Edubridge learn.png

A Project

Report On

**BUG Detector System**

By

Sanjeeva Rayudu

B.TECH

Batch: 2020- 5345

Centre: Bangalore

Under the Guidance of,

Jayant V.

Technical Trainer

EduBridge

(School of coding)

. **ABSTRACT**

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

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

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

**Features:**

* Product and Component based
* Creating & Changing Bugs at ease
* Query Bug List to any depth
* Reporting & Charting in more comprehensive way
* User Accounts to control the access and maintain security
* Simple Status & Resolutions
* Multi-level Priorities & Severities
* Targets & Milestones for guiding the programmers
* Attachments & Additional Comments for more information
* Robust database back-end.

**I developed this Application in Java, JSP, Servlets and My SQL. It’s a web-based project so I have used HTML, CSS.**

**Software Requirements:**

**Front End: Java/J2EE Technologies (Servlet, JSP), HTML, CSS.**

**Back End: My SQL workbench 5.6.23**

**Middleware/Server: Apache Tomcat v 7.0 IDE: Eclipse IDE for Java EE Developer**

**Browser: Best result on Google Chrome**

**Operating System: Windows 7.**

**Data Base:**

**Create database bts;**

**Use bts;**

**DROP TABLE IF EXISTS `adminresponse`;**

**CREATE TABLE `adminresponse` (**

**`f1` varchar (45) NOT NULL default '',**

**`Message` varchar (45) default NULL,**

**`Reassignto` varchar (45) default NULL,**

**`New status` varchar (45) default NULL,**

**`Tested` varchar (45) default NULL,**

**PRIMARY KEY (`f1`)**

**) ENGINE=InnoDB DEFAULT CHARSET=latin1;**

**ALTER TABLE `adminresponse` DISABLE KEYS;**

**CREATE TABLE `assign\_project` (**

**`Sno` int (10) unsigned NOT NULL auto\_increment,**

**`Project name` varchar (45) NOT NULL default '',**

**`Userid` varchar (45) NOT NULL default '',**

**`Role` varchar (45) NOT NULL default '',**

**PRIMARY KEY (`sno`)**

**) ENGINE=InnoDB DEFAULT CHARSET=latin1;**

**CREATE TABLE `bug\_report (**

**`bugno` int(10) unsigned NOT NULL auto\_increment,**

**`bug\_name` varchar (45) NOT NULL default '',**

**`bug\_type` varchar (45) NOT NULL default '',**

**`bug\_level` varchar (45) NOT NULL default '',**

**`Priority` varchar (45) NOT NULL default '',**

**`Pname` varchar (45) NOT NULL default '',**

**`Tester code` varchar (45) NOT NULL default '',**

**`Bugdate` varchar (45) NOT NULL default '',**

**`e\_code` varchar (45) NOT NULL default '',**

**`Status` varchar (45) NOT NULL default '',**

**`bug\_rectifieddate` varchar (45) NOT NULL default '',**

**`status1` varchar (45) NOT NULL default '',**

**PRIMARY KEY (`bugno`)**

**) ENGINE=InnoDB DEFAULT CHARSET=latin1;**

**DROP TABLE IF EXISTS `bug\_solution`;**

**CREATE TABLE `bug\_solution` (**

**`Bugno` int (10) unsigned NOT NULL default '0',**

**`e\_code` varchar (45) NOT NULL default '',**

**`Solution` varchar (100) NOT NULL default '',**

**`Date` varchar (45) NOT NULL default ''**

**) ENGINE=InnoDB DEFAULT CHARSET=latin1;**

**ALTER TABLE `bug\_solution` DISABLE KEYS ;**

**LOCK TABLES `bug\_solution` WRITE;**

**ALTER TABLE `bug\_solution` ENABLE KEYS;**

**--**

**-- Table structure for table `department\_details`**

**--**

**DROP TABLE IF EXISTS `department\_details`;**

**CREATE TABLE `department\_details` (**

**`dept\_no` int (10) unsigned NOT NULL auto\_increment,**

**`dept\_name` varchar (45) NOT NULL default '',**

**`dept\_loc` varchar (45) NOT NULL default '',**

**PRIMARY KEY (`dept\_no`)**

**) ENGINE=InnoDB DEFAULT CHARSET=latin1;**

**--**

**-- Dumping data for table `department\_details`**

**ALTER TABLE `department\_details` ENABLE KEYS ;**

**DROP TABLE IF EXISTS `employee`;**

**CREATE TABLE `employee` (**

**`Username` varchar (45) NOT NULL default '',**

**`Userid` varchar (45) default NULL,**

**`Password` varchar (45) default NULL,**

**`Email` varchar (45) default NULL,**

**`Address` varchar (45) default NULL,**

**`Phno` varchar (45) default NULL,**

**`Status` varchar (45) default NULL,**

**PRIMARY KEY (`username`)**

**) ENGINE=InnoDB DEFAULT CHARSET=latin1;**

**-- Dumping data for table `employee`**

**--**

**ALTER TABLE `employee` DISABLE KEYS;**

**LOCK TABLES `employee` WRITE;**

**Select\*from bts.employee;**

**UNLOCK TABLES;**

**ALTER TABLE `employee` ENABLE KEYS;**

**--**

**-- Table structure for table `employee\_details`**

**--**

**DROP TABLE IF EXISTS `employee\_details`;**

**CREATE TABLE `employee\_details` (**

**`UserID` int (10) unsigned NOT NULL auto\_increment,**

**`Fame` varchar (45) NOT NULL default '',**

**`Lname` varchar (45) NOT NULL default '',**

**`Gender` varchar (45) NOT NULL default '',**

**`DOB` varchar (45) NOT NULL default '',**

**`Qualification` varchar (45) NOT NULL default '',**

**`Address` varchar (45) NOT NULL default '',**

**`Phoneno` varchar (45) NOT NULL default '0',**

**`Mobileno` varchar (45) NOT NULL default '0',**

**`Emailid` varchar (45) NOT NULL default '',**

**`Doj` varchar (45) NOT NULL default '',**

**`Designation` varchar (45) NOT NULL default '',**

**`Martial Status` varchar (45) NOT NULL default '',**

**`Password` varchar (45) NOT NULL default '',**

**`Hint question` varchar (45) NOT NULL default '',**

**`Hint answer` varchar (45) NOT NULL default '',**

**PRIMARY KEY (`UserID`)**

**) ENGINE=InnoDB DEFAULT CHARSET=latin1;**

**--**

**-- Dumping data for table `employee\_details`**

**--**

**ALTER TABLE `employee\_details` DISABLE KEYS;**

**LOCK TABLES `employee\_details` WRITE;**

**Select\*from bts.employee\_details;**

**DELETE FROM Employee\_details WHERE e\_code=15;**

**UNLOCK TABLES;**

**ALTER TABLE `employee\_details` ENABLE KEYS;**

**Select\*from bts.employee\_details;**

**--**

**-- Table structure for table `employee request`**

**--**

**DROP TABLE IF EXISTS `employee request`;**

**CREATE TABLE `employee request` (**

**`f1` varchar (45) NOT NULL default '',**

**`To` varchar (45) default NULL,**

**`From` varchar (45) default NULL,**

**`Subject` varchar (45) default NULL,**

**`Message` varchar (45) default NULL,**

**`Tested` varchar (45) default NULL,**

**PRIMARY KEY (`f1`)**

**) ENGINE=InnoDB DEFAULT CHARSET=latin1;**

**--**

**-- Dumping data for table `employee request`**

**ALTER TABLE `employee request` DISABLE KEYS;**

**LOCK TABLES `employee request` WRITE;**

**UNLOCK TABLES;**

**ALTER TABLE `employee request` ENABLE KEYS;**

**--**

**-- Table structure for table `project\_details`**

**--**

**DROP TABLE IF EXISTS `project\_details`;**

**CREATE TABLE `project\_details` (**

**`project\_no` int (10) unsigned NOT NULL auto\_increment,**

**`project\_name` varchar (45) NOT NULL default '',**

**`project\_description` varchar (1000) NOT NULL default '',**

**`Sdate` varchar (45) NOT NULL default '',**

**`Duration` int (10) unsigned NOT NULL default '',**

**`Client name` varchar (45) NOT NULL default '',**

**`Client address` varchar (45) NOT NULL default '',**

**`Clientphone` varchar (45) NOT NULL default '',**

**`Client email` varchar (45) NOT NULL default '',**

**`Project lead` varchar (45) NOT NULL default '',**

**`Deptname` varchar (45) NOT NULL default '',**

**PRIMARY KEY (`project\_no`)**

**) ENGINE=InnoDB DEFAULT CHARSET=latin1;**

**--**

**-- Dumping data for table `project\_details`**

**--**

**ALTER TABLE `project\_details` DISABLE KEYS;**

**LOCK TABLES `project\_details` WRITE;**

