

IT Trouble Ticket Management System

Prepared By:

Chetan Munegowda(A20334370)

Abhay Girish Manoli(A20332174)

Section:

ITMD411-05

Table of Contents

Project Overview.....	4
IT Trouble Ticket System	4
Project Objective.....	4
Database Specifications	6
Design.....	6
Table Specifications	6
Table Scripts:.....	7
Stored Procedures scripts	9
Usecase Diagram:.....	11
User:.....	11
Admin:.....	11
Project source code with screen shots:	12
Login Code(Main File):	12
Login Screen Shot:.....	18
Create User/Admin Java File:.....	18
Create Admin/User ScreenShot:.....	29
Admin Landing Page:	31
Admin Landing Page Screen Shot:	36
User Landing Page:.....	37
User Landing Page Screen Shot:	43
Create Ticket By User:.....	44
User View/Update/Purge Ticket:.....	52
User View Ticket Snapshot:	60
User Update Ticket Snapshot:	61
User Purge Ticket Snapshot:	62
Admin View/Update Ticket Code:.....	64
Admin View Snapshot:.....	71
Admin Update Snapshot:.....	72
Helper Files and Validations Code:	74
Get SQL Connection:	74
Create Database:.....	75
Integer Field Code:.....	76

Helper File For Validation:	78
Helper File For restricting text field limit:	79
Validations Snapshot:	80
Login Screen:	80
When the fields are empty and the user clicks on login button:	80
About Trouble Ticket Management System:	81
Invalid Credentials:	81
Create User/Admin Validation Screen Shot:	82
Empty Field Validations:	82
Invalid Email Address Validation:	82
Integer Field Validation:	83
Duplicate UserName Validation:	83
Admin Langing Page Sceen Validation:	84
Trouble Ticket Landing Page Validation:	84
Create Ticket Mandatory Field Validation:	85
Session Information: Displaying the logged in user name	85
Future Work:	86
References:	86

Project Overview

IT Trouble Ticket System

Trouble ticket system is computer software package which manages and maintains lists of issues, as needed by an organization. Trouble tracking systems are commonly used in an organization's customer support call center to create, update, and resolve reported customer issues, or even issues reported by that organization's other employees. A support ticket should include vital information for the account involved and the issue encountered. An trouble tracking system is similar to a "bugtracker", and often, a software company will sell both, and some bugtrackers are capable of being used as an issue tracking system, and vice versa.

A ticket element, within an issue tracking system, is a running report on a particular problem, its status, and other relevant data. They are commonly created in a help desk or call center environment and almost always have a unique reference number, also known as a *case*, *issue* or *call log* number which is used to allow the user or help staff to quickly locate, add to or communicate the status of the user's issue or request.

IT Trouble Ticket System allows the User/Admin to register to our system, later User can use the system to create the trouble tickets. Ticket is dynamically allocated to available admin. User can monitor the status of the ticket. Also, admin can login to the trouble ticket system to update the status of the tickets.

Project Objective

IT help desk contacts you for help in creating a help desk trouble ticket system.

Your tasks will incorporate CRUD database techniques. Check the following site for SQL help for using SQL CRUD statements using Java!

Creativity tasks:

- Decide what database table(s)/fields do you need for this
- Design a menu to accommodate users with the following menu items as follows:
 - Start a ticket
 - Close a ticket
 - View a ticket(s) –allow for user to view a particular ticket or multiple tickets

- Update a ticket
- Purge a ticket(s) –allow for user to purge a particular ticket or multiple tickets

Runtime tasks: (Your program should perform the following tasks)

- Input (open up) at least 10 trouble ticket cases
- Show a view of a particular ticket created
- Show a view of all tickets created
- Allow for a closing of at least one ticket. Show pop up message stating that the ticket was closed as follows:
“Ticket #xxxxxxxx has been closed” where xxxxxxxxx represents the ticket number closed
- Update at least one trouble ticket by changing the current description to ‘WH WIFI PROBLEM’
- Purge at least one ticket. Make sure to display a pop up to ask user the following:
‘Are you sure you want to delete this ticket #xxxxxxxx?’ where xxxxxxxxx represents the ticket number in question

Snapshot tasks: Include snapshots in MS Word to include the following:

- A view of a ticket created
- A view of all tickets created
- The trouble ticket pop up message showing that a ticket number was closed
- Show the view of the updated ticket bearing the description ‘WH WIFI PROBLEM’
- The pop up message showing the ticket number requested to be purged
- Final snapshot of a view of all your records with each of your fields

Database Specifications

Design

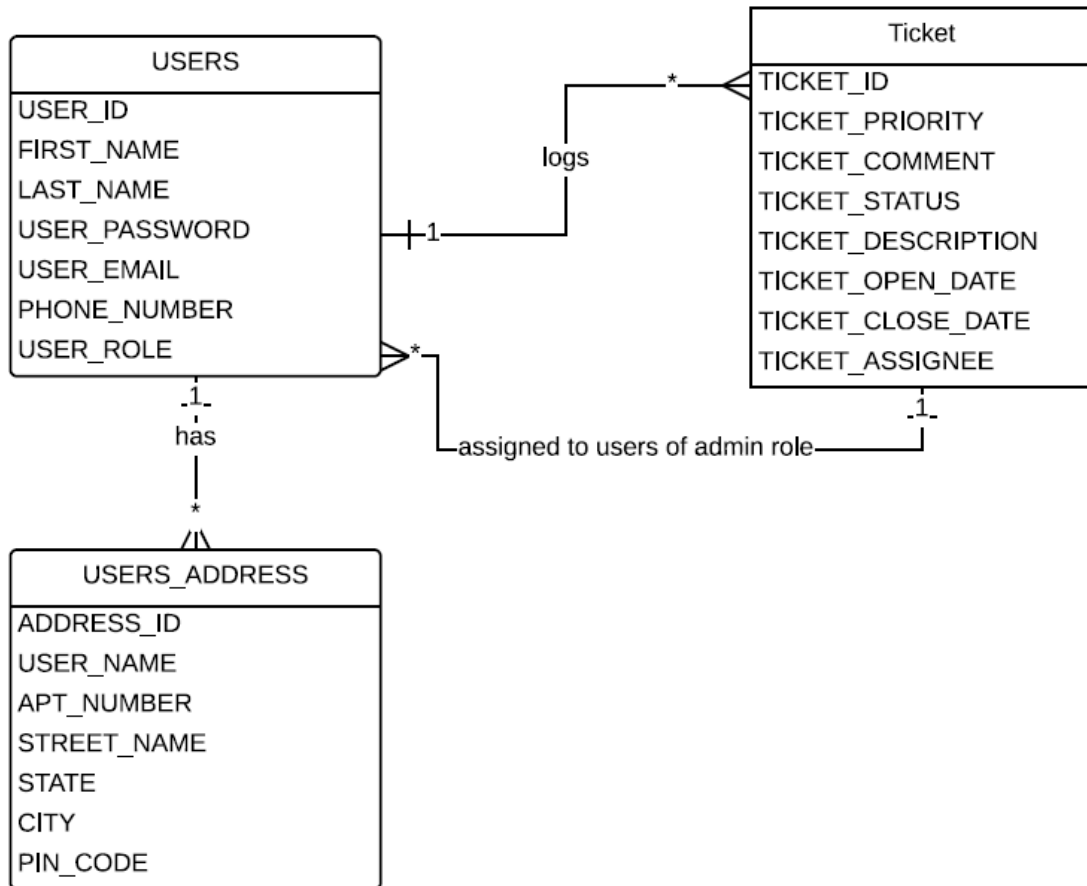


Table Specifications

No	Table Name	Field Name	Field Type
1	CMUNEGOW_TBL_USERS	USER_ID USER_NAME FIRST_NAME LAST_NAME USER_PASSWORD USER_EMAIL PHONE_NUMBER USER_ROLE	INTEGER, Primarykey VARCHAR(20), Unique VARCHAR(40) VARCHAR(40) VARCHAR(30) VARCHAR(50) LONG VARCHAR(10)

2	CMUNEGOW_TBL_USERADDRESS	ADDRESS_ID USER_NAME APT_NUMBER STREET_NAME STATE CITY PIN_CODE	INTEGER,Primarykey VARCHAR(20),Unique INTEGER VARCHAR(50) VARCHAR(25) VARCHAR(25) INTEGER
3	CMUNEGOW_TBL_TICKET	TICKET_ID TICKET_PRIORITY TICKET_COMMENT TICKET_CATEGORY TICKET_STATUS TICKET_DESCRIPTION TICKET_OPEN_DATE TICKET_CLOSE_DATE TICKET_CREATEDBY TICKET_ASSIGNEE TICKET_ISPURGED	INTEGER,PrimaryKey VARCHAR(10) VARCHAR(100) VARCHAR(30) VARCHAR(10) VARCHAR(100) DATE DATE VARCHAR(20),Foreignkey VARCHAR(20),Foreignkey CHAR(1)

Table Scripts:

```

CREATE TABLE CMUNEGOW_TBL_USERS
(USER_ID INTEGER NOT NULL AUTO_INCREMENT ,
USER_NAME VARCHAR(20) NOT NULL UNIQUE ,
FIRST_NAME VARCHAR(40) NOT NULL ,
LAST_NAME VARCHAR(40) NOT NULL ,
USER_PASSWORD VARCHAR(30) NOT NULL ,
USER_EMAIL VARCHAR(50) NOT NULL ,
PHONE_NUMBER LONG NOT NULL ,
USER_ROLE VARCHAR(10) NOT NULL ,
PRIMARY KEY(USER_ID));

```

```
CREATE TABLE CMUNEGOW_TBL_USERADDRESS(  
ADDRESS_ID INTEGER NOT NULL AUTO_INCREMENT ,  
USER_NAME VARCHAR(20) ,  
APT_NUMBER INTEGER ,  
STREET_NAME VARCHAR(50) ,  
STATE VARCHAR(25) ,  
CITY VARCHAR(25) ,  
PIN_CODE INTEGER ,  
PRIMARY KEY(ADDRESS_ID) ,  
FOREIGN KEY (USER_NAME) REFERENCES CMUNEGOW_TBL_USERS(USER_NAME));
```

```
CREATE TABLE CMUNEGOW_TBL_TICKET  
(TICKET_ID INTEGER NOT NULL AUTO_INCREMENT,  
TICKET_PRIORITY VARCHAR(10),  
TICKET_COMMENT VARCHAR(100),  
TICKET_CATEGORY VARCHAR(30),  
TICKET_STATUS VARCHAR(10),  
TICKET_DESCRIPTION VARCHAR(100),  
TICKET_OPEN_DATE DATE,  
TICKET_CLOSE_DATE DATE,  
TICKET_CREATEDBY VARCHAR(20),  
TICKET_ASSIGNEE VARCHAR(20),  
TICKET_ISPURGED CHAR(1),  
PRIMARY KEY(TICKET_ID),  
FOREIGN KEY(TICKET_ASSIGNEE) REFERENCES CMUNEGOW_TBL_USERS(USER_NAME),  
FOREIGN KEY(TICKET_CREATEDBY) REFERENCES CMUNEGOW_TBL_USERS(USER_NAME));
```


[Stored Procedures scripts](#)

Insert/Update procedures scripts:

```
CREATE PROCEDURE cmam_createUserorAdmin
(
  IN username VARCHAR(20),
  IN firstname VARCHAR(40),
  IN lastname VARCHAR(40),
  IN userpassword VARCHAR(30),
  IN useremail VARCHAR(50),
  IN phonenumber LONG,
  IN userrole VARCHAR(10)
)
BEGIN
  insert into CMUNEGOW_TBL_USERS
  (USER_NAME,FIRST_NAME, LAST_NAME ,
  USER_PASSWORD ,USER_EMAIL ,
  PHONE_NUMBER,USER_ROLE)
  VALUES
  (username,firstname,
  lastname,userpassword,
  useremail,onenumber,
  userrole);
END;

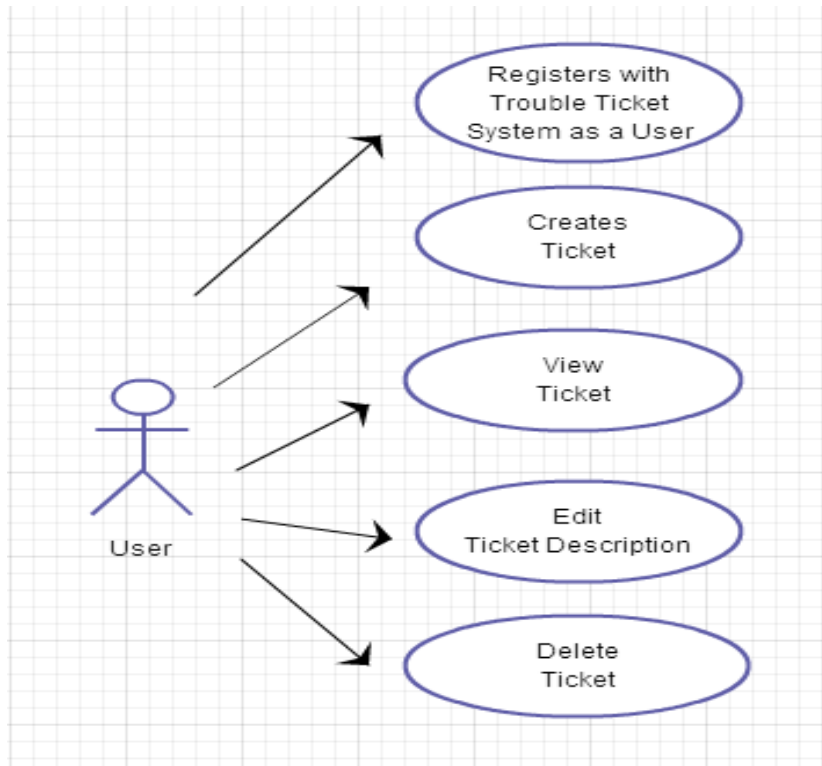
CREATE PROCEDURE cmam_addUserOrAdminAddress (
  username VARCHAR(20),
  aptnumber INTEGER,
  streetname VARCHAR(50),
  state VARCHAR(25),
  city VARCHAR(25),
  pincode INTEGER
)
BEGIN
  insert into CMUNEGOW_TBL_USERADDRESS
  (USER_NAME, APT_NUMBER, STREET_NAME,
  STATE, CITY,PIN_CODE)
  VALUES (username,aptnumber,streetname,
  state,city,pincode);
END;
```

```
CREATE PROCEDURE cmam_addTroubleTicket(  
  IN tpriority VARCHAR(10),  
  IN tComment VARCHAR(100),  
  IN tStatus VARCHAR(10),  
  IN tDiscription VARCHAR(100),  
  IN tCategory VARCHAR(30),  
  IN tOpenDate DATE,  
  IN tCloseDate DATE,  
  IN tCreatedBy VARCHAR(20),  
  IN tAssignee VARCHAR(20),  
  IN tPurgedBy CHAR(1))  
BEGIN  
  insert into CMUNEGOW_TBL_TICKET  
  (TICKET_PRIORITY,TICKET_COMMENT, TICKET_STATUS ,  
  TICKET_DESCRIPTION ,TICKET_CATEGORY,  
  TICKET_OPEN_DATE ,TICKET_CLOSE_DATE,TICKET_CREATEDBY,  
  TICKET_ASSIGNEE,TICKET_ISPURGED)  
  VALUES (tpriority,tComment,tStatus,tDiscription,  
  tCategory,tOpenDate,tCloseDate,tCreatedBy,tAssignee,  
  tPurgedBy);  
END;
```

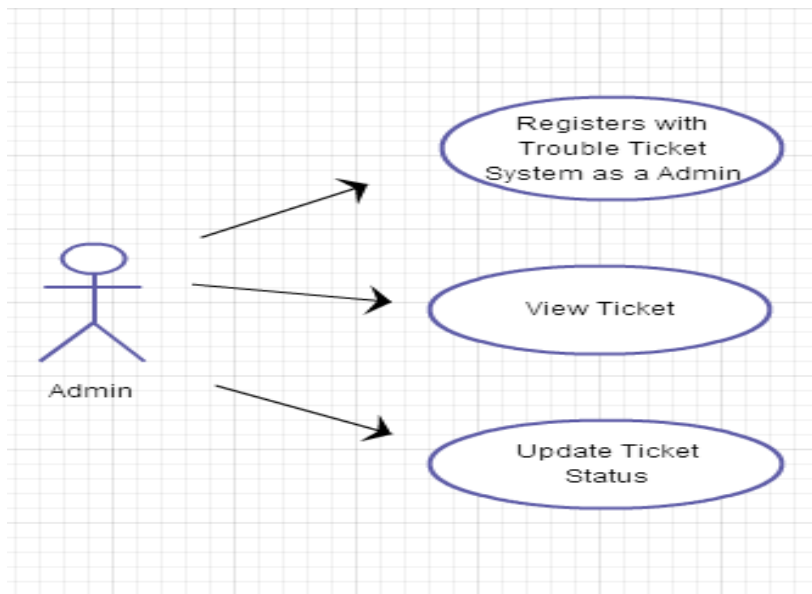
```
CREATE PROCEDURE cmam_updateTroubleTicket (  
  IN tpriority VARCHAR(10),  
  IN tComment VARCHAR(100),  
  IN tCategory VARCHAR(30),  
  IN tDiscription VARCHAR(100),  
  IN ticketId VARCHAR(10)  
  )  
BEGIN  
  Update CMUNEGOW_TBL_TICKET  
  SET TICKET_PRIORITY = tpriority,  
  TICKET_COMMENT = tComment,  
  TICKET_CATEGORY = tCategory,  
  TICKET_DESCRIPTION = tDiscription  
  WHERE TICKET_ID = ticketId;  
END;
```

Usecase Diagram:

User:



Admin:



Project source code with screen shots:

Login Code(Main File):

Allows the user or Admin to login to the IT Trouble Ticket System.

```
package com.iit.tts.loginpage;

import java.awt.Color;
import java.awt.EventQueue;
import java.awt.Font;
import java.awt.Image;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

import javax.swing.DefaultComboBoxModel;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JMenu;
import javax.swing.JMenuBar;
import javax.swing.JMenuItem;
import javax.swing.JOptionPane;
import javax.swing.JPasswordField;
import javax.swing.JTextField;

import com.iit.tts.db.CreateDatabase;
import com.iit.tts.db.SqlConnetion;
import com.iit.tts.troubletickethelper.Troubletickethelper;

/**
 * This Class makes the User to login to the User/Admin
 * to login to Application and later use the
 * trouble ticket management system. If the new
 * user then he can register to the trouble
 * ticket system application
 * @author Chetan Munegowda and Abhay Manoli
 */
public class Login {

    public JFrame mainFrameObj;
    private JTextField textFieldUserName;
    private CreateDatabase createDatabaseObj = null;
    private Connection dbConnection = null;
    private JPasswordField passwordFieldUsrPwd;
    private JComboBox RoleComboBox;
    String strRole = null;
    private StringBuffer strBuffer = null;
```

```
private String strValidationMsg = null;

/**
 * Launch the application.
 */
public static void main(String[] args) {
    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                Login window = new Login();
                window.mainFrameObj.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

/**
 * Create the application.
 */
public Login() {
    setUpDataBase();
    initialize();
}

/*
 * Creates the database tables for the
 * trouble ticket management system
 */
private void setUpDataBase() {

    try{
        dbConnection = SqlConnetion.dbConnector();
        createDatabaseObj = new CreateDatabase();
        createDatabaseObj.CreateTicketsDataBase();
    }
    catch(Exception e)
    {
        JOptionPane.showMessageDialog(null, e.getMessage());
    }
}

/**
 * Gets the user role
 * @return
 */
public String getUserRole()
{
    if (RoleComboBox.getSelectedIndex() != -1)
    {
        strRole = (String) RoleComboBox.getItemAt
            (RoleComboBox.getSelectedIndex());
    }
    return strRole;
}
```

```

/**
 * Initialize the contents of the frame.
 */
private void initialize() {
    mainFrameObj = new JFrame();
    mainFrameObj.getContentPane().setFont(new
        Font("Tahoma", Font.PLAIN, 20));
    mainFrameObj.setBounds(100, 100, 896, 486);
    mainFrameObj.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    mainFrameObj.getContentPane().setLayout(null);

    JMenuBar menuBarObj = new JMenuBar();
    JMenu fileObj = new JMenu("File");
    menuBarObj.add(fileObj);
    JMenuItem exitObj = new JMenuItem("Exit");
    fileObj.add(exitObj);
    exitObj.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent evt) {
            System.exit(0);
        }
    });

    JMenu helpObj = new JMenu("Help");
    menuBarObj.add(helpObj);
    JMenuItem aboutAppObj = new JMenuItem(
        "About Application");
    helpObj.add(aboutAppObj);
    aboutAppObj.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent evt) {
            JOptionPane.showMessageDialog(null,
                "1. This application helps the User/Admin to register
                + "to the Trouble Ticket System.\n"
                + "2. User can use this "
                + "application to create trouble tickets and
                monitor the "
                + "status of the ticket. \n"
                + "3. Admin can update the ticket remarks and
                + "close the ticket which are assigned "
                + "to him" ,
                "Developed By Chetan Munegowda and Abhay
                Manoli",
                JOptionPane.INFORMATION_MESSAGE);
        }
    });

    mainFrameObj.setJMenuBar(menuBarObj);
    mainFrameObj.getContentPane().setBackground(
        Color.getHSBColor(1.4f, 1.0f, 1.6f));

    JLabel lblTroubleTicketSystem = new JLabel("IT "
        + "Trouble Ticket System");

