    if(summary.equals(res2.getString("summary")))
    if(env.equals(res2.getString("env")))
    if(desc.equals(res2.getString("desc")))
    {
        flag=1;
        break;
    }
}
if (flag==1){
y=Integer.parseInt(d);
Statement
stmt4=con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_UPDATABLE);
ResultSet res3=stmt4.executeQuery("select * from issue where iss_id
like '"+y+"' ");
while (res3.next())
{
    %>
    <b><font size=5 color=green>The specified issue already exists.The
details are...</font><br><br><br>
    <table border=1>
        <tr><td><font size=4>ISSUE ID</td>
        <td><font size=4><%=res3.getString("iss_id")%></td>
        </tr>
        <tr><td><font size=4>ISSUE TYPE</td>

```

```

        <td><font size=4><%=res3.getString("iss_type")%></td>
    </tr>
    <tr><td><font size=4>PROJECT TITLE</td>
    <td><font size=4><%=res3.getString("pro")%></td>
    </tr>
    <tr><td><font size=4>PRIORITY</td>
    <td><font size=4><%=res3.getString("prior")%></td>
    </tr>
    <tr><td><font size=4>STATUS</td>
    <td><font size=4><%=res3.getString("status")%></td>
    </tr>
    <tr><td><font size=4>SUMMARY</td>
    <td><font size=4><%=res3.getString("summary")%></td>
    </tr>
</table>
</center><br><br><br>
<%
res3.close(); }
}
else
{
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
Connection
con1=DriverManager.getConnection("jdbc:odbc:project");
Statement
stmt=con1.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_UPDATABLE);

Statement
stmt1=con1.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE
,ResultSet.CONCUR_UPDATABLE);
ResultSet res=stmt1.executeQuery("select user_id from user");
String str=res.getString("user_id");

Statement
stmt2=con1.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE
,ResultSet.CONCUR_UPDATABLE);
ResultSet res1=stmt2.executeQuery("select email_id from admin
where user_id=" +str+ " ");
String str1=res1.getString("email_id");

```

```

stmt.executeUpdate("insert into issue values (" + n + ", " + iss_type + "
, " + str + " , " + str1 + " , " + pro + " , " + prior + " , " + status + " ,
" + comp + " , " + vers + " , " + ass_to + " , " + summary + " ,
" + env + " , " + desc + " , " + period + " , null ) ");
iss_id=n;
con1.close();
%>
<b><font size=6>The details entered are ....</font><br><br>
<table border=1>
  <tr><td><b>ISSUE ID</b></td>
  <td><b><%=iss_id%></b></td>
  </tr>
  <tr><td>ISSUE TYPE</td>
  <td><%=iss_type%></td>
  </tr>
  <tr><td>PROJECT TITLE</td>
  <td><%=pro%></td>
  </tr>
  <tr><td>PRIORITY</td>
  <td><%=prior%></td>
  </tr>
  <tr><td>SUMMARY</td>
  <td><%=summary%></td>
  </tr>
</table>
<% } %>
</body>
</html>

```

### View.jsp :

```

<html>
<head>
  <title>View jsp form</title>
</head>
<% @ page import="java.sql.*" %>
<body>
<%! int i,n;
%>

```

```

<%
String iss=request.getParameter("iss_id");
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
Connection con=DriverManager.getConnection("jdbc:odbc:project");

Statement
stmt1=con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_UPDATABLE);

Statement
stmt2=con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_UPDATABLE);
ResultSet res=stmt1.executeQuery("select iss_id from issue");
ResultSet res1=stmt2.executeQuery("select iss_id from issue");

while(res1.next())
{
    i++;
}
while (res.next())
{
    boolean str=iss.equals(res.getString("iss_id"));
    i--;
    if(!str)
    {
        if(i>1)
            continue;
        else{
            out.println("Invalid Issue ID");
            break;
        }
    }
    else if(str)
    {
        String query="select * from issue where iss_id=?";
        PreparedStatement stmt=con.prepareStatement(query);
        stmt.setInt(1,Integer.parseInt(request.getParameter("iss_id")));
        ResultSet res2=stmt.executeQuery();
        while (res2.next()){
    %>
    <table border=1>

```

```

<tr><td><font size=4>ISSUE ID</td>
<td><font size=4><%=res2.getString("iss_id")%></td></tr>

<tr><td><font size=4>ISSUE TYPE</td>
<td><font size=4><%=res2.getString("iss_type")%></td></tr>

<tr><td><font size=4>PROJECT TITLE</td>
<td><font size=4><%=res2.getString("pro")%></td></tr>

<tr><td><font size=4>PRIORITY</td>
<td><font size=4><%=res2.getString("prior")%></td></tr>

<tr><td><font size=4>SUMMARY</td>
<td><font size=4><%=res2.getString("summary")%></td>
</tr>
</table>
<%
    }
    break;
    }}
con.close();
%>
</body>
</html>

```

### **Browse.jsp :**

