长 沙 学 院

软件工程基础实训Ⅱ 实训

|  |  |
| --- | --- |
| **题目** | **基于C/S架构的餐饮管理系统** |
| **学院** | **计算机科学与工程**  **请注意**：  蓝色字体内容为示例或解释，仅供参考。 |
| **专业(班级)** | **软件工程（21软件01）** |
| **姓名** | **王雅晴** |
| **学号** | **B20210304131** |
| **指导教师** | **杨刚、潘怡** |
| **起止日期** | **2023.06.05～2023.06.16** |

实训任务书

**课程名称：软件工程基础实训II**

**实训题目：基于C/S架构的餐饮管理系统**

### 已知技术参数和设计要求：

1. 问题描述（功能要求）：

本次案例将完成一个基于C/S架构的餐饮管理系统。

（1）.进行餐饮管理系统的功能设计。

（2）.设计餐饮管理系统的数据库。

（3）.利用JAVA的GUI类设计开发友好的人机接口。

（4）.利用JAVA语言设计监听用户操作和JDBC进行数据库的处理。

餐饮管理系统主要有6大模块：

（1）.员工管理：对员工实现增删改查。

（2）.客户管理：对客户实现增删改查。

（3）.餐台管理：对餐台实现增删改查。

（4）.菜品管理：对菜品分类、菜品实现增删改查。

（5）.点菜管理：服务员对某客户、某一空闲餐台实行开台，同时实现点菜，将餐台号与所点的菜品对应起来，分别显示出来，并记录开台时间。

（6）.结账管理：收银员对某一餐台通过统计显示消费的菜品清单统计出消费金额，通过手动输入实收金额进行找零的计算，并显示，完成结账的操作，并记录成统计数据。

基础实训II有2周时间40学时，学生基本必做任务：独立完成基于C/S架构的餐饮管理系统。

选做扩展：菜品推荐、销售统计。

**2. 运行环境要求：**

（1）客户端：

windows操作系统，JDK1.6，Idea或Eclipse。

（2）服务器：

MySQL5.6或 SQL Server 数据库服务器。

3. 技术要求：

1） 掌握软件工程的需求分析和系统设计方法。

2) 掌握数据库的分析与设计，完成餐饮管理系统数据库的设计。

3) 掌握JAVA的GUI设计方法，完成登录、主界面、菜品分类管理界面、菜品管理界面等人机接口。

4） 掌握JAVA面向对象程序设计方法，设计系统所需的各种类。

5) 掌握JDBC方法，对数据库进行操作，完成主程序和菜品分类、菜品管理等功能。

### 实训工作量：

40课时

### 工作计划：

1.班级

21软件1-4

2.课时及教室安排

见实训课程工作计划表。

实训报告

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 学生姓名 | 王雅晴 | 学号 | B20210304131 | 班级 | 21软件01班 |
| 专业 | 软件工程 | | 指导教师姓名 | 杨刚、潘怡 | |
| 项目名称 | 基于C/S架构的餐饮管理系统 | | | | |
| **实训过程**：（说明实训各阶段的工作过程，包括需求分析、系统设计、编码实现、系统测试、交付实施等阶段的主要工作。）  本栏正文排版要求：宋体5号，行距18磅，段首缩进2字符。  本栏可跨多页。  **系统需求分析**  本次案例将完成一个基于C/S架构的餐饮管理系统。  （1）进行餐饮管理系统的功能设计。  （2）设计餐饮管理系统的数据库。  （3）利用JAVA的GUI类设计开发友好的人机接口。  （4）利用JAVA语言设计监听用户操作和JDBC进行数据库的处理。  餐饮管理系统主要有6大模块：  （1）员工管理：对员工实现增删改查。  （2）客户管理：对客户实现增删改查。  （3）餐台管理：对餐台实现增删改查。  （4）菜品管理：对菜品分类、菜品实现增删改查。  （5）点菜管理：服务员对某客户、某一空闲餐台实行开台，同时实现点菜，将餐台号与所点的菜品对应起来，分别显示出来，并记录开台时间。  （6）结账管理：收银员对某一餐台通过统计显示消费的菜品清单统计出消费金额，通过手动输入实收金额进行找零的计算，并显示，完成结账的操作，并记录成统计数据。  **系统设计：**  IMG_256  **编码实现：**  **Idea:**  1686896015916  1686896029352  登陆界面：  package 界面类;  import java.awt.EventQueue; import java.awt.Font; import java.awt.event.ActionEvent; import java.awt.event.ActionListener; import javax.swing.GroupLayout; import javax.swing.GroupLayout.Alignment; 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.JTextField; import javax.swing.LayoutStyle.ComponentPlacement; import javax.swing.border.EmptyBorder;  import 功能类.UserDAO; import 实体类.User; import 工具类.JDBCconnection; import java.awt.Color; import java.sql.Connection; import javax.swing.JPasswordField;  public class LoginFrame extends JFrame {  private JPanel contentPane;  private JTextField UserNameText;   private UserDAO userDao = new UserDAO();  private JPasswordField passwordField; */\*\* \* Launch the application. \*/* public static void main(String[] args) {  EventQueue.*invokeLater*(new Runnable() {  public void run() {  try {  LoginFrame frame = new LoginFrame();  frame.setVisible(true);  } catch (Exception e) {  e.printStackTrace();  }}  });  }  */\*\* \* Create the frame. \*/* public LoginFrame() {  setTitle("管理员登录");  setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  setBounds(100, 100, 565, 353);  contentPane = new JPanel();  contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));  setContentPane(contentPane);   JLabel lblNewLabel = new JLabel("餐饮管理系统");  lblNewLabel.setFont(new Font("黑体", Font.*BOLD*, 20));  JLabel lblNewLabel\_1 = new JLabel("用户名:");  UserNameText = new JTextField();  UserNameText.setColumns(10);   JLabel lblNewLabel\_2 = new JLabel("密码:");   JButton btnNewButton = new JButton("登录");  btnNewButton.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  LoginFrameActionPerformed(e);  }  });  JButton btnNewButton\_1 = new JButton("重置");  btnNewButton\_1.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  UserNameText.setText("");  passwordField.setText("");  }  });  passwordField = new JPasswordField();  GroupLayout gl\_contentPane = new GroupLayout(contentPane);  gl\_contentPane.setHorizontalGroup(  gl\_contentPane.createParallelGroup(Alignment.*TRAILING*)  .addGroup(gl\_contentPane.createSequentialGroup()  .addGap(53)  .addGroup(gl\_contentPane.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_contentPane.createSequentialGroup()  .addComponent(btnNewButton, GroupLayout.*PREFERRED\_SIZE*, 97, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*RELATED*, 200, Short.*MAX\_VALUE*)  .addComponent(btnNewButton\_1, GroupLayout.*PREFERRED\_SIZE*, 97, GroupLayout.*PREFERRED\_SIZE*)  .addGap(94))  .addGroup(gl\_contentPane.createSequentialGroup()  .addGroup(gl\_contentPane.createParallelGroup(Alignment.*TRAILING*)  .addGroup(gl\_contentPane.createSequentialGroup()  .addComponent(lblNewLabel\_1, GroupLayout.*PREFERRED\_SIZE*, 72, GroupLayout.*PREFERRED\_SIZE*)  .addGap(48))  .addGroup(gl\_contentPane.createSequentialGroup()  .addComponent(lblNewLabel\_2, GroupLayout.*PREFERRED\_SIZE*, 63, GroupLayout.*PREFERRED\_SIZE*)  .addGap(57)))  .addGroup(gl\_contentPane.createParallelGroup(Alignment.*LEADING*, false)  .addComponent(passwordField)  .addComponent(UserNameText, GroupLayout.*DEFAULT\_SIZE*, 181, Short.*MAX\_VALUE*))  .addContainerGap(187, Short.*MAX\_VALUE*))))  .addGroup(gl\_contentPane.createSequentialGroup()  .addGap(185)  .addComponent(lblNewLabel, GroupLayout.*DEFAULT\_SIZE*, 179, Short.*MAX\_VALUE*)  .addGap(177))  );  gl\_contentPane.setVerticalGroup(  gl\_contentPane.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_contentPane.createSequentialGroup()  .addGap(30)  .addComponent(lblNewLabel)  .addGap(39)  .addGroup(gl\_contentPane.createParallelGroup(Alignment.*BASELINE*)  .addComponent(UserNameText, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(lblNewLabel\_1))  .addGap(33)  .addGroup(gl\_contentPane.createParallelGroup(Alignment.*BASELINE*)  .addComponent(lblNewLabel\_2)  .addComponent(passwordField, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addGap(53)  .addGroup(gl\_contentPane.createParallelGroup(Alignment.*BASELINE*)  .addComponent(btnNewButton)  .addComponent(btnNewButton\_1))  .addContainerGap())  );  contentPane.setLayout(gl\_contentPane); */\*\* \* 设置界面居中显示 \*/* this.setLocationRelativeTo(null);  }    */\*\* \* 登录事件处理  \* @param e \*/* protected void LoginFrameActionPerformed(ActionEvent e) {  *// TODO Auto-generated method stub* String userName = this.UserNameText.getText();  String password = new String( passwordField.getPassword());   if(userName == null || userName.length() == 0) {  JOptionPane.*showMessageDialog*(null, "用户名不能为空!");  this.UserNameText.setText("");  return ;  }   if(password == null || password.length() == 0) {  JOptionPane.*showMessageDialog*(null, "密码不能为空!");  this.passwordField.setText("");  return ;  }   User user = new User(userName,password);  Connection conn = null;  conn = (Connection) JDBCconnection.*getConn*();  try {  User rs = userDao.login(user);  if(rs != null) {  JOptionPane.*showMessageDialog*(null, "登录成功!");  dispose();  new MainFrame().setVisible(true);  } else {  JOptionPane.*showMessageDialog*(null, "用户名或密码错误!");  }   } catch (Exception e1) {  e1.printStackTrace();  }  } }  存储管理员信息：  package 功能类;  import java.sql.Connection; import java.sql.PreparedStatement; import java.sql.ResultSet;   import 实体类.User; import 工具类.JDBCconnection;public class UserDAO {  */\*\*  \* 登录验证  \* @param user  \* @return 用户所有的信息  \* @throws Exception  \*/* public User login(User user) throws Exception {  Connection conn = null;  conn = (Connection) JDBCconnection.*getConn*();  User resultUser = null;   String sql = "select \* from user where username=? and password=?";  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sql);  pstmt.setString(1, user.getUsername());  pstmt.setString(2, user.getPassword());  ResultSet rs = pstmt.executeQuery();    if(rs.next()) {  resultUser = new User();  resultUser.setId(rs.getInt("id"));  resultUser.setUsername(rs.getString("username"));  resultUser.setPassword(rs.getString("password"));  }  return resultUser;  } }  主界面：  package 界面类;  import java.awt.BorderLayout; import java.awt.EventQueue; import javax.swing.JFrame; import javax.swing.JPanel; import javax.swing.border.EmptyBorder; import javax.swing.JMenuBar; import javax.swing.JMenu; import javax.swing.SpringLayout; import javax.swing.JMenuItem; import javax.swing.ImageIcon; import javax.swing.JDesktopPane; import javax.swing.GroupLayout; import javax.swing.GroupLayout.Alignment; import java.awt.event.ActionListener; import java.sql.SQLException; import java.awt.event.ActionEvent;  public class MainFrame extends JFrame {   private JPanel contentPane;  private JDesktopPane table = null;public static void main(String[] args) {  EventQueue.*invokeLater*(new Runnable() {  public void run() {  try {  MainFrame frame = new MainFrame();  frame.setVisible(true);  } catch (Exception e) {  e.printStackTrace();  } }  });  } public MainFrame() {  setTitle("餐饮管理系统主界面");  setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  setBounds(100, 100, 599, 377);    JMenuBar menuBar = new JMenuBar();  setJMenuBar(menuBar);    JMenu mnNewMenu = new JMenu("餐饮管理系统");   menuBar.add(mnNewMenu);    JMenuBar menuBar\_1 = new JMenuBar();  mnNewMenu.add(menuBar\_1);   JMenu Dish = new JMenu("菜单管理");  mnNewMenu.add(Dish);    JMenuItem DishAdd = new JMenuItem("菜单内容添加");  DishAdd.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  DishAddFrame df = new DishAddFrame();  df.setVisible(true);  table.add(df);  }  });  Dish.add(DishAdd);    JMenuItem mntmNewMenuItem\_14 = new JMenuItem("菜单内容维护");  mntmNewMenuItem\_14.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  DishManagerFrame df = new DishManagerFrame();  df.setVisible(true);  table.add(df);  }  });  Dish.add(mntmNewMenuItem\_14);    JMenu mnNewMenu\_6 = new JMenu("菜品管理");   mnNewMenu.add(mnNewMenu\_6);    JMenuItem mntmNewMenuItem\_7 = new JMenuItem("菜品内容添加");  mntmNewMenuItem\_7.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  CategoryAddFrame cate = new CategoryAddFrame();  cate.setVisible(true);  table.add(cate);  }  });  mnNewMenu\_6.add(mntmNewMenuItem\_7);    JMenuItem mntmNewMenuItem\_6 = new JMenuItem("菜品内容维护");  mntmNewMenuItem\_6.