**Select\*from bts.project\_details;**

**DELETE FROM project\_details WHERE project\_no=18;**

**UNLOCK TABLES;**

**ALTER TABLE `project\_details` ENABLE KEYS;**

**--**

**-- Table structure for table `Edit\_profile`**

**--**

**DROP TABLE IF EXISTS `Edit\_profile`;**

**CREATE TABLE `Edit\_profile` (**

**`UserID` int (10) unsigned NOT NULL auto\_increment,**

**`Fname` varchar (45) NOT NULL default '',**

**`Lname` varchar (45) NOT NULL default '',**

**`Gender` varchar (45) NOT NULL default '',**

**`DOB` varchar (45) NOT NULL default '',**

**`Qualification` varchar (45) NOT NULL default '',**

**`Address` varchar (45) NOT NULL default '',**

**`Phoneno` varchar (45) NOT NULL default '0',**

**`Mobileno` varchar (45) NOT NULL default '0',**

**`Emailed` varchar (45) NOT NULL default '',**

**`Doj` varchar (45) NOT NULL default '',**

**`Designation` varchar (45) NOT NULL default '',**

**`Marital Status` varchar (45) NOT NULL default '',**

**`Password` varchar (45) NOT NULL default '',**

**`Hint question` varchar (45) NOT NULL default '',**

**`Hintanswer` varchar (45) NOT NULL default '',**

**PRIMARY KEY (`UserID`)**

**) ENGINE=InnoDB DEFAULT CHARSET=latin1;**

**--**

**-- Dumping data for table `Edit\_profile`**

**--**

**ALTER TABLE `Edit\_\_profile` DISABLE KEYS ;**

**LOCK TABLES `Edit\_profile` WRITE;**

**Select\*from bts.Edit\_profile;**

**ALTER TABLE `Edit profile` ENABLE KEYS;**

**Select\*from bts.Edit\_profile;**

Table of Contents:

1. Introduction

**1.1 Purpose**

**1.2 Existing System**

**1.3 Proposed System**

2. Software and Hardware Requirements:

**2.1 Software Requirements**

**2.2 Hardware Requirements**

**3. UML Diagrams:**

**3.2.1 Class Diagram**

**3.2.2 Interaction Diagram**

**3.2.2.1 Sequence Diagram**

**3.2.2.2 Collaboration Diagram**

**3.2.3 Use Case Diagram**

**3.2.4 Component Diagram**

**3.2.5 Deployment Diagram**

**List of Photos:**

Home Page

Registration Form

Admin home Page

Tester Home Page

Developer home Page

Admin Add Department

Admin View Department

Admin view employee

Admin Add Project

View Project

Developer Bug Information

Assign Bug

Employee password Changes

# Introduction

## Purpose

The main objective of this system is develop flawless system, which is access real time information from anywhere in the world, 24 hours a day 365 days in a year.  Another aim is that manage hundred of projects in multiple locations or just a few. The another main objective of this system is track the all the defects or bugs in the project and make the project user friendly and bugs free system.

**1.2 Existing System**

In any software development bugs are inevitable. Let it be in any kind of product bugs arise at any phase of development. One has to take a great care in the proper maintenance and resolution of the bugs. In the Existing system the bugs are not properly maintained and they are simply relied on shared lists and email to monitor the bugs.

In this type of system it becomes difficult to track a bug if a bug is over looked then it may cause tremendous errors in the next phase and can improve the cost of project whatever necessary effort spent on the bug maintenance may not be worthy. So bug history has to be maintained properly. And there is no efficient search technique.

One has to search the whole database for the details of particular bug which might have occurred sometime earlier. It is both time consuming and error prone. And it is very difficult to share the bug among several users as there is no proper maintenance of the bugs. In order to have an efficient product bugs must be maintained properly and should be resolved in time both to reduce time and money spent on the development.

**1.3 Proposed System**

* This system maintains the products, Bugs and bug tracking. It has advantage of maintaining bug history it stores all the details from bug origin to bug resolution.
* Each product can have versions for easy maintenance of the product and all the user of the product is stored in the database. It provides the advantage of maintaining users to the bugs and resolutions provided by them.
* Our System provides the searching based on status, priority, and operating system.
* It provides with user and bug hierarchy, which would be helpful in knowing the relation between bugs and users allotted to the bug.
* It is provided with a fully authenticated system with password encryption. And has the facility for storing attachments for a bug.
* One can keep a track of the bug in a product with much lower cost and effort.
* The most advantage of this system is maintaining log records which are helpful in knowing any errors or misuse of the system by other users.

**2. Software and Hardware Requirements**

## 2.1 Software Requirements

A set of programs associated with the operation of a computer is called software. Software is the part of the computer system which enables the user to interact with several physical hardware devices.

The minimum software requirement specifications for developing this project are as follows:

Designing frontend : JSP and Servlets

Backend : My SQL

Scripting : Java Script

UML : Rational Rose

IDE : My Eclipse

Web Server : Tomcat

## 2.2 Hardware Requirement Specification

The Collection of internal electronic circuits and external physical devices used in building a computer is called Hardware.

The minimum hardware requirement specification for developing this project is as follows:

Processor : Pentium IV

RAM : 512MB RAM

Hard Disk : 10GB

**3. UML Diagrams:**

**Unified Modelling Language**

The Unified Modelling Language allows the software engineer to express an analysis model using the modelling notation that is governed by a set of syntactic semantic and pragmatic rules.

This UML diagrams must include the following:

* Class diagram
* Interaction Diagram
* Use case Diagram
* Activity Diagram
* Component Diagram
* Deployment Diagram

**3.2.1Class Diagrams**

The class diagram is the main building block in [object oriented](http://en.wikipedia.org/wiki/Object_oriented) modelling. They are being used both for general [conceptual modelling](http://en.wikipedia.org/wiki/Conceptual_model) of the systematic of the application, and for detailed modelling translating the models into [programming code](http://en.wikipedia.org/wiki/Programming_code).

The classes in a class diagram represent both the main objects and or interactions in the application and the objects to be programmed. In the class diagram these classes are represented with boxes which contain three parts:

* The upper part holds the name of the class
* The middle part contains the attributes of the class, and
* The bottom part gives the methods or operations the class can take or undertake

**An Activity Diagram** shows the flow from activity to activity.

An activity is an ongoing non- atomic execution within a state machine.

**3.2.1Class Diagrams:**



Fig: Class Diagram

**3.2.2 Interaction Diagram**

**Interaction Diagrams**

An interaction diagram shows an interaction, consisting of a set of objects and their relationships, including the messages that may be dispatched among them.

**A sequence diagram** is an interaction diagram that emphasizes the time ordering of messages. Graphically, a sequence diagram is a table that shows objects arranged along x-axis and messages, ordered in increasing time, along the y-axis.

