

HELPDESK MANAGEMENT SYSTEM



A PROJECT REPORT

Submitted by

PREETHI B(2303811710422121)

in partial fulfillment of requirements for the award of the course

CGB1201 - JAVA PROGRAMMING

In

COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM-621112

NOVEMBER- 2024

**K.RAMAKRISHNAN COLLEGE OF TECHNOLOGY
(AUTONOMOUS)**

SAMAYAPURAM-621112

BONAFIDECERTIFICATE

Certified that this project report on “**HELPDESK MANAGEMENT SYSTEM** ” is the bonafide work of **PREETHI B (2303811710422121)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

CGB1201-JAVA PROGRAMMING
Dr.A.DELPHIN CAROLINA RANI, M.E.,Ph.D.,
HEAD OF THE DEPARTMENT
PROFESSOR

SIGNATURE

Dr.A.Delphin Carolina Rani, M.E.,Ph.D.,

HEAD OF THE DEPARTMENT

PROFESSOR

Department of CSE

K.Ramakrishnan College of Technology
(Autonomous)

Samayapuram-621112.

CGB1201-JAVA PROGRAMMING
Mrs.K.VALLI PRIYADHARSHINI, M.E.,(Ph.D.),
SUPERVISOR
ASSISTANT PROFESSOR

SIGNATURE

Mrs.K.Valli Priyadharshini, M.E.,(Ph.D.),

SUPERVISOR

ASSISTANT PROFESSOR

Department of CSE

K.Ramakrishnan College of Technology
(Autonomous)

Samayapuram-621112.

Submitted for the viva-voce examination held on 03.12.2024

CGB1201- JAVA PROGRAMMING
Mr.MANJARMANNAN A, M.E.,
INTERNAL EXAMINER
ASSISTANT PROFESSOR

INTERNAL EXAMINER

CGB1201-JAVA PROGRAMMING
Mrs.P.ARUNA PRIYA, M.E.,
EXTERNAL EXAMINER
ASSISTANT PROFESSOR
8104-DSEC, PERAMBALUR.

EXTERNAL EXAMINER

DECLARATION

I declare that the project report on “**HELPDESK MANAGEMENT SYSTEM**” is the result of original work done by us and best of our knowledge, similar work has not been submitted to “**ANNA UNIVERSITY CHENNAI**” for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **CGB1201- JAVA PROGRAMMING**.



PREETHI B

Place: Samayapuram

Date: 03/12/2024

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution “**K.Ramakrishnan College of Technology (Autonomous)**”, for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN, B.E.**, for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.**, for forwarding to our project and offering adequateduration in completing our project.

I would like to thank **Dr. N. VASUDEVAN, M.Tech., Ph.D.**, Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. A. DELPHIN CAROLINA RANI, M.E.,Ph.D.**, Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project supervisor **Mrs.K.VALLI PRIYADHARSHINI, M.E.,(Ph.D.)**, Department of **COMPUTER SCIENCE AND ENGINEERING**, for his incalculable suggestions, creativity, assistance and patiencewhich motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

VISION OF THE INSTITUTION

To serve the society by offering top-notch technical education on par with global standards

MISSION OF THE INSTITUTION

- Be a center of excellence for technical education in emerging technologies by exceeding the needs of the industry and society.
- Be an institute with world class research facilities
- Be an institute nurturing talent and enhancing the competency of students to transform them as all-round personality respecting moral and ethical values

VISION OF DEPARTMENT

To be a center of eminence in creating competent software professionals with research and innovative skills.

MISSION OF DEPARTMENT

M1: Industry Specific: To nurture students in working with various hardware and software platforms inclined with the best practices of industry.

M2: Research: To prepare students for research-oriented activities.

M3: Society: To empower students with the required skills to solve complex technological problems of society.

PROGRAM EDUCATIONAL OBJECTIVES

1. PEO1: Domain Knowledge

To produce graduates who have strong foundation of knowledge and skills in the field of Computer Science and Engineering.

2. PEO2: Employability Skills and Research

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

3. PEO3: Ethics and Values

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO 1: Domain Knowledge

To analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.

PSO 2: Quality Software

To apply software engineering principles and practices for developing quality software for scientific and business applications.