```

<html>
<head>
<title>browse</title>
</head>
<%@ page import="java.sql.*" %>
<body>
<%
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
Connection con=DriverManager.getConnection("jdbc:odbc:project");
String query="select iss_id,iss_type,ass_to,prior,status,summary from
issue where pro=?";
PreparedStatement stmt=con.prepareStatement(query);
stmt.setString(1,request.getParameter("pro"));

```

```

ResultSet res=stmt.executeQuery( );
%>
Project Name : <%=request.getParameter("pro")%>
<table border=1>
<tr><td>Issue ID </td>
<td>Issue Type</td>
<td>Assigned To</td>
<td>Priority</td>
<td>Status</td>
<td>Summary</td></tr>
<%
while(res.next())
{
%>
<tr>
<td><%=res.getString("iss_id")%></td>
<td><%=res.getString("iss_type")%></td>
<td><%=res.getString("ass_to")%></td>
<td><%=res.getString("prior")%></td>
<td><%=res.getString("status")%></td>
<td><%=res.getString("summary")%></td>
</tr>
<%
}
res.close();
con.close();
%>
</table>
</body>
</html>

```

### Open.jsp :

```

<html>
<%@ page import="java.sql.*" %>
<body>
<%
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
Connection con=DriverManager.getConnection("jdbc:odbc:project");

```



Statement

```
stmt=con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_UPDATABLE);
```

```
ResultSet res= stmt.executeQuery("select * from issue where status='open' ");
```

```
%>
```

```
<center><font size=5 color=teal><b>STATUS : OPEN</b>
```

```
<table border=1>
```

```
<tr><center>
```

```
<td><center><b>&nbsp;Issue ID&nbsp;</td>
```

```
<td><center><b>&nbsp;Issue Type&nbsp;</td>
```

```
<td><center><b>&nbsp;Project Title&nbsp;</td>
```

```
<td><center><b>&nbsp;Raised By&nbsp;</td>
```

```
<td><center><b>&nbsp;Assigned To&nbsp;</td>
```

```
<td><center><b>&nbsp;Priority&nbsp;</td>
```

```
<td><center><b>&nbsp;Summary&nbsp;</td>
```

```
<center>
```

```
</tr>
```

```
<%
```

```
while(res.next())
```

```
{
```

```
%>
```

```
<tr>
```

```
<td><center>&nbsp;<%=res.getString("iss_id")%>&nbsp;</td>
```

```
<td><center>&nbsp;<%=res.getString("iss_type")%>&nbsp;</td>
```

```
<td><center>&nbsp;<%=res.getString("pro")%>&nbsp;</td>
```

```
<td><center>&nbsp;<%=res.getString("user_id")%>&nbsp;</td>
```

```
<td><center>&nbsp;<%=res.getString("ass_to")%>&nbsp;</td>
```

```
<td><center>&nbsp;<%=res.getString("prior")%>&nbsp;</td>
```

```
<td><center>&nbsp;<%=res.getString("summary")%>&nbsp;</td>
```

```
</tr>
```

```
<%
```

```
}
```

```
res.close();
```

```
con.close();
```

```
%>
```

```
</center>
```

```
</body>
```

```
</html>
```

### Basic.jsp :

```
<html>
<head>
<title>basic search</title>
</head>
<body background="bg.jpg" link=blue vlink=teal alink=green>
<%@ page import="java.sql.*" %>
<%!
String iss_type,status;
int i,flag;
int n=1;
%>
<%
String type=request.getParameter("iss_type");
String stat=request.getParameter("status");
String query="select * from issue where pro = ? ";
String str=request.getParameter("pro");
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
Connection con=DriverManager.getConnection("jdbc:odbc:project");

PreparedStatement stmt=con.prepareStatement(query);
stmt.setString(1,request.getParameter("pro"));
ResultSet res=stmt.executeQuery( );

PreparedStatement stmt1=con.prepareStatement(query);
stmt1.setString(1,request.getParameter("pro"));
ResultSet res1=stmt1.executeQuery( );

while(res1.next())
    i++;
i--;
%>
<br><br><font color=green>
The result of your search is as follows.....</font>
<table>

<tr><td><b> Project Name</td>
<td><b> : <font
color=fushcia><%=request.getParameter("pro")%></td></tr>
```

```

<tr><td><b>Issue Type</b></td>
<td><b> : <font
color=fushcia><%=request.getParameter("iss_type")%></td></tr>

```

```

<tr><td><b>Status </b></td>
<td><b> : <font
color=fushcia><%=request.getParameter("status")%></td></tr>
</table>

```

```

<%
while(res.next())
{
iss_type = res.getString("iss_type");
status=res.getString("status");
if (iss_type.equals(type))
{
    if(status.equals(stat))
    {
        flag=1;
        if(n==1){ %>
        <table border=1>
        <tr><td><b>Issue ID </b></td>
        <td><b>Priority</b></td>
        <td><b>Raised by </b></td>
        <td><b>Assigned to </b></td>
        <td><b>Summary</b></td>
        <td><b>Solution</b></td></tr>
        <% n++;}
        %>
        <tr><td><center><%=res.getString("iss_id")%></td>
        <td><center><%=res.getString("prior")%></td>
        <td><center><%=res.getString("user_id")%></td>
        <td><center><%=res.getString("ass_to")%></td>
        <td><center><%=res.getString("summary")%></td>
        <td><center><%=res.getString("solution")%></td></tr>
        <% } }
        else
        {
            i--;
            if(i>0)
                continue;

```