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  CategoryManageFrame cate;  try {  cate = new CategoryManageFrame();  cate.setVisible(true);  table.add(cate);  } catch (SQLException e1) {  *// TODO Auto-generated catch block* e1.printStackTrace();  }  }  });  mnNewMenu\_6.add(mntmNewMenuItem\_6);    JMenu mnNewMenu\_5 = new JMenu("点菜管理");   mnNewMenu.add(mnNewMenu\_5);    JMenuItem mntmNewMenuItem\_9 = new JMenuItem("点菜管理");  mntmNewMenuItem\_9.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  OrderDishFrame of = new OrderDishFrame();  of.setVisible(true);  table.add(of);  }  });  mnNewMenu\_5.add(mntmNewMenuItem\_9);    JMenu mnNewMenu\_4 = new JMenu("餐台管理");   mnNewMenu.add(mnNewMenu\_4);    JMenuItem mntmNewMenuItem\_10 = new JMenuItem("餐台管理添加");  mntmNewMenuItem\_10.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  DeskAddFrame da = new DeskAddFrame();  da.setVisible(true);  table.add(da);  }  });  mnNewMenu\_4.add(mntmNewMenuItem\_10);    JMenuItem mntmNewMenuItem\_11 = new JMenuItem("餐台管理维护");  mntmNewMenuItem\_11.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  DeskManageFrame df;  try {  df = new DeskManageFrame();  df.setVisible(true);  table.add(df);  } catch (Exception e1) {  *// TODO Auto-generated catch block* e1.printStackTrace();  }  }  });  mnNewMenu\_4.add(mntmNewMenuItem\_11);    JMenu mnNewMenu\_3 = new JMenu("客户管理");   mnNewMenu.add(mnNewMenu\_3);    JMenuItem mntmNewMenuItem\_5 = new JMenuItem("客户类别添加");  mntmNewMenuItem\_5.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  CustomerAddFrame cf = new CustomerAddFrame();  cf.setVisible(true);  table.add(cf);  }  });  mnNewMenu\_3.add(mntmNewMenuItem\_5);    JMenuItem mntmNewMenuItem\_4 = new JMenuItem("客户类别维护");  mntmNewMenuItem\_4.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  CustomerManagerFrame cf1 = new CustomerManagerFrame();  cf1.setVisible(true);  table.add(cf1);  }  });  mnNewMenu\_3.add(mntmNewMenuItem\_4);    JMenu mnNewMenu\_2 = new JMenu("员工管理");   mnNewMenu.add(mnNewMenu\_2);    JMenuItem mntmNewMenuItem\_2 = new JMenuItem("员工信息添加");  mntmNewMenuItem\_2.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  EmployeeAddFrame ef = new EmployeeAddFrame();  ef.setVisible(true);  table.add(ef);  }  });  mnNewMenu\_2.add(mntmNewMenuItem\_2);    JMenuItem mntmNewMenuItem\_3 = new JMenuItem("员工信息维护");  mntmNewMenuItem\_3.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  EmployeeManageFrame ef = new EmployeeManageFrame();  ef.setVisible(true);  table.add(ef);  }  });  mnNewMenu\_2.add(mntmNewMenuItem\_3);    contentPane = new JPanel();  contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));  setContentPane(contentPane);  contentPane.setLayout(new BorderLayout(0, 0));    table = new JDesktopPane();  contentPane.add(table, BorderLayout.*CENTER*);  GroupLayout gl\_table = new GroupLayout(table);  gl\_table.setHorizontalGroup(  gl\_table.createParallelGroup(Alignment.*LEADING*)  .addGap(0, 575, Short.*MAX\_VALUE*)   );   gl\_table.setVerticalGroup(  gl\_table.createParallelGroup(Alignment.*LEADING*)  .addGap(0, 307, Short.*MAX\_VALUE*)  );  table.setLayout(gl\_table);    this.setExtendedState(*MAXIMIZED\_BOTH*);  } }  存储客户信息：  package 实体类; public class Customer {  private int id;  private String name;  private String sex;  private String company;  private String tel;  private String cardID;  public Customer() {  super();  }    public Customer(int id,String name,String sex,String company,String tel,String cardID) {  super();  this.id = id;  this.name = name;  this.sex = sex;  this.company = company;  this.tel = tel;  this.cardID = cardID;  }    public Customer(String name,String sex,String company,String tel,String cardID) {  super();  this.name = name;  this.sex = sex;  this.company = company;  this.tel = tel;  this.cardID = cardID;  }    public int getId() {  return id;  }  public void setId(int id) {  this.id = id;  }  public String getName() {  return name;  }  public void setName(String name) {  this.name = name;  }  public String getSex() {  return sex;  }  public void setSex(String sex) {  this.sex = sex;  }  public String getCompany() {  return company;  }  public void setCompany(String company) {  this.company = company;  }  public String getTel() {  return tel;  }  public void setTel(String tel) {  this.tel = tel;  }  public String getCardID() {  return cardID;  }  public void setCardID(String cardID) {  this.cardID = cardID;  }  public String toString() {  return name;  } }  客户信息主界面：  package 界面类;  import java.awt.EventQueue; import java.sql.Connection; import java.sql.ResultSet; import java.util.Vector;  import javax.swing.JInternalFrame; import javax.swing.GroupLayout; import javax.swing.GroupLayout.Alignment; import javax.swing.JScrollPane; import javax.swing.JTable; import javax.swing.table.DefaultTableModel;  import 功能类.CategoryDAO; import 功能类.CustomerDAO; import 实体类.Category; import 实体类.Customer; import 工具类.JDBCconnection;  import javax.swing.LayoutStyle.ComponentPlacement; import javax.swing.JLabel; import javax.swing.JOptionPane; import javax.swing.JTextField; import javax.swing.JButton; import java.awt.event.ActionListener; import java.awt.event.ActionEvent; import javax.swing.JPanel; import javax.swing.border.TitledBorder; import java.awt.event.MouseAdapter; import java.awt.event.MouseEvent;  public class CustomerManagerFrame extends JInternalFrame {  private JTable Customertable;  private JTextField nameTxt;  private JTextField idText;  private JTextField nameText;  private JTextField sexText;  private JTextField telText;  private JTextField cardText;  private JTextField comText;  private CustomerDAO customerdao = new CustomerDAO();public static void main(String[] args) {  EventQueue.*invokeLater*(new Runnable() {  public void run() {  try {  CustomerManagerFrame frame = new CustomerManagerFrame();  frame.setVisible(true);  } catch (Exception e) {  e.printStackTrace();  }  }  });  }public CustomerManagerFrame() {  setIconifiable(true);  setClosable(true);  setTitle("客户管理");  setBounds(100, 100, 625, 607);    JScrollPane scrollPane = new JScrollPane();    JLabel lblNewLabel = new JLabel("客户名称");    nameTxt = new JTextField();  nameTxt.setColumns(10);    JButton btnNewButton = new JButton("查询");  btnNewButton.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  CustomerSerach(e);  }  });    JPanel panel = new JPanel();  panel.setBorder(new TitledBorder(null, "\u8868\u5355\u64CD\u4F5C", TitledBorder.*LEADING*, TitledBorder.*TOP*, null, null));  GroupLayout groupLayout = new GroupLayout(getContentPane());  groupLayout.setHorizontalGroup(  groupLayout.createParallelGroup(Alignment.*TRAILING*)  .addGroup(groupLayout.createSequentialGroup()  .addGap(91)  .addComponent(lblNewLabel, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addGap(52)  .addComponent(nameTxt, GroupLayout.*PREFERRED\_SIZE*, 163, GroupLayout.*PREFERRED\_SIZE*)  .addGap(57)  .addComponent(btnNewButton, GroupLayout.*PREFERRED\_SIZE*, 97, GroupLayout.*PREFERRED\_SIZE*)  .addContainerGap(95, Short.*MAX\_VALUE*))  .addGroup(groupLayout.createSequentialGroup()  .addContainerGap(82, Short.*MAX\_VALUE*)  .addGroup(groupLayout.createParallelGroup(Alignment.*TRAILING*)  .addComponent(panel, Alignment.*LEADING*, GroupLayout.*DEFAULT\_SIZE*, 452, Short.*MAX\_VALUE*)  .addComponent(scrollPane, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addGap(79))  );  groupLayout.setVerticalGroup(  groupLayout.createParallelGroup(Alignment.*LEADING*)  .addGroup(groupLayout.createSequentialGroup()  .addContainerGap(27, Short.*MAX\_VALUE*)  .addGroup(groupLayout.createParallelGroup(Alignment.*BASELINE*)  .addComponent(lblNewLabel)  .addComponent(nameTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(btnNewButton))  .addPreferredGap(ComponentPlacement.*UNRELATED*)  .addComponent(scrollPane, GroupLayout.*PREFERRED\_SIZE*, 260, GroupLayout.*PREFERRED\_SIZE*)  .addGap(26)  .addComponent(panel, GroupLayout.*PREFERRED\_SIZE*, 210, GroupLayout.*PREFERRED\_SIZE*)  .addGap(22))  );    JLabel lblNewLabel\_1 = new JLabel("编号:");    idText = new JTextField();  idText.setEditable(false);  idText.setColumns(10);    JLabel lblNewLabel\_2 = new JLabel("客户姓名:");    nameText = new JTextField();  nameText.setColumns(10);    JLabel lblNewLabel\_3 = new JLabel("性别:");    sexText = new JTextField();  sexText.setColumns(10);    JLabel lblNewLabel\_4 = new JLabel("电话号码:");    telText = new JTextField();  telText.setColumns(10);    JLabel lblNewLabel\_5 = new JLabel("贵宾卡号:");    cardText = new JTextField();  cardText.setColumns(10);    JButton change = new JButton("修改");  change.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  changeActionEvent(e);  }  });    JButton delete = new JButton("删除");  delete.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  DeleteActionPerformed(e);  }  });    JLabel companyText = new JLabel("客户单位:");    comText = new JTextField();  comText.setColumns(10);  GroupLayout gl\_panel = new GroupLayout(panel);  gl\_panel.setHorizontalGroup(  gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_panel.createSequentialGroup()  .addGroup(gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_panel.createSequentialGroup()  .addContainerGap()  .addGroup(gl\_panel.createParallelGroup(Alignment.*LEADING*, false)  .addGroup(gl\_panel.createSequentialGroup()  .addComponent(lblNewLabel\_1, GroupLayout.*PREFERRED\_SIZE*, 44, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*RELATED*)  .addComponent(idText, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addGroup(gl\_panel.createSequentialGroup()  .addComponent(lblNewLabel\_2)  .addGap(14)  .addComponent(nameText, GroupLayout.*PREFERRED\_SIZE*, 91, GroupLayout.*PREFERRED\_SIZE*))  .addGroup(gl\_panel.createSequentialGroup()  .addComponent(lblNewLabel\_3, GroupLayout.*PREFERRED\_SIZE*, 52, GroupLayout.*PREFERRED\_SIZE*)  .addGap(18)  .addComponent(sexText)))  .addGap(37)  .addGroup(gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_panel.createSequentialGroup()  .addComponent(companyText, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addGap(18)  .addComponent(comText, GroupLayout.*DEFAULT\_SIZE*, 138, Short.*MAX\_VALUE*))  .addGroup(gl\_panel.createParallelGroup(Alignment.