PSO 3: Innovation Ideas

To adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing/novel problems

PROGRAM OUTCOMES (POs)

Engineering students will be able to:

- 1. Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
- 2. Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences
- 3. Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations
- 4. Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions

5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

ABSTRACT

The **HelpDesk Management System** is designed to streamline customer support operations for companies, ensuring efficient handling of support requests, systematic tracking of ticket statuses, and improved response times. This system facilitates interaction between customers and company representatives, providing a centralized platform for managing issues. Customers can log in to create and monitor the status of their tickets, while company representatives can view all tickets, update statuses, and resolve issues. Key features include role-based authentication, a user-friendly interface for ticket management, and a workflow that tracks tickets through their lifecycle (Open → In Progress → Resolved). The system enhances transparency, promotes accountability, and helps companies deliver better customer service. By integrating features such as ticket prioritization and real-time status updates, the **HelpDesk Management System** ensures a faster and more organized resolution process, ultimately improving customer satisfaction. Future expansions could include reporting tools, mobile compatibility, and integration with email or messaging platforms for automated notifications.

ABSTRACT WITH POs AND PSOs MAPPING

CO 5 : BUILD JAVA APPLICATIONS FOR SOLVING REAL-TIME PROBLEMS.

| ABSTRACT | POs MAPPED | PSOs MAPPED |
|---|--|--|
| The HelpDesk Management System helps manage customer support requests. Customers can create and track tickets, while company staff can view and resolve them. The system has different logins for customers and staff, ensuring secure access. It helps improve communication and speeds up problem resolution, making customer support more efficient. | PO1 -3 PO2 -3 PO3 -3 PO5 -3 PO6 -3 PO8 -3 PO9 -3 PO10 -3 PO11-3 PO12 -3 | PSO1 -3 PSO2 -3 PSO3 -3 |

Note: 1- Low, 2-Medium, 3- High

TABLE OF CONTENTS

| CHAPTER NO. | TITLE | PAGE NO. |
|-------------|--------------------------------------|----------|
| | ABSTRACT | viii |
| 1 | INTRODUCTION | 1 |
| | 1.1 Objective | 1 |
| | 1.2 Overview | 1 |
| | 1.3 Java Programming concepts | 2 |
| 2 | PROJECT METHODOLOGY | 3 |
| | 2.1 Proposed Work | 3 |
| | 2.2 Block Diagram | 3 |
| 3 | MODULE DESCRIPTION | 4 |
| | 3.1 Login Module | 4 |
| | 3.2 Customer Dashboard Module | 4 |
| | 3.3 Company Dashboard Module | 4 |
| | 3.4 Ticket Management Module | 5 |
| | 3.5 Logout Module | 5 |
| 4 | CONCLUSION & FUTURE SCOPE | 6 |
| | 4.1 Conclusion | 6 |
| | 4.2 Future Scope | 6 |
| | REFERENCES | 20 |
| | APPENDIX A (SOURCE CODE) | 8 |
| | APPENDIX B (SCREENSHOTS) | 16 |

CHAPTER 1

INTRODUCTION

1.1 Objective

The objective of the HelpDesk Management System is to provide an efficient platform for managing customer support requests, ensuring faster response times and improved customer satisfaction. It allows customers to log issues, track their status, and receive timely updates, while enabling company representatives to monitor, prioritize, and resolve tickets systematically. By streamlining the support process and maintaining transparency, the system enhances communication, reduces delays, and improves overall efficiency in handling customer issues.

1.2 Overview

The HelpDesk Management System is a user-friendly application designed to handle customer support requests efficiently. It provides two primary user roles: customers and company representatives. Customers can log in to create tickets for their issues and track the status of those tickets, while company representatives can view all tickets, update their statuses, and ensure timely resolution. The system features role-based authentication, an intuitive interface, and a ticket lifecycle workflow (Open → In Progress → Resolved) to manage requests systematically. By centralizing the support process, it enhances transparency, improves communication, and ensures faster issue resolution, leading to higher customer satisfaction. Future enhancements, such as automated notifications and reporting tools, can further optimize its functionality.