**A Collaboration** is a society of classes, interfaces, and other elements that work together to provide some cooperative behaviour that’s bigger than the sum of all its parts.

**3.2.2.1 Sequence Diagram**

* An interaction diagram shows an interaction, consisting of a set of objects and their relationships, including the messages that may be dispatched among them.
* A sequence diagram is an interaction diagram that emphasizes the time ordering of messages.
* Graphically, a sequence diagram is a table that shows objects arranged along x-axis and messages, ordered in increasing time, along the y-axis.

**Contents**

* Sequence diagrams commonly contain the following:
  + - Objects
    - Links
    - Messages

Like all other diagrams, sequence diagrams may contain notes and constrains.



Fig: Administration Sequence Diagram





Fig: Employee Sequence



Fig: Manager Sequence

**3.2.2.2 Collaborations Diagram**

Collaboration is a society of classes, interfaces, and other elements that work together to provide some cooperative behaviour that’s bigger than the sum of all its parts.

Collaboration is also the specification of how an element, such as a classifier or an operation, is realized by a set of classifiers and associations playing specific roles used in a specific way

**Contents:**

Collaboration diagrams commonly contain the following:

* + - Objects
    - Links
    - Messages

Like all other diagrams, sequence diagrams may contain notes and constrains.



Fig: Administrative Collaboration



Fig: Employee Collaboration



Fig: Manager Collaboration

**3.2.3 Use case Diagram**

**A use case diagram**  is a diagram that shows a set of use cases and actors and relationships.

Use case Diagrams represent the functionality of the system from a user’s point of view. Use cases are used during requirements elicitation and analysis to represent the functionality of the system .Use cases focus on the behaviour of the system from external point of view.