```

```
lblTroubleTicketSystem.setFont(new Font("Tahoma", Font.BOLD, 20));
lblTroubleTicketSystem.setBounds(392, 40, 286, 60);
mainFrameObj.getContentPane().add(lblTroubleTicketSystem);

textFieldUserName = new JTextField();
textFieldUserName.setFont(new Font("Tahoma", Font.BOLD, 11));
textFieldUserName.setBounds(526, 166, 200, 20);
mainFrameObj.getContentPane().add(textFieldUserName);
textFieldUserName.setColumns(10);

JLabel lblNewLabel = new JLabel("User Id");
lblNewLabel.setFont(new Font("Tahoma", Font.BOLD, 14));
lblNewLabel.setBounds(392, 155, 77, 42);
mainFrameObj.getContentPane().add(lblNewLabel);

JLabel lblNewLabel_1 = new JLabel("Password");
lblNewLabel_1.setFont(new Font("Tahoma", Font.BOLD, 14));
lblNewLabel_1.setBounds(392, 205, 68, 35);
mainFrameObj.getContentPane().add(lblNewLabel_1);

JButton btnLogin = new JButton("Login");
btnLogin.setFont(new Font("Tahoma", Font.BOLD, 12));
Image btnImg = new
ImageIcon(this.getClass().getResource("/ok.png")).getImage();
btnLogin.setIcon(new ImageIcon(btnImg));

/**
 * validates the fields authenticate whether the valid user
 * has logged in
 */
btnLogin.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent evt) {
        if(validateFields())
        {
            authenticateRecords();
        }
    }
});

/**
 * Validates the user name,password and user role
 * are entered
 * @return
 */
private boolean validateFields()
{
    strBuffer = new StringBuffer("");
    Boolean bValStatus = true;

    if(TroubleTicketHelper.validateTextField
        (textFieldUserName.getText()))
    {
        strValidationMsg = "Please enter the User Name!";
        strBuffer.append(strValidationMsg+"\n");
        bValStatus = false;
    }
}
```

```

        if(Troubletickethelper.validateTextField(
            passwordFieldUsrPwd.getText()))
        {
            strValidationMsg = "Please enter the Password!";
            strBuffer.append(strValidationMsg+"\n");
            bValStatus = false;
        }

        if(Troubletickethelper.validateDropDown(
            (String) RoleComboBox.getSelectedItem()))
        {
            strValidationMsg = "Please select the User Role!";
            strBuffer.append(strValidationMsg+"\n");
            bValStatus = false;
        }

        if(!(strBuffer.toString().equals(""))){
            JOptionPane.showMessageDialog(null, strBuffer);
        }
        return bValStatus;
    }

    /**
     * Validates user name, user password and user role matches
     * authenticates
     */
    private void authenticateRecords()
    {
        String sql = "SELECT * FROM CMUNEGOW_TBL_USERS WHERE
USER_NAME = ? "
            + "AND USER_PASSWORD = ? "
            + "AND USER_ROLE= ?";
        PreparedStatement prepStmt = null;
        try
        {
            prepStmt = dbConnection.prepareStatement(sql);
            prepStmt.setString(1, textFieldUserName.getText());
            prepStmt.setString(2, passwordFieldUsrPwd.getText());
            prepStmt.setString(3,
                (String)RoleComboBox.getSelectedItem());
            ResultSet rs = prepStmt.executeQuery();
            boolean bUserStatus = false;
            while(rs.next())
            {
                bUserStatus = true;
            }
            if(bUserStatus)
            {
                strRole = getUserRole();
                if(strRole.equalsIgnoreCase("USER"))
                {
                    mainFrameObj.dispose();
                    Troubleticketlandingpage ttlp = new
Troubleticketlandingpage(textFieldUserName.getText());
                    ttlp.setVisible(true);
                }
            }
        }
    }

```



```
        else if(strRole.equalsIgnoreCase("ADMIN"))
        {
            mainFrameObj.dispose();
            Adminlandingpage adlp = new
Adminlandingpage(textFieldUserName.getText());
            adlp.setVisible(true);
        }
    }
    else
    {
        JOptionPane.showMessageDialog(null,
"Username, Password and User Role pair not found."
        + "Please try Again!");
    }
}
catch(SQLException e){
    JOptionPane.showMessageDialog(null,e.getMessage());
}
}
});
btnLogin.setBounds(411, 318, 116, 35);
mainFrameObj.getContentPane().add(btnLogin);

passwordFieldUsrPwd = new JPasswordField();
passwordFieldUsrPwd.setBounds(526, 212, 200, 20);
mainFrameObj.getContentPane().add(passwordFieldUsrPwd);

JLabel lblIcon = new JLabel("");
Image img = new ImageIcon(this.getClass().getResource
("/login.png")).getImage();
lblIcon.setIcon(new ImageIcon(img));
lblIcon.setBounds(36, 88, 284, 249);
mainFrameObj.getContentPane().add(lblIcon);

JLabel lblRole = new JLabel("Role");
lblRole.setFont(new Font("Tahoma", Font.BOLD, 14));
lblRole.setBounds(392, 260, 68, 32);
mainFrameObj.getContentPane().add(lblRole);

final DefaultComboBoxModel roleModel = new DefaultComboBoxModel();

roleModel.addElement("SELECT");
roleModel.addElement("USER");
roleModel.addElement("ADMIN");

RoleComboBox = new JComboBox(roleModel);
RoleComboBox.setSelectedIndex(0);
RoleComboBox.setBounds(526, 260, 200, 32);
mainFrameObj.getContentPane().add(RoleComboBox);

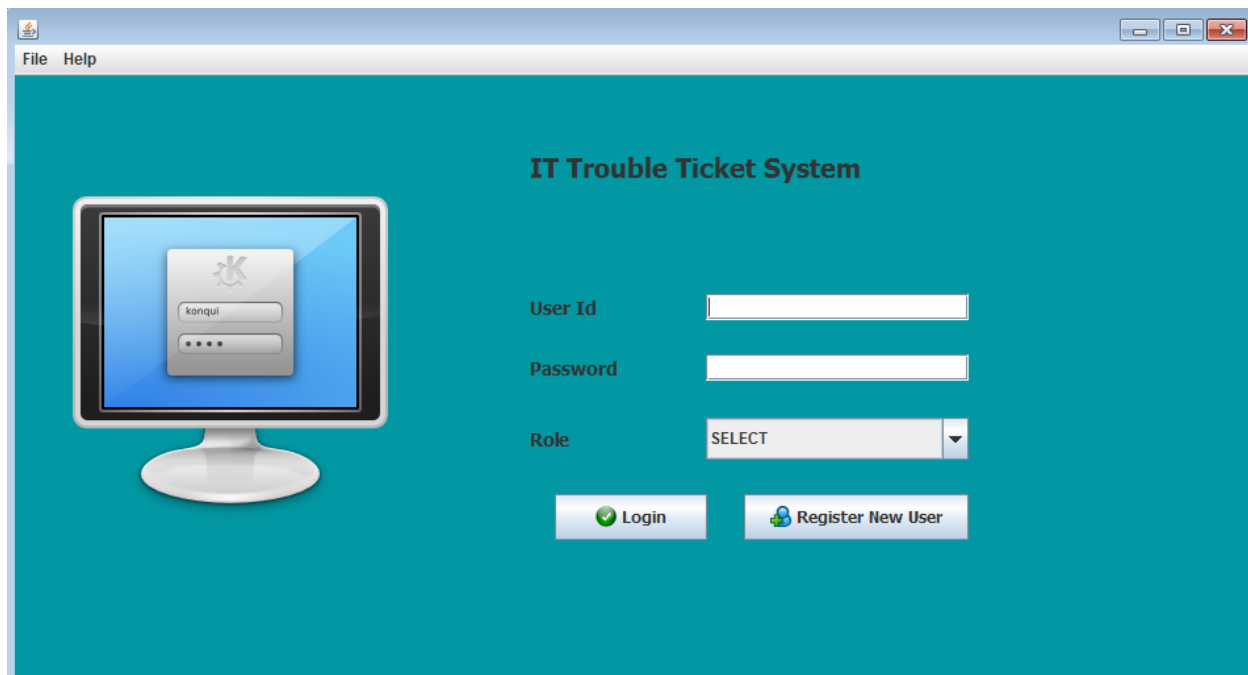
JButton btnNewButton = new JButton("Register New User");
Image btnImg1 = new
ImageIcon(this.getClass().getResource("/regnewuser.png")).getImage();
btnNewButton.setIcon(new ImageIcon(btnImg1));
btnNewButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
```

```

        mainFrameObj.dispose();
        CreateUserOrAdmin createUserAdminObj = new CreateUserOrAdmin();
        createUserAdminObj.setVisible(true);
    }
});
btnNewButton.setFont(new Font("Tahoma", Font.BOLD, 12));
btnNewButton.setBounds(555, 318, 171, 35);
mainFrameObj.getContentPane().add(btnNewButton);
}
}

```

Login Screen Shot:



Create User/Admin Java File:

Creates the un-registered user/admin to register to the IT Trouble Ticket System.

```

package com.iit.tts.loginpage;

import java.awt.Color;
import java.awt.EventQueue;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;
import java.sql.CallableStatement;
import java.sql.Connection;

```

```

import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Enumuration;

import javax.swing.JButton;
import javax.swing.ButtonGroup;
import javax.swing.DefaultComboBoxModel;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JPasswordField;
import javax.swing.JRadioButton;
import javax.swing.JTextField;
import javax.swing.border.EmptyBorder;

import com.iit.tts.db.SqlConnetion;
import com.iit.tts.troubleshootertickethelper.IntegerField;
import com.iit.tts.troubleshootertickethelper.JTextFieldLimit;
import com.iit.tts.troubleshootertickethelper.Troubleshootertickethelper;

import java.sql.PreparedStatement;

/*****
 * Class which allows People to register as a
 * User and Admin to IT Trouble Ticket System
 *
 * @author Chetan Munegowda and Abhay Manoli
 *
 *****/
public class CreateUserOrAdmin extends JFrame {

    private JPanel createUserOrAdminContetPane;
    private JTextField textField_UserName;
    private JTextField textField_FirstName;
    private JTextField textField_LastName;
    private JTextField textField_Email;
    private IntegerField textField_CellularNo;
    private JTextField textField_HouseNo;
    private JTextField textField_StreetName;
    private JTextField textField_Pincode;
    private Connection connectionObj = null;
    private static int count = 10;
    private JTextField textField_City;
    CallableStatement callStObj = null;
    StringBuffer strBuffer = null;
    String strValidationMessage = null;
    private JPasswordField textField_Password;

    /**
     * Launch the application.
     */
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {

```

```

        public void run() {
            try {
                CreateUserOrAmin frame = new CreateUserOrAmin();
                frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

/*****
 * Create the frame. Adds Necessary field to create the user
 * and admin
 * @param usrRole
 *****/
public CreateUserOrAmin() {

    Login loginObj = new Login();
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(100, 100, 741, 479);
    createUserOrAdminContetPane = new JPanel();
    createUserOrAdminContetPane.setBorder(new EmptyBorder(5, 5, 5, 5));
    createUserOrAdminContetPane.setBackground(Color.getHSBColor(1.4f,
1.0f, 1.6f));
    setContentPane(createUserOrAdminContetPane);
    createUserOrAdminContetPane.setLayout(null);

    JLabel label = new JLabel("");
    label.setBounds(5, 5, 424, 0);
    createUserOrAdminContetPane.add(label);

    JLabel lblCreateUseramin = new JLabel("Create User/Amin");
    lblCreateUseramin.setFont(new Font("Tahoma", Font.BOLD, 19));
    lblCreateUseramin.setBounds(265, 5, 199, 36);
    createUserOrAdminContetPane.add(lblCreateUseramin);

    JLabel lblUserNameFld = new JLabel("User Name");
    lblUserNameFld.setFont(new Font("Tahoma", Font.BOLD, 13));
    lblUserNameFld.setBounds(56, 69, 81, 17);
    createUserOrAdminContetPane.add(lblUserNameFld);

    textField_UserName = new JTextField();
    textField_UserName.setBounds(165, 69, 132, 20);
    textField_UserName.setDocument(new JTextFieldLimit(20));
    createUserOrAdminContetPane.add(textField_UserName);
    textField_UserName.setColumns(10);

    JLabel lblFirstNameFld = new JLabel("First Name");
    lblFirstNameFld.setFont(new Font("Tahoma", Font.BOLD, 13));
    lblFirstNameFld.setBounds(357, 71, 72, 15);
    createUserOrAdminContetPane.add(lblFirstNameFld);

    textField_FirstName = new JTextField();
    textField_FirstName.setBounds(503, 68, 137, 20);
    textField_FirstName.setDocument(new JTextFieldLimit(40));

```

```
createUserOrAdminContetPane.add(textField_FirstName);
textField_FirstName.setColumns(10);

JLabel lblLastNameFld = new JLabel("Last Name");
lblLastNameFld.setFont(new Font("Tahoma", Font.BOLD, 13));
lblLastNameFld.setBounds(56, 107, 81, 17);
createUserOrAdminContetPane.add(lblLastNameFld);

textField_LastName = new JTextField();
textField_LastName.setBounds(165, 105, 132, 20);
textField_LastName.setDocument(new JTextFieldLimit(40));
createUserOrAdminContetPane.add(textField_LastName);
textField_LastName.setColumns(10);

JLabel lblPassword = new JLabel("Password");
lblPassword.setFont(new Font("Tahoma", Font.BOLD, 13));
lblPassword.setBounds(56, 146, 81, 14);
createUserOrAdminContetPane.add(lblPassword);

JLabel lblEmailFld = new JLabel("Email");
lblEmailFld.setFont(new Font("Tahoma", Font.BOLD, 13));
lblEmailFld.setBounds(357, 149, 60, 14);
createUserOrAdminContetPane.add(lblEmailFld);

textField_Email = new JTextField();
textField_Email.setBounds(503, 146, 137, 20);
textField_Email.setDocument(new JTextFieldLimit(50));
createUserOrAdminContetPane.add(textField_Email);
textField_Email.setColumns(10);

JLabel lblRoleFld = new JLabel("Role");
lblRoleFld.setFont(new Font("Tahoma", Font.BOLD, 13));
lblRoleFld.setBounds(56, 183, 60, 14);
createUserOrAdminContetPane.add(lblRoleFld);

JLabel lblHouseNumberFld = new JLabel("House Number");
lblHouseNumberFld.setFont(new Font("Tahoma", Font.BOLD, 13));
lblHouseNumberFld.setBounds(357, 184, 96, 14);
createUserOrAdminContetPane.add(lblHouseNumberFld);

JLabel lblCellularNumberFld = new JLabel("Cellular Number");
lblCellularNumberFld.setFont(new Font("Tahoma", Font.BOLD, 13));
lblCellularNumberFld.setBounds(357, 108, 107, 14);
createUserOrAdminContetPane.add(lblCellularNumberFld);

textField_CellularNo = new IntegerField();
textField_CellularNo.addKeyListener(new KeyAdapter() {
    @Override
    public void keyPressed(KeyEvent evt)
    {
        String value = textField_CellularNo.getText();
        int l = value.length();
        if (evt.getKeyChar() >= '0' && evt.getKeyChar() <= '9') {
            textField_CellularNo.setEditable(true);
        }
        else {
            textField_CellularNo.setEditable(true);
        }
    }
});
```

```

JOptionPane.showMessageDialog(null, "* Enter only numeric
digits(0-9)");
        textField_CellularNo.setText("");
    }
}
});
textField_CellularNo.setBounds(503, 105, 137, 20);
textField_CellularNo.setDocument(new JTextFieldsLimit(10));
createUserOrAdminContetPane.add(textField_CellularNo);
textField_CellularNo.setColumns(10);

textField_HouseNo = new JTextFields();
textField_HouseNo.addKeyListener(new KeyAdapter() {
    @Override
    public void keyPressed(KeyEvent evt) {
        String value = textField_HouseNo.getText();
        int l = value.length();
        if (evt.getKeyChar() >= '0' && evt.getKeyChar() <= '9') {
            textField_HouseNo.setEditable(true);
        }
        else {
            textField_HouseNo.setEditable(true);
            JOptionPane.showMessageDialog(null, "* Enter only numeric
digits(0-9)");
            textField_HouseNo.setText("");
        }
    }
});
textField_HouseNo.setBounds(503, 181, 137, 20);
createUserOrAdminContetPane.add(textField_HouseNo);
textField_HouseNo.setColumns(10);

JLabel lblStreetNameFld = new JLabel("Street Name");
lblStreetNameFld.setFont(new Font("Tahoma", Font.BOLD, 13));
lblStreetNameFld.setBounds(56, 216, 96, 14);
createUserOrAdminContetPane.add(lblStreetNameFld);

JLabel lblCityFld = new JLabel("City");
lblCityFld.setFont(new Font("Tahoma", Font.BOLD, 13));
lblCityFld.setBounds(357, 256, 46, 14);
createUserOrAdminContetPane.add(lblCityFld);

JLabel lblStateFld = new JLabel("State");
lblStateFld.setFont(new Font("Tahoma", Font.BOLD, 13));
lblStateFld.setBounds(357, 216, 46, 14);
createUserOrAdminContetPane.add(lblStateFld);
final DefaultComboBoxModel cityModel = new DefaultComboBoxModel();

/*
 * Contains the state name where this trouble ticket system is
 * supported
 */
final DefaultComboBoxModel stateModel = new DefaultComboBoxModel(
    new String[]{"Select","Alabama","Arizona","California",
"Colorado","Florida","Iowa","Illinois","Kansas","Maryland","Missouri",

```

```

        "Nebraska","Nevada","New Jersey","New York","Ohio",
        "Okhlahoma","Pennsylvania","Texas","Virginia","Wisconsin"});
        JComboBox comboBox_State = new JComboBox(stateModel);

        comboBox_State.setBounds(503, 210, 137, 26);

        createUserOrAdminContetPane.add(comboBox_State);

        textField_StreetName = new JTextField();
        textField_StreetName.setBounds(165, 214, 132, 20);
        textField_StreetName.setDocument(new JTextFieldLimit(25));
        createUserOrAdminContetPane.add(textField_StreetName);
        textField_StreetName.setColumns(10);

        ButtonGroup rolebtnGroup = new ButtonGroup();
        JRadioButton rdbtnUser = new JRadioButton("User");
        rdbtnUser.setFont(new Font("Tahoma", Font.BOLD, 12));
        rdbtnUser.setBounds(165, 179, 60, 23);
        rolebtnGroup.add(rdbtnUser);
        createUserOrAdminContetPane.add(rdbtnUser);

        JRadioButton rdbtnAdmin = new JRadioButton("Admin");
        rdbtnAdmin.setFont(new Font("Tahoma", Font.BOLD, 12));
        rdbtnAdmin.setBounds(227, 179, 70, 23);
        rolebtnGroup.add(rdbtnAdmin);
        createUserOrAdminContetPane.add(rdbtnAdmin);

        JLabel lblGenderFld = new JLabel("Pincode");
        lblGenderFld.setFont(new Font("Tahoma", Font.BOLD, 13));
        lblGenderFld.setBounds(56, 256, 60, 14);
        createUserOrAdminContetPane.add(lblGenderFld);

        JButton btnSubmt = new JButton("Submit");
        btnSubmt.setFont(new Font("Tahoma", Font.BOLD, 14));

        /*****
        * Action Listener event for the Submit Button, which
        * inserts the records to the database
        *****/
        btnSubmt.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent arg0) {

                if(validateFieldSatus()){

                    insertRecordsIntoUser();

                }

            }

        /****
        * Validates the Mandatory fields and throws validation
        * message if the field is empty
        * @return