*LEADING*, false)  .addGroup(gl\_panel.createSequentialGroup()  .addComponent(lblNewLabel\_4, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*UNRELATED*)  .addComponent(telText, GroupLayout.*PREFERRED\_SIZE*, 156, GroupLayout.*PREFERRED\_SIZE*))  .addGroup(gl\_panel.createSequentialGroup()  .addComponent(lblNewLabel\_5, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*UNRELATED*)  .addComponent(cardText)))))  .addGroup(gl\_panel.createSequentialGroup()  .addComponent(change, GroupLayout.*PREFERRED\_SIZE*, 97, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*RELATED*, 236, Short.*MAX\_VALUE*)  .addComponent(delete, GroupLayout.*PREFERRED\_SIZE*, 97, GroupLayout.*PREFERRED\_SIZE*)))  .addContainerGap())  );  gl\_panel.setVerticalGroup(  gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_panel.createSequentialGroup()  .addGap(18)  .addGroup(gl\_panel.createParallelGroup(Alignment.*BASELINE*)  .addComponent(lblNewLabel\_1)  .addComponent(idText, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(companyText)  .addComponent(comText, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addGap(18)  .addGroup(gl\_panel.createParallelGroup(Alignment.*BASELINE*)  .addComponent(nameText, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(lblNewLabel\_2)  .addComponent(lblNewLabel\_4)  .addComponent(telText, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addGap(18)  .addGroup(gl\_panel.createParallelGroup(Alignment.*BASELINE*)  .addComponent(lblNewLabel\_3)  .addComponent(lblNewLabel\_5)  .addComponent(cardText, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(sexText, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addPreferredGap(ComponentPlacement.*RELATED*, 25, Short.*MAX\_VALUE*)  .addGroup(gl\_panel.createParallelGroup(Alignment.*BASELINE*)  .addComponent(change)  .addComponent(delete))  .addGap(22))  );  panel.setLayout(gl\_panel);    Customertable = new JTable();  Customertable.addMouseListener(new MouseAdapter() {  @Override  public void mousePressed(MouseEvent e) {  customerTablePressed(e);  }  });  Customertable.setModel(new DefaultTableModel(  new Object[][] {  },  new String[] {  }  ) {  boolean[] columnEditables = new boolean[] {  false, false, true, true, true, true  };  public boolean isCellEditable(int row, int column) {  return columnEditables[column];  }  });  Customertable.getColumnModel().getColumn(0).setResizable(false);  Customertable.getColumnModel().getColumn(4).setPreferredWidth(115);  scrollPane.setViewportView(Customertable);  getContentPane().setLayout(groupLayout);  this.fillTable(new Customer());   } private void customerTablePressed(MouseEvent e) {  *// TODO Auto-generated method stub* int row = this.Customertable.getSelectedRow();*//选中获得的行号;* Integer id = (Integer) Customertable.getValueAt(row, 0);  idText.setText(id.toString());    this.nameText.setText((String) Customertable.getValueAt(row, 1));  this.sexText.setText((String)Customertable.getValueAt(row, 2));  this.comText.setText((String) Customertable.getValueAt(row, 3));  this.telText.setText((String)Customertable.getValueAt(row, 4));  this.cardText.setText((String)Customertable.getValueAt(row, 5));      } private void DeleteActionPerformed(ActionEvent e) {  *// TODO Auto-generated method stub* String id = this.idText.getText();  if(id == null || id.length() == 0) {  JOptionPane.*showMessageDialog*(null, "请选择要删除的记录!");  return ;  }    int flag = JOptionPane.*showConfirmDialog*(null, "确定要删除记录吗？");  if(flag == 0) {  Connection conn = null;  try {  conn = (Connection) JDBCconnection.*getConn*();  int yes = customerdao.delete(Integer.*parseInt*(id));  if(yes == 1) {  JOptionPane.*showMessageDialog*(null, "删除成功!");  this.resetValue();  this.fillTable(new Customer());  } else {  JOptionPane.*showMessageDialog*(null, "删除失败!");  this.resetValue();  }  } catch(Exception e1) {  e1.printStackTrace();  } finally {  JDBCconnection.*close*(conn, null, null);  }  }  }private void changeActionEvent(ActionEvent e) {  *// TODO Auto-generated method stub* String id = this.idText.getText();  String name = this.nameText.getText();  String sex = this.sexText.getText();  String company = this.comText.getText();  String tel = this.telText.getText();  String cardid = this.cardText.getText();    if(name == null || name.length() == 0) {  JOptionPane.*showMessageDialog*(null, "请选择要修改的记录!");  return ;  }  if(cardid == null || cardid.length() == 0) {  JOptionPane.*showMessageDialog*(null, "贵宾卡号不能为空!");  return ;  }    Customer c = new Customer(Integer.*parseInt*(id),name,sex,company,tel,cardid);  Connection conn = (Connection) JDBCconnection.*getConn*();  try {  int flag = customerdao.update(c);  if(flag == 1) {  JOptionPane.*showMessageDialog*(null, "修改成功!");  resetValue();  this.fillTable(new Customer());  } else {  JOptionPane.*showMessageDialog*(null, "修改失败!");  resetValue();  }  } catch (Exception e1) {  e1.printStackTrace();  }    }private void CustomerSerach(ActionEvent evt) {  *// TODO Auto-generated method stub* String name = this.nameTxt.getText();  Customer c = new Customer();  c.setName(name);  this.fillTable(c);  }   private void fillTable(Customer customer) {  DefaultTableModel dtm = (DefaultTableModel)Customertable.getModel();  dtm.setRowCount(0); *//设置成0行 相当于清空表格* Connection conn = null;    try {  conn = (Connection )JDBCconnection.*getConn*();  try {  ResultSet rs = CustomerDAO.*list*(customer);  while(rs.next()) {  Vector v = new Vector();  v.add(rs.getInt("id"));  v.add(rs.getString("name"));  v.add(rs.getString("sex"));  v.add(rs.getString("company"));  v.add(rs.getString("tel"));  v.add(rs.getString("cardID"));  dtm.addRow(v);  }  } catch (Exception e) {  *// TODO Auto-generated catch block* e.printStackTrace();  }  } catch (Exception e) {  e.printStackTrace();  } finally {  JDBCconnection.*close*(conn, null, null);  }  }    private void resetValue() {  this.cardText.setText("");  this.telText.setText("");  this.nameText.setText("");  this.sexText.setText("");  this.comText.setText("");  this.idText.setText("");  }  }  客户信息功能实现：  package 功能类;  import java.sql.Connection; import java.sql.PreparedStatement; import java.sql.ResultSet; import java.sql.SQLException;  import 实体类.Category; import 实体类.Customer; import 实体类.Employee; import 工具类.JDBCconnection;  public class CustomerDAO {  */\*\*  \* 客户类被添加;  \* @param customer  \* @return  \* @throws SQLException  \*/* public int add(Customer customer) throws SQLException {  Connection conn = (Connection)JDBCconnection.*getConn*();    String sql = "insert into customer(name,sex,company,tel,cardID) values(?,?,?,?,?)";  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sql);  pstmt.setString(1, customer.getName());  pstmt.setString(2, customer.getSex());  pstmt.setString(3, customer.getCompany());  pstmt.setString(4, customer.getTel());  pstmt.setString(5, customer.getCardID());   return pstmt.executeUpdate();  }  */\*\*  \* 客户类别查询  \*/* public static ResultSet list(Customer c) throws Exception { *//这里可能不一样!* Connection conn = (Connection)JDBCconnection.*getConn*();  StringBuffer sb = new StringBuffer("select \* from customer");  if(c.getName() != null) {  if(c.getName().length() != 0)  sb.append(" and name like '%"+c.getName()+"%'");  }  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sb.toString().replaceFirst("and", "where"));  return pstmt.executeQuery();  }   */\*\*  \* 客户类别删除  \* @param id  \* @return  \* @throws Exception  \*/* public int delete(int id) throws Exception{  String sql = "delete from customer where id=?";  Connection conn = (Connection)JDBCconnection.*getConn*();  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sql);  pstmt.setInt(1, id);  return pstmt.executeUpdate();  }   */\*\*  \* 更新客户类别  \* @param c  \* @return  \* @throws Exception  \*/* public int update(Customer c) throws Exception{  String sql = "update customer set name =?,sex =?,company=?,tel=?,cardID=? where id =?";  Connection conn = (Connection)JDBCconnection.*getConn*();  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sql);  pstmt.setString(1, c.getName());  pstmt.setString(2, c.getSex());  pstmt.setString(3, c.getCompany());  pstmt.setString(4, c.getTel());  pstmt.setString(5, c.getCardID());  pstmt.setInt(6, c.getId());  return pstmt.executeUpdate();  }   public static void main(String[] args) {  try {  ResultSet rs = CustomerDAO.*list*(new Customer());  while(rs.next()) {  System.*out*.println(rs.getString("name"));  }  } catch (Exception e) {  *// TODO Auto-generated catch block* e.printStackTrace();  }   } }  连接数据库：  package 工具类;  import java.sql.\*; public class JDBCconnection {    private static final String *DRIVER* = "com.mysql.cj.jdbc.Driver";  private static final Connection *conn* = null;  private static final String *URL* = "jdbc:mysql://localhost:3306/canyin?useUnicode=true&characterEncoding=utf8";  private static final String *USERNAME* = "root";  private static final String *PASSWORD* = "1234";    static {  try {  Class.*forName*(*DRIVER*);  } catch (ClassNotFoundException e) {  e.printStackTrace();  }  }   public static void main(String[] args) {  System.*out*.println(*getConn*());  }    public static Connection getConn() {  Connection conn = null;  try {  conn = DriverManager.