```

        else if(i==0)
            out.println("NO records found");
    }
}
res.close();
con.close();
%>
</table>
<%
if(flag==1){ %>
<a href="view.html">Click here to view the details of the
issue</a><br><br></center>
<% } %>
<a href="filter10.html">BACK<<</a><br><br>
</body>
</html>

```

### Adv.jsp :

```

<html>
<head>
<title>advanced search</title>
</head>
<% @ page import="java.sql.*" %>
<% @ page import="java.util.*" %>
<body background="bg.jpg" link=green vlink=black alink=blue>
<%!
String t,s,p,c,v,a,d;
Random randomValue=new Random();
int fil_id=Math.abs(randomValue.nextInt())%100+1;
int n,i,y,flag;
%>
<%
String pro=request.getParameter("pro");
String type=request.getParameter("iss_type");
String stat=request.getParameter("status");
String prior=request.getParameter("prior");
String comp=request.getParameter("comp");
String aff=request.getParameter("vers");
String ass=request.getParameter("ass_to");

```

```

fil_id++ ;
n=fil_id++;
String query="select * from issue where pro = ? ";
String str=request.getParameter("pro");
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
Connection con=DriverManager.getConnection("jdbc:odbc:project");
PreparedStatement stmt=con.prepareStatement(query);
stmt.setString(1,request.getParameter("pro"));
ResultSet res=stmt.executeQuery( );

PreparedStatement stmt2=con.prepareStatement(query);
stmt2.setString(1,request.getParameter("pro"));
ResultSet res1=stmt2.executeQuery( );

while(res1.next())
    i++;
i--;
%>
<%
while(res.next())
{
t=res.getString("iss_type");
p=res.getString("prior");
s=res.getString("status");
c=res.getString("comp");
v=res.getString("vers");
a=res.getString("ass_to");
if (type.equals(t))
    if(prior.equals(p))
        if(stat.equals(s))
            if(comp.equals(c))
                if(aff.equals(v))
                    if(ass.equals(a)){
                        i--;
%>
<b> Project Name : <%=request.getParameter("pro")%>
<table border=1>
<tr><td><b>Issue ID </td>
<td><b>Summary</td></tr>

```

```

<tr><td><%=res.getString("iss_id")%></td>
<td><%=res.getString("summary")%></td></tr>

```

```

</table>

```

```

 <input type=button
value=" Save it as a filter" onClick="save()" >

```

```

<%

```

```

break;

```

```

}

```

```

else{ i--;

```

```

    if(i>0)

```

```

        continue;

```

```

        else if(i==0)

```

```

out.println("NO records found");

```

```

}

```

```

}

```

```

res.close();

```

```

con.close();

```

```

%>

```

```

<script language="JavaScript">

```

```

function save()

```

```

{

```

```

<%

```

```

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

```

```

Connection

```

```

con1=DriverManager.getConnection("jdbc:odbc:project");

```

```

Statement

```

```

stmt3=con1.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE
,ResultSet.CONCUR_UPDATABLE);

```

```

ResultSet res2=stmt3.executeQuery("select * from filter");

```

```

while(res2.next())

```

```

{

```

```

if(pro.equals(res2.getString("pro")))

```

```

if(type.equals(res2.getString("iss_type")))

```

```

if(prior.equals(res2.getString("prior")))

```

```

if(stat.equals(res2.getString("status")))

```

```

if(comp.equals(res2.getString("comp")))

```

```

if(aff.equals(res2.getString("vers")))

```

```

if(ass.equals(res2.getString("ass_to")))
{
d=res2.getString("fil_id");
flag=1;
break;
}
}
y=Integer.parseInt(d);
if (flag==1)
{ %>
<!--
var popup1="Filter Duplicate.";
alert(popup1 );
//-->
<% }
else
{
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
Connection
con2=DriverManager.getConnection("jdbc:odbc:project");
Statement
stmt1=con2.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE
,ResultSet.CONCUR_UPDATABLE);
stmt1.executeUpdate("insert into filter values ('"+n+"', '"+pro+"', '"+
type +"' , '"+prior+"', '"+stat+"', '"+comp+"', '"+aff+"', '"+
ass+"'"));
fil_id=n;
%>
<!--
var popup="Filter Saved.\n\n";
alert(popup);
//-->
<% }
%>
}
</script>
</body>
</html>

```

## InputsForm.jsp :

```
<%@ page contentType="text/html;charset=WINDOWS-1252"%>

<html>
<head>
<title>InputsForm.jsp</title>
</head>
<body >

<script language="JavaScript1.1">
function submission1()
{
if(document.forms[0].p_from.value == "" ||
document.forms[0].p_to.value == "" ||
document.forms[0].p_smtpServer.value == "") {
alert("Host, From and To fields are mandatory.");
return;
}
document.forms[0].action = "SendMail.jsp";
document.forms[0].submit();
}
</script>