```

```
*****/
private boolean validateFieldStatus() {

    strBuffer = new StringBuffer("");
    strValidationMessage = new String();
    Boolean bBoolFldStatus = true;

    if(Troubletickethelper.validateTextField(textField_UserName.getText()))
    {
        strValidationMessage = "Please provide User Name!";
        strBuffer.append(strValidationMessage+"\n");
        bBoolFldStatus = false;
    }

    if(Troubletickethelper.validateTextField(textField_FirstName.getText()))
    {
        strValidationMessage = "Please provide First Name!";
        strBuffer.append(strValidationMessage+"\n");
        bBoolFldStatus = false;
    }

    if(Troubletickethelper.validateTextField(textField_LastName.getText()))
    {
        strValidationMessage = "Please provide Last Name!";
        strBuffer.append(strValidationMessage+"\n");
        bBoolFldStatus = false;
    }

    if(Troubletickethelper.validateTextField(getSelectedButtonText(rolebtnGroup))
    )
    {
        strValidationMessage = "Please provide Role!";
        strBuffer.append(strValidationMessage+"\n");
        bBoolFldStatus = false;
    }

    if(Troubletickethelper.validateTextField(textField_Email.getText()))
    {
        strValidationMessage = "Please provide Email Address!";
        strBuffer.append(strValidationMessage+"\n");
        bBoolFldStatus = false;
    }

    if(!Troubletickethelper.validateEmail(textField_Email.getText()))
    {
        strValidationMessage = "Please provide valid E-mail
Address!";

        strBuffer.append(strValidationMessage+"\n");
        bBoolFldStatus = false;
    }
}
```



```

    }

    if(Troubletickethelper.validateTextField(textField_City.getText()))
    {
        strValidationMessage = "Please provide City!";
        strBuffer.append(strValidationMessage+"\n");
        bBoolFldStatus = false;
    }

    if(Troubletickethelper.validateTextField(textField_City.getText()))
    {
        strValidationMessage = "Please provide Street Name!";
        strBuffer.append(strValidationMessage+"\n");
        bBoolFldStatus = false;
    }

    if(Troubletickethelper.validateDropDown((String)comboBox_State.getSelectedItem()))
    {
        strValidationMessage = "Please select proper State!";
        strBuffer.append(strValidationMessage+"\n");
        bBoolFldStatus = false;
    }

    if(checkForUserName(textField_UserName.getText()))
    {
        strValidationMessage = "User Name already exists. "
            + "Please enter another user name!";
        strBuffer.append(strValidationMessage+"\n");
        bBoolFldStatus = false;
    }

    if(!(strBuffer.toString().equals(""))
    {
        JOptionPane.showMessageDialog(null, strBuffer);
    }

    return bBoolFldStatus;
}

private boolean checkForUserName(String strUserName) {
    boolean bStatus = false;
    try
    {
        connectionObj = SqlConnection.dbConnector();
        String sqlString = "SELECT * FROM CMUNEGOW_TBL_USERS
WHERE "
            + "USER_NAME = ?";

        PreparedStatement pStatementObj =
connectionObj.prepareStatement(sqlString);
        pStatementObj.setString(1, strUserName);

```

```

        ResultSet rs = pStatementObj.executeQuery();

        while(rs.next())
        {
            bStatus = true;
        }
    }
    catch (SQLException e) {
        JOptionPane.showMessageDialog(null, e.getMessage());
    }

    return bStatus;
}

/*****
 * Calls the stored procedure to insert the records to the
 * database
 *****/
private void insertRecordsIntoUser() {
    try{
        connectionObj = SqlConnetion.dbConnector();
        connectionObj.setAutoCommit(false);

        String sqlInsertUserProcedure = "{ "
            + "call cmam_createUserorAdmin(?,?,?,?,?,?,?) "
            + "}";

        callStObj = connectionObj.prepareCall(
            sqlInsertUserProcedure);

        callStObj.setString(1,
textField_UserName.getText().toUpperCase());
        callStObj.setString(2, textField_FirstName.getText());
        callStObj.setString(3, textField_LastName.getText());
        callStObj.setString(4, textField_Password.getText());
        callStObj.setString(5, textField_Email.getText());

        callStObj.setLong(6,
Long.parseLong(textField_CellularNo.getText().trim()));

        callStObj.setString(7,
getSelectedButtonText(rolebtnGroup).toUpperCase());
        callStObj.execute();

        String sqlInsertUserAddress = "{ "
            + "call cmam_addUserOrAdminAddress(?,?,?,?,?,?) "
            + "}";

        callStObj = connectionObj.prepareCall(
            sqlInsertUserAddress);

        Integer iHNumber =
parseInteger(textField_HouseNo.getText());
        callStObj.setString(1, textField_UserName.getText());

```

```

        callStObj.setInt(2, iHNumber);
        callStObj.setString(3, textField_StreetName.getText());
        callStObj.setString(4, (String)
comboBox_State.getSelectedItem());
        callStObj.setString(5, textField_City.getText());

callStObj.setInt(6,parseInteger(textField_Pincode.getText()));
        callStObj.execute();
        connectionObj.commit();

if(getSelectedButtonText(rolebtnGroup).equalsIgnoreCase("USER"))
    {
        JOptionPane.showMessageDialog(null, "User "
+textField_FirstName.getText()
        +" successfully registered!");
    }
    else
    {
        JOptionPane.showMessageDialog(null, "Admin
"+textField_FirstName.getText()
        +" successfully registered!");
    }

    loginObj.mainFrameObj.setVisible(true);
}
catch (SQLException e ) {

    if (connectionObj != null) {
        try {
            JOptionPane.showMessageDialog(null, "Transaction
is being rolled back");
            connectionObj.rollback();
            JOptionPane.showMessageDialog(null,
e.getMessage());

            callStObj.close();

        } catch (SQLException excep) {
            System.out.println(excep.getMessage());
        }
    }
}
catch (Exception e) {
    System.out.println(e.getMessage());
}
}

/*****
 * Parses the input text to the integer
 * @param s
 * @return
 *****/
private Integer parseInteger(String s)
{
    if(s == null || s.isEmpty())
        return 0;
    else
        return Integer.parseInt(s);
}

```

```

    }

});
btnSubmt.setBounds(235, 349, 107, 26);
createUserOrAdminContetPane.add(btnSubmt);

JButton btnCancel = new JButton("Cancel");
btnCancel.setFont(new Font("Tahoma", Font.BOLD, 14));
btnCancel.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        createUserOrAdminContetPane.invalidate();
        loginObj.mainFrameObj.setVisible(true);
    }
});
btnCancel.setBounds(356, 349, 108, 26);
createUserOrAdminContetPane.add(btnCancel);

textField_Pincode = new JTextField();
/**
 * Action Listener which listens and
 * of characters entered are only digits from
 * 0 to 9
 */
textField_Pincode.addKeyListener(new KeyAdapter() {
    @Override
    public void keyPressed(KeyEvent evt) {
        String value = textField_Pincode.getText();
        int l = value.length();
        if (evt.getKeyChar() >= '0' && evt.getKeyChar() <= '9') {
            textField_Pincode.setEditable(true);
        }
        else {
            textField_Pincode.setEditable(true);
            JOptionPane.showMessageDialog(null, "* Enter only numeric
digits(0-9)");
            textField_Pincode.setText("");
        }
    }
});
textField_Pincode.setBounds(165, 253, 132, 20);
createUserOrAdminContetPane.add(textField_Pincode);
textField_Pincode.setColumns(10);

textField_City = new JTextField();
textField_City.setDocument(new JTextFieldLimit(25));
textField_City.setBounds(503, 253, 137, 20);
createUserOrAdminContetPane.add(textField_City);
textField_City.setColumns(10);

textField_Password = new JPasswordField();
textField_Password.setBounds(165, 144, 132, 20);
createUserOrAdminContetPane.add(textField_Password);
}

/*****

```

```
* Method which gets the text from the selected radio button
* @param rolebtnGroup
* @return

*****/
private String getSelectedButtonText(ButtonGroup rolebtnGroup) {

    for (Enumeration<AbstractButton> buttons =
rolebtnGroup.getElements(); buttons.hasMoreElements();) {
        AbstractButton button = buttons.nextElement();

        if (button.isSelected()) {
            return button.getText();
        }
    }
    return null;
}
}
```

Create Admin/User ScreenShot:

Create User/Amin

User Name	<input type="text"/>	First Name	<input type="text"/>
Last Name	<input type="text"/>	Cellular Number	<input type="text"/>
Password	<input type="text"/>	Email	<input type="text"/>
Role	<input type="radio"/> User <input type="radio"/> Admin	House Number	<input type="text"/>
Street Name	<input type="text"/>	State	Select ▼
Pincode	<input type="text"/>	City	<input type="text"/>

The screenshot shows a web application window titled "Create User/Amin". The form contains the following fields and values:

Field	Value
User Name	abhay
First Name	Abhay
Last Name	Manoli
Cellular Number	7125387129
Password
Role	<input checked="" type="radio"/> Admin
Street Name	Cottage Grove Avenue
Pincode	60616
City	Chicago

A modal message box is displayed in the center, stating: "Admin Abhay successfully registered!". The message box has an "OK" button.

At the bottom of the form, there are two buttons: "Submit" and "Cancel".

The screenshot shows the same "Create User/Amin" web application window. The form contains the following fields and values:

Field	Value
User Name	luke
First Name	Luke
Last Name	Papademas
Cellular Number	23875198
Password
Role	<input checked="" type="radio"/> User
Street Name	Halsted Street
Pincode	60615
City	Chicago
State	Illinois

A modal message box is displayed in the center, stating: "User Luke successfully registered!". The message box has an "OK" button.

At the bottom of the form, there are two buttons: "Submit" and "Cancel".

Admin Landing Page:

When the admin successfully login to IT Trouble Ticket System, admin can view all the allocated tickets under his name. He can view, update, refresh the ticket information or logout from the application.

```
package com.iit.tts.loginpage;

import java.awt.Color;
import java.awt.EventQueue;
import java.awt.Font;
import java.awt.Image;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.border.EmptyBorder;
import javax.swing.table.DefaultTableModel;

import net.proteanit.sql.DbUtils;

import com.iit.tts.db.SqlConnetion;
/**
 * This class allows Admin the to view all
 * tickets allocated to him, so he can view,
 * update the tickets.
 * @author Chetan Munegowda and Abhay Manoli
 */
public class Adminlandingpage extends JFrame {

    private JPanel contentPane;
    private JTable tableObjct;
    private JScrollPane scrollPaneObj;
    private JButton btn_View;
    private static String strUserNme = null;
    private static Connection connectionObj;
    private String ticket_Id = "";
    private String strAdminFlg ;
    private JButton btnLogOut;

    /**
     * Launch the application.
     */
}
```

```

    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Adminlandingpage frame = new
Adminlandingpage(strUserNme);
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

/**
 * Create the frame.
 * @param string
 */
public Adminlandingpage(String strUserName) {
    setUserName(strUserName);
    connectionObj = SqlConnection.dbConnector();
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(100, 100, 1016, 685);
    contentPane = new JPanel();
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
    contentPane.setBackground(Color.getHSBColor(1.4f, 1.0f, 1.6f));
    setContentPane(contentPane);
    contentPane.setLayout(null);

    JLabel lblAdminLandgPage = new JLabel("Hello " +strUserNme +"!
Welcome "
        + "To Trouble Ticket Admin Landing Page");
    lblAdminLandgPage.setFont(new Font("Tahoma", Font.BOLD, 14));
    lblAdminLandgPage.setBounds(295, 11, 479, 27);
    contentPane.add(lblAdminLandgPage);

    scrollPaneObj = new JScrollPane();

    scrollPaneObj.setBounds(32, 62, 943, 394);
    contentPane.add(scrollPaneObj);

/**
 * Action handler for mouse clicked event,
 * when user clicks on the row, it selects the
 * ticket id, which is used by the button to
 * identify any rows are selected
 */
    tableObjct = new JTable();
    tableObjct.addMouseListener(new MouseAdapter() {

        @Override
        public void mouseClicked(MouseEvent evt) {
            try
            {
                int row = tableObjct.getSelectedRow();
                ticket_Id = (tableObjct.getModel().getValueAt(row,
0)).toString();

```



```

    }
    catch(Exception e){
        JOptionPane.showMessageDialog(null, e.getMessage());
    }

}

});

scrollPaneObj.setViewportView(tableObjct);

//instance table model
DefaultTableModel tableModel = new DefaultTableModel() {

    @Override
    public boolean isCellEditable(int row, int column) {
        //all cells false
        return false;
    }
};

tableObjct.setModel(tableModel);

displayJTable();

btn_View = new JButton("View");
btn_View.setFont(new Font("Tahoma", Font.BOLD, 14));
Image viewImg = new
ImageIcon(this.getClass().getResource("/view.png")).getImage();
btn_View.setIcon(new ImageIcon(viewImg));

/*Action listener for the view button,
 * when the admin clicks on the view button
 * without selecting any record then alert to select
 * any record other wise display the records selected
 */

btn_View.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        if(ticket_Id.equalsIgnoreCase(""))
        {
            JOptionPane.showMessageDialog(null, "Please select a
row");
        }
        else
        {
            strAdminFlg = "VIEW";
            contentPane.setVisible(false);
            Adminupdate adminUpdate = new
Adminupdate(ticket_Id,strAdminFlg,strUserNme);
            adminUpdate.setVisible(true);
        }
    }
});

btn_View.setBounds(270, 493, 107, 23);

```

```

contentPane.add(btn_View);

/*
 * If the user clicks the update button without
 * selecting any record then alert the user to select record
 * other display the update page
 */
JButton btn_Update = new JButton("Update");
btn_Update.setFont(new Font("Tahoma", Font.BOLD, 13));
Image updateImg = new
ImageIcon(this.getClass().getResource("/update.png")).getImage();
btn_Update.setIcon(new ImageIcon(updateImg));
btn_Update.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0)
    {
        if(ticket_Id.equalsIgnoreCase(""))
        {
            JOptionPane.showMessageDialog(null, "Please select a
row");
        }
        else
        {
            strAdminFlg = "UPDATE";
            contentPane.setVisible(false);
            Adminupdate adminUpdate = new
Adminupdate(ticket_Id,strAdminFlg,strUserNme);
            adminUpdate.setVisible(true);
        }
    }
});
btn_Update.setFont(new Font("Tahoma", Font.BOLD, 14));
btn_Update.setBounds(401, 493, 118, 23);
contentPane.add(btn_Update);

//Refresh button which is used for the user to refresh the page
//each and evry time he updates status.Our application will not
support
//auto refresh
JButton btn_Refresh = new JButton("Refresh");
Image refreshImg = new
ImageIcon(this.getClass().getResource("/refresh.png")).getImage();
btn_Refresh.setIcon(new ImageIcon(refreshImg));
btn_Refresh.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        try{
            String query = "SELECT
TICKET_ID,TICKET_PRIORITY,TICKET_COMMENT,"
+
"TICKET_CATEGORY,TICKET_STATUS,TICKET_DESCRIPTION,"
+ "TICKET_CREATEDBY,TICKET_OPEN_DATE "
+ " FROM CMUNEGOW_TBL_TICKET WHERE "
+ "TICKET_ASSIGNEE= ? "
+ "AND TICKET_ISPURGED = ?";

            PreparedStatement pstmtObj =
connectionObj.prepareStatement(query);
            pstmtObj.setString(1, strUserNme);

```

```

        pstmtObj.setString(2, "N");
        ResultSet rs = pstmtObj.executeQuery();
        tableObjct.setModel(DbUtils.resultSetToTableModel(rs));
    }
    catch(Exception e)
    {
        System.out.println(e.getMessage());
    }
}

});
btn_Refresh.setFont(new Font("Tahoma", Font.BOLD, 12));
btn_Refresh.setBounds(549, 493, 112, 25);
contentPane.add(btn_Refresh);

btnLogOut = new JButton("Log Out");
Image logoutImg = new
ImageIcon(this.getClass().getResource("/logout.png")).getImage();
btnLogOut.setIcon(new ImageIcon(logoutImg));

btnLogOut.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent evt) {
        Login loginPageObj = new Login();
        loginPageObj.mainFrameObj.setVisible(true);
        contentPane.setVisible(false);
    }
});
btnLogOut.setBounds(857, 15, 118, 23);
contentPane.add(btnLogOut);
}

//Query to displays the records in the landing page
//and displays all the rcords to the user.
private void displayJTable() {

    try{
        String query = "SELECT TICKET_ID,TICKET_PRIORITY,TICKET_COMMENT,"
            + "TICKET_CATEGORY,TICKET_STATUS,TICKET_DESCRIPTION,"
            + "TICKET_CREATEDBY,TICKET_OPEN_DATE "
            + " FROM CMUNEGOW_TBL_TICKET WHERE "
            + "TICKET_ASSIGNEE= ? "
            + "AND TICKET_ISPURGED = ?";

        PreparedStatement pstmtObj =
connectionObj.prepareStatement(query);
        pstmtObj.setString(1, strUserNme);
        pstmtObj.setString(2, "N");
        ResultSet rs = pstmtObj.executeQuery();
        tableObjct.setModel(DbUtils.resultSetToTableModel(rs));
    }
    catch(Exception e)
    {
        System.out.println(e.getMessage());
    }
}

/*
 * Methods which sets the String user name

```

```

    */
    private void setUsername(String strUserName) {
        this.strUserNme = strUserName;
    }
}

```

Admin Landing Page Screen Shot:

Hello dipen! Welcome To Trouble Ticket Admin Landing Page Log Out

TICKET_ID	TICKET_PRIORITY	TICKET_CATEGORY	TICKET_STATUS	TICKET_DESCRIPTION	TICKET_CREATEDBY	TICKET_OPEN_DATE
1	MEDIUM	OS ISSUE	OPEN	please resolve the iss...	chetan	2014-11-29
2	LOW	NETWORKING ISSUE	OPEN	Wifi Notworking	chetan	2014-11-29
3	MEDIUM	HARDWARE ISSUE	OPEN	Motherboard Issue.	chetan	2014-11-29
4	MEDIUM	SOFTWARE INSTALL	OPEN	Install Visual Studio fo...	chetan	2014-11-29
5	LOW	INTENET CONNECTI...	OPEN	Wifi Not working.	chetan	2014-11-29
6	HIGH	LOGIN ISSUE	OPEN	Not able to login to my...	chetan	2014-11-29
7	HIGH	COMPUTER ALLOCA...	OPEN	Allocate Computer.	chetan	2014-11-29
8	LOW	COMPUTER RESTART ...	OPEN	Computer restarts aut...	chetan	2014-11-29
9	MEDIUM	OTHERS	OPEN	Need the credentials f...	chetan	2014-11-29
10	MEDIUM	NETWORKING ISSUE	OPEN	Slow Internet Speed.	chetan	2014-11-29
11	LOW	SOFTWARE INSTALL	OPEN	Install Notepad ++.	chetan	2014-11-29

View Update Refresh

User Landing Page:

When the user successfully logs in to IT Trouble Ticket System, user can view all the tickets created by him. He can view,update,purge,refresh the ticket information or logout from the application.

```
package com.iit.tts.loginpage;

import java.awt.Color;
import java.awt.EventQueue;
import java.awt.Font;
import java.awt.Image;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

import javax.swing.DefaultComboBoxModel;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.border.EmptyBorder;
import javax.swing.table.DefaultTableModel;

import net.proteanit.sql.DbUtils;

import com.iit.tts.db.SqlConnetion;

/**
 * Landing page for the user trouble ticketing system,
 * where user can view the details of the tickets
 * created by him and also naviagate to add,edit,
 * update and purge screens
 * @author Chetan Munegowda and Abhay Manoli
 */
public class Troubleticketlandingpage extends JFrame {

    private JPanel contentPaneObj;
    private JTable tableObj;
    private Connection connectionObj= null;
    private JButton btn_CreateButton;
    private JButton btn_ViewButton;
    private JLabel lblWelcomeToTroble;
    private JButton btn_UpdateButton;
    private JButton btn_PurgeButton;
    private static String strLoginUserName;
    private String ticket_Id = "" ;
```

```

private String strUserFlag ;

/**
 * Launch the application.
 */
public static void main(String[] args) {
    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                Troubleticketlandingpage frame = new
Troubleticketlandingpage(strLoginUserName);
                frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

/**
 * Create the frame.
 * @param string
 */
public Troubleticketlandingpage(String strUsrName) {

    setLoginUserName(strUsrName);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(100, 100, 1012, 534);
    contentPaneObj = new JPanel();
    contentPaneObj.setBorder(new EmptyBorder(5, 5, 5, 5));
    contentPaneObj.setBackground(Color.getHSBColor(1.4f, 1.0f, 1.6f));
    setContentPane(contentPaneObj);
    contentPaneObj.setLayout(null);

    connectionObj = SqlConnection.dbConnector();

    JScrollPane scrollPaneObj = new JScrollPane();
    scrollPaneObj.setBounds(10, 65, 963, 322);
    contentPaneObj.add(scrollPaneObj);

    /**
     * Action handler for mouse clicked event,
     * when user clicks on the row, it selects the
     * ticket id, which is used by the button to
     * identify any rows are selected
     */
    tableObj = new JTable();
    tableObj.addMouseListener(new MouseAdapter() {
        @Override
        public void mouseClicked(MouseEvent arg0) {

            try{
                int row = tableObj.getSelectedRow();
                ticket_Id = (tableObj.getModel().getValueAt(row,
0)).toString();