1.3 Java Programming Concepts

- **Encapsulation:** The HelpDesk Management System uses encapsulation by grouping related data and methods within classes, such as User and Ticket. Private fields are accessed through public methods, ensuring data integrity and security.
- **Inheritance:** Although not directly implemented in the current version, the system could extend functionality using inheritance, such as creating specific classes for different user roles (Customer, Admin) that inherit from a base User class. **Polymorphism:** The system can demonstrate polymorphism by overriding methods, such as customizing ticket handling behaviors for different roles.
- **Abstraction:** GUI components like buttons and dialogs abstract the internal logic from the user, showing only options like "Create Ticket" or "View Tickets". The login system abstracts validation details, allowing users to simply enter credentials.
- **Swing and AWT:** Components like JFrame, JLabel, JButton, and JOptionPane are used to build the user interface. Layout managers (GridLayout, BorderLayout) organize GUI components.
- **Event handling:** It includes ActionListener for managing button actions like login, logout, ticket creation, and resolution. ItemListener handles checkbox state changes for selecting login type.
- **Error handling:** It uses try-catch to manage invalid or null inputs during ticket resolution safely.

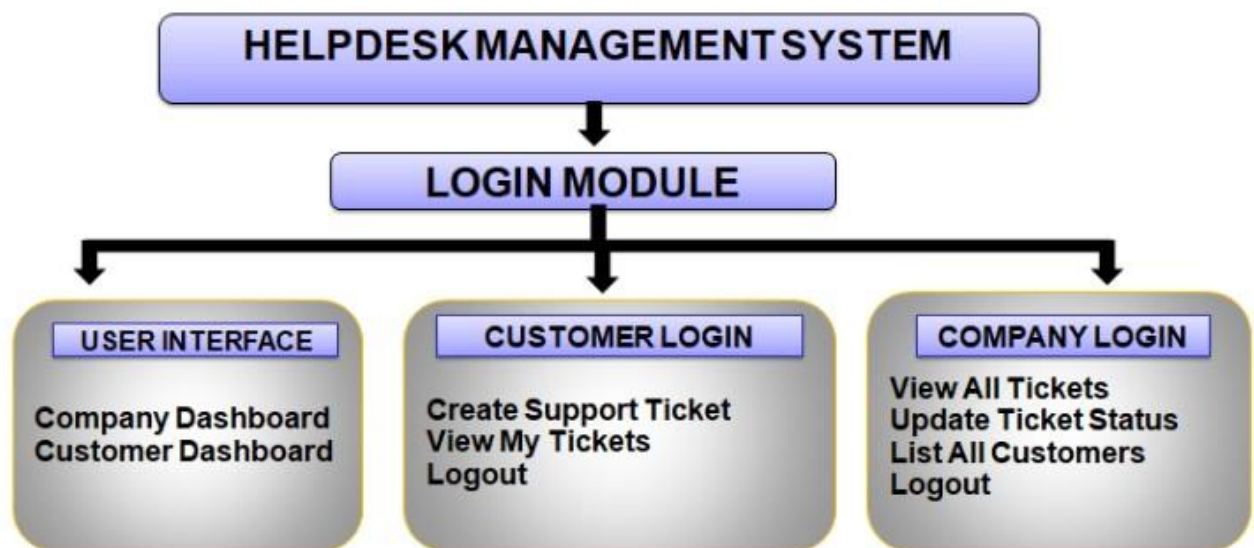
CHAPTER 2

PROJECT METHODOLOGY

2.1 Proposed Work

The proposed work focuses on developing a HelpDesk Management System with a user-friendly interface and efficient modules for both customer and company operations. The system includes a login module that differentiates user roles into customer and company categories. The customer login module enables users to create support tickets, view their tickets, and securely log out. The company login module provides functionalities to view all tickets, update ticket statuses, list all customers, and log out. The interface integrates dashboards tailored to customers and companies, ensuring seamless interaction and functionality. This structured approach enhances ticket management and communication between users and the support team.

2.2 Block Diagram



CHAPTER 3

MODULE DESCRIPTION

3.1 Login Module

The Login Module is designed to authenticate users and grant access based on their roles, either as a Customer or a Company representative. Users provide their credentials through username and password fields, and their roles are selected via checkboxes. Upon clicking the login button, the system validates the credentials and directs the user to the appropriate dashboard. An exit button is also provided to close the application. This module ensures secure access control and role-specific functionalities.

3.2 Customer Dashboard Module

The Customer Dashboard Module provides customers with a dedicated interface to manage their support tickets. After successful login, customers can create new tickets by describing their issues, view the status of their existing tickets, and securely log out of the system. It includes interactive buttons such as "Create Support Ticket," "View My Tickets," and "Logout," ensuring a seamless experience for users to resolve their concerns efficiently.