<table bgcolor=black width=100% cellpadding="1">
<tr>
<td><font color=red size=3>Send Mail</td></tr>
</table>

<table border=0 >
<tr><td><form method=post>

<table border=0 cellpadding=4>
<tr><td align=right width="124"><b>Mail Server Host</b></td>

<td align=left width="298">
<% String l_svr = (String)session.getAttribute("smtpServer"); %>
<input type="text" name="p_smtpServer"size=60 value="<%=
l_svr!=null ? l_svr : "" %>" >
</td></tr>
```



```

<tr><td align=right width="124"><b>From</b></td>

<td align=left width="298">
<input type=text size=60 maxlength=60 name="p_from"></td></tr>

<tr><td align=right width="124"><b>to</b></td>
<td align=left width="298">
<input type=text size=60 maxlength=200 name="p_to"></td></tr>

<tr><td align=right width="124"><b>cc</b></td>
<td align=left width="298">
<input type=text size=60 maxlength=200 name="p_cc"></td></tr>

<tr><td align=right width="124"><b>bcc</b></td>
<td align=left width="298">
<input type=text size=60 maxlength=200 name="p_bcc"></td></tr>

<tr><td align=right width="124"><b>subject</b></td>
<td align=left width="298">
<input type=text size=60 maxlength=100
name="p_subject"></td></tr>

</table>
<b>message</b><br>
<textarea name="p_message" rows=10 cols=66 size=2000
wrap=hard></textarea>
<br><br>
<input type=button value=" Send " onClick="submission1();">
<input type=reset value="Reset Form"><br>
</form>
</td></tr>
</table>
</body>
</html>

```

## SendMail.jsp

```
<%@ page contentType="text/html"%>
<%@ page import="javax.mail.*"%>
<%@ page import="javax.mail.internet.*"%>
<%@ page import="java.util.*"%>

<html>
<head>
<title>SendMail.jsp</title>
</head>
<body >

<%!
public String send(String p_from, String p_to, String p_cc, String
p_bcc,
String p_subject, String p_message, String p_smtpServer) {
String l_result = "<BR><BR><BR><BR><BR><BR><BR>";

String l_host = p_smtpServer;

Properties l_props = System.getProperties();

l_props.put("mail.smtp.host", l_host);

Session l_session = Session.getDefaultInstance(l_props, null);

l_session.setDebug(true); // Enable the debug mode

try {
MimeMessage l_msg = new MimeMessage(l_session); // Create a
New message

l_msg.setFrom(new InternetAddress(p_from)); // Set the From
address

l_msg.setRecipients(Message.RecipientType.TO,
InternetAddress.parse(p_to, false));
```

```

l_msg.setRecipients(Message.RecipientType.CC,
InternetAddress.parse(p_cc, false));

l_msg.setRecipients(Message.RecipientType.BCC,
InternetAddress.parse(p_bcc, false));

l_msg.setSubject(p_subject); // Sets the Subject

MimeBodyPart l_mbp = new MimeBodyPart();
l_mbp.setText(p_message);

Multipart l_mp = new MimeMultipart();
l_mp.addBodyPart(l_mbp);

l_msg.setContent(l_mp);

l_msg.setSentDate(new Date());

Transport.send(l_msg);

l_result = l_result + "<font size=4
color=\"blue\"><B>Success!</B>"+
"<font size=4 color=\"black\"> "+

"<hr><font color=green><B>Mail was successfully sent to
</b></font>: "+p_to+"<br>";
if (!p_cc.equals(""))
l_result = l_result + "<FONT color=green><B>CCed To
</B></FONT>: "+p_cc+"<BR>";

if (!p_bcc.equals(""))
l_result = l_result + "<FONT color=green><B>BCCed To
</B></FONT>: "+p_bcc ;

l_result = l_result+"<BR><HR>";
} catch (MessagingException mex) {
l_result = l_result + "<FONT SIZE=4 COLOR=\"blue\"> <B>Error :
</B><BR><HR> "+
"<FONT SIZE=3
COLOR=\"black\">"+mex.toString()+"<BR><HR>";
}

```

```

} catch (Exception e) {

l_result = l_result + "<FONT SIZE=4 COLOR=\"blue\"> <B>Error :
</B><BR><HR> "+
"<FONT SIZE=3 COLOR=\"black\">"+e.toString()+"<BR><HR>";