Fig: Over all Use Case Diagram



Fig: Administrator Use Case



Fig: Manager Use Case

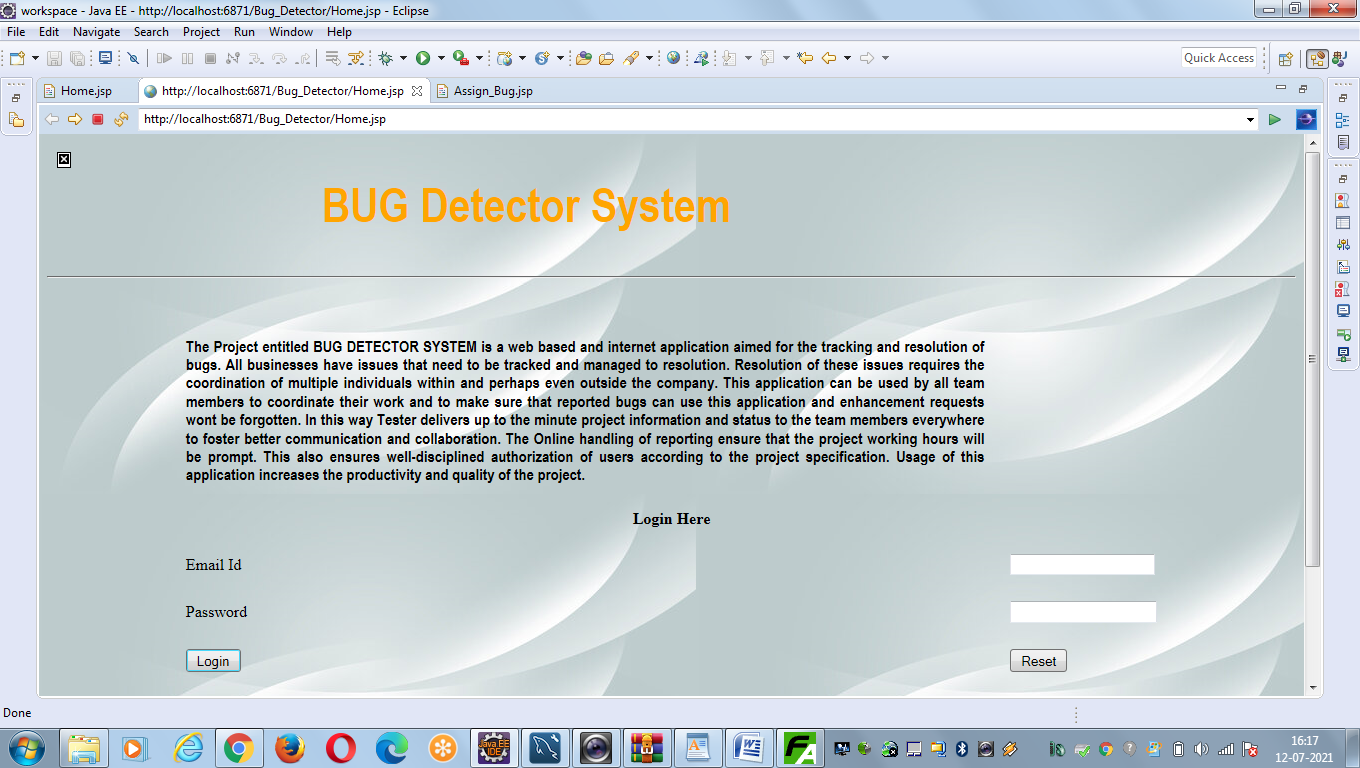


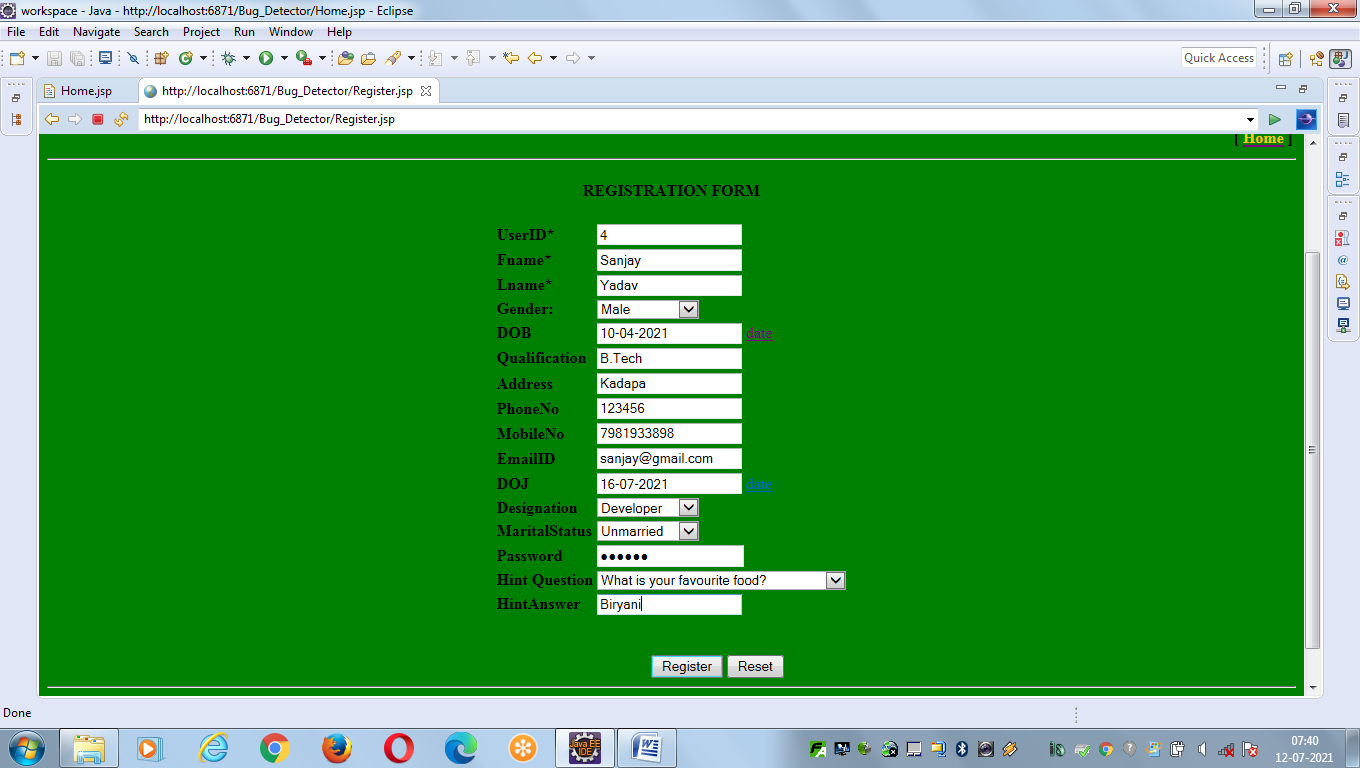
Fig: Employee Use Case



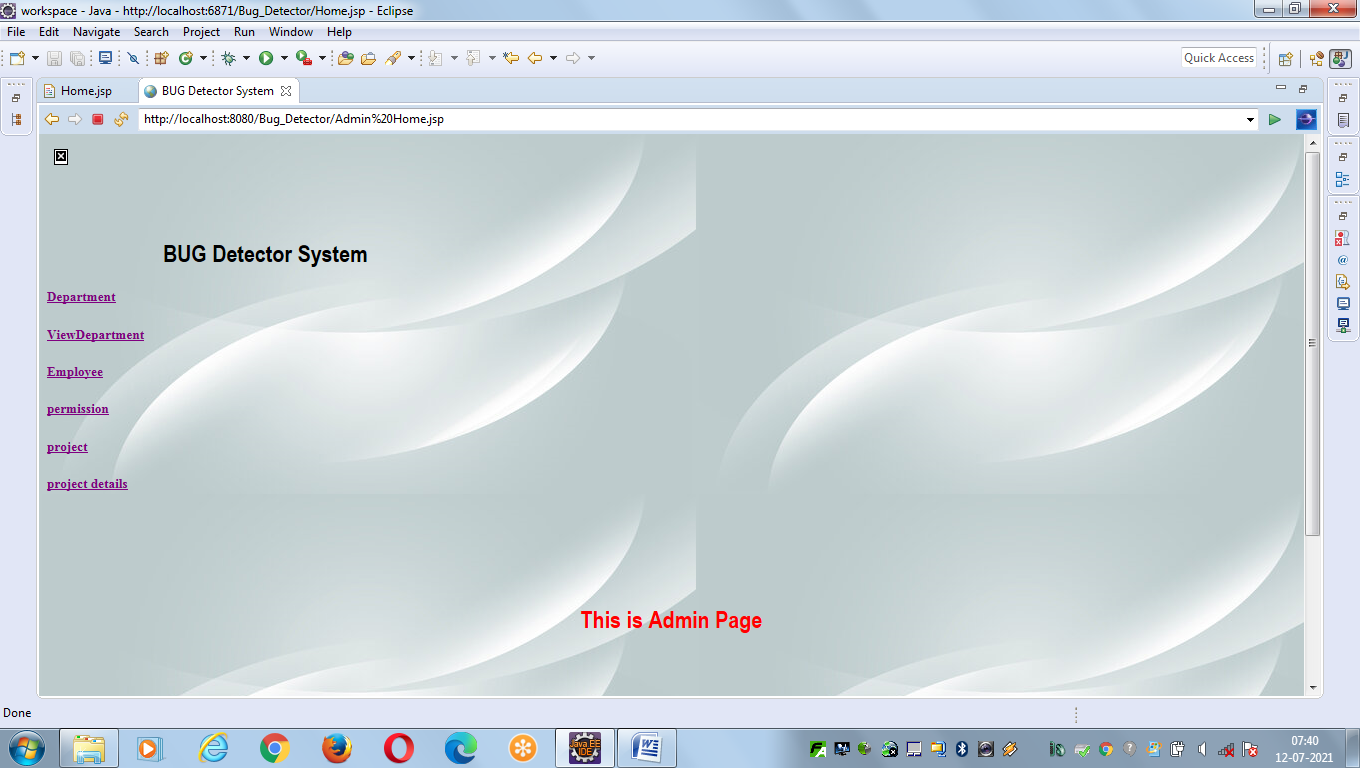
Fig: Component Diagram

Screen Shot Images:

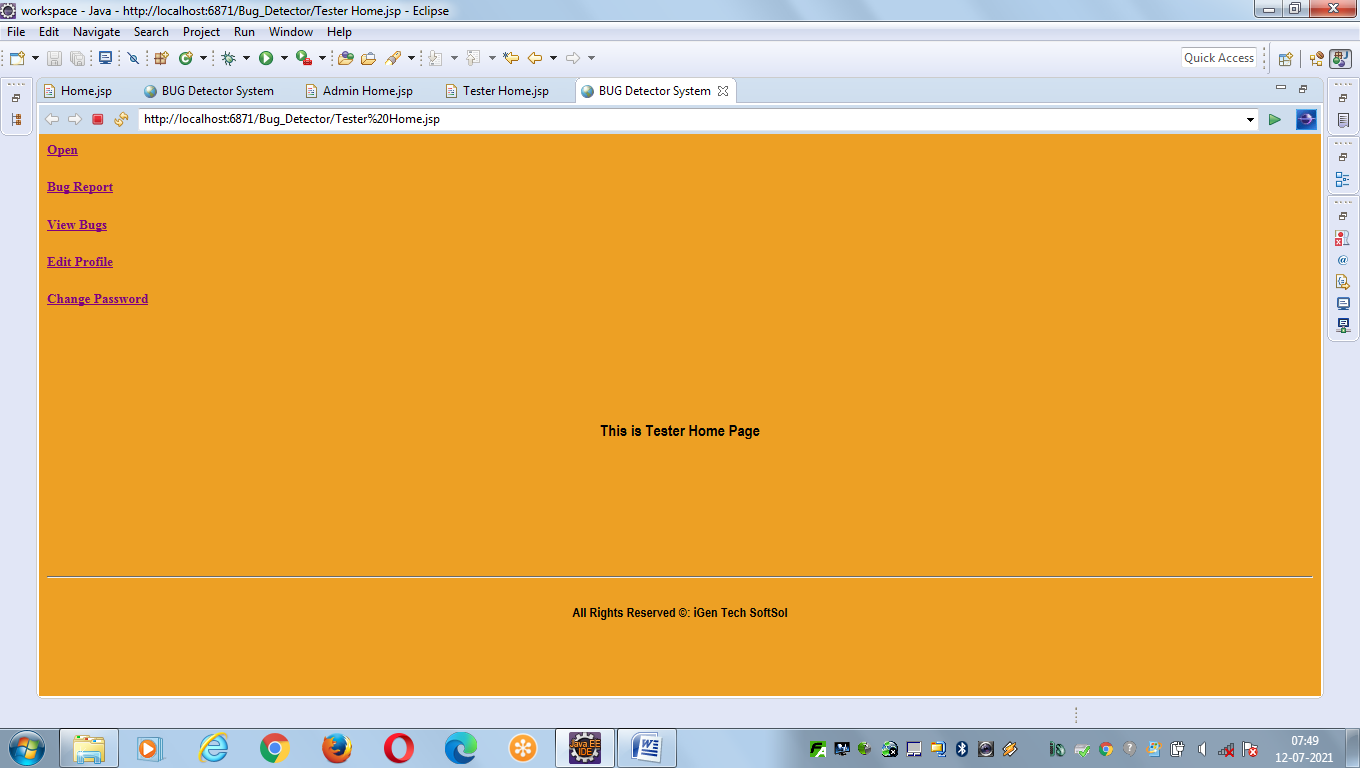
Home Page: 