3.3 Company Dashboard Module

The Company Dashboard Module allows company representatives to oversee and manage all customer tickets. This module provides options to view all submitted tickets, update their statuses (such as Open, In Progress, or Resolved), and access a list of all customers who have submitted tickets. The logout feature ensures secure session termination. Through this module, company representatives can efficiently handle customer issues and maintain an organized workflow.

3.4 Ticket Management Module

The Ticket Management Module is a comprehensive feature that handles all aspects of ticket processing. Customers can create new tickets via a simple dialog box where they describe their issues, and both customers and company representatives can view the status of tickets through a ticket viewing panel. Additionally, company representatives can update ticket statuses to reflect the progress of issue resolution. This module bridges communication and ensures transparency between customers and the company.

3.5 Logout Module

The Logout Module is designed to manage user sessions and enhance security. By clicking the logout button, users can end their current session and return to the login screen. This module clears session data and ensures that unauthorized access is prevented once a user has logged out, maintaining the integrity and confidentiality of the system.

CHAPTER 4

CONCLUSION AND FUTURE SCOPE

4.1 CONCLUSION

The HelpDesk Management System is an essential tool for managing customer service interactions effectively. It ensures streamlined communication between customers and company representatives, enhancing support efficiency. The system provides a clear structure, starting from the login module to various dashboards, allowing users to easily create and manage support tickets. The customer dashboard allows users to track ticket progress, while the company dashboard ensures that tickets are addressed promptly and efficiently. By managing ticket statuses and customer queries, the system ensures that issues are resolved in a timely manner. Security is prioritized through authentication and role-based access. The user-friendly interface simplifies navigation, making it easy for both customers and company representatives to interact with the system. This system will improve service quality, customer satisfaction, and support resolution time. In conclusion, it plays a crucial role in enhancing the customer support process. It is a vital step towards building a more efficient support framework for businesses.

4.2 FUTURE SCOPE

The HelpDesk Management System has the potential for several future improvements to further enhance its functionality and user experience. Integration of AI and machine learning could automate ticket categorization, saving time for both customers and support agents. Real-time chat features would enhance communication by providing instant responses, improving customer satisfaction. Mobile application development would make the system more accessible, allowing users to manage tickets from anywhere. Expanding the system to support multiple languages would make it more inclusive for global customers. Adding voice recognition for ticket creation could improve accessibility for users with disabilities.

Advanced analytics and reporting features would provide valuable insights into ticket trends, helping companies optimize their support processes. Enhanced integration with other platforms, such as email or social media, would broaden communication channels. The system could also be scaled to accommodate businesses with larger customer bases. Finally, more customization options for companies to tailor the system to their specific needs would make it even more versatile.

APPENDIX A

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.util.ArrayList;

public class HelpDeskManagementSystem {
    private static final String[] companyAccounts = {"admin"};
    private static final String[] customerAccounts = {"Preethi", "Sneha", "Vinitha",
"Hassani", "Mirna"};
    private static final String[] companyPasswords = {"admin123"};
    private static final String[] customerPasswords = {"preethi123", "sneha123",
"vinitha123", "hassani123", "mirna123"};
    private static final ArrayList<String> tickets = new ArrayList<>();
    private static final ArrayList<String> ticketOwners = new ArrayList<>();
    private static boolean isLoggedIn = false;
    private static String loggedInRole = "";
    private static String loggedInUsername = "";
    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            JFrame frame = new JFrame("HelpDesk Management System");
            frame.setSize(600, 400);
            frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            frame.getContentPane().setBackground(new Color(173, 216, 230)); // Light
blue background
            frame.setLayout(new BorderLayout());
            JLabel statusLabel = new JLabel("Welcome to the HelpDesk Management
System", JLabel.CENTER);
```