e.printStackTrace();
}
finally {
return l_result;
}
}
%>

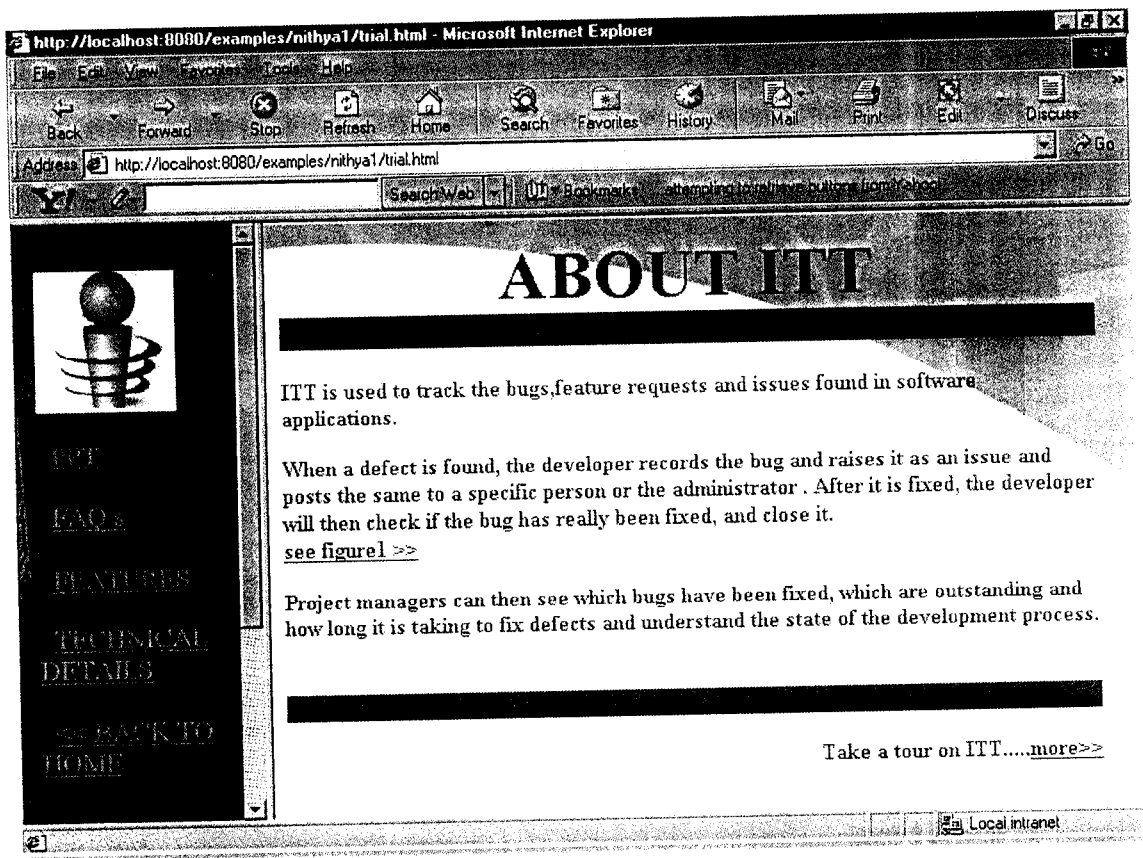
<%
String l_from = request.getParameter("p_from");
String l_to = request.getParameter("p_to");
String l_cc = request.getParameter("p_cc");
String l_bcc = request.getParameter("p_bcc");
String l_subject = request.getParameter("p_subject");
String l_message = request.getParameter("p_message");

String l_smtpSvr = request.getParameter("p_smtpServer");
session.setAttribute("smtpServer",l_smtpSvr);
String l_result =
send(l_from,l_to,l_cc,l_bcc,l_subject,l_message,l_smtpSvr);
%>
<%= l_result %>
<font size=3 color="blue">
<a href="InputsForm.jsp">Compose Mail</A>
</center>
</body>
</html>

