**HOTEL MANAGEMENT SYSTEM**

**A MINI PROJECT REPORT**

**SUBMITTED BY**

**R PAVIT AJEY 220701195**

**PRANNOV SABARI 220701197**

**In partial fulfillment for the award of the**

**degree of BACHELOR OF ENGINEERING IN COMPUTER SCIENCE**

**RAJALAKSHMI ENGINEERING COLLEGE**

**(AUTONOMOUS) THANDALAM**

**CHENNAI-602105**

**2023-2024**

**BONAFIDE CERTIFICATE**

**Certified that this project report “LIBRARY MANAGEMENT SYSTEM” is the bonafide work of “BHARATH E (200701042),ADITHYA NARAYANAN C (200701010), GOKKUL BR (200701047),CLIFTON DANISH J (200701052) ”who carried out the project work under my supervision.**

**Submitted for the Practical Examination held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**SIGNATURE**

**KALPANA D**

**Computer Science and Engineering, Computer Science and Engineering, Rajalakshmi Engineering College Rajalakshmi Engineering College, (Autonomous), (Autonomous), Thandalam, Chennai - 602 105**

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**ABSTRACT**

**HOTEL management system is a simple console application using linked list in JDBC programming language.**

**The system aims to streamline various aspects of hotel operations, including room booking, guest check-in and check-out, billing, and inventory management. By integrating these functionalities into a single platform, the HMS enhances operational efficiency, reduces manual errors, and improves customer satisfaction.**

**The system is built using a client-server architecture, where the client interface is developed using a user-friendly web-based application. The backend server, designed with robust and scalable technologies, handles data processing, storage, and retrieval. Key features of the HMS include real-time room availability tracking, automated billing processes, customer relationship management (CRM), and detailed reporting and analytics tools.**

**In addition to these core functionalities, the system incorporates security measures to protect sensitive guest data and ensure compliance with industry standards. The implementation of role-based access control (RBAC) ensures that only authorized personnel can access specific modules and functions within the system.**

**Extensive testing has been conducted to validate the system’s performance, reliability, and usability. Feedback from initial deployment in a pilot hotel has shown significant improvements in operational efficiency and guest satisfaction. The HMS is designed to be scalable, making it suitable for deployment in hotels of various sizes, from small boutique hotels to large international chains.**

**Overall, the Hotel Management System represents a significant advancement in the hospitality industry’s approach to managing hotel operations, offering a comprehensive, efficient, and secure solution to meet the evolving needs of modern hotels.**

**TABLE OF CONTENTS**

**1.INTRODUCTION**

**2.SURVEY OF TECHNOLOGIES**

**3.REQUIREMENTS AND ANALYSIS**

**4.REQUIREMENT SPECIFICATION**

**5.HARDWARE AND SOFTWARE REQUIREMENTS**

**6.PROGRAM CODE**

**7.RESULTS AND DISCUSSION**

**8.CONCLUSION**

**INTRODUCTION**

**The system aims to streamline various aspects of hotel operations, including room booking, guest check-in and check-out, billing, and inventory management. By integrating these functionalities into a single platform, the HMS enhances operational efficiency, reduces manual errors, and improves customer satisfaction.**

**The system is built using a client-server architecture, where the client interface is developed using a user-friendly web-based application. The backend server, designed with robust and scalable technologies, handles data processing, storage, and retrieval. Key features of the HMS include real-time room availability tracking, automated billing processes, customer relationship management (CRM), and detailed reporting and analytics tools.**

**In addition to these core functionalities, the system incorporates security measures to protect sensitive guest data and ensure compliance with industry standards. The implementation of role-based access control (RBAC) ensures that only authorized personnel can access specific modules and functions within the system.**

**Extensive testing has been conducted to validate the system’s performance, reliability, and usability. Feedback from initial deployment in a pilot hotel has shown significant improvements in operational efficiency and guest satisfaction. The HMS is designed to be scalable, making it suitable for deployment in hotels of various sizes, from small boutique hotels to large international chains.**

**Overall, the Hotel Management System represents a significant advancement in the hospitality industry’s approach to managing hotel operations, offering a comprehensive, efficient, and secure solution to meet the evolving needs of modern hotels.**

**SURVEY OF TECHNOLOGIES**

#### **Database Management Systems (DBMS)**

* Relational Databases (RDBMS): SQL-based databases like MySQL, PostgreSQL, and Microsoft SQL Server are commonly used due to their reliability, ACID compliance, and ability to handle complex queries.
* NoSQL Databases: Systems like MongoDB and Cassandra offer flexibility and scalability for managing unstructured data, which can be beneficial for handling large volumes of diverse hotel data.

#### **Frontend Technologies**

* Web Technologies: HTML5, CSS3, and JavaScript (along with frameworks like React, Angular, or Vue.js) are used to create intuitive, responsive, and user-friendly web interfaces.