```

        statusLabel.setOpaque(true);
        statusLabel.setBackground(new Color(255, 182, 193)); // Light pink
        statusLabel.setFont(new Font("Arial", Font.BOLD, 14));
        frame.add(statusLabel, BorderLayout.SOUTH);
        JPanel loginPanel = new JPanel(new GridLayout(7, 2, 10, 10));
        loginPanel.setBackground(new Color(173, 216, 230)); // Light blue
        frame.add(loginPanel, BorderLayout.CENTER);
        setupLoginPanel(loginPanel, statusLabel, frame);
        frame.setVisible(true);
    });
}

private static void setupLoginPanel(JPanel loginPanel, JLabel statusLabel,
JFrame frame) {
    loginPanel.removeAll();
    JLabel usernameLabel = new JLabel("Username:");
    JLabel passwordLabel = new JLabel("Password:");
    JTextField usernameField = new JTextField();
    JPasswordField passwordField = new JPasswordField();
    // Checkboxes for Customer and Company login
    JCheckBox customerLoginCheckbox = new JCheckBox("Customer Login");
    JCheckBox companyLoginCheckbox = new JCheckBox("Company Login");
    // Disable the other checkbox when one is selected
    customerLoginCheckbox.addItemListener(e->
companyLoginCheckbox.setEnabled(!customerLoginCheckbox.isSelected()));
    companyLoginCheckbox.addItemListener(e ->
customerLoginCheckbox.setEnabled(!companyLoginCheckbox.isSelected()));
    JButton loginButton = new JButton("Login");
    JButton exitButton = new JButton("Exit");
    loginButton.addActionListener(e -> {

```

```

String username = usernameField.getText();
String password = new String(passwordField.getPassword());
if (customerLoginCheckbox.isSelected()) {
    if (loginCustomer(username, password)) {
        statusLabel.setText("Welcome, " + loggedInUsername + "!");
        switchToCustomerDashboard(loginPanel, statusLabel, frame);
    } else {
        statusLabel.setText("Invalid customer username or password. Try
again!");
    }
} else if (companyLoginCheckbox.isSelected()) {
    if (loginCompany(username, password)) {
        statusLabel.setText("Welcome, " + loggedInUsername + "!");
        switchToCompanyDashboard(loginPanel, statusLabel, frame);
    } else {
        statusLabel.setText("Invalid company username or password. Try
again!");
    }
} else {
    statusLabel.setText("Please select a login option.");
}
});
exitButton.addActionListener(e -> System.exit(0));
loginPanel.add(usernameLabel);
loginPanel.add(usernameField);
loginPanel.add(passwordLabel);
loginPanel.add(passwordField);
loginPanel.add(customerLoginCheckbox);
loginPanel.add(companyLoginCheckbox);

```

```

loginPanel.add(loginButton);
loginPanel.add(exitButton);
loginPanel.revalidate();
loginPanel.repaint();
}

private static boolean loginCustomer(String username, String password) {
    for (int i = 0; i < customerAccounts.length; i++) {
        if (customerAccounts[i].equals(username) &&
customerPasswords[i].equals(password)) {
            isLoggedIn = true;
            loggedInRole = "customer";
            loggedInUsername = username;
            return true;
        }
    }
    return false;
}

private static boolean loginCompany(String username, String password) {
    for (int i = 0; i < companyAccounts.length; i++) {
        if (companyAccounts[i].equals(username) &&
companyPasswords[i].equals(password)) {
            isLoggedIn = true;
            loggedInRole = "company";
            loggedInUsername = username;
            return true;
        }
    }
    return false;
}

```

```

private static void switchToCustomerDashboard(JPanel loginPanel, JLabel
statusLabel, JFrame frame) {
    loginPanel.removeAll();
    JButton createTicketButton = new JButton("Create Ticket");
    JButton viewTicketsButton = new JButton("View My Tickets");
    JButton logoutButton = new JButton("Logout");
    createTicketButton.addActionListener(e -> createTicketDialog(statusLabel,
frame));
    viewTicketsButton.addActionListener(e -> viewCustomerTickets(statusLabel,
frame));
    logoutButton.addActionListener(e -> logout(loginPanel, statusLabel, frame));
    loginPanel.add(createTicketButton);
    loginPanel.add(viewTicketsButton);
    loginPanel.add(logoutButton);
    loginPanel.revalidate();
    loginPanel.repaint();
}

private static void switchToCompanyDashboard(JPanel loginPanel, JLabel
statusLabel, JFrame frame) {
    loginPanel.removeAll();
    JButton viewAllTicketsButton = new JButton("View All Tickets");
    JButton resolveTicketButton = new JButton("Resolve Ticket");
    JButton logoutButton = new JButton("Logout");
    viewAllTicketsButton.addActionListener(e -> viewAllTickets(statusLabel,
frame));
    resolveTicketButton.addActionListener(e -> resolveTicketDialog(statusLabel,
frame));
    logoutButton.addActionListener(e -> logout(loginPanel, statusLabel, frame));
    loginPanel.add(viewAllTicketsButton);

```