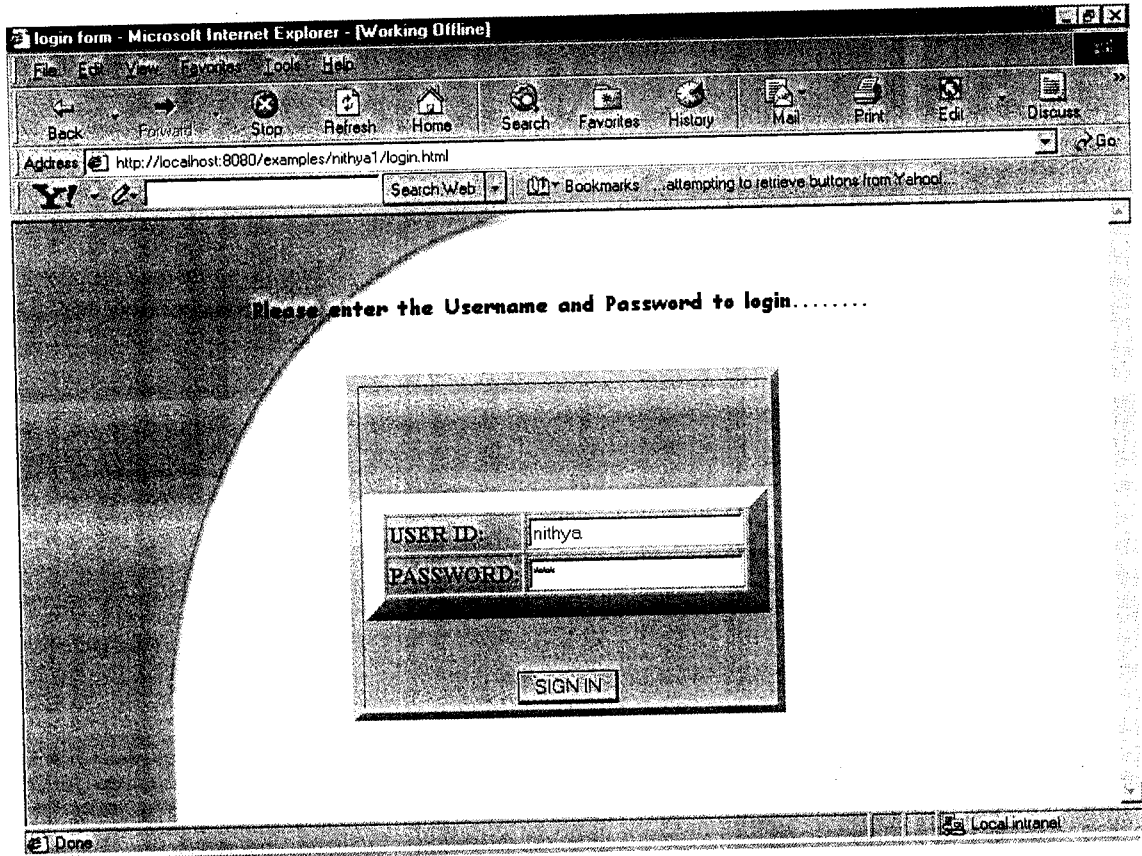
```

## A.2 RESULTS

### INTRODUCTION FORM




# LOGIN FORM



# CREATE FORM

ISSUE INTRODUCTION FORM - Microsoft Internet Explorer - [Working Offline]

Back Forward Stop Refresh Home Search Favorites History Mail Print Edit Discuss



CREATE ISSUES  
VIEWING ISSUES  
BROWSING PROJECT  
SEARCHING ISSUES  
FILTERING ISSUES  
PROPOSING SOLUTION

ISSUE ID: Bug  
PROJECT: WORLD BANK PROJECT  
CATEGORY: BLOCKER  
SEVERITY: XXX  
PRIORITY: XXX  
ASSIGNED TO: vidya  
SUMMARY: Illegal State Exception  
ENVIRONMENT: Java Server Pages using Tomcat web server  
DESCRIPTION: Jasper exception arises when the values are passed. It further generates a general error.

CREATE RESET

Local intranet

# DISPLAY FORM

The screenshot shows a Microsoft Internet Explorer browser window with the address bar displaying `http://localhost:8080/examples/nithya1/issue.html`. The browser's menu bar includes File, Edit, View, Favorites, Tools, and Help. The toolbar contains icons for Back, Forward, Stop, Refresh, Home, Search, Favorites, History, Mail, Print, Edit, and Discuss. The main content area features a dark sidebar on the left with a logo of a sphere on a stand and several menu items: CREATE ISSUES, VIEWING ISSUES, DROWNING