```

```

    }
    catch(Exception e){
        JOptionPane.showMessageDialog(null, e.getMessage());
    }
}

});

//instance table model
DefaultTableModel tableModel = new DefaultTableModel() {

    @Override
    public boolean isCellEditable(int row, int column) {
        //all cells false
        return false;
    }
};

tableObj.setModel(tableModel);

scrollPaneObj.setViewportViewView(tableObj);

btn_CreateButton = new JButton("Create");
btn_CreateButton.setFont(new Font("Tahoma", Font.BOLD, 13));
Image createImg = new
ImageIcon(this.getClass().getResource("/create.png")).getImage();
btn_CreateButton.setIcon(new ImageIcon(createImg));
btn_CreateButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        contentPaneObj.setVisible(false);
        CreateTicket createTicktObj = new CreateTicket(strUserName);
        createTicktObj.setVisible(true);
    }
});
btn_CreateButton.setBounds(115, 420, 107, 28);
contentPaneObj.add(btn_CreateButton);

btn_ViewButton = new JButton("View");
btn_ViewButton.setFont(new Font("Tahoma", Font.BOLD, 13));
Image viewImg = new
ImageIcon(this.getClass().getResource("/view.png")).getImage();
btn_ViewButton.setIcon(new ImageIcon(viewImg));
btn_ViewButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        //Checks the user has selected any record if
        //not prompt the user to select the record
        if(ticket_Id.equalsIgnoreCase(""))
        {
            JOptionPane.showMessageDialog(null, "Please select a
row");
        }
        //Else navigate to the screen as per the action flag, make
sure to
        //send the ticket id, user flag and user name for the
appropriate
        //session tracking
    }
});

```

```

        {
            strUserFlag = "VIEW";
            contentPaneObj.setVisible(false);
            ViewTicket viewTickt = new
ViewTicket(ticket_Id, strUserFlag, strUsrName);
            viewTickt.setVisible(true);
        }
    }
});
btn_ViewButton.setBounds(270, 420, 99, 28);
contentPaneObj.add(btn_ViewButton);

//Welcome the user with logged in user name
lblWelcomeToTroble = new JLabel("Hello "+strLoginUserName +"! Welcome
to Troble Ticket Landing Page");
lblWelcomeToTroble.setFont(new Font("Tahoma", Font.BOLD, 17));
lblWelcomeToTroble.setBounds(226, 11, 512, 28);
contentPaneObj.add(lblWelcomeToTroble);

//Combo box values for the ticket status
final DefaultComboBoxModel ticktStatusModel = new
DefaultComboBoxModel(
    new String[]{"Select", "OPEN", "CLOSED"});

//Display the records in the landing page
displayJtable();

btn_UpdateButton = new JButton("Update");
btn_UpdateButton.setFont(new Font("Tahoma", Font.BOLD, 13));
Image updateImg = new
ImageIcon(this.getClass().getResource("/update.png")).getImage();
btn_UpdateButton.setIcon(new ImageIcon(updateImg));

//Check the user has selected any row, if so the update the action
flag navigate
//to the page else prompt him to select a row
btn_UpdateButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        if(ticket_Id.equalsIgnoreCase(""))
        {
            JOptionPane.showMessageDialog(null, "Please select a
row");
        }
        else
        {
            strUserFlag = "UPDATE";
            contentPaneObj.setVisible(false);
            ViewTicket viewTickt = new
ViewTicket(ticket_Id, strUserFlag, strUsrName);
            viewTickt.setVisible(true);
        }
    }
});
btn_UpdateButton.setBounds(406, 420, 107, 28);
contentPaneObj.add(btn_UpdateButton);

```



```

/*
 * If the user clicks purge button without selecting row, then prompt
 * him to select the row, otherwise display the record and make the
flag
 * as purge
 */
btn_PurgeButton = new JButton("Purge");
btn_PurgeButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        if(ticket_Id.equalsIgnoreCase(""))
        {
            JOptionPane.showMessageDialog(null, "Please select a
row");
        }
        else
        {
            strUserFlag = "PURGE";
            contentPaneObj.setVisible(false);
            ViewTicket viewTickt = new
ViewTicket(ticket_Id,strUserFlag,strUsrName);
            viewTickt.setVisible(true);
        }
    }
});
btn_PurgeButton.setFont(new Font("Tahoma", Font.BOLD, 13));
Image purgeImg = new
ImageIcon(this.getClass().getResource("/delete.png")).getImage();
btn_PurgeButton.setIcon(new ImageIcon(purgeImg));

btn_PurgeButton.setBounds(554, 420, 95, 28);
contentPaneObj.add(btn_PurgeButton);
//When the user clicks the refresh he can see all
//updated changes in the session
JButton btnRefresh = new JButton("Refresh");
btnRefresh.setFont(new Font("Tahoma", Font.BOLD, 12));

Image refreshImg = new
ImageIcon(this.getClass().getResource("/refresh.png")).getImage();
btnRefresh.setIcon(new ImageIcon(refreshImg));
btnRefresh.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0)
    {
        try{

            String query = "SELECT
TICKET_ID,TICKET_PRIORITY,TICKET_COMMENT,"
+
"TICKET_CATEGORY,TICKET_STATUS,TICKET_DESCRIPTION,"
+
"TICKET_ASSIGNEE,TICKET_OPEN_DATE,TICKET_CLOSE_DATE "
+ " FROM CMUNEGOW_TBL_TICKET WHERE "
+ "TICKET_CREATEDBY= ? "
+ "AND TICKET_ISPURGED = ?";

            PreparedStatement pstmtObj =
connectionObj.prepareStatement(query);

```

```

        pstmtObj.setString(1, strLoginUserName);
        pstmtObj.setString(2, "N");
        ResultSet rs = pstmtObj.executeQuery();
        tableObj.setModel(DbUtils.resultSetToTableModel(rs));

    }
    catch(Exception e)
    {
        System.out.println(e.getMessage());
    }
}

});
btnRefresh.setBounds(683, 420, 99, 28);
contentPaneObj.add(btnRefresh);

JButton btnLogOut = new JButton("Log Out");
Image logoutImg = new
ImageIcon(this.getClass().getResource("/logout.png")).getImage();
btnLogOut.setIcon(new ImageIcon(logoutImg));
btnLogOut.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent evt) {
        Login loginPageObj = new Login();
        loginPageObj.mainFrameObj.setVisible(true);
        contentPaneObj.setVisible(false);
    }
});
btnLogOut.setBounds(837, 17, 121, 23);
contentPaneObj.add(btnLogOut);
}

/**
 * Method which displays the records in the table
 */
private void displayJtable() {

    try{

        String query = "SELECT TICKET_ID,TICKET_PRIORITY,TICKET_COMMENT,"
            + "TICKET_CATEGORY,TICKET_STATUS,TICKET_DESCRIPTION,"
            + "TICKET_ASSIGNEE,TICKET_OPEN_DATE,TICKET_CLOSE_DATE "
            + " FROM CMUNEGOW_TBL_TICKET WHERE "
            + "TICKET_CREATEDBY= ? "
            + "AND TICKET_ISPURGED = ?";

        PreparedStatement pstmtObj =
connectionObj.prepareStatement(query);
        pstmtObj.setString(1, strLoginUserName);
        pstmtObj.setString(2, "N");
        ResultSet rs = pstmtObj.executeQuery();
        tableObj.setModel(DbUtils.resultSetToTableModel(rs));

    }
    catch(Exception e)
    {
        System.out.println(e.getMessage());
    }
}

```

```

    }

    /*
    * Method which sets the user name which is used for
    * the session tracking
    */
    private void setLoginUserName(String strUsrName) {
        this.strLoginUserName = strUsrName;
    }
}

```

User Landing Page Screen Shot:

Hello chetan! Welcome to Troble Ticket Landing Page Log Out

TICKET_ID	TICKET_PRIORITY	TICKET_COMME...	TICKET_CATEG...	TICKET_STATUS	TICKET_DESCRI...	TICKET_ASSIGN...	TICKET_OPEN_...	TICKET_CLOSE...
1	MEDIUM		OS ISSUE	OPEN	please resolve th...	DIPEN	2014-11-29	
2	LOW		NETWORKING I...	OPEN	Wifi Notworking	DIPEN	2014-11-29	
3	MEDIUM		HARDWARE ISS...	OPEN	Motherboard Iss...	DIPEN	2014-11-29	
4	MEDIUM		SOFTWARE INS...	OPEN	Install Visual Stu...	DIPEN	2014-11-29	
5	LOW		INTENET CONN...	OPEN	Wifi Not working.	DIPEN	2014-11-29	
6	HIGH		LOGIN ISSUE	OPEN	Not able to login t...	DIPEN	2014-11-29	
7	HIGH		COMPUTER ALL...	OPEN	Allocate Computer.	DIPEN	2014-11-29	
8	LOW		COMUTER REST...	OPEN	Computer restart...	DIPEN	2014-11-29	
9	MEDIUM		OTHERS	OPEN	Need the credent...	DIPEN	2014-11-29	
10	MEDIUM		NETWORKING I...	OPEN	Slow Internet Sp...	DIPEN	2014-11-29	
11	LOW		SOFTWARE INS...	OPEN	Install Notepad ++.	DIPEN	2014-11-29	

Create View Update Purge Refre...

Create Ticket By User:

User can create the trouble ticket by clicking the create button from the trouble ticket landing page. Tickets will be allocated to the available admin randomly.

```
package com.iit.tts.loginpage;

import java.awt.Color;
import java.awt.EventQueue;
import java.awt.Font;

import javax.crypto.spec.PSource;
import javax.swing.DefaultComboBoxModel;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.border.EmptyBorder;

import com.iit.tts.db.SqlConnetion;
import com.iit.tts.troubleshootertickethelper.Troubleshootertickethelper;

import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.sql.CallableStatement;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.Random;

/**
 * This class helps user to create IT trouble
 * ticket
 * @author Chetan Munegowda and Abhay Manoli
 */
public class CreateTicket extends JFrame {

    private JPanel contentPane;
    private JTextField text_UserNameFld;
    private static String strUserNme;
    private Connection connectionObj = null;
    private Random randomGenerator ;
    private Troubleshooterticketlandingpage ttLangPage ;
    private StringBuffer strBuffer;
    private String strValidationMessage;

    /**
```

```
* Launch the application.
*/
public static void main(String[] args) {
    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                CreateTicket frame = new CreateTicket(strUserName);
                frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

/**
 * Create the frame.
 * @param strUsrName
 */
public CreateTicket(String strUsrName) {
    setUserName(strUsrName);

    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(100, 100, 728, 475);
    contentPane = new JPanel();
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
    contentPane.setBackground(Color.getHSBColor(1.4f, 1.0f, 1.6f));
    setContentPane(contentPane);
    contentPane.setLayout(null);

    JLabel lblCreateTicket = new JLabel("Create Trouble Ticket");
    lblCreateTicket.setFont(new Font("Tahoma", Font.BOLD, 17));
    lblCreateTicket.setBounds(299, 29, 184, 39);
    contentPane.add(lblCreateTicket);

    JLabel lblUserName = new JLabel("User Name");
    lblUserName.setFont(new Font("Tahoma", Font.BOLD, 14));
    lblUserName.setBounds(97, 102, 80, 26);
    contentPane.add(lblUserName);

    JLabel lblTicketCategory = new JLabel("Ticket Category");
    lblTicketCategory.setFont(new Font("Tahoma", Font.BOLD, 14));
    lblTicketCategory.setBounds(97, 145, 118, 20);
    contentPane.add(lblTicketCategory);

    JLabel lblTicketPriority = new JLabel("Ticket Priority");
    lblTicketPriority.setFont(new Font("Tahoma", Font.BOLD, 14));
    lblTicketPriority.setBounds(97, 188, 95, 20);
    contentPane.add(lblTicketPriority);

    JLabel lblTicketDescription = new JLabel("Ticket Description");
    lblTicketDescription.setFont(new Font("Tahoma", Font.BOLD, 14));
    lblTicketDescription.setBounds(97, 234, 131, 19);
    contentPane.add(lblTicketDescription);

    text_UserNameFld = new JTextField();
    text_UserNameFld.setBounds(262, 99, 221, 20);
}
```

```

text_UserNameFld.setText(strUserNme);
text_UserNameFld.setEditable(false);
contentPane.add(text_UserNameFld);
text_UserNameFld.setColumns(10);

/*
 * Initialize the ticket category
 */
final DefaultComboBoxModel ticketCategoryModel = new
DefaultComboBoxModel(
    new String[]{"SELECT","OS ISSUE",
        "NETWORKING ISSUE","HARDWARE ISSUE",
        "SOFTWARE INSTALL","SOFTWARE UNINSTALL",
        "INTENET CONNECTIVITY ISSUE",
        "LOGIN ISSUE","COMPUTER ALLOCATE REQUEST",
        "COMUTER RESTART ISSUE","OTHERS"}});

JComboBox comboBox_TicktCategory = new JComboBox(ticketCategoryModel);
comboBox_TicktCategory.setBounds(262, 139, 221, 26);
contentPane.add(comboBox_TicktCategory);

/*
 * Initialize the ticket priority
 */
final DefaultComboBoxModel ticketPriorityModel = new
DefaultComboBoxModel(
    new String[]{"SELECT","LOW","MEDIUM","HIGH"});

JComboBox comboBox_TicktPriority = new
JComboBox(ticketPriorityModel);
comboBox_TicktPriority.setBounds(262, 182, 221, 26);
contentPane.add(comboBox_TicktPriority);

JTextArea textArea_ticktDesc = new JTextArea();
textArea_ticktDesc.setBounds(262, 234, 402, 52);
contentPane.add(textArea_ticktDesc);

JButton btntickt_SubmitBtn = new JButton("Submit");
/*
 * Action listener for submit button which validates the
 * records before inserting
 */
btntickt_SubmitBtn.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {

        if(validateFieldSatus())
        {
            //Inserts the record into trouble ticket
            //table
            insertRecordsIntoTroubleTicket();
        }
    }
});

/**
 * Validates whether the user has given all
 * the mandatory fields which are required to
 * create the ticket
 */

```

```

        * @return
        */
private boolean validateFieldStatus() {

    StringBuffer = new StringBuffer("");
    strValidationMessage = new String();
    Boolean bBoolFldStatus = true;

    if(Troubletickethelper.validateDropDown((String)
comboBox_TicktCategry.getSelectedItem()))
    {
        strValidationMessage = "Please select valid ticket
category!";

        StringBuffer.append(strValidationMessage+"\n");
        bBoolFldStatus = false;
    }

    if(Troubletickethelper.validateDropDown((String)
comboBox_TicktPriority.getSelectedItem()))
    {
        strValidationMessage = "Please select valid ticket
priority!";

        StringBuffer.append(strValidationMessage+"\n");
        bBoolFldStatus = false;
    }

    if(Troubletickethelper.validateTextField(textArea_ticktDesc.getText()))
    {
        strValidationMessage = "Please provide ticket
description!";

        StringBuffer.append(strValidationMessage+"\n");
        bBoolFldStatus = false;
    }

    if(!(StringBuffer.toString().equalsIgnoreCase(""))){
        JOptionPane.showMessageDialog(null, StringBuffer);
    }

    return bBoolFldStatus;
}

/**
 * Method which calls the insert stored procedure to
 * insert the records into the trouble ticket table
 */
private void insertRecordsIntoTroubleTicket() {

    try{
        connectionObj = SqlConnection.dbConnector();
        connectionObj.setAutoCommit(false);

        String sqlInsertProcedure = "{ "
            + "call
cmam_addTroubleTicket(?,?,?,?,?,?,?,?,?,?) "

```

```

        + "}";

        //PreparedStatement pstObj1 =
connectionObj.prepareStatement(sqlInsert);
        CallableStatement cs =
connectionObj.prepareCall(sqlInsertProcedure);

        cs.setString(1, (String)
comboBox_TicketPriority.getSelectedItem());
        cs.setString(2, null);
        //Make the default ticket create status open
        cs.setString(3, "OPEN");
        cs.setString(4, textArea_ticketDesc.getText());
        cs.setString(5, (String)
comboBox_TicketCategory.getSelectedItem());

SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-
MM-dd HH:mm:ss");

        Date date = new Date();
        String strtickOpneDate = dateFormat.format(date);

        cs.setString(6, strtickOpneDate);
        cs.setString(7, null);
        cs.setString(8, strUsrName);
        cs.setString(10, "N");

        //SQL query which queries all the user names
        //whose role is admin
        String sqlSelect = "SELECT USER_NAME FROM "+
            "CMUNEGOW_TBL_USERS "+
            "WHERE USER_ROLE = ?";

        ArrayList<String> arrUserAdmin = new ArrayList<String>();

        PreparedStatement pstObj2 =
connectionObj.prepareStatement(sqlSelect);
        pstObj2.setString(1, "ADMIN");
        ResultSet rs = pstObj2.executeQuery();

        //Store it in the arraylist
        while(rs.next())
        {
            arrUserAdmin.add(rs.getString("USER_NAME"));
        }
        randomGenerator = new Random();

        //If there arraylist size is zero, then display message
to user saying
        //currently no one to solve the ticket
        if(arrUserAdmin.size() == 0)
        {
            JOptionPane.showMessageDialog(null, "Sorry, currently
there are no admin"
            + "for the trouble ticket system.Your ticket
cannot be solved!");
        }
    }

```



```

        //Else using the random generator assign the ticket
        else
        {
            int index =
randomGenerator.nextInt(arrUserAdmin.size());
            cs.setString(9, arrUserAdmin.get(index));
            cs.execute();
            JOptionPane.showMessageDialog(null, "Your Trouble
Ticket "

                                + "is successfully Created");
            connectionObj.commit();
        }
        contentPane.setVisible(false);

        ttLangPage.setVisible(true);

    }catch(SQLException e)
    {
        JOptionPane.showMessageDialog(null, e.getMessage());
    }
    catch (Exception e)
    {
        JOptionPane.showMessageDialog(null, e.getMessage());
    }

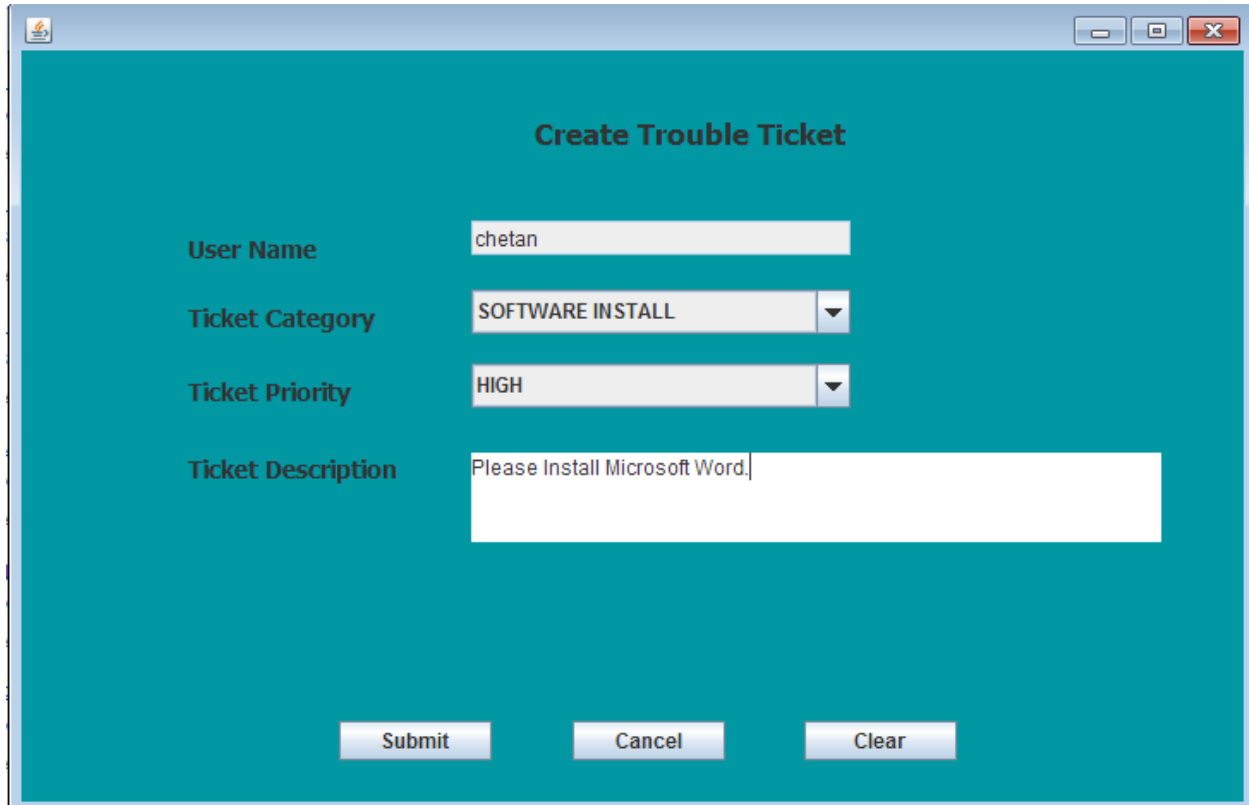
    }
});
bntticket_SubmitBtn.setBounds(185, 390, 89, 23);
contentPane.add(bntticket_SubmitBtn);

//User when clicked on cancel button, navigate to the
//previous page
JButton bntticket_CancelBtn = new JButton("Cancel");
bntticket_CancelBtn.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0)
    {
        contentPane.setVisible(true);
        ttLangPage.setVisible(true);
    }
});
bntticket_CancelBtn.setBounds(321, 390, 89, 23);
contentPane.add(bntticket_CancelBtn);

//User when clicked on clear button, clears the
//fields
JButton bntticket_Clear = new JButton("Clear");
bntticket_Clear.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        textArea_ticktDesc.setText("");
        comboBox_TicktCategry.setSelectedIndex(0);
        comboBox_TicktPriority.setSelectedIndex(0);
    }
});
bntticket_Clear.setBounds(456, 390, 89, 23);
contentPane.add(bntticket_Clear);
}

```

```
/**
 * Initialize the current user name
 * @param strUserName
 */
private void setUsername(String strUserName) {
    this.strUserName = strUserName;
    ttLangPage = new TroubleTicketLandingPage(strUserName);
}
}
```



The screenshot shows a web application window titled "Create Trouble Ticket". The form has a teal background and contains the following fields and controls:

- User Name:** A text input field containing the value "chetan".
- Ticket Category:** A dropdown menu with "SOFTWARE INSTALL" selected.
- Ticket Priority:** A dropdown menu with "HIGH" selected.
- Ticket Description:** A text area containing the text "Please Install Microsoft Word.".
- Buttons:** At the bottom, there are three buttons: "Submit", "Cancel", and "Clear".