```

loginPanel.add(resolveTicketButton);
loginPanel.add(logoutButton);

loginPanel.revalidate();
loginPanel.repaint();
}

private static void createTicketDialog(JLabel statusLabel, JFrame frame) {
    String issue = JOptionPane.showInputDialog(frame, "Enter your issue:",
"Create Ticket", JOptionPane.PLAIN_MESSAGE);
    if (issue != null && !issue.trim().isEmpty()) {
        tickets.add("Open - " + issue);
        ticketOwners.add(loggedInUsername);
        statusLabel.setText("Ticket created successfully!");
    } else {
        statusLabel.setText("Ticket creation cancelled.");
    }
}

private static void viewCustomerTickets(JLabel statusLabel, JFrame frame) {
    StringBuilder ticketList = new StringBuilder();
    for (int i = 0; i < tickets.size(); i++) {
        if (ticketOwners.get(i).equals(loggedInUsername)) {
            ticketList.append("Ticket ").append(i + 1).append(":
").append(tickets.get(i)).append("\n");
        }
    }
    if (ticketList.length() == 0) {
        ticketList.append("No tickets found.");
    }
    JOptionPane.showMessageDialog(frame, ticketList.toString(), "My Tickets",

```

```

OptionPane.INFORMATION_MESSAGE);
    }
    private static void viewAllTickets(JLabel statusLabel, JFrame frame) {
        StringBuilder ticketList = new StringBuilder();
        for (int i = 0; i < tickets.size(); i++) {
            ticketList.append("Ticket ").append(i + 1).append(":
").append(tickets.get(i)).append(" (Owner: ")
                .append(ticketOwners.get(i)).append(")\n");
        }
        if (ticketList.length() == 0) {
            ticketList.append("No tickets found.");
        }
        JOptionPane.showMessageDialog(frame, ticketList.toString(), "All Tickets",
OptionPane.INFORMATION_MESSAGE);
    }
    private static void resolveTicketDialog(JLabel statusLabel, JFrame frame) {
        String ticketIdStr = JOptionPane.showInputDialog(frame, "Enter Ticket ID to
resolve:", "Resolve Ticket", JOptionPane.PLAIN_MESSAGE);
        try {
            int ticketId = Integer.parseInt(ticketIdStr) - 1;
            if (ticketId >= 0 && ticketId < tickets.size()) {
                tickets.set(ticketId, "Resolved - " + tickets.get(ticketId).split(" - ")[1]);
                statusLabel.setText("Ticket resolved!");
            } else {
                statusLabel.setText("Invalid ticket ID.");
            }
        } catch (NumberFormatException | NullPointerException e) {
            statusLabel.setText("Operation cancelled or invalid input.");
        }
    }


```