*getConnection*(*URL*,*USERNAME*,*PASSWORD*);  } catch (Exception e) {  e.printStackTrace();  }  return conn;  }    public static void close(Connection conn,Statement st,ResultSet rs) {  try {  if(rs != null) rs.close();  if(st != null) st.close();  if(conn != null) conn.close();  } catch (Exception e) {  e.printStackTrace();  }  } }  菜品类别管理界面：  package 界面类;  import java.awt.EventQueue; import java.sql.Connection; import java.sql.ResultSet; import java.sql.SQLException; import java.util.Vector;  import javax.swing.JInternalFrame; import javax.swing.GroupLayout; import javax.swing.GroupLayout.Alignment; import javax.swing.JScrollPane; import javax.swing.JTable; import javax.swing.table.DefaultTableModel;  import 功能类.CategoryDAO; import 功能类.DishDAO; import 实体类.Category; import 工具类.JDBCconnection; import javax.swing.JLabel; import javax.swing.JOptionPane; import javax.swing.LayoutStyle.ComponentPlacement; import javax.swing.JTextField; import javax.swing.JButton; import javax.swing.ImageIcon; import java.awt.event.ActionListener; import java.awt.event.ActionEvent; import javax.swing.JPanel; import javax.swing.border.TitledBorder; import javax.swing.JTextArea; import java.awt.event.MouseAdapter; import java.awt.event.MouseEvent; */\*\* \* 查询菜品种类 \*@author yuoga \* \*/* public class CategoryManageFrame extends JInternalFrame {  private JTable CategoryTable;  private CategoryDAO categoryDAO = new CategoryDAO();  private final JTextField Scategory = new JTextField();  private final JButton btnNewButton\_2 = new JButton("查询");  private final JPanel panel = new JPanel();  private final JTextField idText = new JTextField();  private final DishDAO dishdao = new DishDAO();  private final JLabel lblNewLabel\_1 = new JLabel("编号:");  private final JLabel lblNewLabel\_2 = new JLabel("菜品名称：");  private final JLabel lblNewLabel\_3 = new JLabel("菜品描述:");  private final JTextField CategoryNameTxt = new JTextField();  private final JTextArea categoryArea = new JTextArea();  private final JButton deleteButton = new JButton("删除");  private final JButton ChangeButton = new JButton("修改");  */\*\*  \* Launch the application.  \*/* public static void main(String[] args) {  EventQueue.*invokeLater*(new Runnable() {  public void run() {  try {  CategoryManageFrame frame = new CategoryManageFrame();  frame.setVisible(true);  } catch (Exception e) {  e.printStackTrace();  }  }  });  }   */\*\*  \* Create the frame.  \* @throws SQLException   \*/* public CategoryManageFrame() throws SQLException {  panel.setBorder(new TitledBorder(null, "\u83DC\u54C1\u79CD\u7C7B\u64CD\u4F5C", TitledBorder.*LEADING*, TitledBorder.*TOP*, null, null));  btnNewButton\_2.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  SearchCategoryActionPerformed(e);  }  });  setClosable(true);  setIconifiable(true);  setTitle("菜品类别管理");  setBounds(100, 100, 551, 590);    JScrollPane scrollPane = new JScrollPane();  scrollPane.addMouseListener(new MouseAdapter() {  @Override  */\*\*  \* 鼠标点击事件;  \* @param e  \*/* public void mousePressed(MouseEvent e) {  CategoryTableMousePressed(e);  }  });    JLabel lblNewLabel = new JLabel("菜品类别名称：");  Scategory.setColumns(10);  GroupLayout groupLayout = new GroupLayout(getContentPane());  groupLayout.setHorizontalGroup(  groupLayout.createParallelGroup(Alignment.*LEADING*)  .addGroup(groupLayout.createSequentialGroup()  .addGap(45)  .addComponent(lblNewLabel, GroupLayout.*PREFERRED\_SIZE*, 102, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*RELATED*)  .addComponent(Scategory, GroupLayout.*PREFERRED\_SIZE*, 216, GroupLayout.*PREFERRED\_SIZE*)  .addGap(18)  .addComponent(btnNewButton\_2)  .addContainerGap(133, Short.*MAX\_VALUE*))  .addGroup(Alignment.*TRAILING*, groupLayout.createSequentialGroup()  .addGap(23)  .addGroup(groupLayout.createParallelGroup(Alignment.*TRAILING*)  .addComponent(panel, Alignment.*LEADING*, GroupLayout.*DEFAULT\_SIZE*, 522, Short.*MAX\_VALUE*)  .addComponent(scrollPane, Alignment.*LEADING*, 0, 0, Short.*MAX\_VALUE*))  .addGap(50))  );  groupLayout.setVerticalGroup(  groupLayout.createParallelGroup(Alignment.*LEADING*)  .addGroup(groupLayout.createSequentialGroup()  .addGap(42)  .addGroup(groupLayout.createParallelGroup(Alignment.*BASELINE*)  .addComponent(lblNewLabel)  .addComponent(Scategory, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(btnNewButton\_2))  .addPreferredGap(ComponentPlacement.*UNRELATED*)  .addComponent(scrollPane, GroupLayout.*PREFERRED\_SIZE*, 172, GroupLayout.*PREFERRED\_SIZE*)  .addGap(18)  .addComponent(panel, GroupLayout.*PREFERRED\_SIZE*, 257, GroupLayout.*PREFERRED\_SIZE*)  .addContainerGap(240, Short.*MAX\_VALUE*))  );  idText.setEditable(false);  idText.setColumns(10);  CategoryNameTxt.setColumns(10);  GroupLayout gl\_panel = new GroupLayout(panel);  gl\_panel.setHorizontalGroup(  gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_panel.createSequentialGroup()  .addGap(19)  .addGroup(gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_panel.createSequentialGroup()  .addGroup(gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_panel.createParallelGroup(Alignment.*TRAILING*, false)  .addComponent(lblNewLabel\_3, Alignment.*LEADING*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, Short.*MAX\_VALUE*)  .addComponent(lblNewLabel\_2, Alignment.*LEADING*, GroupLayout.*DEFAULT\_SIZE*, 79, Short.*MAX\_VALUE*))  .addComponent(lblNewLabel\_1, GroupLayout.*PREFERRED\_SIZE*, 69, GroupLayout.*PREFERRED\_SIZE*))  .addGap(18)  .addGroup(gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addComponent(idText, GroupLayout.*PREFERRED\_SIZE*, 95, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(CategoryNameTxt, GroupLayout.*DEFAULT\_SIZE*, 307, Short.*MAX\_VALUE*)  .addComponent(categoryArea, GroupLayout.*DEFAULT\_SIZE*, 304, Short.*MAX\_VALUE*))  .addContainerGap())  .addGroup(gl\_panel.createSequentialGroup()  .addComponent(ChangeButton, GroupLayout.*PREFERRED\_SIZE*, 77, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*RELATED*, 266, Short.*MAX\_VALUE*)  .addComponent(deleteButton, GroupLayout.*PREFERRED\_SIZE*, 67, GroupLayout.*PREFERRED\_SIZE*)  .addGap(25))))  );  ChangeButton.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  categoryUpdateActionEvent(e);  }  });  deleteButton.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  CategoryDeleteActionEvent(e);  String id = idText.getText();  int ID = Integer.*parseInt*(id);    if(id == null || id.length() == 0) {  JOptionPane.*showMessageDialog*(null, "请选择要删除的记录!");  return;  }   int n = JOptionPane.*showConfirmDialog*(null, "确定删除该条记录吗?");  if(n == 0) {  Connection conn = null;  try {  conn = (Connection) JDBCconnection.*getConn*();  boolean help = dishdao.existDish(ID);    if(help) {  JOptionPane.*showMessageDialog*(null, "当前菜单类别下有菜品，不能删除此类别!");  return ;  }   int flag = categoryDAO.delete(ID);  if(flag == 1) {  JOptionPane.*showMessageDialog*(null, "删除成功!");  resetValue();  fillTable(new Category());  } else {  JOptionPane.*showMessageDialog*(null, "删除失败!");  resetValue();  }  } catch (Exception e1) {  e1.printStackTrace();  JOptionPane.*showMessageDialog*(null, "删除失败!");  } finally {  try {  JDBCconnection.*close*(conn, null, null);  } catch(Exception evt) {  evt.printStackTrace();  }  }  }  }  });  gl\_panel.setVerticalGroup(  gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_panel.createSequentialGroup()  .addContainerGap()  .addGroup(gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_panel.createParallelGroup(Alignment.*BASELINE*)  .addComponent(lblNewLabel\_1)  .addComponent(idText, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addGroup(gl\_panel.createSequentialGroup()  .addGap(21)  .addGroup(gl\_panel.createParallelGroup(Alignment.*BASELINE*)  .addComponent(lblNewLabel\_2, GroupLayout.*PREFERRED\_SIZE*, 38, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(CategoryNameTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addPreferredGap(ComponentPlacement.*RELATED*)  .addGroup(gl\_panel.createParallelGroup(Alignment.*BASELINE*)  .addComponent(lblNewLabel\_3, GroupLayout.*PREFERRED\_SIZE*, 29, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(categoryArea, GroupLayout.*PREFERRED\_SIZE*, 78, GroupLayout.*PREFERRED\_SIZE*))))  .addGap(18)  .addGroup(gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addComponent(ChangeButton)  .addComponent(deleteButton))  .addGap(15))  );  panel.setLayout(gl\_panel);    CategoryTable = new JTable();  CategoryTable.addMouseListener(new MouseAdapter() {  @Override  public void mousePressed(MouseEvent e) {  CategoryTableMousePressed(e);  }  });  CategoryTable.setModel(new DefaultTableModel(  new Object[][] {  },  new String[] {  "\u7F16\u53F7", "\u83DC\u54C1\u7C7B\u522B\u540D\u79F0", "\u83DC\u54C1\u7C7B\u522B\u63CF\u8FF0"  }  ) {  boolean[] columnEditables = new boolean[] {  false, false, true  };  public boolean isCellEditable(int row, int column) {  return columnEditables[column];  }  });  CategoryTable.getColumnModel().getColumn(0).setPreferredWidth(37);  CategoryTable.getColumnModel().getColumn(2).setPreferredWidth(103);  scrollPane.setViewportView(CategoryTable);  getContentPane().setLayout(groupLayout);  fillTable(new Category());  }  */\*\*  \* 删除表格记录;  \* @param e  \*/* private void CategoryDeleteActionEvent(ActionEvent e) {  *// TODO Auto-generated method stub* }  */\*\*  \* 修改表格;  \* @param e  \*/* private void categoryUpdateActionEvent(ActionEvent e) {  *// TODO Auto-generated method stub* String id = idText.getText();  String Name = CategoryNameTxt.getText();  String Descirb = categoryArea.getText();    if(id == null || id.length() == 0) {  JOptionPane.*showMessageDialog*(null, "请选择要修改的记录!");  return ;  }  if(Name == null || Name.length() == 0){  JOptionPane.*showMessageDialog*(null, "菜品类别名称不能为空!");  return ;  }   Category category = new Category(Integer.*parseInt*(id),Name,Descirb);  Connection conn = null;  try {  conn = (Connection) JDBCconnection.*getConn*();  int flag = categoryDAO.update(category);  if(flag == 1) {  JOptionPane.*showMessageDialog*(null, "修改成功!");  resetValue();  this.fillTable(new Category());  } else {  JOptionPane.*showMessageDialog*(null, "修改失败!");  resetValue();  }  } catch(Exception e1) {  e1.printStackTrace();  } finally {  try {  JDBCconnection.*close*(conn, null, null);  } catch (Exception e2) {  e2.printStackTrace();  }  }  }  */\*\*  \* 表格行点击事件处理   \*/* private void CategoryTableMousePressed(MouseEvent evt) {  *// TODO Auto-generated method stub* int row = CategoryTable.getSelectedRow();  Integer id = (Integer) CategoryTable.getValueAt(row, 0);  idText.setText(id.toString());  CategoryNameTxt.setText((String) CategoryTable.getValueAt(row, 1));  categoryArea.setText((String)CategoryTable.getValueAt(row, 2));  }   */\*\*  \* 菜品种类搜索事件处理;  \* @param evt  \*/* private void SearchCategoryActionPerformed(ActionEvent evt) {  *// TODO Auto-generated method stub* String categoryName = this.Scategory.getText();  Category category = new Category();  category.setName(categoryName);  try {  this.fillTable(category);  } catch (SQLException e) {  *// TODO Auto-generated catch block* e.printStackTrace();  }   }   */\*\*  \* 清空表格;  \* @param category  \* @throws SQLException  \*/* private void fillTable(Category category) throws SQLException {  DefaultTableModel dtm = (DefaultTableModel)CategoryTable.getModel();  dtm.setRowCount(0); *//设置成0行 相当于清空表格* Connection conn = null;   try {  conn = (Connection )JDBCconnection.