Create Trouble Ticket

User Name: chetan

Ticket Category: SOFTWARE INSTALL

Ticket Priority: HIGH

Ticket Description: Please I

Message
Your Trouble Ticket is successfully Created
OK

Submit Cancel Clear

Hello chetan! Welcome to Troble Ticket Landing Page [Log Out](#)

TICKET_ID	TICKET_PRIORITY	TICKET_C...	TICKET_CATEGORY	TICKET_STA...	TICKET_DESCRIPTION	TICKET_AS...	TICKET_OPEN_D...	TICKET_CLOSE...
1	MEDIUM		OS ISSUE	OPEN	please resolve the issue	DIPEN	2014-11-29	
2	LOW		NETWORKING ISSUE	OPEN	Wifi Networking	DIPEN	2014-11-29	
3	MEDIUM		HARDWARE ISSUE	OPEN	Motherboard Issue.	DIPEN	2014-11-29	
4	MEDIUM		SOFTWARE INSTALL	OPEN	Install Visual Studio for C#.	DIPEN	2014-11-29	
5	LOW		INTENET CONNECTIV...	OPEN	Wifi Not working.	DIPEN	2014-11-29	
6	HIGH		LOGIN ISSUE	OPEN	Not able to login to my system.	DIPEN	2014-11-29	
7	HIGH		COMPUTER ALLOCAT...	OPEN	Allocate Computer.	DIPEN	2014-11-29	
8	LOW		COMUTER RESTART I...	OPEN	Computer restarts automaticall...	DIPEN	2014-11-29	
9	MEDIUM		OTHERS	OPEN	Need the credentials for Oracle...	DIPEN	2014-11-29	
10	MEDIUM		NETWORKING ISSUE	OPEN	Slow Internet Speed.	DIPEN	2014-11-29	
11	LOW		SOFTWARE INSTALL	OPEN	Install Notepad ++.	DIPEN	2014-11-29	
12	HIGH		SOFTWARE INSTALL	OPEN	Please Install Microsoft Word.	DIPEN	2014-11-29	

Create View Update Purge Refre...

User View/Update/Purge Ticket:

User can view the ticket information by selecting the row and then click the view button from the user landing page. Here, all the fields are in disabled and non editable. Similarly, user the update the ticket by clicking the update button, here some fields are editable and Finally user can purge the ticket by clicking the purge button, here user is alerted with confirm dialog box whether he is sure to delete the ticket. Finally, the confirmation message is displayed to the user. Later, He can click the refresh button to see the updated changes in the session.

```
package com.iit.tts.loginpage;

import java.awt.Color;
import java.awt.EventQueue;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.CallableStatement;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

import javax.swing.DefaultComboBoxModel;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.border.EmptyBorder;

import com.iit.tts.db.SqlConnetion;

/*****
 * This class allows us to view the tickets
 * @author Chetan Munegowda and Abhay Manoli
 */
*****/
public class ViewTicket extends JFrame {

    private JPanel contentPane;
    private JTextField txtField_UsrNme;
    private JTextField textField_ticketAssignee;
    private static String strActionFlag;
    private static String strTicketId;
    private static String strUsrNme;
    private Connection conectionObj;
    private JComboBox comboBox_ticketCategory;
    private JComboBox comboBox_ticketPriority;
    private JTextArea textArea_ticketDesc;
    private JTextArea textArea_adminRemarks;
    private String strTCategory;
    private String strTPriority;
    private String strTCreatedUsrName;
```

```

private String strTAssignee;
private String strTComment;
private String strTStatus;
private String strTDesc;
private Long lPhoneNumber;
private Connection connectionObj;
private JComboBox comboBox_ticketStatus;
private JTextField textField_AsgnePhoneNo;

/**
 * Launch the application.
 */
public static void main(String[] args) {
    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                ViewTicket frame = new
ViewTicket(strTicketId,strActionFlag, strUsrNme);
                frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

/**
 * Create the frame.
 * @param strUsrName
 * @param ticket_Id
 * @param strUsrName
 */
public ViewTicket(String ticket_Id, String strActionFlg, String
strUsrName) {

    initializeFields(ticket_Id,strActionFlg,strUsrName);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(100, 100, 891, 629);
    contentPane = new JPanel();
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
    contentPane.setBackground(Color.getHSBColor(1.4f, 1.0f, 1.6f));
    setContentPane(contentPane);
    contentPane.setLayout(null);

    JLabel lblTroubleTicket = new JLabel("View Trouble Ticket");
    lblTroubleTicket.setFont(new Font("Tahoma", Font.BOLD, 17));
    lblTroubleTicket.setBounds(280, 24, 196, 38);
    contentPane.add(lblTroubleTicket);

    JLabel lblUserName = new JLabel("User Name");
    lblUserName.setFont(new Font("Tahoma", Font.BOLD, 14));
    lblUserName.setBounds(153, 87, 87, 21);
    contentPane.add(lblUserName);

    JLabel lblTicketCategory = new JLabel("Ticket Category");
    lblTicketCategory.setFont(new Font("Tahoma", Font.BOLD, 14));
    lblTicketCategory.setBounds(153, 133, 125, 21);

```

```
contentPane.add(lblTicketCategory);

JLabel lblticketPriority = new JLabel("Ticket Priority");
lblticketPriority.setFont(new Font("Tahoma", Font.BOLD, 14));
lblticketPriority.setBounds(153, 177, 125, 27);
contentPane.add(lblticketPriority);

JLabel lblticketDesc = new JLabel("Ticket Description");
lblticketDesc.setFont(new Font("Tahoma", Font.BOLD, 14));
lblticketDesc.setBounds(153, 376, 125, 21);
contentPane.add(lblticketDesc);

JLabel lblNewLabel = new JLabel("Ticket Assignee");
lblNewLabel.setFont(new Font("Tahoma", Font.BOLD, 14));
lblNewLabel.setBounds(153, 271, 125, 21);
contentPane.add(lblNewLabel);

JLabel lblNewLabel_1 = new JLabel("Admin Remarks");
lblNewLabel_1.setFont(new Font("Tahoma", Font.BOLD, 14));
lblNewLabel_1.setBounds(153, 444, 125, 21);
contentPane.add(lblNewLabel_1);

txtField_UsrNme = new JTextField();
txtField_UsrNme.setFont(new Font("Tahoma", Font.BOLD, 12));
txtField_UsrNme.setEditable(false);
txtField_UsrNme.setEnabled(false);
txtField_UsrNme.setText(strUsrName);
txtField_UsrNme.setBounds(335, 89, 196, 21);
contentPane.add(txtField_UsrNme);
txtField_UsrNme.setColumns(10);

/**
 * Drop down values for ticket category
 */
final DefaultComboBoxModel ticketCategoryModel = new
DefaultComboBoxModel(
    new String[]{"SELECT","OS ISSUE",
        "NETWORKING ISSUE","HARDWARE ISSUE",
        "SOFTWARE INSTALL","SOFTWARE UNINSTALL",
        "INTENET CONNECTIVITY ISSUE",
        "LOGIN ISSUE","COMPUTER ALLOCATE REQUEST",
        "COMUTER RESTART ISSUE","OTHERS"});

comboBox_ticketCategory = new JComboBox(ticketCategoryModel);
comboBox_ticketCategory.setFont(new Font("Tahoma", Font.BOLD, 12));
comboBox_ticketCategory.setBounds(335, 135, 196, 21);
contentPane.add(comboBox_ticketCategory);

final DefaultComboBoxModel ticketPriorityModel = new
DefaultComboBoxModel(
    new String[]{"SELECT","LOW","MEDIUM","HIGH"});

comboBox_ticketPriority = new JComboBox(ticketPriorityModel);
comboBox_ticketPriority.setFont(new Font("Tahoma", Font.PLAIN, 12));
comboBox_ticketPriority.setBounds(335, 180, 196, 24);
contentPane.add(comboBox_ticketPriority);
```

```
textField_ticketAssignee = new JTextField();
textField_ticketAssignee.setFont(new Font("Tahoma", Font.BOLD, 12));
textField_ticketAssignee.setBounds(335, 273, 196, 21);
contentPane.add(textField_ticketAssignee);
textField_ticketAssignee.setColumns(10);

textArea_ticketDesc = new JTextArea();
textArea_ticketDesc.setFont(new Font("Monospaced", Font.BOLD, 13));
textArea_ticketDesc.setBounds(335, 376, 390, 38);
contentPane.add(textArea_ticketDesc);

textArea_adminRemarks = new JTextArea();
textArea_adminRemarks.setFont(new Font("Monospaced", Font.BOLD, 13));
textArea_adminRemarks.setEditable(false);
textArea_adminRemarks.setBounds(335, 444, 390, 38);
contentPane.add(textArea_adminRemarks);

JLabel lbltickt_Status = new JLabel("Ticket Status");
lbltickt_Status.setFont(new Font("Tahoma", Font.BOLD, 14));
lbltickt_Status.setBounds(153, 227, 117, 21);
contentPane.add(lbltickt_Status);

/**
 * Action listener for cancel button, when the user clicks
 * cancel button it navigates back to landing page
 */
JButton btn_CancelBtn = new JButton("Cancel");
btn_CancelBtn.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        contentPane.setVisible(false);
        Troubleticketlandingpage ttLangPage = new
Troubleticketlandingpage(strUserName);
        ttLangPage.setVisible(true);
    }
});
btn_CancelBtn.setFont(new Font("Tahoma", Font.BOLD, 12));
btn_CancelBtn.setBounds(359, 530, 117, 27);
contentPane.add(btn_CancelBtn);

/**
 * When the user clicks on the submit button, it checks
 * for the user flag UPDATE or PURGE, depending on it
 * the action is taken
 */
JButton btn_SubmitBtn = new JButton("Submit");
btn_SubmitBtn.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0)
    {
        //If the user clicked update button,
        //then it updates the record
        if(strActionFlg.equalsIgnoreCase("UPDATE"))
        {
            updateTicketInformation();
        }
        //If the user clicked the purge button,
        //then the record gets purged
    }
});
```

```

        else if(strActionFlg.equalsIgnoreCase("PURGE"))
        {
            purgeTicketInformation();
        }
        //finally going back to the landing page
        Troubleticketlandingpage ttLangPage = new
Troubleticketlandingpage(strUsrName);
        contentPane.setVisible(false);
        ttLangPage.setVisible(true);
    }

    /*
     * Method which purges the user record, its a soft delete where
     * me make the isdeleted flag yes
     */
    private void purgeTicketInformation() {

        int iAction = JOptionPane.showConfirmDialog(null, "Do you
really"
            + " want to
Delete","Delete",JOptionPane.YES_NO_OPTION);

        if(iAction == 0)
        {
            try{
                conectionObj = SqlConnection.dbConnector();
                conectionObj.setAutoCommit(false);

                String strQuery = "Update CMUNEGOW_TBL_TICKET "
                    + "SET TICKET_ISPURGED = ? "
                    + "WHERE TICKET_ID = ? ";

                PreparedStatement psObj =
conectionObj.prepareStatement(strQuery);
                psObj.setString(1, "Y");
                psObj.setString(2, strTicketId);
                psObj.execute();
                conectionObj.setAutoCommit(true);
                JOptionPane.showMessageDialog(null, "Your Trouble
Ticket is"
                    + " successfully deleted!");
            }

            catch(Exception e){
                JOptionPane.showMessageDialog(null, e.getMessage());
            }
        }
    }

    /**
     * Method which updates the ticket record
     */
    private void updateTicketInformation() {

```



```

        try{
            conectionObj = SqlConnection.dbConnector();
            conectionObj.setAutoCommit(false);

            JOptionPane.showMessageDialog(null, "Updating the
ticketId: " +strTicktId);

            String sqlInsertTicketProcedure = "{ "
                + "call cmam_updateTroubleTicket(?,?,?,?) "
                + "}";

            CallableStatement callStObj =
conectionObj.prepareCall(sqlInsertTicketProcedure);
            callStObj.setString(1, (String)
comboBox_ticketPriority.getSelectedItemAt());
            callStObj.setString(2, textArea_adminRemarks.getText());
            callStObj.setString(3, (String)
comboBox_ticketCategory.getSelectedItemAt());
            callStObj.setString(4, textArea_ticketDesc.getText());
            callStObj.setString(5, strTicktId);

            callStObj.execute();
            conectionObj.setAutoCommit(true);
            JOptionPane.showMessageDialog(null, "Your Trouble Ticket
"
                + "is successfully updated!");

        }

        catch(Exception e){
            JOptionPane.showMessageDialog(null, e.getMessage() );
        }
    }

});
btn_SubmitBtn.setFont(new Font("Tahoma", Font.BOLD, 12));
btn_SubmitBtn.setBounds(209, 531, 117, 25);
contentPane.add(btn_SubmitBtn);

/**
 * Initialize the ticket status
 */
final DefaultComboBoxModel ticketStatusModel = new
DefaultComboBoxModel(
    new String[]{"SELECT", "OPEN", "CLOSED"});
comboBox_ticketStatus = new JComboBox(ticketStatusModel);
comboBox_ticketStatus.setFont(new Font("Tahoma", Font.BOLD, 12));
comboBox_ticketStatus.setBounds(335, 227, 196, 22);
contentPane.add(comboBox_ticketStatus);

JLabel lblTicketAssigneePhone = new JLabel("Assignee Phone Number");
lblTicketAssigneePhone.setFont(new Font("Tahoma", Font.BOLD, 14));
lblTicketAssigneePhone.setBounds(153, 325, 175, 21);
contentPane.add(lblTicketAssigneePhone);

textField_AsgnePhoneNo = new JTextField();
textField_AsgnePhoneNo.setFont(new Font("Tahoma", Font.BOLD, 12));
textField_AsgnePhoneNo.setEnabled(false);

```

```
textField_AsgnePhoneNo.setEditable(false);
textField_AsgnePhoneNo.setBounds(335, 327, 196, 21);
contentPane.add(textField_AsgnePhoneNo);
textField_AsgnePhoneNo.setColumns(10);

//Fetches the record
fetchRecords();

//If the user pressed View button, then make the fields
//non editable and disabled so user will not have the option
//to change
if(strActionFlag.equalsIgnoreCase("VIEW"))
{
    lblTroubleTicket.setText("View Trouble Ticket");
    textField_ticketAssignee.setEnabled(false);
    txtField_UsrNme.setEnabled(false);
    textArea_adminRemarks.setEnabled(false);
    textArea_ticketDesc.setEnabled(false);
    comboBox_ticketCategory.setEnabled(false);
    comboBox_ticketPriority.setEnabled(false);
    comboBox_ticketStatus.setEnabled(false);
    textField_AsgnePhoneNo.setEnabled(false);
    btn_SubmitBtn.setEnabled(false);
    btn_SubmitBtn.setContentAreaFilled(false);
}

//If the user clicks on Purge button, the records
//are non editable but the submit button is made
//enabled
else if(strActionFlag.equalsIgnoreCase("PURGE"))
{
    lblTroubleTicket.setText("Purge Trouble Ticket");
    textField_ticketAssignee.setEnabled(false);
    txtField_UsrNme.setEnabled(false);
    textArea_adminRemarks.setEnabled(false);
    textArea_ticketDesc.setEnabled(false);
    comboBox_ticketCategory.setEnabled(false);
    comboBox_ticketPriority.setEnabled(false);
    comboBox_ticketStatus.setEnabled(false);
}

//If the user clicks on Update button the he can
//update ticket description,ticket category and
//submit button is made editable
else if(strActionFlg.equalsIgnoreCase("UPDATE"))
{
    lblTroubleTicket.setText("Update Trouble Ticket");
    textField_ticketAssignee.setEnabled(false);
    txtField_UsrNme.setEnabled(false);
    textArea_adminRemarks.setEnabled(false);
    textArea_ticketDesc.setEnabled(true);
    comboBox_ticketCategory.setEnabled(true);
    comboBox_ticketPriority.setEnabled(true);
    comboBox_ticketStatus.setEnabled(false);
    btn_SubmitBtn.setEnabled(true);
    btn_SubmitBtn.setContentAreaFilled(true);
}
}
```

```
/**
 * This method fetches the fields from the database
 * of the selected ticket Id from the previous screen
 */
private void fetchRecords() {

    try
    {
        conectionObj = SqlConnection.dbConnector();

        String selectTicktInfoQuery = "SELECT TICKET_CATEGORY,
TICKET_PRIORITY,TICKET_COMMENT,"
            + "TICKET_STATUS,TICKET_DESCRIPTION,TICKET_ASSIGNEE "
            + "FROM CMUNEGOW_TBL_TICKET "
            + "WHERE TICKET_ID = ?";

        PreparedStatement psObj =
conectionObj.prepareStatement(selectTicktInfoQuery);
        psObj.setString(1, strTicketId);

        ResultSet rs = psObj.executeQuery();
        while(rs.next())
        {
            strTAssignee = rs.getString("TICKET_ASSIGNEE");
            strTCategory = rs.getString("TICKET_CATEGORY");
            strTComment = rs.getString("TICKET_COMMENT");
            strTPriority = rs.getString("TICKET_PRIORITY");
            strTStatus = rs.getString("TICKET_STATUS");
            strTDesc = rs.getString("TICKET_DESCRIPTION");
        }

        comboBox_ticketCategory.setSelectedItem(strTCategory);
        comboBox_ticketPriorty.setSelectedItem(strTPriority);
        comboBox_ticketStatus.setSelectedItem(strTStatus);
        textField_ticketAssignee.setText(strTAssignee);
        textArea_ticketDesc.setText(strTDesc);
        textArea_adminRemarks.setText(strTComment);

        String selectTickAssigneeQuery = "SELECT PHONE_NUMBER "
            + "FROM CMUNEGOW_TBL_USERS "
            + "WHERE USER_NAME = ?";

        psObj = conectionObj.prepareStatement(selectTickAssigneeQuery);
        psObj.setString(1, strTAssignee);
        ResultSet rsObj = psObj.executeQuery();

        while(rsObj.next())
        {
            lPhoneNumber = (Long)rsObj.getLong("PHONE_NUMBER");
            textField_AsgnePhoneNo.setText(Long.toString(lPhoneNumber));
        }
    }
    catch(Exception e)
    {
    }
```