```
}  
private static void logout(JPanel loginPanel, JLabel statusLabel, JFrame frame) {  
    isLoggedIn = false;  
    loggedInRole = "";  
    loggedInUsername = "";  
    statusLabel.setText("Logged out successfully.");  
    setupLoginPanel(loginPanel, statusLabel, frame);  
}  
}
```

APPENDIX B

Customer Login



The screenshot shows a window titled "HelpDesk Management System". It contains a login form with the following elements:

- Username:** A text input field containing the text "Preethi".
- Password:** A password input field with masked characters ".....".
- Customer Login:** A checkbox that is checked.
- Company Login:** An unchecked checkbox.
- Login:** A button to submit the login information.
- Exit:** A button to close the window.

At the bottom of the window, a pink banner displays the text: "Welcome to the HelpDesk Management System".

Customer Dashboard



The screenshot shows a window titled "HelpDesk Management System". It displays a dashboard with the following elements:

- Create Ticket:** A button to create a new ticket.
- View My Tickets:** A button to view the user's tickets.
- Logout:** A button to log out of the system.

At the bottom of the window, a pink banner displays the text: "Welcome, Preethi!".

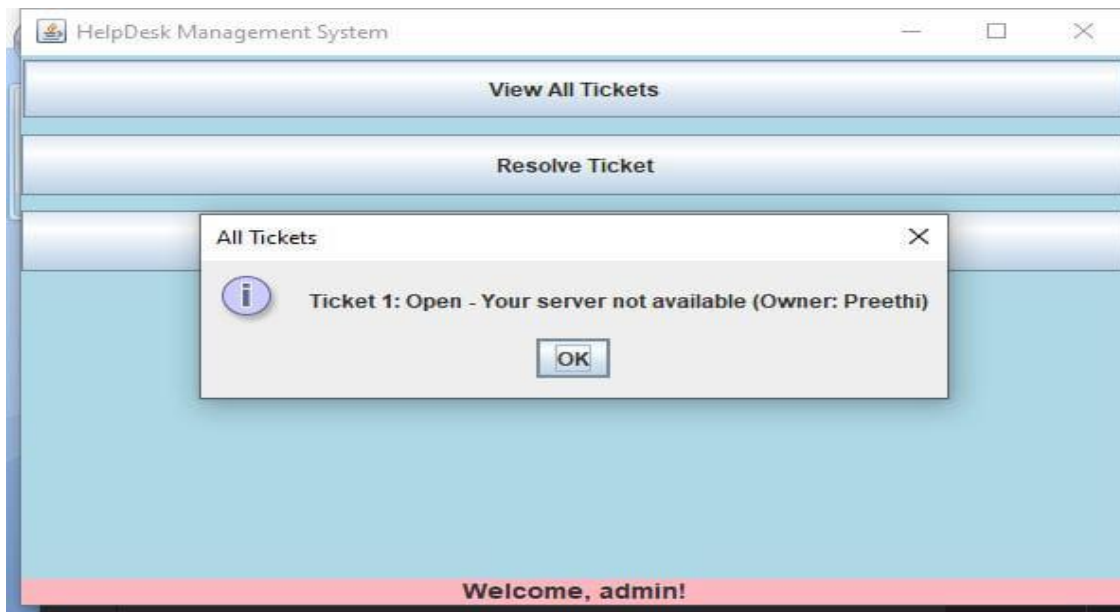
Create Ticket