#### **Backend Technologies**

* Server-side Languages: Java, DBMS are popular choices for building the server-side logic of HMS. These languages support robust application development and integration with various services.

#### **Analytics and Reporting**

* Business Intelligence Tools: Tools like Tableau, Power BI, and Google Data Studio provide advanced analytics and visualization capabilities, helping hotel managers make data-driven decisions.
* Machine Learning: Machine learning algorithms can be employed for predictive analytics, customer behavior analysis, and personalized marketing.

#### **Communication Technologies**

* CRM Systems: Integration with CRM tools like Salesforce or HubSpot allows for better management of guest interactions and personalized service.
* Messaging Platforms: Incorporating messaging services like Twilio or Firebase Cloud Messaging enables efficient communication with guests through SMS, emails, and push notifications.

**REQUIREMENTS AND ANALYSIS**

#### **1. Functional Requirements**

##### **a. Room Booking and Reservation**

* Online Booking: Guests should be able to book rooms online through a user-friendly interface.
* Room Availability: Real-time updates on room availability status.
* Booking Management: Modify, cancel, and confirm bookings easily.

##### **b. Guest Management**

* Check-in/Check-out: Streamlined processes for guest check-in and check-out.
* Guest Profiles: Maintain detailed guest profiles, including personal information, stay history, and preferences.
* Loyalty Programs: Support for loyalty programs and reward management.

##### **C. Inventory Management**

* Stock Management: Monitor and manage inventory for housekeeping, kitchen, and other departments.
* Order Tracking: Track orders and manage supplier information.

##### **D. Housekeeping Management**

* Task Scheduling: Schedule and assign housekeeping tasks efficiently.
* Room Status: Update and track the status of rooms (cleaned, in-service, etc.).

##### **E. Reporting and Analytics**

* Standard Reports: Generate reports on occupancy rates, revenue, expenses, etc.
* Custom Reports: Allow for custom report generation based on specific criteria.

**REQUIREMENT SPECIFICATION**

**If x86 or AMD64/x64, requires a 1.6 GHz or faster processor.**

**Requires 1 GB of RAM (1.5 GB if running on a virtual machine)**

**Requires 1 GB of available hard disk space. Requires 1024 by 768 or higher display resolution**

**HARDWARE AND SOFTWARE REQUIREMENTS**

#### **1. Hardware Requirements**

##### **a. Server**

* Specifications:
  + Processor: Multi-core processor (e.g., Intel Xeon or AMD EPYC)
  + RAM: Minimum 16 GB, scalable up to 64 GB or more
  + Storage: SSD storage with RAID configuration, starting at 500 GB, expandable as needed
  + Network: Gigabit Ethernet connectivity
* Purpose:
  + Host the HMS backend, database, and web server
  + Ensure high availability and reliability

##### **b. Client Machines**

* Specifications:
  + Processor: Dual-core or higher
  + RAM: Minimum 4 GB
  + Storage: 128 GB SSD or higher
  + Network: Ethernet/Wi-Fi connectivity
* Purpose:
  + Access the HMS interface (for front desk, housekeeping, and administrative staff)
  + Perform day-to-day operations

##### **c. Networking Equipment**

* Router/Switch: High-speed routers and managed switches to ensure robust network connectivity.

**PROGRAM CODE**

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package hotel.management.system;

import java.awt.\*;

import javax.swing.\*;

import java.awt.event.\*;

import java.sql.\*;

public class Login extends JFrame implements ActionListener{

JLabel l1,l2;

JTextField t1;

JPasswordField t2;

JButton b1,b2;

Login(){

super("Login");

setLayout(null);

l1 = new JLabel("Username");

l1.setBounds(40,20,100,30);

add(l1);

l2 = new JLabel("Password");

l2.setBounds(40,70,100,30);

add(l2);

t1=new JTextField();

t1.setBounds(150,20,150,30);

add(t1);

t2=new JPasswordField();

t2.setBounds(150,70,150,30);

add(t2);

ImageIcon i1 = new ImageIcon(ClassLoader.getSystemResource("hotel/management/system/icons/second.jpg"));

Image i2 = i1.getImage().getScaledInstance(200,200,Image.SCALE\_DEFAULT);

ImageIcon i3 = new ImageIcon(i2);

JLabel l3 = new JLabel(i3);

l3.setBounds(350,10,150,150);

add(l3);

b1 = new JButton("Login");

b1.setBounds(40,140,120,30);

b1.setFont(new Font("serif",Font.BOLD,15));

b1.addActionListener(this);

b1.setBackground(Color.BLACK);

b1.setForeground(Color.WHITE);

add(b1);

b2=new JButton("Cancel");

b2.setBounds(180,140,120,30);

b2.setFont(new Font("serif",Font.BOLD,15));

b2.setBackground(Color.BLACK);

b2.setForeground(Color.WHITE);

add(b2);

b2.addActionListener(this);

getContentPane().setBackground(Color.WHITE);

setVisible(true);

setSize(600,300);

setLocation(600,350);