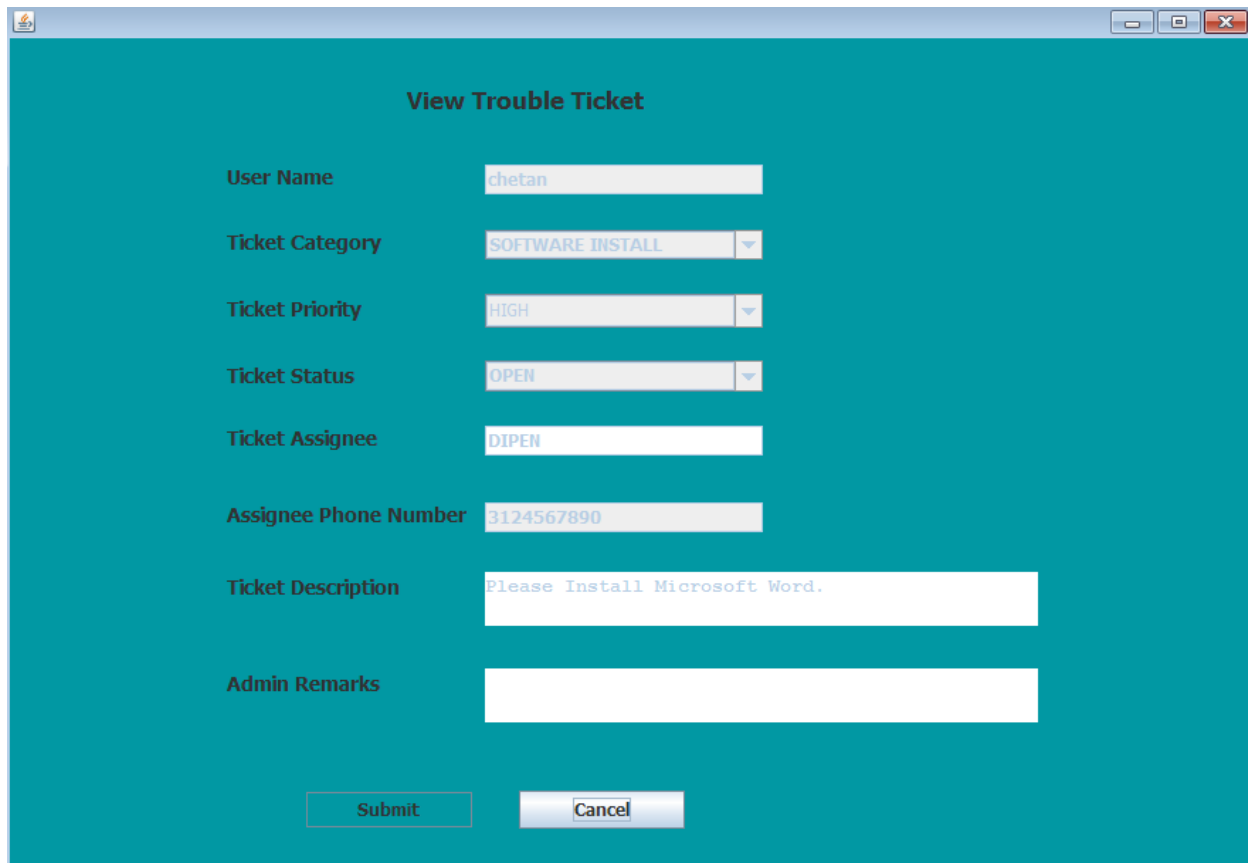
```
JOptionPane.showMessageDialog(null, e.getMessage());
    }

}

/*****
 * Initializes the user fields ticket Id, action Flag and
 * username from the are receievd from the previous screen
 * @param ticket_Id
 * @param actionFlg
 * @param strUsrName
 *****/
private void initializeFields(String ticket_Id, String actionFlg, String
strUsrName) {

    this.strActionFlag = actionFlg;
    this.strTicketId = ticket_Id;
    this.strUsrNme = strUsrName;
}
}
```

User View Ticket Snapshot:

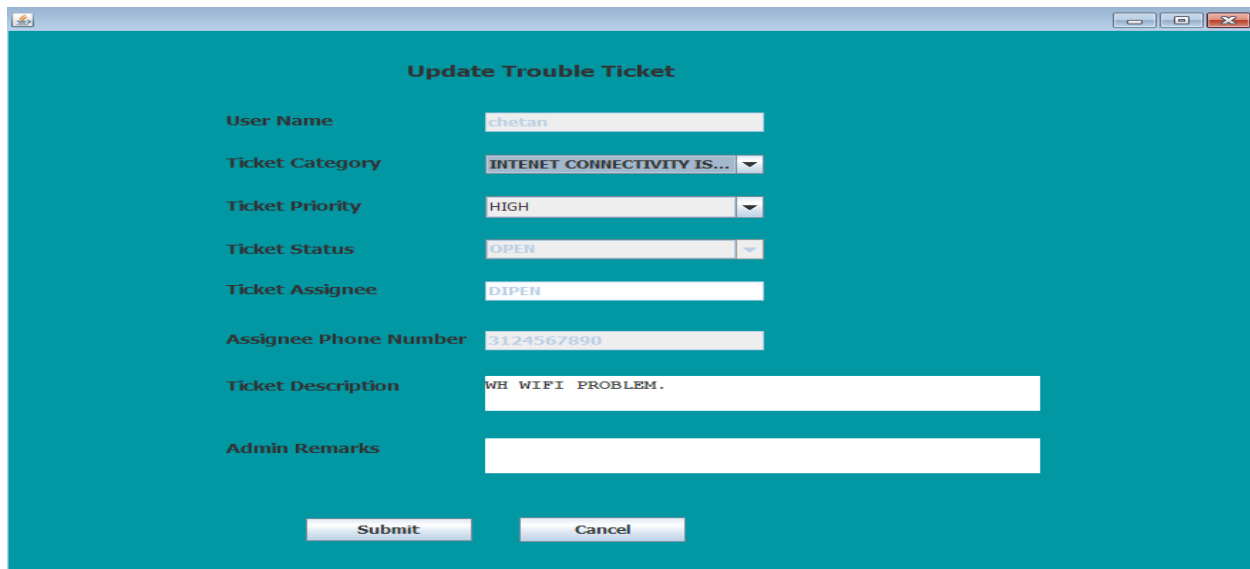


The screenshot shows a Java Swing window titled "View Trouble Ticket" with a teal background. It contains several form fields for displaying ticket information:

- User Name:** chetan
- Ticket Category:** SOFTWARE INSTALL (dropdown menu)
- Ticket Priority:** HIGH (dropdown menu)
- Ticket Status:** OPEN (dropdown menu)
- Ticket Assignee:** DIPEN
- Assignee Phone Number:** 3124567890
- Ticket Description:** Please Install Microsoft Word.
- Admin Remarks:** (empty text area)

At the bottom of the window, there are two buttons: "Submit" and "Cancel".

User Update Ticket Snapshot:



Update Trouble Ticket

User Name:

Ticket Category:

Ticket Priority:

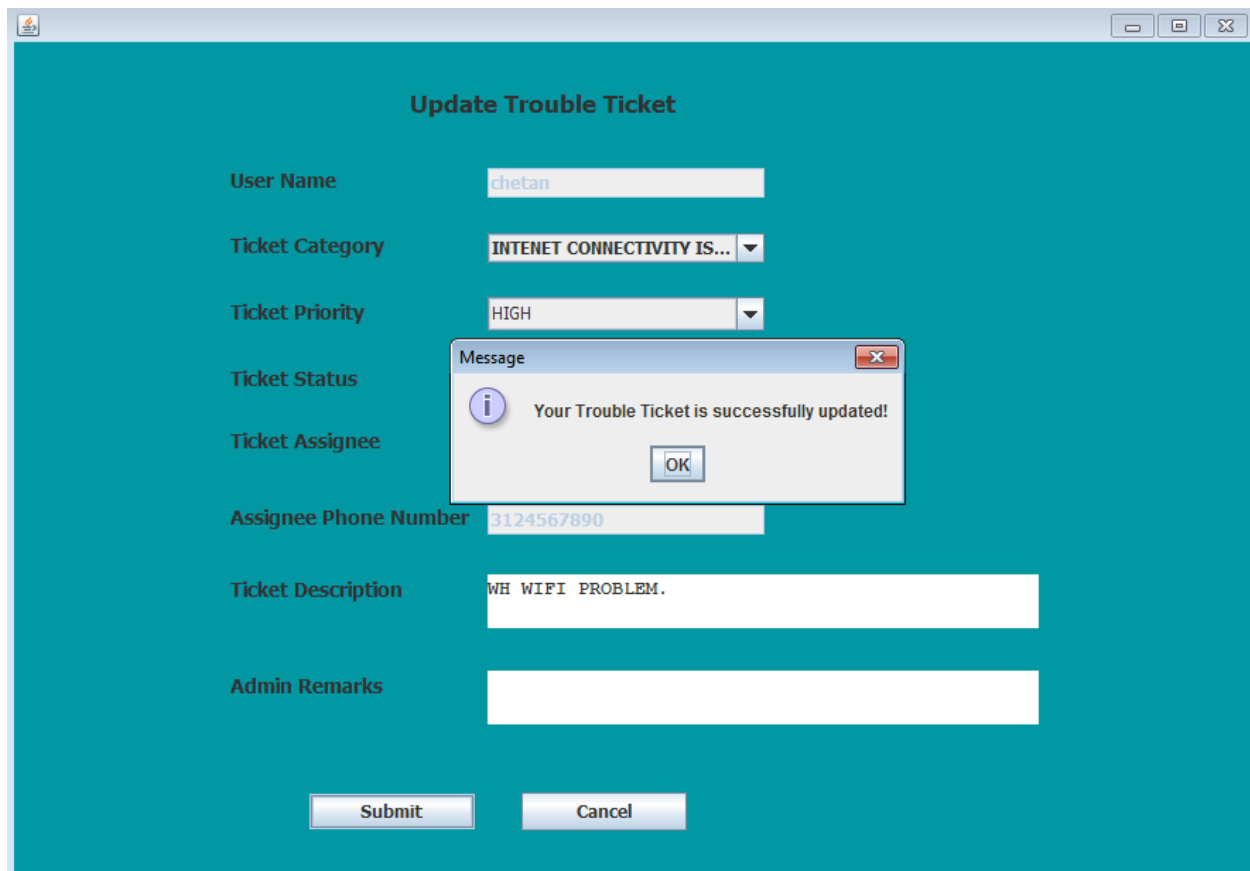
Ticket Status:

Ticket Assignee:

Assignee Phone Number:

Ticket Description:

Admin Remarks:



Update Trouble Ticket

User Name:

Ticket Category:

Ticket Priority:

Ticket Status:


Ticket Assignee:

Assignee Phone Number:

Ticket Description:

Admin Remarks:

Message

 Your Trouble Ticket is successfully updated!

Hello chetan! Welcome to Troble Ticket Landing Page Log Out

TICKET_ID	TICKET_PRIORITY	TICKET_...	TICKET_CATEGORY	TICKET_ST...	TICKET_DESCRIPTION	TICKET_ASSIGN...	TICKET_O...	TICKET_C...
1	MEDIUM		OS ISSUE	OPEN	please resolve the issue	DIPEN	2014-11-29	
2	LOW		NETWORKING ISSUE	OPEN	Wifi Notworking	DIPEN	2014-11-29	
3	MEDIUM		HARDWARE ISSUE	OPEN	Motherboard Issue.	DIPEN	2014-11-29	
4	MEDIUM		SOFTWARE INSTALL	OPEN	Install Visual Studio for C#.	DIPEN	2014-11-29	
5	LOW		INTENET CONNECTIVITY ISSUE	OPEN	Wifi Not working.	DIPEN	2014-11-29	
6	HIGH		LOGIN ISSUE	OPEN	Not able to login to my system.	DIPEN	2014-11-29	
7	HIGH		COMPUTER ALLOCATE REQUEST	OPEN	Allocate Computer.	DIPEN	2014-11-29	
8	LOW		COMPUTER RESTART ISSUE	OPEN	Computer restarts automaticall...	DIPEN	2014-11-29	
9	MEDIUM		OTHERS	OPEN	Need the credentials for Oracle ...	DIPEN	2014-11-29	
10	MEDIUM		NETWORKING ISSUE	OPEN	Slow Internet Speed.	DIPEN	2014-11-29	
11	LOW		SOFTWARE INSTALL	OPEN	Install Notepad ++.	DIPEN	2014-11-29	
12	HIGH		INTENET CONNECTIVITY ISSUE	OPEN	WH WIFI PROBLEM.	DIPEN	2014-11-29	

Create View Update Purge Refre...

User Purge Ticket Snapshot:

Purge Trouble Ticket

User Name

Ticket Category

Ticket Priority

Ticket Status

Ticket Assignee

Assignee Phone Number

Ticket Description

Admin Remarks

Submit Cancel

Purge Trouble Ticket

User Name:

Ticket Category:

Ticket Priority:

Ticket Status:

Ticket Assignee:

Assignee Phone Number:

Ticket Description:

Admin Remarks:

Delete

Are you sure you want to delete this ticket 8

Purge Trouble Ticket

User Name:

Ticket Category:

Ticket Priority:

Ticket Status:

Ticket Assignee:

Assignee Phone Number:

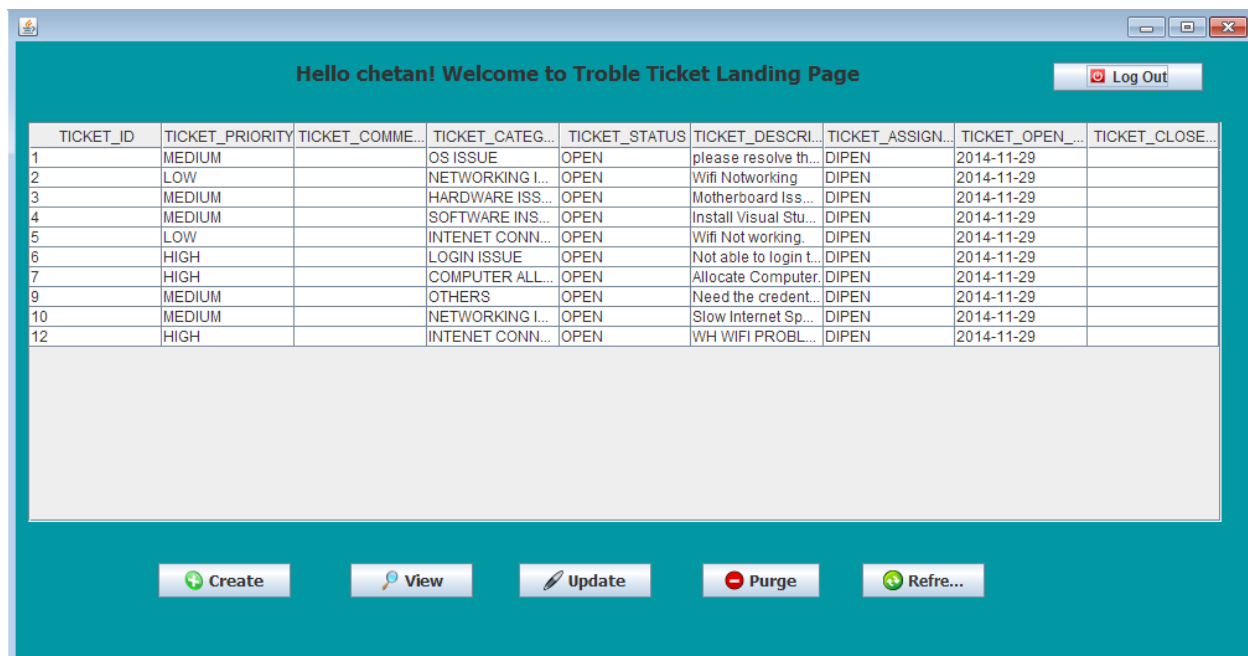
Ticket Description:

Admin Remarks:

Message

Your Trouble Ticket is successfully deleted!

8th record is successfully deleted!



Admin View/Update Ticket Code:

Admin can view the ticket information by selecting the row and then click the view button from the admin landing page. Here, all the fields are in disabled and non editable. Similarly, admin can update or close the ticket by clicking the update button, here some fields are editable. Confirmation message is displayed to the user. He can click the refresh button to see the updated changes in the current session.

```
package com.iit.tts.loginpage;

import java.awt.Color;
import java.awt.EventQueue;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.text.SimpleDateFormat;
import java.util.Date;

import javax.swing.DefaultComboBoxModel;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
```



```

import javax.swing.JPanel;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.border.EmptyBorder;

import com.iit.tts.db.SqlConnetion;

/*****
 * This class allows us to Admin to view and
 * update the status of the Tickets and enter the
 * admin remarks.
 *
 * @author Chetan Munegowda and Abhay Manoli
 */
*****/
public class Adminupdate extends JFrame {

    private JPanel contentPane;
    private JTextField txtFld_ticketCreatedBy;
    private static String ticket_Id;
    private static String strAdminFlg;
    private static String strUserName;
    private Connection conectionObj;
    private String strTCategory;
    private String strTPriority;
    private String strTCreatedBy;
    private String strTComment;
    private String strTStatus;
    private String strTDesc;
    private JComboBox comboBox_ticketCatgry;
    private JComboBox comboBox_ticketPriorty;
    private JComboBox comboBox_ticketStatus;
    private JTextArea textArea_ticketDesc;
    private JTextArea textArea_adminRemarks;

    /**
     * Launch the application.
     */
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Adminupdate frame = new
Adminupdate(ticket_Id,strAdminFlg,strUserName);
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    /**
     * Create the frame.
     * @param strUserNme

```

```
* @param strAdminFlg
* @param ticket_Id
*/
public Adminupdate(String ticket_Id, String strAdminFlg, String
strUserNme) {
    setAdminFields(ticket_Id,strAdminFlg,strUserNme);
    Adminlandingpage adminLandngObj = new Adminlandingpage(strUserName);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(100, 100, 866, 513);
    contentPane = new JPanel();
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
    contentPane.setBackground(Color.getHSBColor(1.4f, 1.0f, 1.6f));
    setContentPane(contentPane);
    contentPane.setLayout(null);

    JLabel lblViewupdateTicketBy = new JLabel("");
    lblViewupdateTicketBy.setFont(new Font("Tahoma", Font.BOLD, 14));
    lblViewupdateTicketBy.setBounds(352, 11, 244, 34);
    contentPane.add(lblViewupdateTicketBy);

    JLabel lblTicketCreatedBy = new JLabel("Ticket Created By");
    lblTicketCreatedBy.setFont(new Font("Tahoma", Font.BOLD, 12));
    lblTicketCreatedBy.setBounds(182, 110, 117, 21);
    contentPane.add(lblTicketCreatedBy);

    JLabel lblTicketCategory = new JLabel("Ticket Category");
    lblTicketCategory.setFont(new Font("Tahoma", Font.BOLD, 12));
    lblTicketCategory.setBounds(182, 161, 107, 14);
    contentPane.add(lblTicketCategory);

    JLabel lblTicketPriority = new JLabel("Ticket Priority");
    lblTicketPriority.setFont(new Font("Tahoma", Font.BOLD, 12));
    lblTicketPriority.setBounds(182, 211, 107, 14);
    contentPane.add(lblTicketPriority);

    JLabel lblTicketStatus = new JLabel("Ticket Status");
    lblTicketStatus.setFont(new Font("Tahoma", Font.BOLD, 12));
    lblTicketStatus.setBounds(182, 259, 90, 14);
    contentPane.add(lblTicketStatus);

    JLabel lblNewLabel = new JLabel("Ticket Description");
    lblNewLabel.setFont(new Font("Tahoma", Font.BOLD, 12));
    lblNewLabel.setBounds(182, 312, 117, 14);
    contentPane.add(lblNewLabel);

    JLabel lblAdminRemarks = new JLabel("Admin Remarks");
    lblAdminRemarks.setFont(new Font("Tahoma", Font.BOLD, 12));
    lblAdminRemarks.setBounds(182, 365, 107, 14);
    contentPane.add(lblAdminRemarks);

    //Lable text which welcomes the logged in user
    JLabel lblNewLabel_1 = new JLabel("Welcome " +strUserName);
    lblNewLabel_1.setFont(new Font("Tahoma", Font.BOLD, 14));
    lblNewLabel_1.setBounds(384, 56, 155, 14);
    contentPane.add(lblNewLabel_1);

    txtFld_ticketCreatedBy = new JTextField();
```

```
txtFld_ticketCreatedBy.setEditable(false);
txtFld_ticketCreatedBy.setEnabled(false);
txtFld_ticketCreatedBy.setBounds(362, 110, 211, 21);
contentPane.add(txtFld_ticketCreatedBy);
txtFld_ticketCreatedBy.setColumns(10);
txtFld_ticketCreatedBy.setText(strUserName);

/*
 * Initializes the ticket category
 */
final DefaultComboBoxModel ticketCategoryModel = new
DefaultComboBoxModel(
    new String[]{"SELECT","OS ISSUE",
        "NETWORKING ISSUE","HARDWARE ISSUE",
        "SOFTWARE INSTALL","SOFTWARE UNINSTALL",
        "INTENET CONNECTIVITY ISSUE",
        "LOGIN ISSUE","COMPUTER ALLOCATE REQUEST",
        "COMUTER RESTART ISSUE","OTHERS"});

comboBox_ticketCatgry = new JComboBox(ticketCategoryModel);
comboBox_ticketCatgry.setEnabled(false);
comboBox_ticketCatgry.setBounds(362, 158, 211, 21);
contentPane.add(comboBox_ticketCatgry);

/**
 * Initializes the ticket priority
 */
final DefaultComboBoxModel ticketPriorityModel = new
DefaultComboBoxModel(
    new String[]{"SELECT","LOW","MEDIUM","HIGH"});

comboBox_ticketPriorty = new JComboBox(ticketPriorityModel);
comboBox_ticketPriorty.setEnabled(false);
comboBox_ticketPriorty.setBounds(362, 208, 211, 21);
contentPane.add(comboBox_ticketPriorty);

/*
 * Initializes the ticket status
 */
final DefaultComboBoxModel ticketStatusModel = new
DefaultComboBoxModel(
    new String[]{"SELECT","OPEN","CLOSED"});

comboBox_ticketStatus = new JComboBox(ticketStatusModel);
comboBox_ticketStatus.setEditable(true);
comboBox_ticketStatus.setBounds(362, 254, 211, 23);
contentPane.add(comboBox_ticketStatus);

textArea_ticketDesc = new JTextArea();
textArea_ticketDesc.setEditable(false);
textArea_ticketDesc.setEnabled(false);
textArea_ticketDesc.setBounds(362, 308, 329, 39);
contentPane.add(textArea_ticketDesc);

textArea_adminRemarks = new JTextArea();
textArea_adminRemarks.setBounds(362, 361, 329, 39);
```

```

contentPane.add(textArea_adminRemarks);

/*
 * Action listener for the submit button which
 * validates the all the records before updating
 * it to the database
 */
JButton btnSubmit = new JButton("Submit");
btnSubmit.setFont(new Font("Tahoma", Font.BOLD, 12));

/*
 * If the user action is UPDATE, then just update
 * the records to the db
 */
btnSubmit.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0)
    {
        if(strAdminFlg.equalsIgnoreCase("UPDATE"))
        {
            updateTicketInformation();

            contentPane.setVisible(false);
            adminLandngObj.setVisible(true);
        }

        /*****
        * Using this method admin updates the ticket information
        * for the selected ticket from the previous screen
        *****/
        private void updateTicketInformation() {

            try{
                conectionObj = SqlConnection.dbConnector();
                conectionObj.setAutoCommit(false);

                String strQuery = "Update CMUNEGOW_TBL_TICKET "
                    + "SET TICKET_STATUS = ? , "
                    + "TICKET_COMMENT = ? , "
                    + "TICKET_CLOSE_DATE = ? "
                    + "WHERE TICKET_ID = ?";

                PreparedStatement psObj =
conectionObj.prepareStatement(strQuery);
                psObj.setString(1, (String)
comboBox_ticketStatus.getSelectedItem());
                SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-
MM-dd HH:mm:ss");
                Date date = new Date();
                String strtickCloseDate = dateFormat.format(date);
                psObj.setString(2, textArea_adminRemarks.getText());
                psObj.setString(3, strtickCloseDate);
                psObj.setString(4, ticket_Id);

                psObj.execute();