*getConn*();  try {  ResultSet rs = CategoryDAO.*list*(category);  while(rs.next()) {  Vector v = new Vector();  v.add(rs.getInt("id"));  v.add(rs.getString("name"));  v.add(rs.getString("describ"));  dtm.addRow(v);  }  } catch (Exception e) {  *// TODO Auto-generated catch block* e.printStackTrace();  }  } catch (Exception e) {  e.printStackTrace();  } finally {  JDBCconnection.*close*(conn, null, null);  }  }  private static class \_\_Tmp {  private static void \_\_tmp() {  javax.swing.JPanel \_\_wbp\_panel = new javax.swing.JPanel();  }  }  */\*\*  \* 重置表单;  \*/* private void resetValue() {  this.categoryArea.setText("");  this.CategoryNameTxt.setText("");  this.idText.setText("");  } }  菜品类别添加界面：  package 界面类;  import java.awt.EventQueue;  import javax.swing.JInternalFrame; import javax.swing.GroupLayout; import javax.swing.GroupLayout.Alignment; import javax.swing.JLabel; import javax.swing.JOptionPane; import javax.swing.LayoutStyle.ComponentPlacement; import 功能类.CategoryDAO; import 实体类.Category; import 工具类.JDBCconnection; import javax.swing.JTextField; import javax.swing.JTextArea; import javax.swing.JButton; import java.awt.event.ActionListener; import java.sql.Connection; import java.sql.SQLException; import java.awt.event.ActionEvent; import javax.swing.ImageIcon;  public class CategoryAddFrame extends JInternalFrame {  private JTextField CategoryNameTxt;  private JTextArea CategoryDescrib;  private CategoryDAO categorydao = new CategoryDAO();  */\*\*  \* Launch the application.  \*/* public static void main(String[] args) {  EventQueue.*invokeLater*(new Runnable() {  public void run() {  try {  CategoryAddFrame frame = new CategoryAddFrame();  frame.setVisible(true);  } catch (Exception e) {  e.printStackTrace();  }  }  });  }  */\*\*  \* Create the frame.  \*/* public CategoryAddFrame() {  setClosable(true);  setIconifiable(true);  setTitle("菜品种类添加");  setBounds(100, 100, 450, 300);    JLabel lblNewLabel = new JLabel("菜品类别名称:");    JLabel lblNewLabel\_1 = new JLabel("菜品类别描述:");    CategoryNameTxt = new JTextField();  CategoryNameTxt.setColumns(10);   CategoryDescrib = new JTextArea();    JButton btnNewButton = new JButton("添加");   btnNewButton.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  try {  CategoryAddActionPerformed(e);  } catch (SQLException e1) {  *// TODO Auto-generated catch block* e1.printStackTrace();  }  }  });   JButton btnNewButton\_1 = new JButton("重置");  btnNewButton\_1.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  resetValueActionPerformed(e);  }   });  GroupLayout groupLayout = new GroupLayout(getContentPane());  groupLayout.setHorizontalGroup(  groupLayout.createParallelGroup(Alignment.*LEADING*)  .addGroup(groupLayout.createSequentialGroup()  .addGap(45)  .addGroup(groupLayout.createParallelGroup(Alignment.*LEADING*)  .addComponent(lblNewLabel\_1, GroupLayout.*PREFERRED\_SIZE*, 107, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(lblNewLabel, GroupLayout.*PREFERRED\_SIZE*, 116, GroupLayout.*PREFERRED\_SIZE*))  .addPreferredGap(ComponentPlacement.*UNRELATED*)  .addGroup(groupLayout.createParallelGroup(Alignment.*LEADING*, false)  .addComponent(CategoryNameTxt)  .addComponent(CategoryDescrib, GroupLayout.*DEFAULT\_SIZE*, 240, Short.*MAX\_VALUE*))  .addContainerGap(27, Short.*MAX\_VALUE*))  .addGroup(Alignment.*TRAILING*, groupLayout.createSequentialGroup()  .addGap(37)  .addComponent(btnNewButton, GroupLayout.*PREFERRED\_SIZE*, 97, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*RELATED*, 161, Short.*MAX\_VALUE*)  .addComponent(btnNewButton\_1, GroupLayout.*PREFERRED\_SIZE*, 97, GroupLayout.*PREFERRED\_SIZE*)  .addGap(46))  );  groupLayout.setVerticalGroup(  groupLayout.createParallelGroup(Alignment.*LEADING*)  .addGroup(groupLayout.createSequentialGroup()  .addContainerGap(37, Short.*MAX\_VALUE*)  .addGroup(groupLayout.createParallelGroup(Alignment.*BASELINE*)  .addComponent(lblNewLabel)  .addComponent(CategoryNameTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addGap(18)  .addGroup(groupLayout.createParallelGroup(Alignment.*BASELINE*)  .addComponent(lblNewLabel\_1)  .addComponent(CategoryDescrib, GroupLayout.*PREFERRED\_SIZE*, 86, GroupLayout.*PREFERRED\_SIZE*))  .addGap(18)  .addGroup(groupLayout.createParallelGroup(Alignment.*BASELINE*)  .addComponent(btnNewButton\_1)  .addComponent(btnNewButton))  .addContainerGap(27, Short.*MAX\_VALUE*))  );  getContentPane().setLayout(groupLayout);  }  */\*\*  \* 菜品类别添加事件处理  \* @param e  \* @throws SQLException   \*/* private void CategoryAddActionPerformed(ActionEvent e) throws SQLException {  *// TODO Auto-generated method stub* String CateName = this.CategoryNameTxt.getText();  String CateDescrib = this.CategoryDescrib.getText();    if(CateName == null || CateName.length() == 0) {  JOptionPane.*showMessageDialog*(null, "菜品类别名称不能为空!");  resetValue();  return ;  }    Category category = new Category(CateName,CateDescrib);  Connection conn = null;  try {  int flag = categorydao.add(category);  if(flag == 1) {  JOptionPane.*showMessageDialog*(null, "菜品类别添加成功!");  resetValue();  } else {  JOptionPane.*showMessageDialog*(null, "菜品类别添加失败!");  resetValue();  }  } catch (Exception e1) {  e1.printStackTrace();  } finally {  try {  conn.close();  } catch(SQLException e2) {  e2.printStackTrace();  }  }    }  */\*\*  \* 重置事件处理  \* @param e  \*/* private void resetValueActionPerformed(ActionEvent e) {  *// TODO Auto-generated method stub* this.resetValue();  }  */\*\*  \* 重置表单  \*/* private void resetValue() {  this.CategoryNameTxt.setText("");  this.CategoryDescrib.setText("");  } }  菜品类别功能实现：  package 功能类;  import 实体类.Category; import 工具类.JDBCconnection;  import java.sql.Connection; import java.sql.PreparedStatement; import java.sql.ResultSet; */\*\* \* 菜品类别Dao类 \* @author yuoga \* \*/* public class CategoryDAO {  */\* \* 菜品类别添加 \*/* public int add(Category c) throws Exception{  Connection conn = (Connection)JDBCconnection.*getConn*();    String sql = "insert into category(name,describ) values(?,?)";  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sql);  pstmt.setString(1, c.getName());  pstmt.setString(2, c.getDescrib());   return pstmt.executeUpdate();  } */\*\* \* 查询菜品类别集合  \* @param c  \* @return  \* @throws Exception \*/* public static ResultSet list(Category c) throws Exception { *//这里可能不一样!* Connection conn = (Connection)JDBCconnection.*getConn*();  StringBuffer sb = new StringBuffer("select \* from category");  if(c.getName() != null) {  if(c.getName().length() != 0)  sb.append(" and name like '%"+c.getName()+"%'");  }  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sb.toString().replaceFirst("and", "where"));  return pstmt.executeQuery();  }  */\*\* \* 删除菜品类别  \* @param id  \* @return  \* @throws Exception \*/* public int delete(int id) throws Exception{  String sql = "delete from category where id=?";  Connection conn = (Connection)JDBCconnection.*getConn*();  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sql);  pstmt.setInt(1, id);  return pstmt.executeUpdate();  }  */\*\* \* 更新菜品类别  \* @param c  \* @return  \* @throws Exception \*/* public int update(Category c) throws Exception{  String sql = "update category set name =?,describ =? where id =?";  Connection conn = (Connection)JDBCconnection.*getConn*();  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sql);   pstmt.setString(1, c.getName());   pstmt.setString(2, c.getDescrib());   pstmt.setInt(3, c.getId());  return pstmt.executeUpdate();  }  }   员工管理界面：  package 界面类;  import java.awt.EventQueue; import java.sql.ResultSet; import java.util.Vector;  import javax.swing.JInternalFrame; import javax.swing.GroupLayout; import javax.swing.GroupLayout.Alignment; import javax.swing.JScrollPane; import javax.swing.JTable; import javax.swing.LayoutStyle.ComponentPlacement; import javax.swing.table.DefaultTableModel;  import 功能类.EmployeeDAO; import 实体类.Employee;  import javax.swing.JLabel; import javax.swing.JOptionPane; import javax.swing.JTextField; import javax.swing.JButton; import javax.swing.JPanel; import javax.swing.border.TitledBorder; import java.awt.event.MouseAdapter; import java.awt.event.MouseEvent; import java.awt.event.ActionListener; import java.awt.event.ActionEvent;  public class EmployeeManageFrame extends JInternalFrame {  private JTable employeeTable;  private JTextField SearchTxt;  private EmployeeDAO employeedao = new EmployeeDAO();  private JTextField EidTxt;  private JTextField EnameTxt;  private JTextField EsexTxt;  private JTextField yearTxt;  private JTextField monthTxt;  private JTextField dayTxt;  private JTextField addressTxt;  private JTextField TelTxt;  private JTextField PositionTxt;  private JTextField StatusTxt;  private JTextField identity;  */\*\*  \* Launch the application.  \*/* public static void main(String[] args) {  EventQueue.*invokeLater*(new Runnable() {  public void run() {  try {  EmployeeManageFrame frame = new EmployeeManageFrame();  frame.setVisible(true);  } catch (Exception e) {  e.printStackTrace();  }  }  });  }  */\*\*  \* Create the frame.  \*/* public EmployeeManageFrame() {  setIconifiable(true);  setClosable(true);  setTitle("员工管理");  setBounds(100, 100, 837, 737);    JScrollPane scrollPane = new JScrollPane();    JLabel lblNewLabel = new JLabel("员工姓名:");    SearchTxt = new JTextField();  SearchTxt.setColumns(10);    JButton btnNewButton = new JButton("查询");  btnNewButton.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  EmployeeSearchActionEvent(e);  }  });    JPanel panel = new JPanel();  panel.setBorder(new TitledBorder(null, "\u8868\u5355\u64CD\u4F5C", TitledBorder.*LEADING*, TitledBorder.*TOP*, null, null));  GroupLayout groupLayout = new GroupLayout(getContentPane());  groupLayout.setHorizontalGroup(  groupLayout.createParallelGroup(Alignment.