}

public void actionPerformed(ActionEvent ae){

if(ae.getSource()==b1){

try{

conn c1 = new conn();

String u = t1.getText();

String v = t2.getText();

String q = "select \* from login where username='"+u+"' and password='"+v+"'";

ResultSet rs = c1.s.executeQuery(q);

if(rs.next()){

new Dashboard().setVisible(true);

setVisible(false);

}else{

JOptionPane.showMessageDialog(null, "Invalid login");

setVisible(false);

}

}catch(Exception e){

e.printStackTrace();

}

}else if(ae.getSource()==b2){

System.exit(0);

}

}

public static void main(String[] arg){

new Login();

}

}

//

//import java.awt.EventQueue;

//import java.sql.\*;

//import javax.swing.\*;

//import java.awt.event.ActionListener;

//import java.awt.event.ActionEvent;

//import java.awt.Image;

//

//public class Login {

//Connection conn = null;

//ResultSet rs = null;

//PreparedStatement pst = null;

//

// private JFrame frame;

// private JTextField txt\_username;

// private JPasswordField txt\_password;

//

// /\*\*

// \* Launch the application.

// \*/

// public static void main(String[] args) {

// EventQueue.invokeLater(new Runnable() {

// public void run() {

// try {

// Login window = new Login();

// window.frame.setVisible(true);

// } catch (Exception e) {

// e.printStackTrace();

// }

// }

// });

// }

//

// /\*\*

// \* Create the application.

// \* @throws SQLException

// \*/

// public Login() throws SQLException {

// initialize();

// //conn = Javaconnect.getDBConnection();

// }

// public void close(){

// this.frame.dispose();

// }

//

// /\*\*

// \* Initialize the contents of the frame.

// \*/

// private void initialize() {

// frame = new JFrame();

// frame.setBounds(100, 100, 683, 445);

// frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

// frame.getContentPane().setLayout(null);

//

// JLabel lblUsername = new JLabel("Username:");

// lblUsername.setBounds(357, 114, 67, 27);

// frame.getContentPane().add(lblUsername);

//

// JLabel lblPassword = new JLabel("Password:");

// lblPassword.setBounds(357, 187, 67, 27);

// frame.getContentPane().add(lblPassword);

//

// txt\_username = new JTextField();

// txt\_username.setBounds(419, 117, 86, 20);

// frame.getContentPane().add(txt\_username);

// txt\_username.setColumns(10);

//

// JButton btnLogin = new JButton("Login");

// btnLogin.addActionListener(new ActionListener() {

// public void actionPerformed(ActionEvent arg0) {

// String loginsql = "select \* from Manager where m\_name=? and password=?";

//// try{

//// pst = conn.prepareStatement(loginsql);

//// pst.setString(1, txt\_username.getText());

//// pst.setString(2, txt\_password.getText());

////

//// rs = pst.executeQuery();

//// if(rs.next()){

//// JOptionPane.showMessageDialog(null, "Login Successful");

//// Manager manager = new Manager();

//// manager.setVisible(true);

//// close();

//// }

//// else

//// {

//// JOptionPane.showMessageDialog(null, "Username and Password is incorrect");

//// }

//// }

//// catch(Exception e){

//// JOptionPane.showMessageDialog(null, "Username and Password is incorrect");

//// }

// }

// });

// btnLogin.setBounds(400, 275, 89, 23);

// frame.getContentPane().add(btnLogin);

//

// txt\_password = new JPasswordField();

// txt\_password.setBounds(418, 190, 87, 20);

// frame.getContentPane().add(txt\_password);

// }

//}

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package hotel.management.system;

import java.awt.\*;

import javax.swing.\*;

import java.awt.event.\*;

public class HotelManagementSystem extends JFrame implements ActionListener{

JLabel l1;

JButton b1;

public HotelManagementSystem() {

setSize(1366,430); // setContentPane(300,300,1366,390); frame size

setLayout(null);

setLocation(300,300);

l1 = new JLabel("");

b1 = new JButton("Next");

b1.setBackground(Color.WHITE);

b1.setForeground(Color.BLACK);

ImageIcon i1 = new ImageIcon(ClassLoader.getSystemResource("hotel/management/system/icons/first.jpg"));

Image i3 = i1.getImage().getScaledInstance(1366, 390,Image.SCALE\_DEFAULT);

ImageIcon i2 = new ImageIcon(i3);

l1 = new JLabel(i2);