PROJECT, SEARCHING ISSUES, FILTERING ISSUES, and PROGRESSING SOLUTION. The main content area displays the text "details entered are ...." above a table of issue details. Below the table is a "Send Mail" button. The status bar at the bottom indicates "Local intranet".


ISSUE ID	63
ISSUE TYPE	Bug
PROJECT TITLE	WORLD BANK PROJEC
PRIORITY	BLOCKER
STATUS	open
ASSIGNED TO	vidya
PERIOD	5
SUMMARY	Illegal State Exception
DESCRIPTION	Jasper exception arises



## COMPOSE MAIL FORM

http://localhost:8080/examples/nithya1/issue.html - Microsoft Internet Explorer

File Edit View Favorites Tools Help



[CREATE ISSUES](#)

[VIEWING ISSUES](#)

[BROWNING PROJECT](#)

[SEARCHING ISSUES](#)

[FILTERING ISSUES](#)

[PROPOSING SOLUTION](#)

[EXAMINING SOLUTION](#)

Server Host

Priority !

From

To

CC

BCC

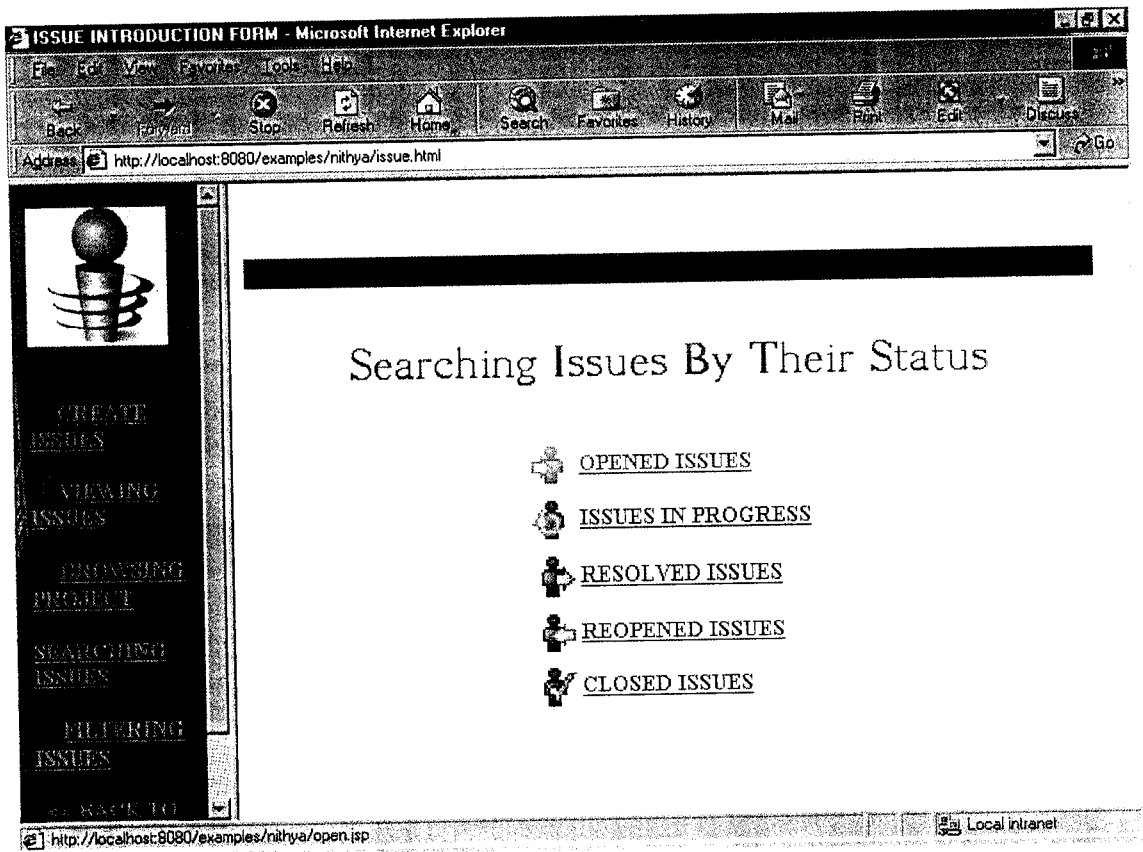
Subject

Message

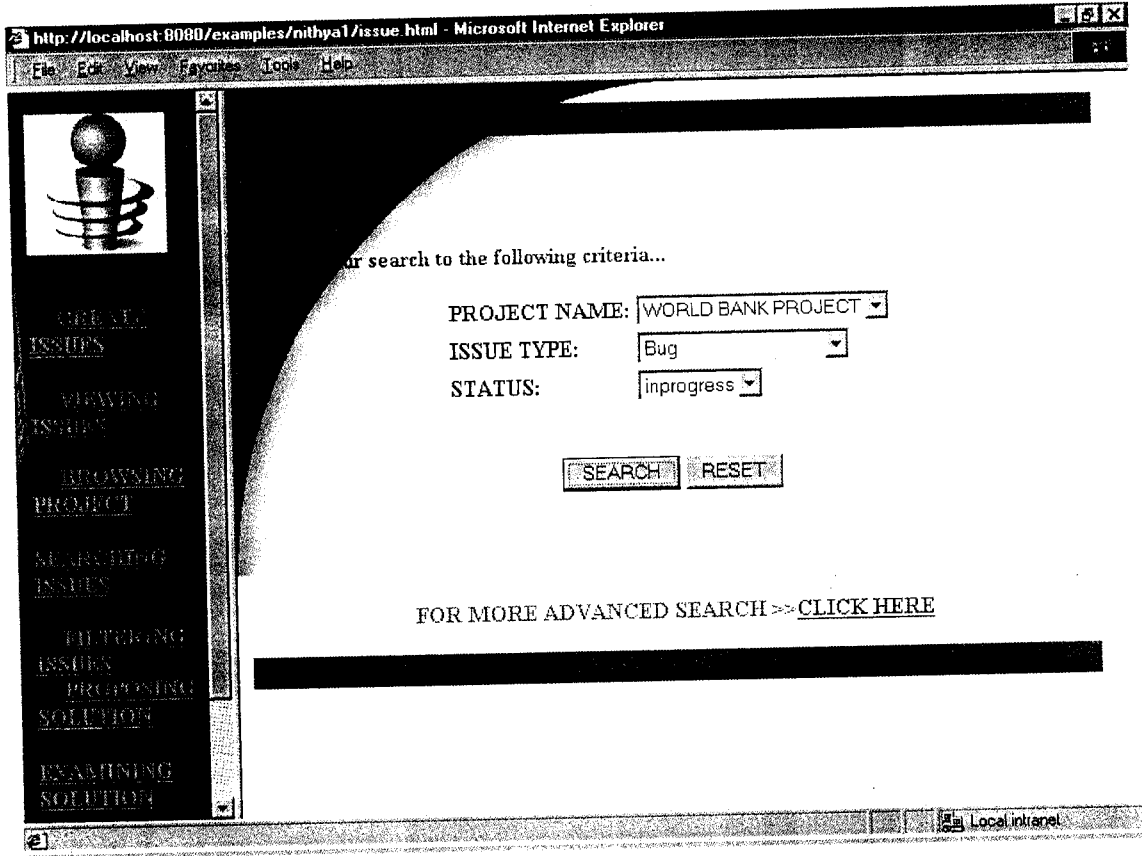
Jasper Exception arises and a general error message is displayed.

Local intranet

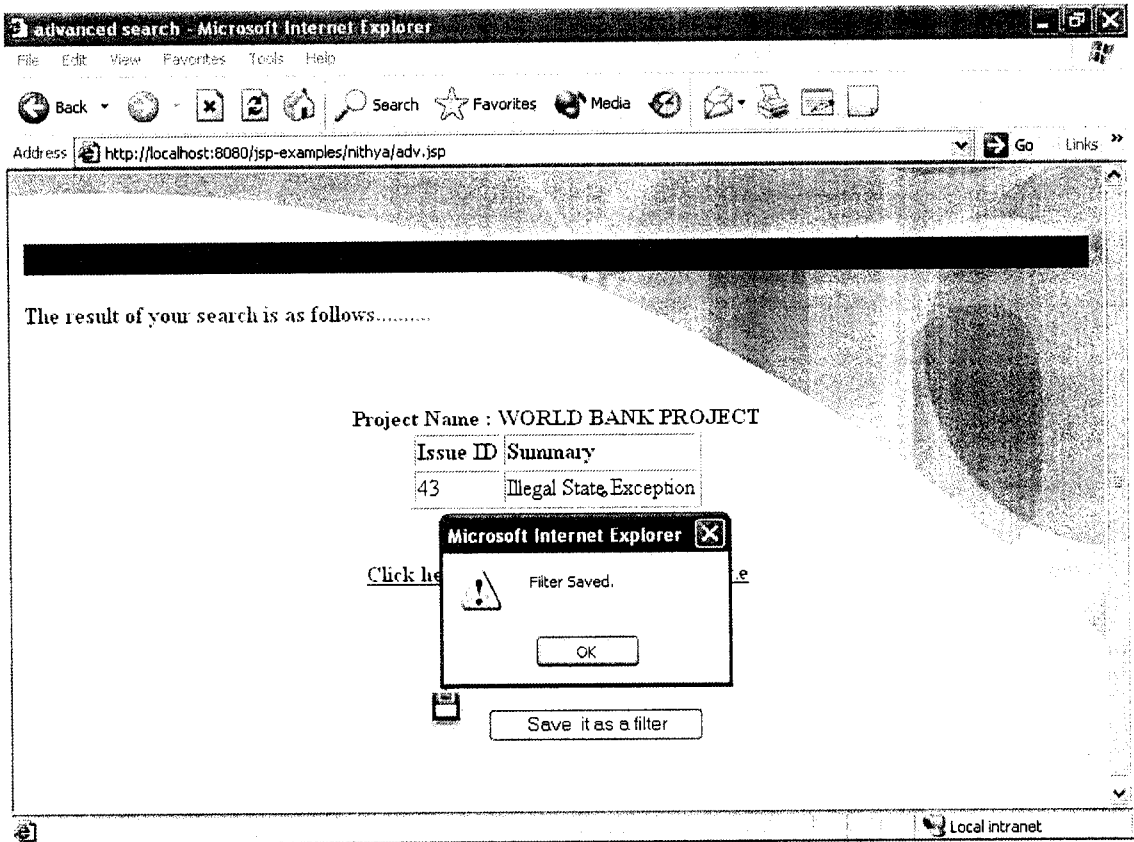
# SEARCH FORM



# BASIC SEARCH FORM



# ADVANCE SEARCH FORM




# SOLUTION FORM

ISSUE INTRODUCTION FORM - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Refresh Home Search Favorites History Mail Print Edit Discuss