*LEADING*)  .addGroup(groupLayout.createSequentialGroup()  .addGroup(groupLayout.createParallelGroup(Alignment.*LEADING*, false)  .addGroup(groupLayout.createSequentialGroup()  .addGap(48)  .addComponent(lblNewLabel, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addGap(26)  .addComponent(SearchTxt, GroupLayout.*PREFERRED\_SIZE*, 206, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*RELATED*, GroupLayout.*DEFAULT\_SIZE*, Short.*MAX\_VALUE*)  .addComponent(btnNewButton, GroupLayout.*PREFERRED\_SIZE*, 97, GroupLayout.*PREFERRED\_SIZE*))  .addGroup(groupLayout.createSequentialGroup()  .addGap(37)  .addGroup(groupLayout.createParallelGroup(Alignment.*LEADING*, false)  .addComponent(scrollPane)  .addComponent(panel, GroupLayout.*PREFERRED\_SIZE*, 758, Short.*MAX\_VALUE*))))  .addContainerGap(30, Short.*MAX\_VALUE*))  );  groupLayout.setVerticalGroup(  groupLayout.createParallelGroup(Alignment.*LEADING*)  .addGroup(groupLayout.createSequentialGroup()  .addGap(21)  .addGroup(groupLayout.createParallelGroup(Alignment.*BASELINE*)  .addComponent(lblNewLabel)  .addComponent(SearchTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(btnNewButton))  .addGap(18)  .addComponent(scrollPane, GroupLayout.*PREFERRED\_SIZE*, 243, GroupLayout.*PREFERRED\_SIZE*)  .addGap(18)  .addComponent(panel, GroupLayout.*PREFERRED\_SIZE*, 316, GroupLayout.*PREFERRED\_SIZE*)  .addContainerGap(69, Short.*MAX\_VALUE*))  );    JLabel lblNewLabel\_1 = new JLabel("工号:");    EidTxt = new JTextField();  EidTxt.setEditable(false);  EidTxt.setColumns(10);    JLabel lblNewLabel\_2 = new JLabel("姓名:");    EnameTxt = new JTextField();  EnameTxt.setColumns(10);    JLabel lblNewLabel\_3 = new JLabel("性别:");    EsexTxt = new JTextField();  EsexTxt.setColumns(10);    JLabel lblNewLabel\_4 = new JLabel("出生年份:");    yearTxt = new JTextField();  yearTxt.setColumns(10);    JLabel lblNewLabel\_5 = new JLabel("出生月份:");    monthTxt = new JTextField();  monthTxt.setColumns(10);    JLabel lblNewLabel\_6 = new JLabel("出生日:");    dayTxt = new JTextField();  dayTxt.setColumns(10);    JLabel lblNewLabel\_7 = new JLabel("家庭住址:");    addressTxt = new JTextField();  addressTxt.setColumns(10);    JLabel lblNewLabel\_8 = new JLabel("电话号码:");    TelTxt = new JTextField();  TelTxt.setColumns(10);    JLabel lblNewLabel\_9 = new JLabel("职位:");    PositionTxt = new JTextField();  PositionTxt.setColumns(10);    JLabel lblNewLabel\_10 = new JLabel("是否在职:");    StatusTxt = new JTextField();  StatusTxt.setColumns(10);    JButton Change = new JButton("修改");  Change.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  EmployeeUpdateActionEvent(e);  }  });    JButton delete = new JButton("删除");  delete.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  EmployeeDeleteActionEvent(e);  }  });    JLabel idTxt = new JLabel("身份证号：");    identity = new JTextField();  identity.setColumns(10);  GroupLayout gl\_panel = new GroupLayout(panel);  gl\_panel.setHorizontalGroup(  gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_panel.createSequentialGroup()  .addGap(19)  .addGroup(gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_panel.createSequentialGroup()  .addComponent(Change, GroupLayout.*PREFERRED\_SIZE*, 97, GroupLayout.*PREFERRED\_SIZE*)  .addGap(476)  .addComponent(delete, GroupLayout.*PREFERRED\_SIZE*, 97, GroupLayout.*PREFERRED\_SIZE*)  .addGap(103))  .addGroup(gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_panel.createSequentialGroup()  .addComponent(lblNewLabel\_9, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addGap(18)  .addComponent(PositionTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addGap(82)  .addComponent(lblNewLabel\_10, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*RELATED*)  .addComponent(StatusTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addGap(421))  .addGroup(gl\_panel.createSequentialGroup()  .addGroup(gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_panel.createSequentialGroup()  .addGroup(gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_panel.createSequentialGroup()  .addPreferredGap(ComponentPlacement.*RELATED*)  .addComponent(lblNewLabel\_4, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addGap(18)  .addComponent(yearTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addGap(31)  .addComponent(lblNewLabel\_5, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addGap(18)  .addComponent(monthTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addGroup(gl\_panel.createSequentialGroup()  .addComponent(lblNewLabel\_7, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addGap(18)  .addComponent(addressTxt, GroupLayout.*PREFERRED\_SIZE*, 185, GroupLayout.*PREFERRED\_SIZE*)))  .addGap(35)  .addGroup(gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addComponent(lblNewLabel\_8, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(idTxt, GroupLayout.*PREFERRED\_SIZE*, 81, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(lblNewLabel\_6, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)))  .addGroup(gl\_panel.createSequentialGroup()  .addGroup(gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_panel.createSequentialGroup()  .addComponent(lblNewLabel\_2, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addGap(18)  .addComponent(EnameTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addGap(47)  .addComponent(lblNewLabel\_3, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, Short.*MAX\_VALUE*)  .addPreferredGap(ComponentPlacement.*UNRELATED*)  .addComponent(EsexTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addGap(77))  .addGroup(gl\_panel.createSequentialGroup()  .addComponent(lblNewLabel\_1, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addGap(18)  .addComponent(EidTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)))  .addGap(63)))  .addPreferredGap(ComponentPlacement.*RELATED*)  .addGroup(gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addComponent(dayTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(identity, GroupLayout.*PREFERRED\_SIZE*, 184, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(TelTxt, GroupLayout.*PREFERRED\_SIZE*, 187, GroupLayout.*PREFERRED\_SIZE*))  .addGap(149)))))  );  gl\_panel.setVerticalGroup(  gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_panel.createSequentialGroup()  .addGap(32)  .addGroup(gl\_panel.createParallelGroup(Alignment.*BASELINE*)  .addComponent(lblNewLabel\_1)  .addComponent(EidTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addGap(30)  .addGroup(gl\_panel.createParallelGroup(Alignment.*BASELINE*)  .addComponent(lblNewLabel\_2)  .addComponent(EnameTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(lblNewLabel\_3)  .addComponent(EsexTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(idTxt)  .addComponent(identity, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addGap(18)  .addGroup(gl\_panel.createParallelGroup(Alignment.*BASELINE*)  .addComponent(yearTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(lblNewLabel\_5)  .addComponent(monthTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(lblNewLabel\_6)  .addComponent(dayTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(lblNewLabel\_4))  .addGap(33)  .addGroup(gl\_panel.createParallelGroup(Alignment.*BASELINE*)  .addComponent(lblNewLabel\_7)  .addComponent(addressTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(lblNewLabel\_8)  .addComponent(TelTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addGap(18)  .addGroup(gl\_panel.createParallelGroup(Alignment.*BASELINE*)  .addComponent(lblNewLabel\_9)  .addComponent(lblNewLabel\_10)  .addComponent(StatusTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(PositionTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addPreferredGap(ComponentPlacement.*RELATED*, 15, Short.*MAX\_VALUE*)  .addGroup(gl\_panel.createParallelGroup(Alignment.*BASELINE*)  .addComponent(Change)  .addComponent(delete))  .addGap(19))  );  panel.setLayout(gl\_panel);    employeeTable = new JTable();  employeeTable.addMouseListener(new MouseAdapter() {  @Override  public void mousePressed(MouseEvent e) {  EmployeeTableMousePressed(e);  }  });  employeeTable.setModel(new DefaultTableModel(  new Object[][] {  },  new String[] {  "\u7F16\u53F7", "\u59D3\u540D", "\u6027\u522B", "\u51FA\u751F\u5E74\u6708", "\u8EAB\u4EFD\u8BC1\u53F7", "\u5BB6\u5EAD\u4F4F\u5740", "\u7535\u8BDD\u53F7\u7801", "\u804C\u4F4D", "\u662F\u5426\u5728\u804C"  }  ) {  boolean[] columnEditables = new boolean[] {  false, false, false, true, true, true, true, true, false  };  public boolean isCellEditable(int row, int column) {  return columnEditables[column];  }  });  employeeTable.getColumnModel().getColumn(0).setPreferredWidth(43);  employeeTable.getColumnModel().getColumn(1).setPreferredWidth(47);  employeeTable.getColumnModel().getColumn(2).setPreferredWidth(45);  employeeTable.getColumnModel().getColumn(7).setPreferredWidth(43);  scrollPane.setViewportView(employeeTable);  getContentPane().setLayout(groupLayout);  this.fillTable(new Employee());  }    */\*\*  \* 员工删除事件;  \* @param e  \*/* private void EmployeeDeleteActionEvent(ActionEvent evt) {  *// TODO Auto-generated method stub* String id = this.EidTxt.getText();  if(id == null || id.length() == 0) {  JOptionPane.*showMessageDialog*(null, "请选择要删除的记录!");  return ;  }    int flag = JOptionPane.*showConfirmDialog*(null, "确定要该删除吗？");  if(flag == 0) {  try {  int isDelete = employeedao.delete(Integer.*parseInt*(id));  if(isDelete == 1) {  JOptionPane.*showMessageDialog*(null, "删除成功!");  resetValue();  this.fillTable(new Employee());  }  } catch(Exception e1) {  e1.printStackTrace();  }  }  }   */\*\*  \* 员工更新类别事件  \* @param evt  \*/* private void EmployeeUpdateActionEvent(ActionEvent evt) {  *// TODO Auto-generated method stub* String id = this.EidTxt.getText();  String name = this.EnameTxt.getText();  String sex = this.EsexTxt.getText();    String year = this.yearTxt.getText();  String month = this.monthTxt.getText();  String day = this.dayTxt.getText();  String birth = null;    if(isRight(Integer.*parseInt*(year),Integer.