```

```

        conectionObj.setAutoCommit(true);

        if (comboBox_ticketStatus.getSelectedItem().equals("CLOSED"))
        {
            JOptionPane.showMessageDialog(null, "Ticket Id "
+ticket_Id
                                +"successfully closed and updated!");
        }
        else
        {
            JOptionPane.showMessageDialog(null, "Ticket "
+ticket_Id
                                + " successfully updated!");
        }

    }

    catch (Exception e) {
        JOptionPane.showMessageDialog(null, e.getMessage() );
    }

}

});
btnSubmit.setBounds(318, 441, 89, 23);
contentPane.add(btnSubmit);

/*
 * If the user clicks the Cancel button, then navigate
 * to the admin landing page
 */
JButton btnNewButton = new JButton("Cancel");
btnNewButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent evt) {

        contentPane.setVisible(false);
        adminLandngObj.setVisible(true);

    }
});
btnNewButton.setFont(new Font("Tahoma", Font.BOLD, 12));
btnNewButton.setBounds(438, 441, 89, 23);
contentPane.add(btnNewButton);

fetchRecords();

//If th user clicks the view button, then
//make the fields non editable and disabled
if (strAdminFlg.equalsIgnoreCase("VIEW"))
{

    lblViewupdateTicketBy.setText("View Ticket By Admin");
    comboBox_ticketStatus.setEnabled(false);
    textArea_adminRemarks.setEnabled(false);
}

//If the user clicks the update button, the
//make the fields editable

```

```
        else if(strAdminFlg.equalsIgnoreCase("UPDATE"))
        {
            lblViewupdateTicketBy.setText("Update Ticket By Admin");
            comboBox_ticketStatus.setEnabled(true);
            textArea_adminRemarks.setEnabled(true);
        }
    }

    /*
     * Method which initialized the ticket id,admin action flag and
     * user name
     */
    private void setAdminFields(String ticketId, String strAdminFlag,
        String strUserNme)
    {
        ticket_Id = ticketId;
        strAdminFlg = strAdminFlag;
        strUserName = strUserNme;
    }

    /**
     * Fetch the records and display to the user in the
     * fields based on the ticket id selected from the
     * previous screen
     */
    private void fetchRecords() {

        try
        {
            conectionObj = SqlConnection.dbConnector();

            String selectQuery = "SELECT TICKET_CATEGORY,
TICKET_PRIORITY,TICKET_COMMENT,"
                + "TICKET_STATUS,TICKET_DESCRIPTION,TICKET_CREATEDBY "
                + "FROM CMUNEGOW_TBL_TICKET "
                + "WHERE TICKET_ID = ?";

            PreparedStatement psObj =
conectionObj.prepareStatement(selectQuery);
            psObj.setString(1, ticket_Id);

            ResultSet rs = psObj.executeQuery();
            while(rs.next())
            {
                strTCreatedBy = rs.getString("TICKET_CREATEDBY");
                strTCategory = rs.getString("TICKET_CATEGORY");
                strTComment = rs.getString("TICKET_COMMENT");
                strTPriority = rs.getString("TICKET_PRIORITY");
                strTStatus = rs.getString("TICKET_STATUS");
                strTDesc = rs.getString("TICKET_DESCRIPTION");
            }

            comboBox_ticketCatgry.setSelectedItem(strTCategory);
            comboBox_ticketPriority.setSelectedItem(strTPriority);
        }
    }
}
```

```

        comboBox_ticketStatus.setSelectedItem(strTStatus);
        txtFld_ticketCreatedBy.setText(strTCreatedBy);
        textArea_ticketDesc.setText(strTDesc);
        textArea_adminRemarks.setText(strTComment);

    }
    catch (Exception e)
    {
        JOptionPane.showMessageDialog(null, e.getMessage());
    }

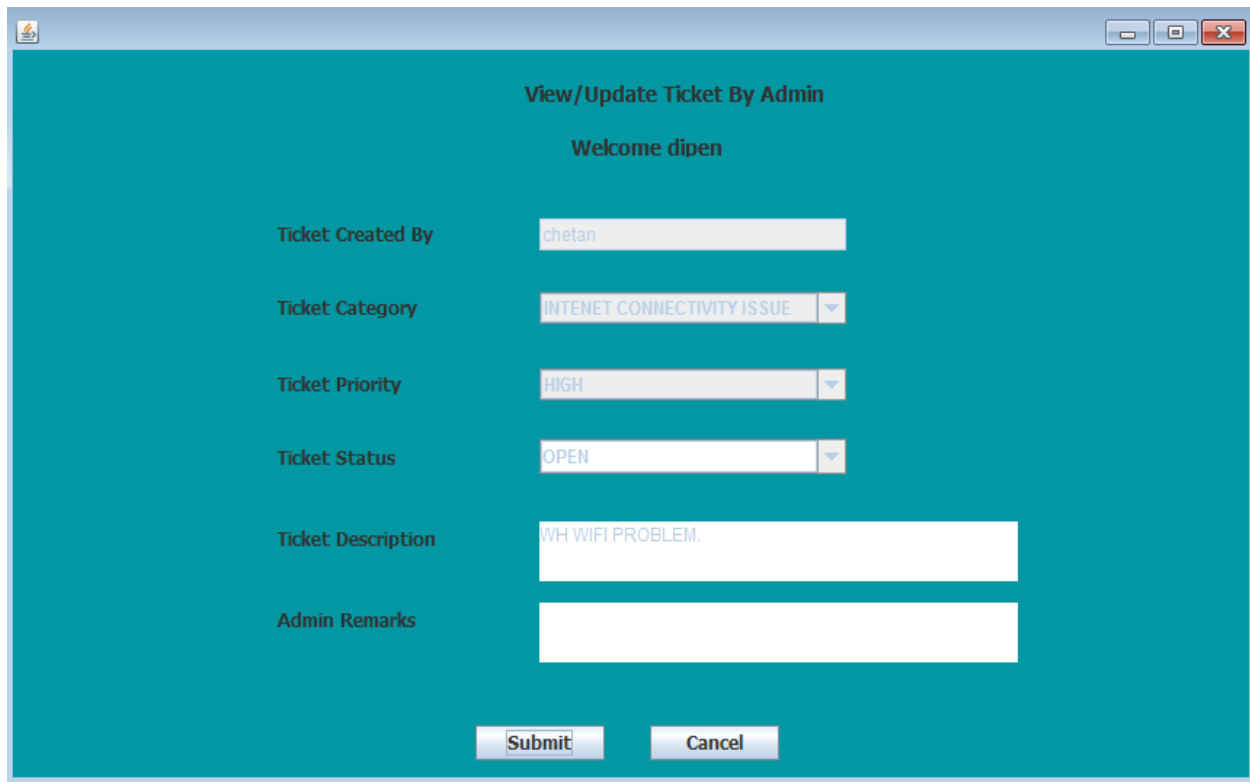
}
}

```

Admin View Snapshot:



TICKET_ID	TICKET_PRIORITY	TICKET_CATEGORY	TICKET_STATUS	TICKET_DESCRIPTION	TICKET_CREATEDBY	TICKET_OPEN_DATE
1	MEDIUM	OS ISSUE	OPEN	please resolve the iss...	chetan	2014-11-29
2	LOW	NETWORKING ISSUE	OPEN	Wifi Notworking	chetan	2014-11-29
3	MEDIUM	HARDWARE ISSUE	OPEN	Motherboard Issue.	chetan	2014-11-29
4	MEDIUM	SOFTWARE INSTALL	OPEN	Install Visual Studio fo...	chetan	2014-11-29
5	LOW	INTENET CONNECTI...	OPEN	Wifi Not working.	chetan	2014-11-29
6	HIGH	LOGIN ISSUE	OPEN	Not able to login to my...	chetan	2014-11-29
7	HIGH	COMPUTER ALLOCA...	OPEN	Allocate Computer.	chetan	2014-11-29
9	MEDIUM	OTHERS	OPEN	Need the credentials f...	chetan	2014-11-29
10	MEDIUM	NETWORKING ISSUE	OPEN	Slow Internet Speed.	chetan	2014-11-29
12	HIGH	INTENET CONNECTI...	OPEN	WH WIFI PROBLEM.	chetan	2014-11-29



View/Update Ticket By Admin

Welcome dipen

Ticket Created By: chetan

Ticket Category: INTERNET CONNECTIVITY ISSUE

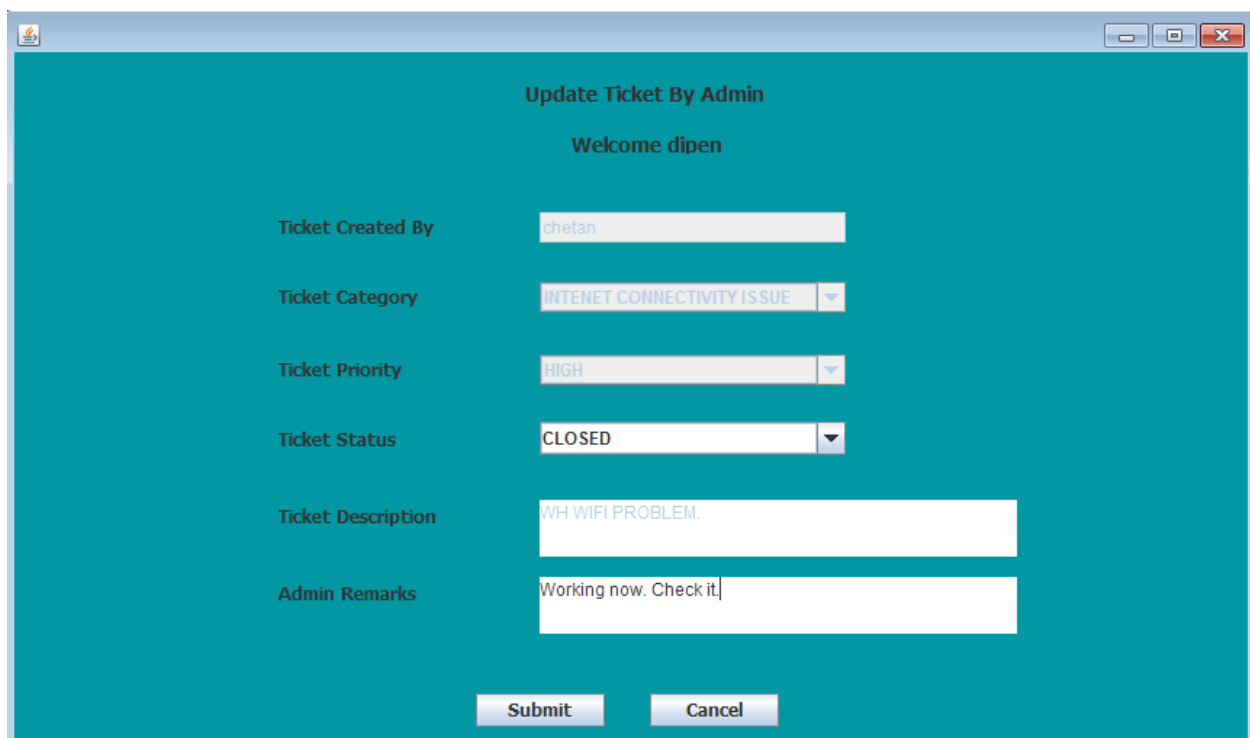
Ticket Priority: HIGH

Ticket Status: OPEN

Ticket Description: WH WIFI PROBLEM.

Admin Remarks:

Admin Update Snapshot:



Update Ticket By Admin

Welcome dipen

Ticket Created By: chetan

Ticket Category: INTERNET CONNECTIVITY ISSUE

Ticket Priority: HIGH

Ticket Status: CLOSED

Ticket Description: WH WIFI PROBLEM.

Admin Remarks: Working now. Check it

Update Ticket By Admin

Welcome dipen

Ticket Created By: chetan

Ticket Category: INTERNET CONNECTIVITY ISSUE

Ticket Priority: HIGH

Ticket Status: CLOSED

Ticket Description: WH WIFI PROBLEM.

Admin Remarks: Working now. Check it.

Submit Cancel

Message: Ticket Id 12 successfully closed and updated! OK

Hello dipen! Welcome To Trouble Ticket Admin Landing Page

Log Out

TICKET_ID	TICKET_PRIORITY	TICKET_CATEGORY	TICKET_STATUS	TICKET_DESCRIP...	TICKET_CREATED...	TICKET_OPEN_DA...	TICKET_CLOSE_D...
1	MEDIUM	OS ISSUE	OPEN	please resolve the ...	chetan	2014-11-29	
2	LOW	NETWORKING ISS...	OPEN	Wifi Notworking	chetan	2014-11-29	
3	MEDIUM	HARDWARE ISSUE	OPEN	Motherboard Issue.	chetan	2014-11-29	
4	MEDIUM	SOFTWARE INSTA...	OPEN	Install Visual Studi...	chetan	2014-11-29	
5	LOW	INTERNET CONNE...	OPEN	Wifi Not working.	chetan	2014-11-29	
6	HIGH	LOGIN ISSUE	OPEN	Not able to login to ...	chetan	2014-11-29	
7	HIGH	COMPUTER ALLO...	OPEN	Allocate Computer.	chetan	2014-11-29	
9	MEDIUM	OTHERS	OPEN	Need the credentia...	chetan	2014-11-29	
10	MEDIUM	NETWORKING ISS...	OPEN	Slow Internet Speed.	chetan	2014-11-29	
12	HIGH	INTERNET CONNE...	CLOSED	WH WIFI PROBLEM.	chetan	2014-11-29	2014-11-29

View Update Refresh

Helper Files and Validations Code:

Get SQL Connection:

Contains the JDBC connection code to get the SQL connection to the MYSQL papademas/tickets database.

```
package com.iit.tts.db;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

import javax.swing.JOptionPane;

/****
 * This class loads the mysql Jdbc driver and
 * returns the SQL Connection object
 * @author Chetan Munegowda and Abhay Manoli
 */

public class SqlConnetion {

    private static Connection conn = null;

    /****
     * Static method which returns the connection
     * to the database url "jdbc:mysql://www.papademas.net/tickets"
     * @return Connection Object conn
     */
    public static Connection dbConnector()
    {
        try
        {
            // This will load the MySQL driver,
            //each DB has its own driver
            Class.forName("com.mysql.jdbc.Driver");

            conn = DriverManager
                .getConnection("jdbc:mysql://www.papademas.net"
                    + "/tickets?"
                    + "user=root&password=jamesp");

            return conn;

        }
        catch(SQLException e)
        {

            JOptionPane.showMessageDialog(null, e.getMessage());
            return null;

        }
        catch(ClassNotFoundException e)
```

```

        {
            JOptionPane.showMessageDialog(null, e.getMessage());
            return null;
        }
    }
}

```

Create Database:

Creates the necessary code to create tables required for the IT trouble ticket system

```

package com.iit.tts.db;

import java.sql.Connection;
import java.sql.Statement;

import javax.swing.JOptionPane;

/*****
 * This class creates the database tables which are
 * required for the Trouble Ticket Management System
 *
 * @author Chetan Munegowda and Abhay Manoli
 *
 *****/
public class CreateDatabase {

    Connection conn = null;
    Statement statement = null;

    public CreateDatabase()
    {
        conn = SqlConnection.dbConnector();
    }

    public void CreateTicketsDataBase() throws Exception
    {
        try
        {
            statement = conn.createStatement();

            String sql = "CREATE TABLE CMUNEGOW_TBL_USERS " +
                "(USER_ID INTEGER NOT NULL AUTO_INCREMENT , "+
                " USER_NAME VARCHAR(20) NOT NULL UNIQUE , " +
                " FIRST_NAME VARCHAR(40) NOT NULL , " +
                " LAST_NAME VARCHAR(40) NOT NULL , "+
                " USER_PASSWORD VARCHAR(30) NOT NULL , "+
                " USER_EMAIL VARCHAR(50) NOT NULL , "+
                " PHONE_NUMBER LONG NOT NULL , "+
                " USER_ROLE VARCHAR(10) NOT NULL , " +
                " PRIMARY KEY(USER_ID)) ";

            statement.executeUpdate(sql);
        }
    }
}

```

```

sql = "CREATE TABLE CMUNEGOW_TBL_USERADDRESS (" +
      "ADDRESS_ID INTEGER NOT NULL AUTO_INCREMENT , "+
      "USER_NAME VARCHAR(20) , "+
      "APT_NUMBER INTEGER , "+
      "STREET_NAME VARCHAR(50) , "+
      "STATE VARCHAR(25) , "+
      "CITY VARCHAR(25) , "+
      "PIN_CODE INTEGER , "+
      "PRIMARY KEY (ADDRESS_ID) , "+
      "FOREIGN KEY (USER_NAME) REFERENCES
CMUNEGOW_TBL_USERS (USER_NAME) ) ";
statement.executeUpdate(sql);

sql = "CREATE TABLE CMUNEGOW_TBL_TICKET "+
      "(TICKET_ID INTEGER NOT NULL AUTO_INCREMENT, "+
      "TICKET_PRIORITY VARCHAR(10), "+
      "TICKET_COMMENT VARCHAR(100), "+
      "TICKET_CATEGORY VARCHAR(30), "+
      "TICKET_STATUS VARCHAR(10), "+
      "TICKET_DESCRIPTION VARCHAR(100), "+
      "TICKET_OPEN_DATE DATE, "+
      "TICKET_CLOSE_DATE DATE, "+
      "TICKET_CREATEDBY VARCHAR(20), "+
      "TICKET_ASSIGNEE VARCHAR(20), "+
      "TICKET_ISPURGED CHAR(1), "+
      "PRIMARY KEY (TICKET_ID), "+
      "FOREIGN KEY (TICKET_ASSIGNEE) REFERENCES
CMUNEGOW_TBL_USERS (USER_NAME), "+
      "FOREIGN KEY (TICKET_CREATEDBY) REFERENCES
CMUNEGOW_TBL_USERS (USER_NAME) ) ";
statement.executeUpdate(sql);

JOptionPane.showMessageDialog(null, "Created table "
      + "in given database...");

//end create table
}
catch (Exception e)
{
    JOptionPane.showMessageDialog(null, e.getMessage());
}
}
}

```

Integer Field Code:

Class which helps us to create the integer field.

```

package com.iit.tts.troubletickethelper;
import javax.swing.JTextField;
import javax.swing.text.AttributeSet;
import javax.swing.text.BadLocationException;