CREATE ISSUES  
VIEW ALL ISSUES  
UPDATING PROJECT  
SEARCHING ISSUES  
UPDATING ISSUES  
PROPOSING SOLUTION

Please enter the following .....

ISSUE ID: 63

Check for the consistency of data types.

SOLUTION:

SUBMIT RESET


Local intranet

# EXAMINING SOLUTION FORM

http://localhost:8080/examples/realworld/Issue.html - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Refresh Home Search Favorites History Mail Print Edit Discuss



CREATE ISSUES

VIEWING ISSUES

BROWSING PROJECTS

SEARCHING ISSUES

FILTERING ISSUES

PROPOSING SOLUTIONS

ISSUE ID	63
ISSUE TYPE	Bug
PROJECT TITLE	WORLD BANK PROJEC
BRIEF DESCRIPTION	Jasper exception arises
SOLUTION	Check for the consistency

Microsoft Internet Explorer

Issue closed

OK

Done Local intranet

## REFERENCES

## REFERENCES



- Aaron Tavistock, Damon Hougland, "CORE JSP", Prentice Hall PTR, Upper Saddle River, NJ 07458.
- Avedal (Karl) "Professional JSP", Wrox press Ltd., Hungry Minds, IDG Books, Worldwide Inc., Foster City, California, USA.
- Hans Bergsten, "Java Server Pages", O'reilly publications, July 2000, USA.
- Patrick Naughton, Herbert Schildt, "The Complete Reference JAVA2", Tata McGraw Hill Publishing Company, 3<sup>rd</sup> ed.,.
- Paul Whitehead, "Java Server Pages Programming".