*parseInt*(month),Integer.*parseInt*(day))) {  birth = year + "-"+month+"-"+day;  } else {  JOptionPane.*showMessageDialog*(null, "输入的出生日期非法!");  this.yearTxt.setText("");  this.monthTxt.setText("");  this.dayTxt.setText("");  }    String indentityId = this.identity.getText();  String address = this.addressTxt.getText();  String tel = this.TelTxt.getText();  String position = this.PositionTxt.getText();  String status = this.StatusTxt.getText();    if(id == null || id.length() == 0) {  JOptionPane.*showMessageDialog*(null, "请选择要修改的记录!");  return ;  }    if(name == null || name.length() == 0) {  JOptionPane.*showMessageDialog*(null, "员工姓名不能为空!");  return ;  }    if(position == null || position.length() == 0) {  JOptionPane.*showMessageDialog*(null, "员工职位不能为空!");  return ;  }    if(status == null || status.length() == 0) {  JOptionPane.*showMessageDialog*(null, "员工在职状态不能为空!");  return ;  }      Employee e = new Employee(Integer.*parseInt*(id),name,sex,birth,indentityId,address,tel,position,status);  *//System.out.println(e);* try {  int flag = employeedao.update(e);  if(flag == 1) {  JOptionPane.*showMessageDialog*(null, "修改成功!");  resetValue();  this.fillTable(new Employee());  } else {  JOptionPane.*showMessageDialog*(null, "修改失败!");  }  } catch(Exception e1) {  e1.printStackTrace();  }  }   private void resetValue() {  *// TODO Auto-generated method stub* this.EidTxt.setText("");  this.EnameTxt.setText("");  this.EsexTxt.setText("");  this.monthTxt.setText("");  this.yearTxt.setText("");  this.dayTxt.setText("");  this.TelTxt.setText("");  this.addressTxt.setText("");  this.PositionTxt.setText("");  this.SearchTxt.setText("");  }   */\*\*  \* 员工类别搜索事件处理;  \* @param e  \*/* private void EmployeeSearchActionEvent(ActionEvent evt) {  *// TODO Auto-generated method stub* String name = this.SearchTxt.getText();  Employee e = new Employee();  e.setName(name);  this.fillTable(e);  }   */\*\*  \* 表格行点击事件处理;  \* @param e  \*/* private void EmployeeTableMousePressed(MouseEvent e) {  *// TODO Auto-generated method stub* int row = employeeTable.getSelectedRow();  this.EidTxt.setText((String)employeeTable.getValueAt(row, 0));  this.EnameTxt.setText((String)employeeTable.getValueAt(row, 1));  this.EsexTxt.setText((String)employeeTable.getValueAt(row, 2));    */\*\*  \* 这里尝试用了String自带的分割方法!  \*/* try {  String birth = (String)employeeTable.getValueAt(row, 3);  String[] strs = birth.split("-");  String year = strs[0];  String month = strs[1];  String day = strs[2];  this.yearTxt.setText(strs[0]);  this.monthTxt.setText(strs[1]);  this.dayTxt.setText(strs[2]);    } catch(Exception evt) {  evt.printStackTrace();  }  this.identity.setText((String)employeeTable.getValueAt(row, 4));  this.addressTxt.setText((String)employeeTable.getValueAt(row, 5));  this.TelTxt.setText((String)employeeTable.getValueAt(row, 6));  this.PositionTxt.setText((String)employeeTable.getValueAt(row, 7));  this.StatusTxt.setText((String)employeeTable.getValueAt(row, 8));    }   private void fillTable(Employee e) {  DefaultTableModel dtm = (DefaultTableModel) this.employeeTable.getModel();  dtm.setRowCount(0); *//设置为0行;* try {  ResultSet rs = employeedao.*list*(e);    while(rs.next()) {  Vector v = new Vector();  v.add(rs.getString("id"));  v.add(rs.getString("name"));  v.add(rs.getString("sex"));  v.add(rs.getString("birthday"));  v.add(rs.getString("identityID"));  v.add(rs.getString("address"));  v.add(rs.getString("tel"));  v.add(rs.getString("position"));  v.add(rs.getString("freeze"));  dtm.addRow(v);  }  } catch(Exception evt) {  evt.printStackTrace();  }    }  */\*\*  \* 判断输入的出生年月日是否合法  \* @param year  \* @param month  \* @param day  \* @return  \*/* private boolean isRight(int year, int month, int day) {  int [] days = new int [] {0,31,28,31,30,31,30,31,31,30,31,30,31};  int leap = 0;  if(month <= 0 || month > 12) return false;  if(day == 0) return false;    if(month != 2){  if(day > days[month]) return false;  } else {  if(year % 4 == 0 && year % 100 != 0 || year % 400 == 0)  leap = 1;  else leap = 0;     if(day > leap + 28) return false;  }   return true;  } }  员工功能实现：  package 功能类;  import java.sql.Connection; import java.sql.PreparedStatement; import java.sql.ResultSet;  import 实体类.Category; import 实体类.Employee; import 工具类.JDBCconnection;  public class EmployeeDAO {  public int add(Employee e) throws Exception{  Connection conn = (Connection)JDBCconnection.*getConn*();   String sql = "insert into employee(name,sex,birthday,identityID,address,tel,position,freeze) values(?,?,?,?,?,?,?,?)";  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sql);  pstmt.setString(1, e.getName());  pstmt.setString(2, e.getSex());  pstmt.setString(3, e.getBrithday());  pstmt.setString(4,e.getIdentityID());  pstmt.setString(5, e.getAddress());  pstmt.setString(6, e.getTel());  pstmt.setString(7, e.getPosition());  pstmt.setString(8, e.getFreeze());  return pstmt.executeUpdate();  }   */\*\*  \* 查询员工类别集合  \*/* public static ResultSet list(Employee e) throws Exception { *//这里可能不一样!* Connection conn = (Connection)JDBCconnection.*getConn*();  StringBuffer sb = new StringBuffer("select \* from employee");    if(e.getName() != null) {  if(e.getName().length() != 0)  sb.append(" and name like '%"+e.getName()+"%'");  }  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sb.toString().replaceFirst("and", "where"));  return pstmt.executeQuery();  }   */\*\*  \* 删除员工类别  \* @param id  \* @return  \* @throws Exception  \*/* public int delete(int id) throws Exception{  String sql = "delete from employee where id=?";  Connection conn = (Connection)JDBCconnection.*getConn*();  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sql);  pstmt.setInt(1, id);  return pstmt.executeUpdate();  }  */\*\*  \* 更新员工类别  \* @param  \* @return  \* @throws Exception  \*/* public int update(Employee e) throws Exception{  String sql = "update employee set name =?,sex =?,birthday=?,identityID=?,address=?,tel=?,position=?,freeze=? where id =?";  Connection conn = (Connection)JDBCconnection.*getConn*();  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sql);  *//System.out.println(e);* pstmt.setString(1, e.getName());  pstmt.setString(2, e.getSex());  pstmt.setString(3, e.getBrithday());  pstmt.setString(4, e.getIdentityID());  pstmt.setString(5, e.getAddress());  pstmt.setString(6, e.getTel());  pstmt.setString(7, e.getPosition());  pstmt.setString(8, e.getFreeze());  pstmt.setInt(9, e.getId());  return pstmt.executeUpdate();  } }  餐台管理界面：  package 功能类;  import java.sql.Connection; import java.sql.PreparedStatement; import java.sql.ResultSet;  import 实体类.Category; import 实体类.Employee; import 工具类.JDBCconnection;  public class EmployeeDAO {  public int add(Employee e) throws Exception{  Connection conn = (Connection)JDBCconnection.*getConn*();    String sql = "insert into employee(name,sex,birthday,identityID,address,tel,position,freeze) values(?,?,?,?,?,?,?,?)";  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sql);  pstmt.setString(1, e.getName());  pstmt.setString(2, e.getSex());  pstmt.setString(3, e.getBrithday());  pstmt.setString(4,e.getIdentityID());  pstmt.setString(5, e.getAddress());  pstmt.setString(6, e.getTel());  pstmt.setString(7, e.getPosition());  pstmt.setString(8, e.getFreeze());  return pstmt.executeUpdate();  }   */\*\*  \* 查询员工类别集合   \*/* public static ResultSet list(Employee e) throws Exception { *//这里可能不一样!* Connection conn = (Connection)JDBCconnection.*getConn*();  StringBuffer sb = new StringBuffer("select \* from employee");    if(e.getName() != null) {  if(e.getName().length() != 0)  sb.append(" and name like '%"+e.getName()+"%'");  }  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sb.toString().replaceFirst("and", "where"));  return pstmt.executeQuery();  }   */\*\*  \* 删除员工类别  \* @param id  \* @return  \* @throws Exception  \*/* public int delete(int id) throws Exception{  String sql = "delete from employee where id=?";  Connection conn = (Connection)JDBCconnection.*getConn*();  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sql);  pstmt.setInt(1, id);  return pstmt.executeUpdate();  }  */\*\*  \* 更新员工类别  \* @param  \* @return  \* @throws Exception  \*/* public int update(Employee e) throws Exception{  String sql = "update employee set name =?,sex =?,birthday=?,identityID=?,address=?,tel=?,position=?,freeze=? where id =?";  Connection conn = (Connection)JDBCconnection.*getConn*();  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sql);  *//System.out.println(e);* pstmt.setString(1, e.getName());  pstmt.setString(2, e.getSex());  pstmt.setString(3, e.getBrithday());  pstmt.setString(4, e.getIdentityID());  pstmt.setString(5, e.getAddress());  pstmt.setString(6, e.getTel());  pstmt.setString(7, e.getPosition());  pstmt.setString(8, e.getFreeze());  pstmt.setInt(9, e.getId());  return pstmt.executeUpdate();  } }  餐台功能实现：  package 功能类;  import java.sql.Connection; import java.sql.PreparedStatement; import java.sql.ResultSet;  import 实体类.Category; import 实体类.Desk; import 工具类.JDBCconnection;  public class DeskDAO {  */\*\*  \* 实现对桌台的添加  \* @param d  \* @return  \* @throws Exception  \*/* public int add(Desk d) throws Exception {  Connection conn = JDBCconnection.*getConn*();    String sql = "insert into desk(dno,seating,status) values(?,?,?)";  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sql);  pstmt.setString(1, d.getNo());  pstmt.setInt(2, d.getSeating());  pstmt.setString(3, d.getStatus());  return pstmt.executeUpdate();  }   */\*\*  \* 返回所有的桌台；  \* @param d  \* @return  \* @throws Exception  \*/* public static ResultSet list(Desk d) throws Exception{  Connection conn = JDBCconnection.*getConn*();  StringBuffer sb = new StringBuffer("select \* from desk");    if(d.getNo() != null) {  if(d.getNo().length() != 0) {  sb.append(" and dno like '%"+d.getNo()+"%'");  }  }   PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sb.toString().replaceFirst("and", "where"));  return pstmt.executeQuery();  }  */\*\*  \* 删除选定的台桌  \* @param id  \* @return  \* @throws Exception  \*/* public int delete(int id) throws Exception{  String sql = "delete from desk where id=?";  Connection conn = (Connection)JDBCconnection.