```

```
import javax.swing.text.Document;
import javax.swing.text.PlainDocument;

/**
 * A JTextField that accepts only integers.
 *
 * @author Chetan Munegowda and Abhay Manoli
 */
public class IntegerField extends JTextField {

    public IntegerField() {
        super();
    }

    public IntegerField( int cols ) {
        super( cols );
    }

    @Override
    protected Document createDefaultModel() {
        return new UpperCaseDocument();
    }

    static class UpperCaseDocument extends PlainDocument {

        @Override
        public void insertString( int offs, String str, AttributeSet a )
            throws BadLocationException {

            if ( str == null ) {
                return;
            }

            char[] chars = str.toCharArray();
            boolean ok = true;

            for ( int i = 0; i < chars.length; i++ ) {

                try {
                    Integer.parseInt( String.valueOf( chars[i] ) );
                } catch ( NumberFormatException exc ) {
                    ok = false;
                    break;
                }

            }

            if ( ok )
                super.insertString( offs, new String( chars ), a );
        }
    }
}
```

Helper File For Validation:

Helper file to validate Email is of valid format using pattern matching and validate text fields and drop downs are not left empty.

```
package com.iit.tts.troubletickethelper;

import java.util.regex.Matcher;
import java.util.regex.Pattern;

/*****
 * This class is a helper utility file which
 * validates the Text fields
 *
 * @author Chetan Munegowda and Abhay Manoli
 *
 *****/
public class Troubletickethelper
{

    private static final String EMAIL_PATTERN =
        "^[_A-Za-z0-9-\\+]+(\\.\\[_A-Za-z0-9-\\+]+)*@" +
        "[A-Za-z0-9-]+(\\.\\[_A-Za-z0-9-\\+]+)*\\.\\[_A-Za-z]{2,}\\$";

    /**
     * Returns the status of the field passed as parameter. If the field
     * is null it returns true else false
     * @param strField
     * @return
     */
    public static boolean validateTextField(String strField)
    {
        if((strField == null) || (strField.equals("")))
        {
            return true;
        }
        else
        {
            return false;
        }
    }

    /**
     * Checks the email field is of pattern mentioned in
     * the Email Pattern
     * @param strEmail
     * @return
     *****/
    public static boolean validateEmail(String strEmail)
    {
        Boolean status = false;
        if(strEmail!=null)
        {
            Pattern pattern = Pattern.compile(EMAIL_PATTERN);
            Matcher matcher = pattern.matcher(strEmail);
            status = matcher.matches();
        }
    }
}
```

```

    }
    return status;
}

/*****
 * Validates the Drop down field is selected to proper
 * value ir not
 * @param strField
 * @return
 *****/
public static boolean validateDropDown(String strField) {

    if(strField.equalsIgnoreCase("SELECT"))
    {
        return true;
    }
    else
    {
        return false;
    }
}
}

```

Helper File For restricting text field limit:

Helper file for restricting the text field to certain limit.

```

package com.iit.tts.troubletickethelper;

import javax.swing.text.AttributeSet;
import javax.swing.text.BadLocationException;
import javax.swing.text.PlainDocument;

/*****
 * Class which restricts the fields in the document
 * to the integer limit set
 * @author Chetan Munegowda and Abhay Manoli
 *****/
public class JTextFieldLimit extends PlainDocument {
    private int limit;
    public JTextFieldLimit(int limit) {
        super();
        this.limit = limit;
    }

    JTextFieldLimit(int limit, boolean upper) {
        super();
        this.limit = limit;
    }

    public void insertString(int offset, String str, AttributeSet attr)
    throws BadLocationException {
        if (str == null)
            return;

        if ((getLength() + str.length()) <= limit) {

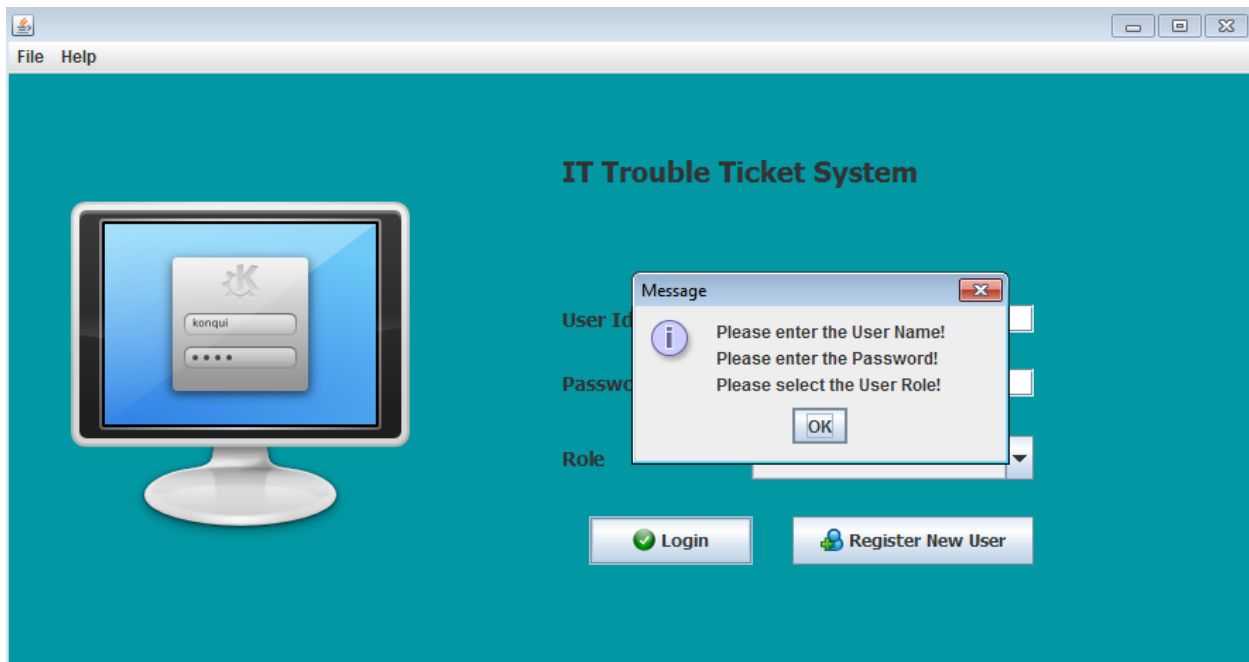
```

```
        super.insertString(offset, str, attr);  
    }  
}
```

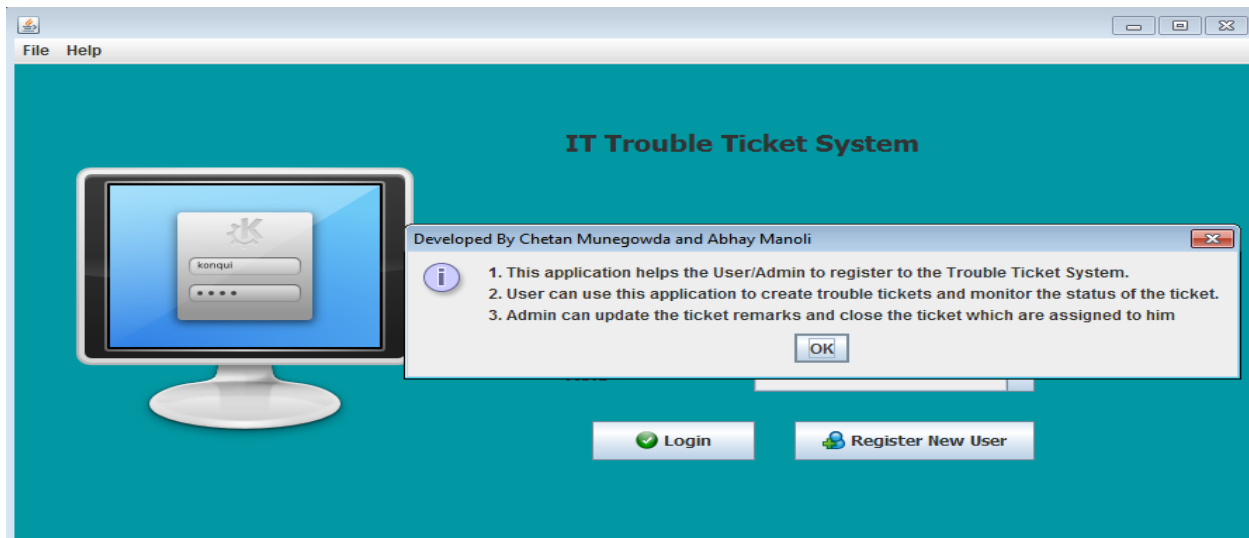
Validations Snapshot:

Login Screen:

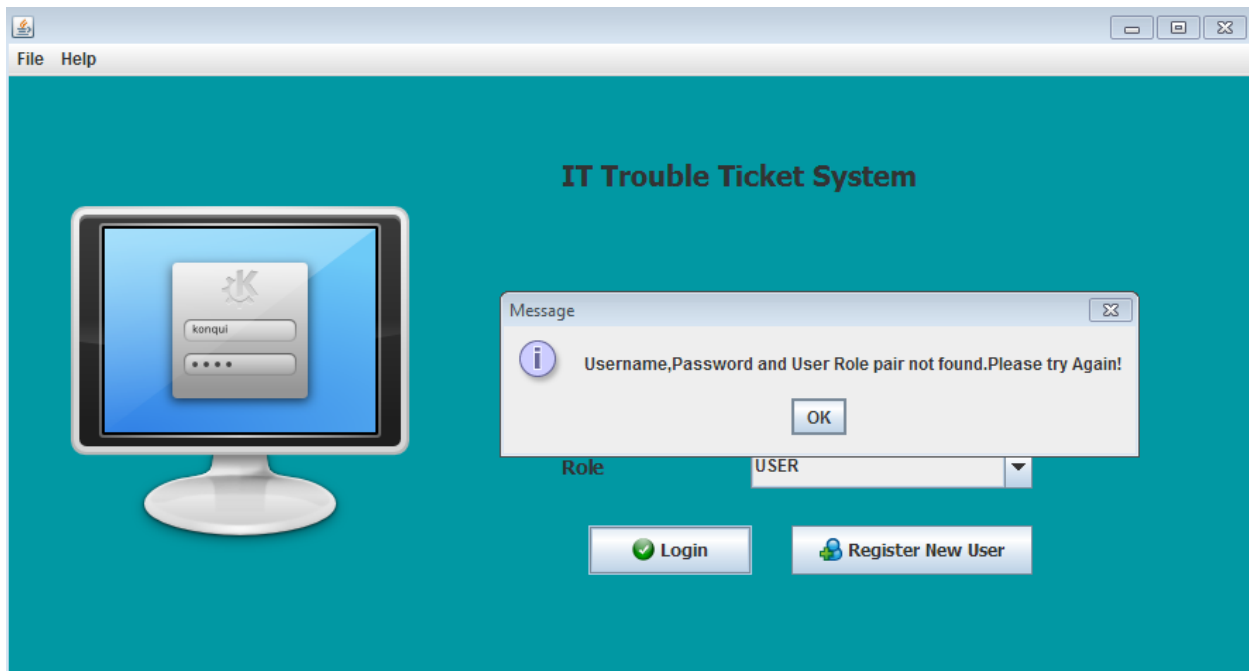
When the fields are empty and the user clicks on login button:



About Trouble Ticket Management System:



Invalid Credentials:



Create User/Admin Validation Screen Shot:

Empty Field Validations:

The screenshot shows a web application window titled "Create User/Amin". The form contains several input fields: "User Name", "Last Name", "Password", "Role" (with radio buttons for "User" and "Admin"), "Street Name", and "Pincode". A modal dialog box titled "Message" is displayed in the center, containing the following text: "Please provide User Name!", "Please provide First Name!", "Please provide Last Name!", "Please provide Role!", "Please provide Email Address!", "Please provide valid E-mail Address!", "Please provide City!", "Please provide Street Name!", and "Please select proper State!". The dialog has an "OK" button. Below the form are "Submit" and "Cancel" buttons.

Invalid Email Address Validation:

The screenshot shows the same "Create User/Amin" form, but with some fields filled: "User Name" is "abhara", "Last Name" is "manoli", "Password" is masked with dots, "Role" is "User", "Street Name" is "Cottage Go", and "Pincode" is "78908". The "First Name" field is "abhay", "Cellular Number" is "1636916361", "Email" is "abhay909", "House Number" is "214", and the "State" dropdown is set to "California". A modal dialog box titled "Message" is displayed, containing the text: "Please provide valid E-mail Address!". The dialog has an "OK" button. Below the form are "Submit" and "Cancel" buttons.

Integer Field Validation:

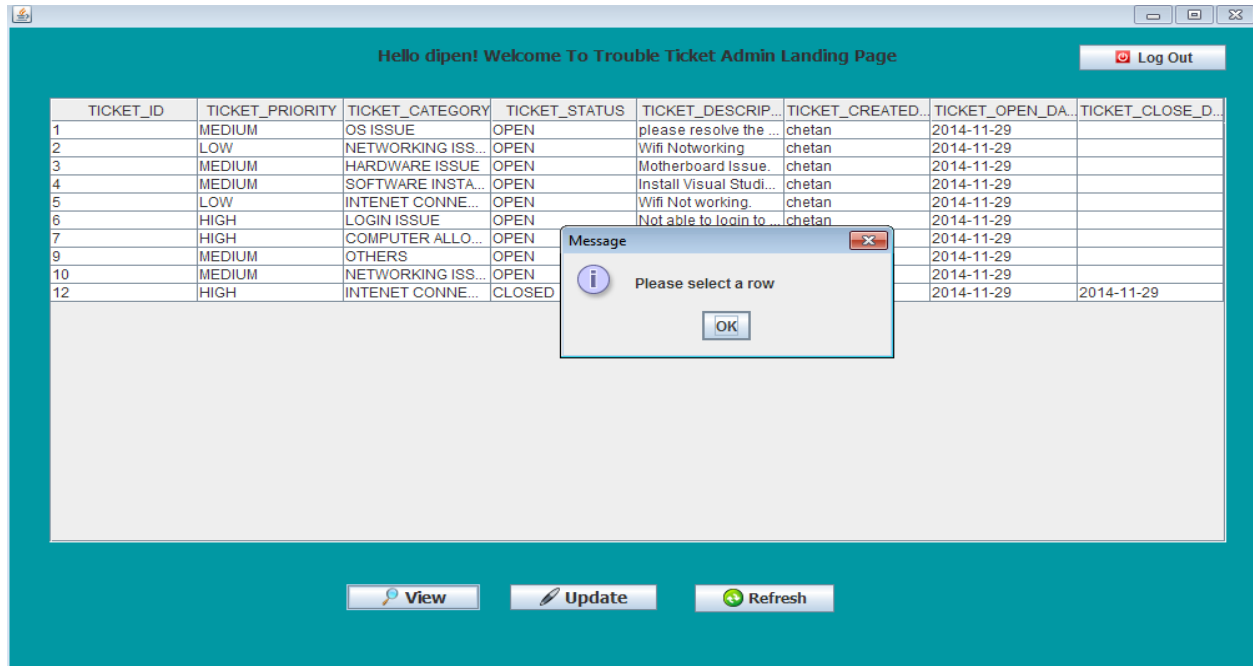
The screenshot shows a web form titled "Create User/Amin" with a teal background. The form contains several input fields: User Name (abhara), First Name (abhay), Last Name (manoli), Cellular Number (empty), Password (masked with dots), Role (radio buttons for User and Admin, with User selected), Street Name (Cottage Gorge), Pincode (78908), City (San Francisco), and Email (abhay@gmail.com). A modal message box is displayed in the center, stating: "Message * Enter only numeric digits(0-9)". The message box has an information icon, a close button (X), and an OK button. Below the form are "Submit" and "Cancel" buttons.

Duplicate UserName Validation:

The screenshot shows the same "Create User/Amin" form, but with different values: User Name (abhay), First Name (abhay), Last Name (manoli), Cellular Number (3124780987), Password (masked with dots), Role (radio buttons for User and Admin, with User selected), Street Name (Cottage Gorge), Pincode (78908), City (San Francisco), and Email (abhay@gmail.com). A modal message box is displayed in the center, stating: "Message User Name already exists. Please enter another user name!". The message box has an information icon, a close button (X), and an OK button. Below the form are "Submit" and "Cancel" buttons.

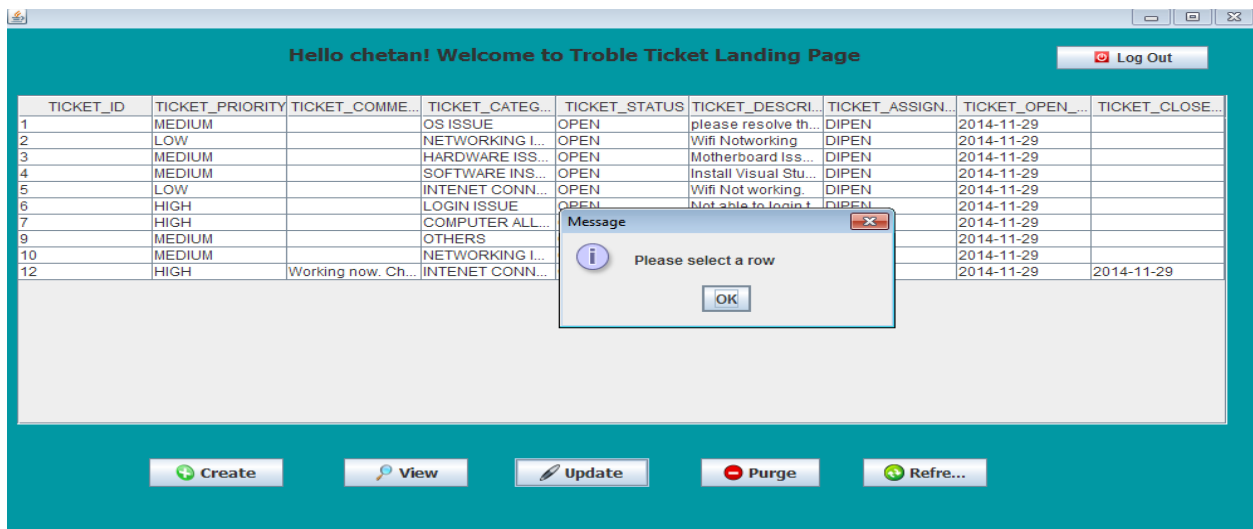
Admin Langing Page Screen Validation:

If admin clicks view or update button without selecting record, then alert is popped to User.



Trouble Ticket Landing Page Validation:

If the user clicks view or update or purge button without selecting record, then alert is thrown.



Create Ticket Mandatory Field Validation:

Create Trouble Ticket

User Name: chetan

Ticket Category: SELECT

Ticket Priority: SELECT

Ticket Description:

Message

- Please select valid ticket category!
- Please select valid ticket priority!
- Please provide ticket description!

OK

Submit Cancel Clear

Session Information: Displaying the logged in user name

Hello dipen! Welcome To Trouble Ticket Admin Landing Page

Log Out

TICKET_ID	TICKET_PRIORITY	TICKET_CATEGORY	TICKET_STATUS	TICKET_DESCRIP...	TICKET_CREATED...	TICKET_OPEN_DA...	TICKET_CLOSE_D...
1	MEDIUM	OS ISSUE	OPEN	please resolve the ...	chetan	2014-11-29	
2	LOW	NETWORKING ISS...	OPEN	Wifi Notworking	chetan	2014-11-29	
3	MEDIUM	HARDWARE ISSUE	OPEN	Motherboard Issue...	chetan	2014-11-29	
4	MEDIUM	SOFTWARE INSTA...	OPEN	Install Visual Studi...	chetan	2014-11-29	
5	LOW	INTENET CONNE...	OPEN	Wifi Not working.	chetan	2014-11-29	
6	HIGH	LOGIN ISSUE	OPEN	Not able to login to ...	chetan	2014-11-29	
7	HIGH	COMPUTER ALLO...	OPEN	Allocate Computer.	chetan	2014-11-29	
9	MEDIUM	OTHERS	OPEN	Need the credentia...	chetan	2014-11-29	
10	MEDIUM	NETWORKING ISS...	OPEN	Slow Internet Speed.	chetan	2014-11-29	
12	HIGH	INTENET CONNE...	CLOSED	WH WIFI PROBLEM.	chetan	2014-11-29	2014-11-29
13	LOW	INTENET CONNE...	OPEN	internet speed slow	agm	2014-11-29	

View Update Refresh

Future Work:

- Developing the IT trouble ticket system completely using android framework.
- Adding more Field level validations.
- Making the look and feel of the IT trouble ticket system much better.
- Instead of using refresh button, user can see the updated changes when the page loads.
- Adding the triggers.
- Transferring the ticket from one admin to other admin.
- Giving the preference for the user to open the ticket instead of re-raising the ticket, if the ticket is not solved.
- Giving max duration for the admin, to close the ticket.

References:

- <http://stackoverflow.com/>
- https://www.youtube.com/watch?v=oeswfZz4IW0&src_vid=r8Qiz9Bn1Ag&feature=iv&annotation_id=annotation_5868455
- ITMD 411-05 Intermediate Software Development course content-
<https://blackboard.iit.edu/>