Registration Form: 

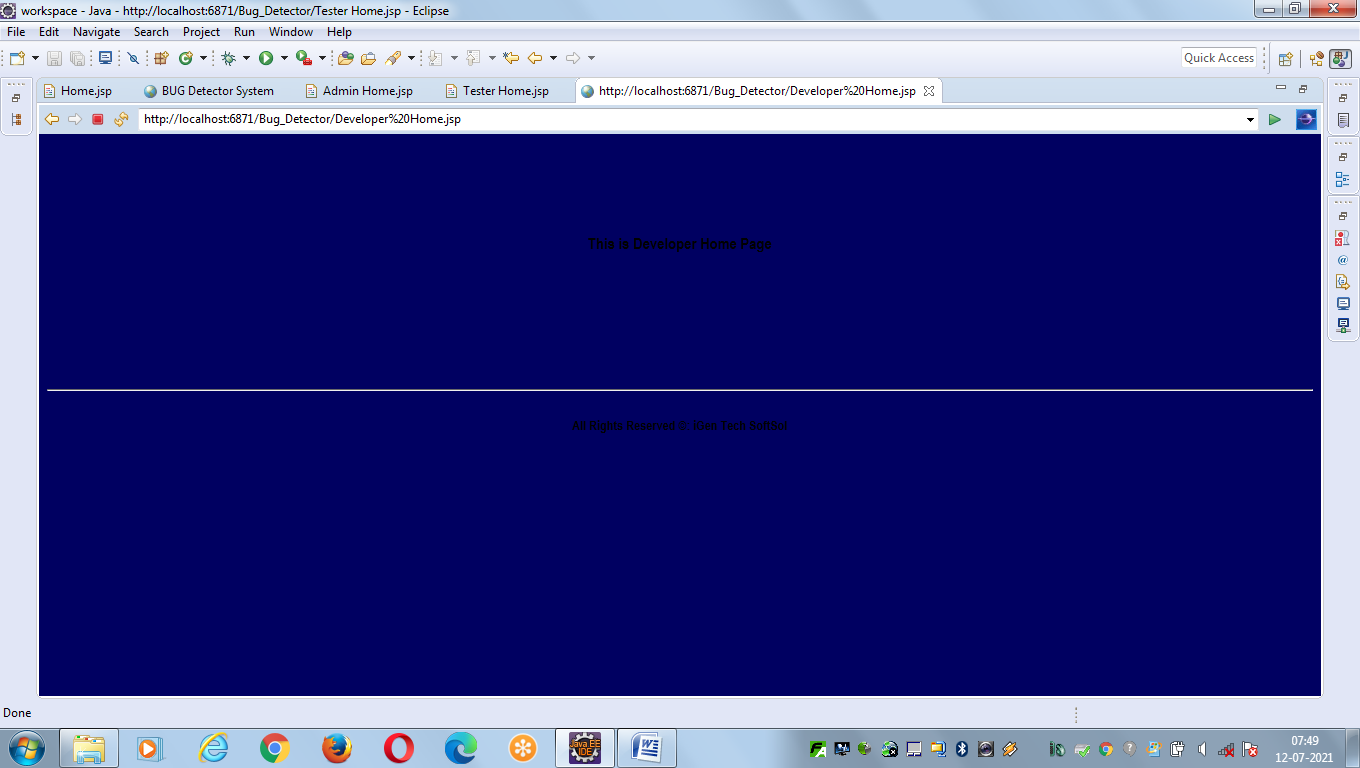
Admin Page:



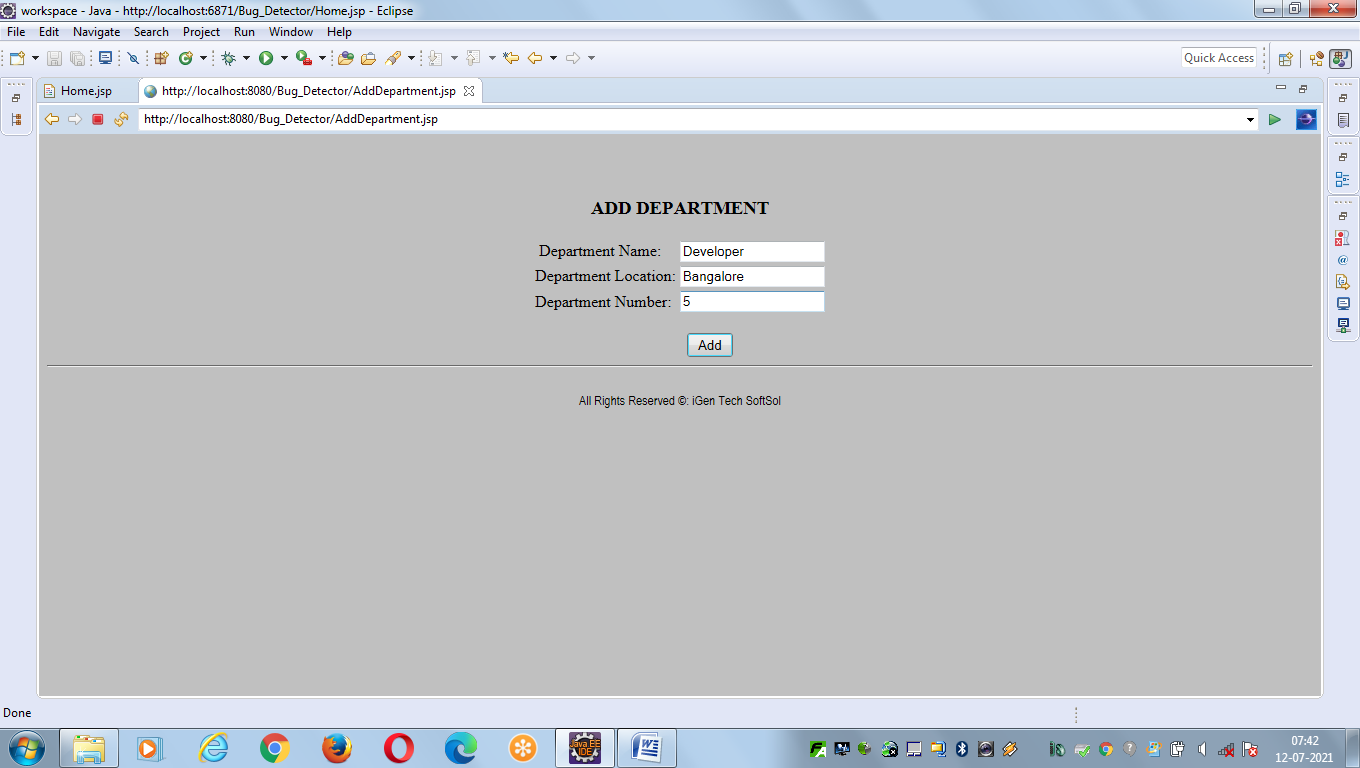
Tester Home Page:



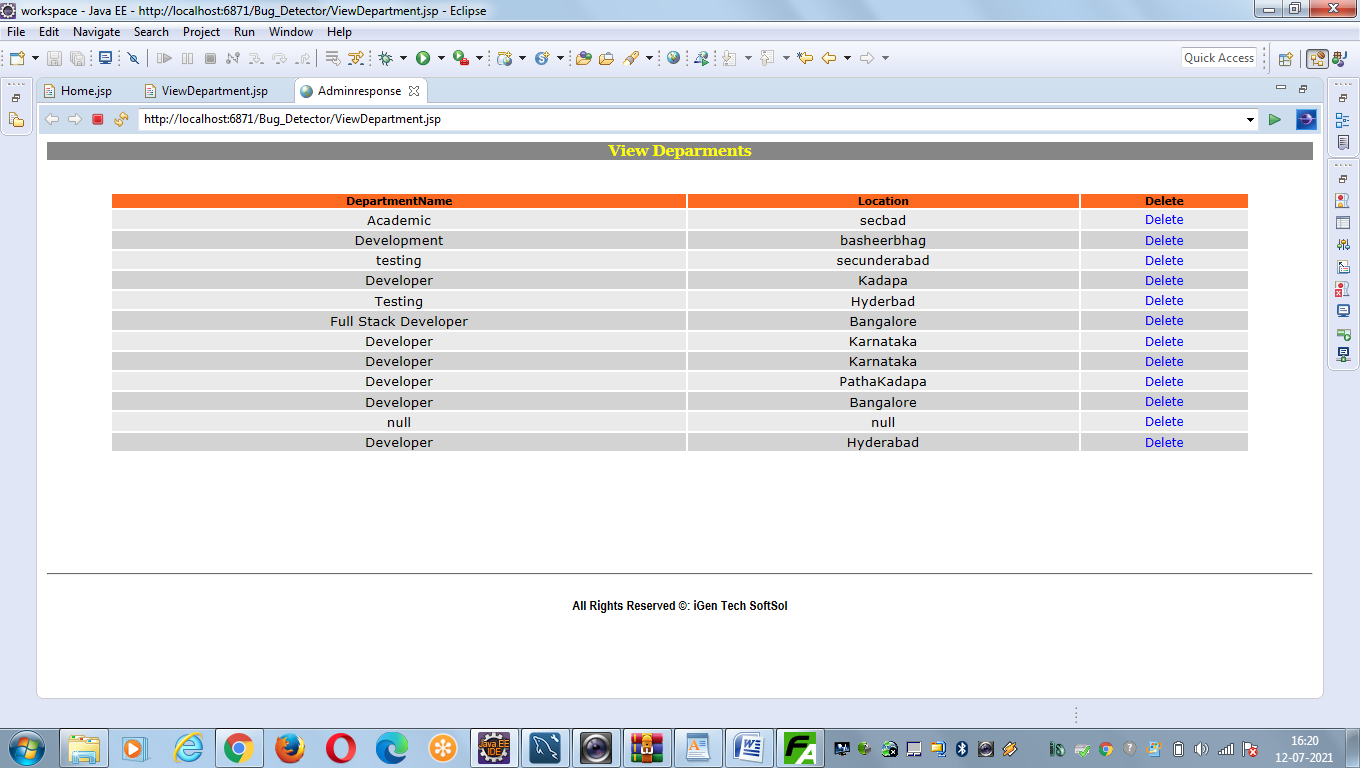
Developer Home Page:



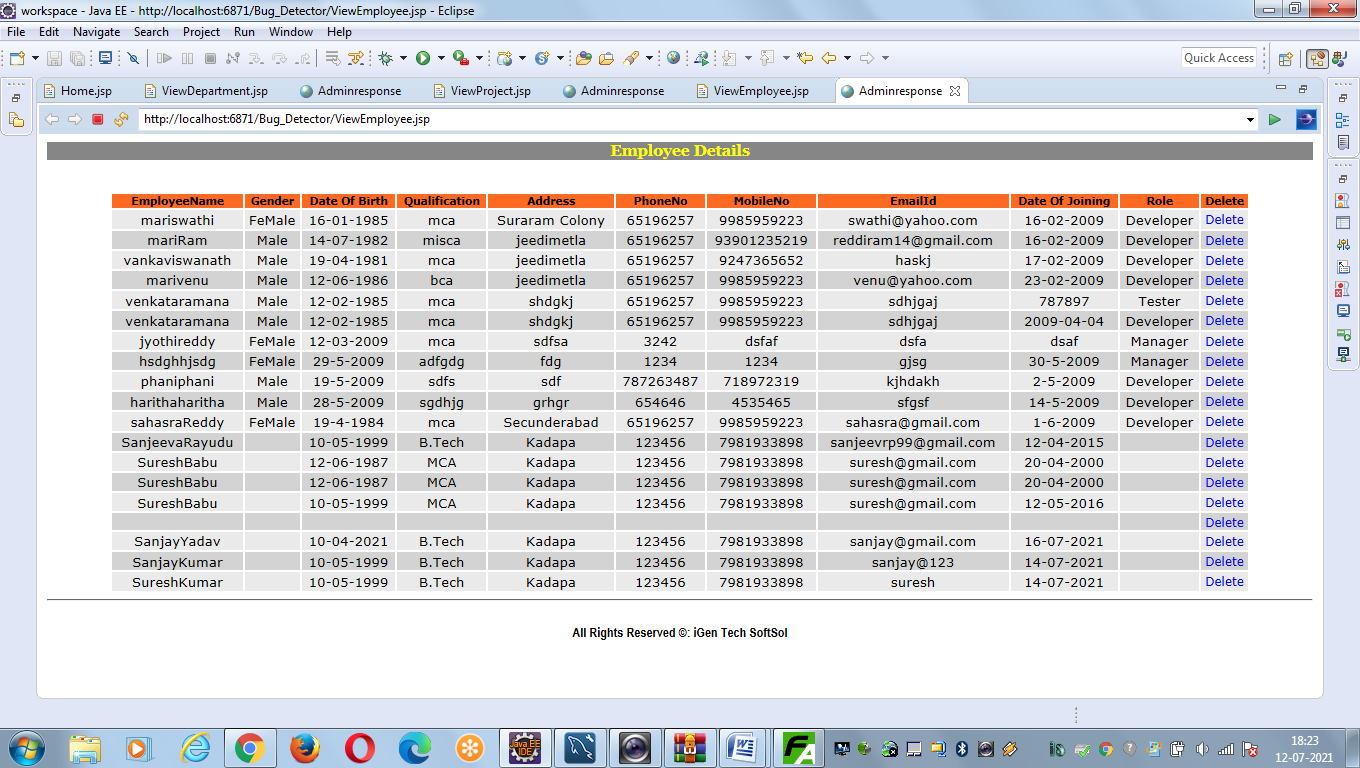
Add Department:



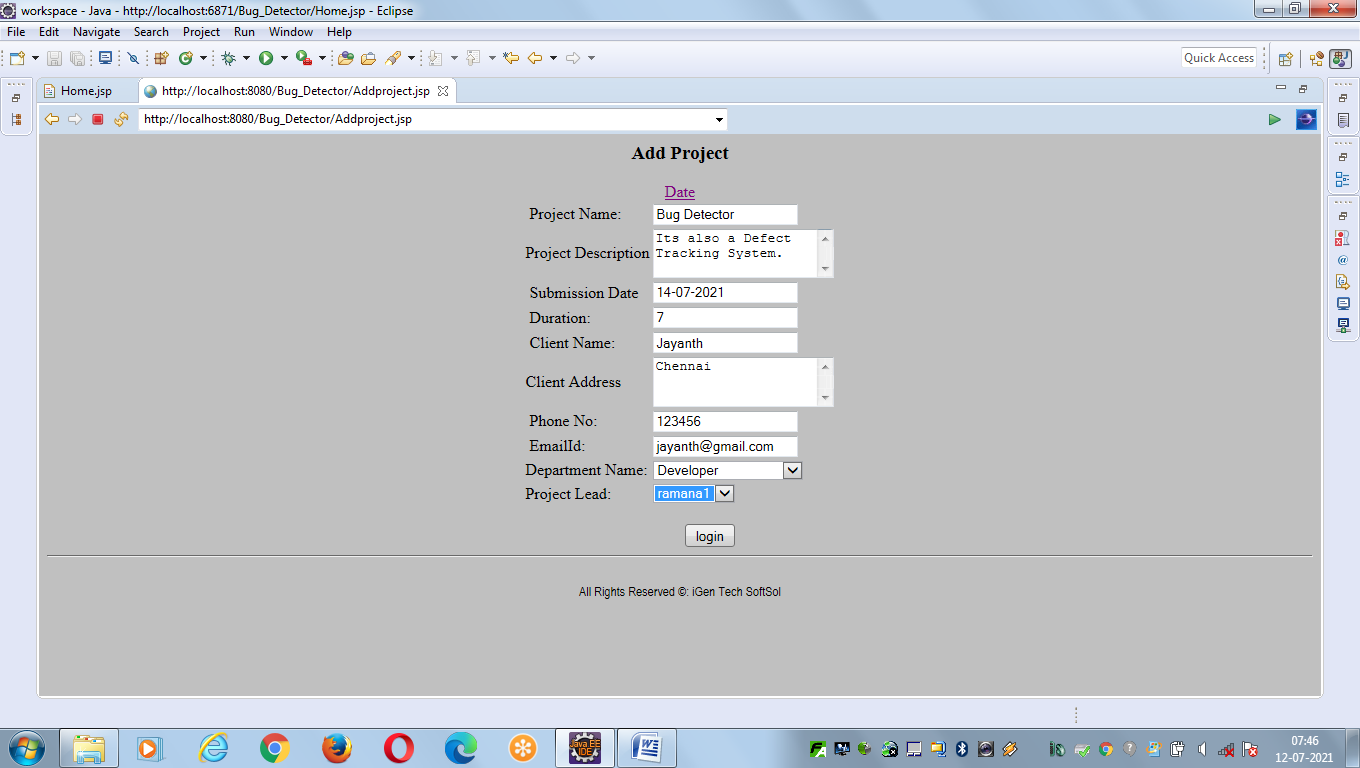
View Department:

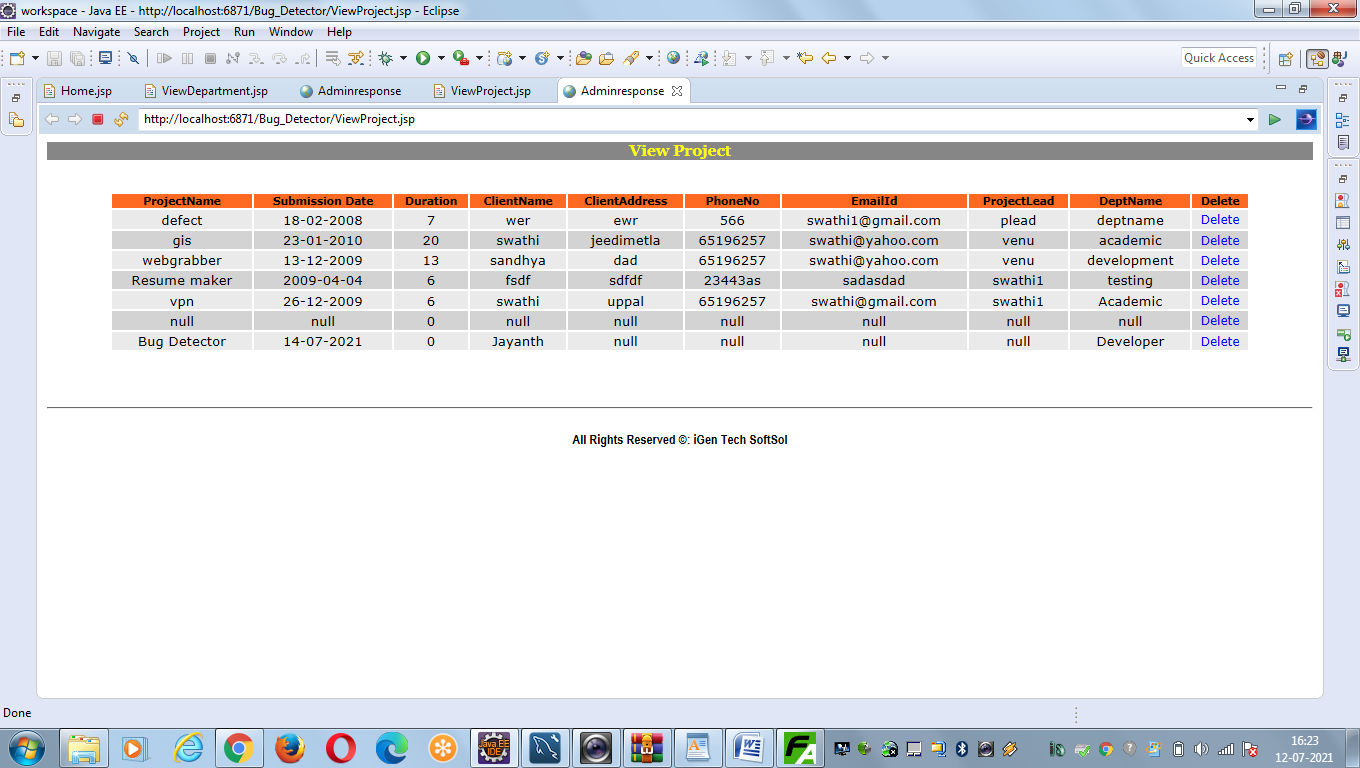


Admin View Employee:

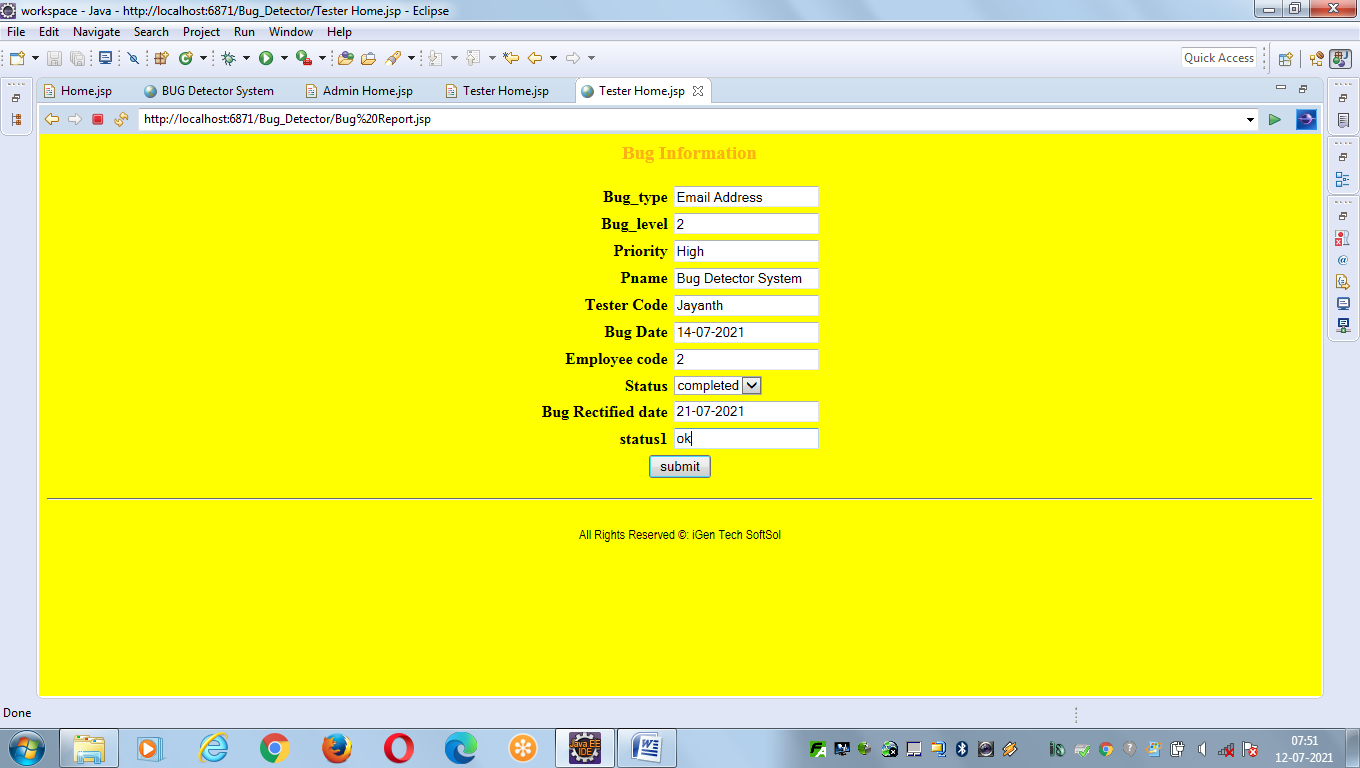


Add Project:

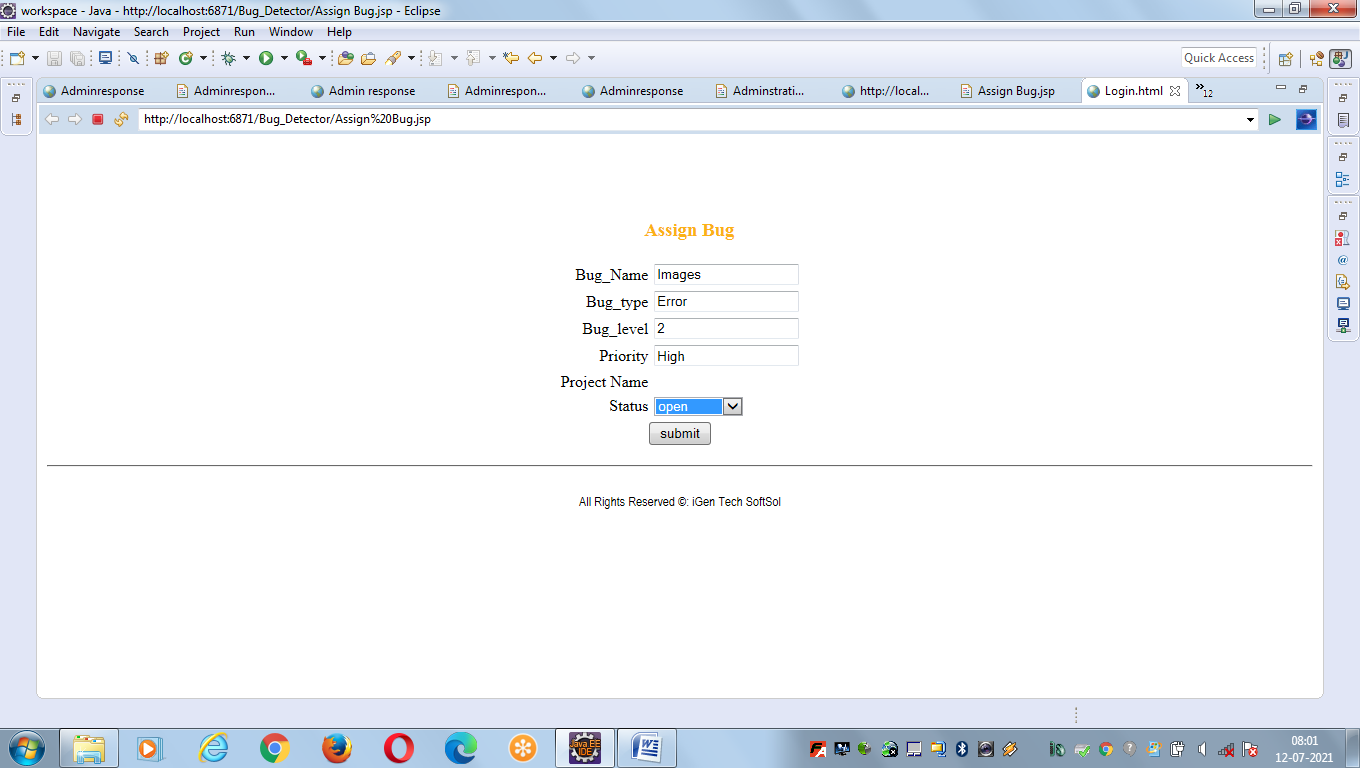


View Project: 

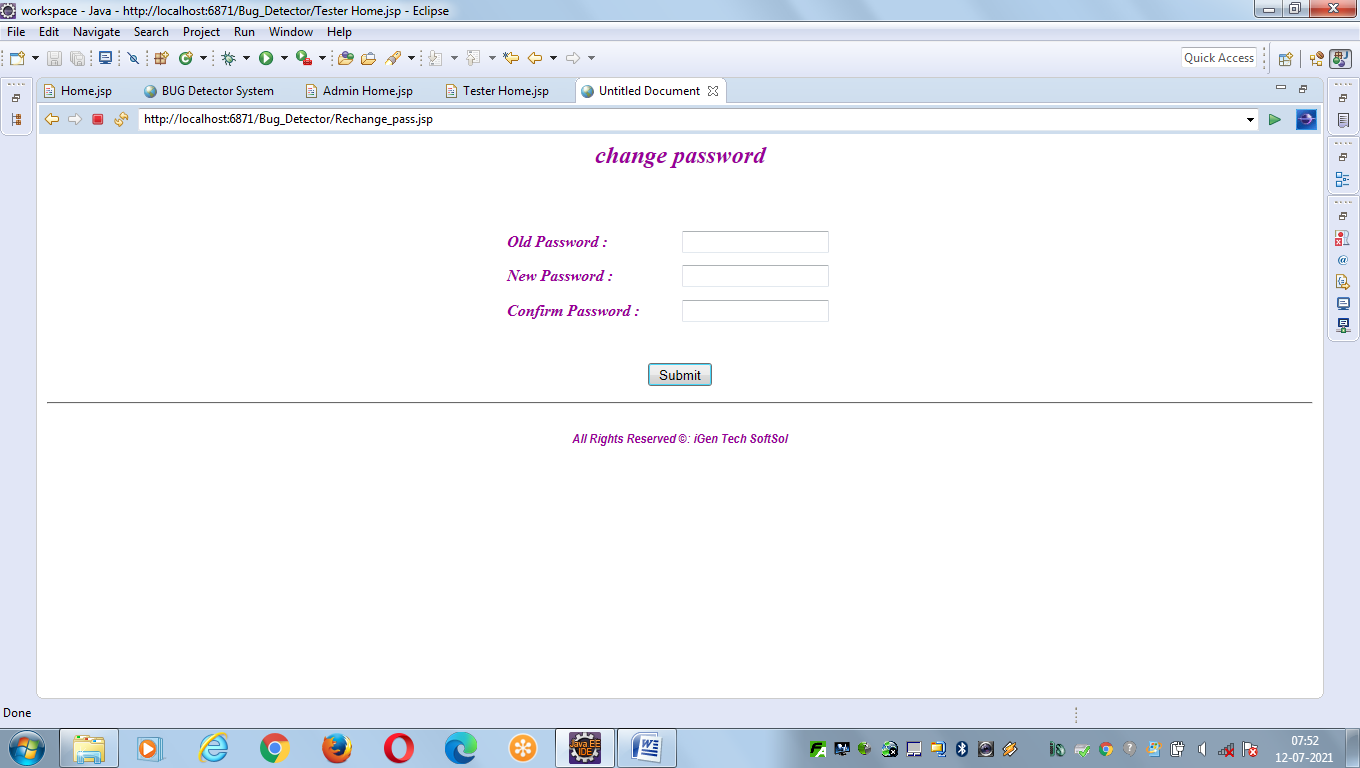
Developers Bug Information:



Assign Bug:



Employee Password Changes:



THANK YOU