The screenshot shows a window titled "HelpDesk Management System" with a "Create Ticket" button. A modal dialog box titled "Create Ticket" is open, containing the text "Enter your issue:" and a text input field with the text "Your server not available". Below the input field are "OK" and "Cancel" buttons. The background window also shows a "View My Tickets" button and a pink footer bar with the text "Welcome, Preethi!".

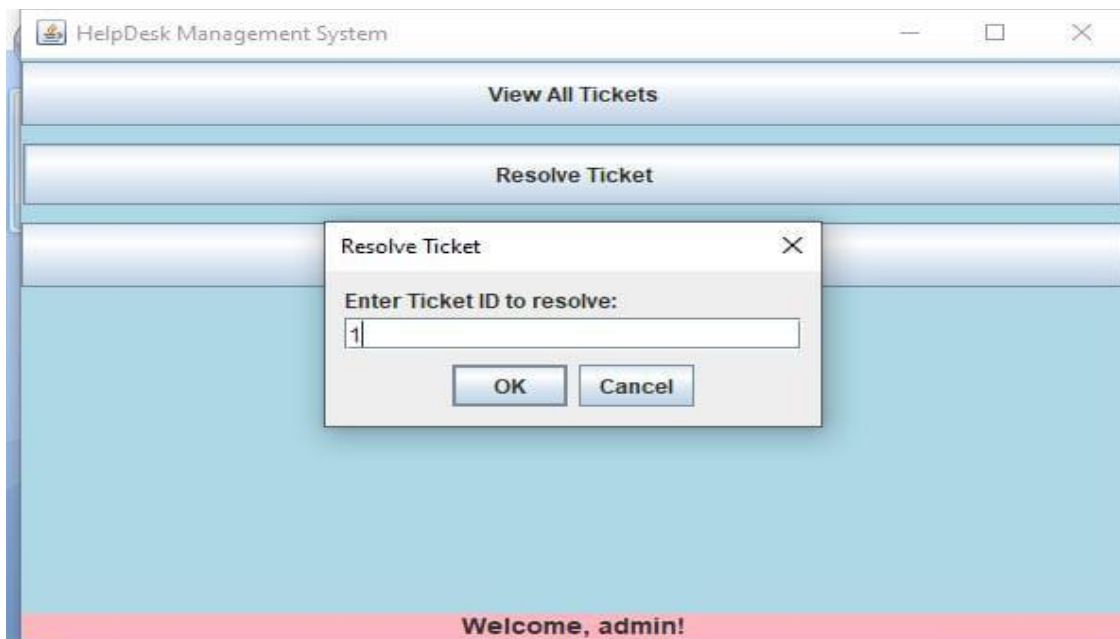
Company Login

The screenshot shows a window titled "HelpDesk Management System" with a login form. The form includes fields for "Username:" (containing "admin") and "Password:" (containing "....."). Below these fields are two checkboxes: "Customer Login" (unchecked) and "Company Login" (checked). At the bottom of the form are "Login" and "Exit" buttons. A pink footer bar at the bottom of the window displays the message "Invalid company username or password. Try again!".

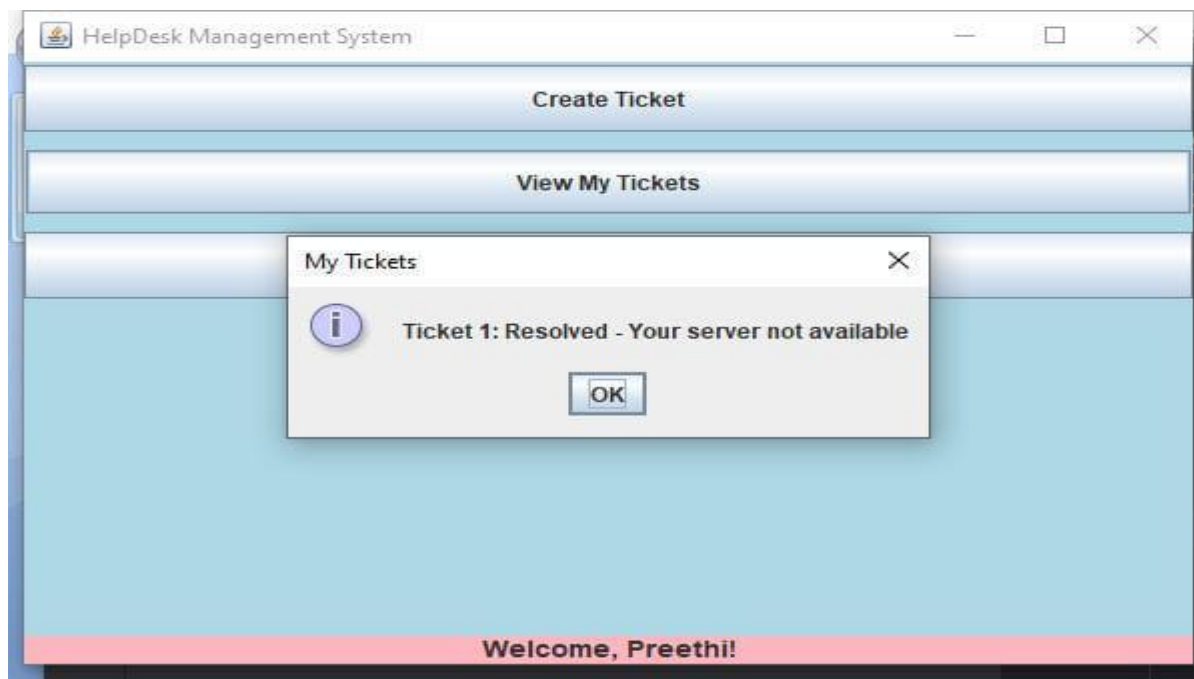
View All Tickets



Resolve Ticket



Resolved message



REFERENCES

WEBSITES

- <https://www.helpdesk.com/>
- <https://projectsgeek.com/>
- <https://stackoverflow.com/questions/1176282/java-open-source-helpdesk-workflow-project>

YOUTUBE LINK

- <https://www.youtube.com/watch?v=sV3cigTJvG4>
- <https://www.youtube.com/watch?v=EBY-FOFddW8>

BOOK

- K. R. Chandran, S. D. Narayanan, and M. M. Srinivasan. (2020). "A Review of HelpDesk Management Systems .
- Horstmann, C. (2018). Core Java Volume I—Fundamentals (11th ed.).