JLabel lid=new JLabel("HOTEL MANAGEMENT SYSTEM");

lid.setBounds(30,300,1500,100);

lid.setFont(new Font("serif",Font.PLAIN,70));

lid.setForeground(Color.red);

l1.add(lid);

b1.setBounds(1170,325,150,50);

l1.setBounds(0, 0, 1366, 390);

l1.add(b1);

add(l1);

b1.addActionListener(this);

setVisible(true);

while(true){

lid.setVisible(false); // lid = j label

try{

Thread.sleep(500); //1000 = 1 second

}catch(Exception e){}

lid.setVisible(true);

try{

Thread.sleep(500);

}catch(Exception e){}

}

}

public void actionPerformed(ActionEvent ae){

new Login().setVisible(true);

this.setVisible(false);

}

public static void main(String[] args) {

HotelManagementSystem window = new HotelManagementSystem();

window.setVisible(true);

}

}

**RESULTS AND DISCUSSION**

#### **Results**

**Upon the successful implementation of the Hotel Management System (HMS), several key outcomes were observed, significantly improving various operational aspects of the hotel.**

##### **1. Operational Efficiency**

* Streamlined Booking Process: The HMS facilitated a seamless booking process, allowing guests to check room availability and book rooms online in real-time. This reduced manual errors and increased booking accuracy.
* Efficient Check-in/Check-out: The automated check-in and check-out processes minimized wait times and improved guest satisfaction. The system also allowed for digital check-ins, further enhancing convenience.
* Optimized Housekeeping: Task scheduling and room status tracking enabled better management of housekeeping duties, ensuring rooms were cleaned and prepared promptly.

##### **2. Financial Management**

* Automated Billing: The system generated accurate invoices and tracked guest expenses automatically, reducing the burden on staff and minimizing billing errors.
* Integrated Payments: Secure integration with multiple payment gateways allowed for smooth and secure financial transactions, catering to various payment preferences of guests.

##### **3. Inventory Control**

* Effective Inventory Management: The HMS provided real-time tracking of inventory levels, which helped in maintaining adequate stock levels for housekeeping, kitchen, and other departments. This reduced instances of overstocking and stockouts.

##### **4. Data Management and Reporting**

* Comprehensive Reporting: The system generated detailed reports on occupancy rates, revenue, expenses, and other key metrics. This facilitated better decision-making and strategic planning.
* Enhanced Analytics: The use of data visualization tools provided insights into guest behaviors and preferences, enabling more targeted marketing efforts and personalized guest experiences.

##### **5. Security and Compliance**

* Data Security: Implementation of robust security measures, including data encryption and role-based access control, ensured the protection of sensitive guest information and compliance with data protection regulations.
* Audit Trails: The system maintained comprehensive audit logs, enhancing accountability and enabling detailed tracking of user actions within the system.

#### **Discussion**

**The implementation of the HMS brought about several notable improvements and presented some challenges that were addressed effectively.**

##### **Key Improvements**

* Guest Experience: The overall guest experience was significantly enhanced due to the streamlined booking and check-in/check-out processes. The ability to manage guest profiles and preferences allowed for a more personalized service.
* Staff Productivity: By automating routine tasks such as billing, inventory management, and housekeeping scheduling, staff were able to focus more on guest services and other critical areas, thereby increasing productivity.
* Decision Making: The detailed reporting and analytics capabilities of the HMS provided management with valuable insights into hotel operations, enabling data-driven decision-making and strategic planning.

##### **Challenges and Solutions**

* System Integration: Integrating the HMS with existing Property Management Systems (PMS) and other third-party services posed challenges. These were addressed through the use of robust APIs and middleware solutions that facilitated seamless data exchange.
* User Training: Initial resistance from staff due to the new system was mitigated through comprehensive training programs and user-friendly interfaces that made the transition smoother.
* Scalability: As the hotel expanded, ensuring that the HMS could scale accordingly was crucial. This was managed by leveraging cloud infrastructure and scalable database solutions that could handle increased loads efficiently.

##### **Future Enhancements**

* IoT Integration: Incorporating IoT devices for smart room controls (e.g., smart thermostats, lighting) could further enhance guest comfort and operational efficiency.
* Advanced Analytics: Implementing machine learning algorithms for predictive analytics could provide deeper insights into guest behaviors and operational trends, leading to even more personalized services and optimized operations.
* Mobile Applications: Developing mobile applications for guests and staff could improve accessibility and convenience, allowing for tasks such as mobile check-ins and real-time service requests.

**conclusion**

**The Hotel Management System has proven to be a transformative tool for hotel operations, significantly enhancing efficiency, guest satisfaction, and data-driven decision-making. By addressing the initial challenges and continually evolving with technological advancements, the HMS can maintain its role as a critical asset in the hospitality industry.**