*getConn*();  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sql);  pstmt.setInt(1, id);  return pstmt.executeUpdate();  }  */\*\*  \* 更改台桌的设置  \* @param d  \* @return  \* @throws Exception  \*/* public int update(Desk d) throws Exception{  String sql = "update desk set dno=?,seating=?,status=? where id =?";  Connection conn = (Connection)JDBCconnection.*getConn*();  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sql);  pstmt.setString(1, d.getNo());  pstmt.setInt(2, d.getSeating());  pstmt.setString(3, d.getStatus());  pstmt.setInt(4, d.getId());  return pstmt.executeUpdate();  }  }  订单界面：  package 界面类;  import java.awt.EventQueue; 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.ResultSet; import java.util.Vector;  import javax.swing.\*; import javax.swing.GroupLayout.Alignment; import javax.swing.JInternalFrame; import javax.swing.LayoutStyle.ComponentPlacement; import javax.swing.border.EmptyBorder; import javax.swing.table.DefaultTableModel;   import 功能类.DeskDAO; import 功能类.DishDAO; import 功能类.OrderDAO; import 功能类.OrderItemDAO; import 实体类.Desk; import 实体类.Dish; import 实体类.OrderItem; import 工具类.JDBCconnection;  public class OrderDishFrame extends JInternalFrame {   private JPanel contentPane;  private JComboBox deskNoJcb;  private DeskDAO deskdao = new DeskDAO();  private DishDAO dishdao = new DishDAO();  private OrderItemDAO orderitemdao = new OrderItemDAO();  private OrderDAO orderdao = new OrderDAO();    private JLabel lblNewLabel\_1;  private JScrollPane scrollPane;  private JTable ordertable;  private JButton btnNewButton;  private JLabel lblNewLabel\_2;  private JTextField nameTxt;  private JLabel lblNewLabel\_3;  private JTextField moneyTxt;  */\*\*  \* Launch the application.  \*/* public static void main(String[] args) {  EventQueue.*invokeLater*(new Runnable() {  public void run() {  try {  OrderDishFrame frame = new OrderDishFrame();  frame.setVisible(true);  } catch (Exception e) {  e.printStackTrace();  }  }  });  }  */\*\*  \* Create the frame.  \*/* public OrderDishFrame() {  setIconifiable(true);  setClosable(true);  *//setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);* setBounds(100, 100, 823, 537);  contentPane = new JPanel();  contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));  setContentPane(contentPane);    JLabel lblNewLabel = new JLabel("");    deskNoJcb = new JComboBox();    lblNewLabel\_1 = new JLabel("餐台号");    scrollPane = new JScrollPane();    btnNewButton = new JButton("点菜");  btnNewButton.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  orderdishActionPerformed(e);  }  });   lblNewLabel\_2 = new JLabel("菜品名");    nameTxt = new JTextField();  nameTxt.setColumns(10);    lblNewLabel\_3 = new JLabel("价格");    moneyTxt = new JTextField();  moneyTxt.setColumns(10);  GroupLayout gl\_contentPane = new GroupLayout(contentPane);  gl\_contentPane.setHorizontalGroup(  gl\_contentPane.createParallelGroup(Alignment.*TRAILING*)  .addGroup(gl\_contentPane.createSequentialGroup()  .addGroup(gl\_contentPane.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_contentPane.createSequentialGroup()  .addGap(10)  .addGroup(gl\_contentPane.createParallelGroup(Alignment.*LEADING*)  .addComponent(lblNewLabel, GroupLayout.*PREFERRED\_SIZE*, 44, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(lblNewLabel\_1, GroupLayout.*PREFERRED\_SIZE*, 44, GroupLayout.*PREFERRED\_SIZE*))  .addGap(4)  .addComponent(deskNoJcb, GroupLayout.*PREFERRED\_SIZE*, 69, GroupLayout.*PREFERRED\_SIZE*))  .addGroup(gl\_contentPane.createSequentialGroup()  .addGap(18)  .addComponent(scrollPane, GroupLayout.*DEFAULT\_SIZE*, 771, Short.*MAX\_VALUE*)))  .addContainerGap())  .addGroup(gl\_contentPane.createSequentialGroup()  .addGap(50)  .addComponent(lblNewLabel\_2, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*RELATED*)  .addComponent(nameTxt, GroupLayout.*PREFERRED\_SIZE*, 89, GroupLayout.*PREFERRED\_SIZE*)  .addGap(18)  .addComponent(lblNewLabel\_3, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*RELATED*)  .addComponent(moneyTxt, GroupLayout.*PREFERRED\_SIZE*, 96, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*RELATED*, 128, Short.*MAX\_VALUE*)  .addComponent(btnNewButton, GroupLayout.*PREFERRED\_SIZE*, 97, GroupLayout.*PREFERRED\_SIZE*)  .addGap(197))  );  gl\_contentPane.setVerticalGroup(  gl\_contentPane.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_contentPane.createSequentialGroup()  .addGap(10)  .addGroup(gl\_contentPane.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_contentPane.createSequentialGroup()  .addComponent(lblNewLabel)  .addGap(4)  .addComponent(lblNewLabel\_1))  .addComponent(deskNoJcb, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addGap(36)  .addComponent(scrollPane, GroupLayout.*PREFERRED\_SIZE*, 200, GroupLayout.*PREFERRED\_SIZE*)  .addGap(48)  .addGroup(gl\_contentPane.createParallelGroup(Alignment.*BASELINE*)  .addComponent(btnNewButton)  .addComponent(lblNewLabel\_2)  .addComponent(nameTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(lblNewLabel\_3)  .addComponent(moneyTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addContainerGap(150, Short.*MAX\_VALUE*))  );   ordertable = new JTable();  ordertable.addMouseListener(new MouseAdapter() {  @Override   *// 鼠标点击事件处理* public void mousePressed(MouseEvent e) {  orderdishtableMousePressed(e);  }  });  ordertable.setModel(new DefaultTableModel(  new Object[][] {  },  new String[] {  "\u5E8F\u53F7", "\u83DC\u54C1\u540D", "\u7C7B\u522B\u7F16\u53F7", "\u5355\u4F4D", "\u4EF7\u683C", "\u72B6\u6001"  }  ));  scrollPane.setViewportView(ordertable);  contentPane.setLayout(gl\_contentPane);  this.fillTable(new Dish());*//显示表格* this.filldeskno("search");  }   protected void orderdishtableMousePressed(MouseEvent evt) {  *// TODO Auto-generated method stub* int row = ordertable.getSelectedRow();  String id = String.*valueOf*(ordertable.getValueAt(row, 0));    String name = String.*valueOf*(ordertable.getValueAt(row, 1));   nameTxt.setText(name);    String categoryid = String.*valueOf*(ordertable.getValueAt(row, 2));      String unit = String.*valueOf*(ordertable.getValueAt(row,3));   String money = String.*valueOf*(ordertable.getValueAt(row, 4));  moneyTxt.setText(money);    String status = String.*valueOf*(ordertable.getValueAt(row, 5));   }   *//点菜事件* protected void orderdishActionPerformed(ActionEvent evt) {  *// TODO Auto-generated method stub* String s=(String)deskNoJcb.getSelectedItem();    String name = this.nameTxt.getText();    String money = this.moneyTxt.getText();   double dishmoney = Double.*valueOf*(money);    OrderItem orderitem = new OrderItem(s,name,dishmoney);    Connection con =null;  try {    int n = orderitemdao.add(orderitem);  System.*out*.println(n);  if(n == 1) {  JOptionPane.*showMessageDialog*(null, "添加成功");  resetValue();  }else {  JOptionPane.*showMessageDialog*(null, "添加失败");  }  }catch(Exception e){  e.printStackTrace();  JOptionPane.*showMessageDialog*(null, "添加失败2");  }finally {  try {  JDBCconnection.*close*(con, null, null);  } catch (Exception e) {  *// TODO Auto-generated catch block* e.printStackTrace();  }  }   orderdishminiFrame orderdishmini2 = new orderdishminiFrame(s);  new orderdishminiFrame(s).setVisible(true);    }   private void resetValue() {  *// TODO Auto-generated method stub* this.nameTxt.setText("");  this.moneyTxt.setText("");  }   private void filldeskno(String type) {  Connection con = null;  Desk desk = null;  try {  ResultSet rs = deskdao.*list*(new Desk());  if("search".equals(type)) {  desk = new Desk();   desk.setNo("请选择...");  }  while(rs.next()) {  desk = new Desk();   desk.setNo(rs.getString("dno"));  if("search".equals(type)){    this.deskNoJcb.addItem(rs.getString("dno"));  }  }  }catch(Exception e) {  e.printStackTrace();  }finally {  try {  JDBCconnection.*close*(con, null, null);  } catch (Exception e) {  *// TODO Auto-generated catch block* e.printStackTrace();  }  }  }  *//表格初始化* private void fillTable(Dish dish) {  DefaultTableModel dtm = (DefaultTableModel) ordertable.getModel();  dtm.setRowCount(0);  Connection con = null;  try {  ResultSet rs = dishdao.*list*(dish);  while(rs.next()) {  Vector v = new Vector();  v.add(rs.getInt("id"));  v.add(rs.getString("Dish\_name"));  v.add(rs.getInt("categoryid"));  v.add(rs.getString("unit"));  v.add(rs.getDouble("price"));  v.add(rs.getString("status"));   dtm.addRow(v);  }  }catch(Exception e) {  e.printStackTrace();  }finally {  try {  JDBCconnection.*close*(con, null, null);  } catch (Exception e) {  *// TODO Auto-generated catch block* e.printStackTrace();  }  }  } }  订单功能实现：  package 功能类;  import java.sql.Connection; import java.sql.PreparedStatement; import java.sql.ResultSet;  import 实体类.Order; import 工具类.JDBCconnection;  public class OrderDAO {  */\*\*  \* 增加订单  \* @param order  \* @return  \* @throws Exception  \*/* public int add(Order order) throws Exception{  String sql = "insert into ordertable values(?,?,?,?,?,?,?,?)";  Connection conn = JDBCconnection.*getConn*();  PreparedStatement pstmt = conn.prepareStatement(sql);  pstmt.setInt(1,order.getId());  pstmt.setString(2,order.getOrderNo());  pstmt.setString(3, order.getDeskId());  pstmt.setString(4,order.getCreatetime());  pstmt.setDouble(5,order.getMoney());  pstmt.setInt(6,order.getCustomerId());  pstmt.setString(7,order.getStatus());  pstmt.setInt(8,order.getNumber());   return pstmt.executeUpdate();   }  */\*\*  \* 查询订单  \* @param order  \* @return  \* @throws Exception  \*/* public ResultSet list(Order order)throws Exception {  Connection conn = JDBCconnection.*getConn*();  StringBuffer a = new StringBuffer("select \* from ordertable ");  if(order.getId() != 0) {  a.append(" and dno like '%"+order.getId()+"%'"); */\* 拼接字符串 a %部分查找 \*/* }   PreparedStatement pstmt=conn.prepareStatement(a.toString().replace("and", "where"));  return pstmt.executeQuery();  }  */\*\*  \* 删除订单  \* @param id  \* @return  \* @throws Exception  \*/* public int delete(int id)throws Exception {  Connection conn = JDBCconnection.*getConn*();  String sql = "delete from ordertable where id =?"; */\* 从ordertable中删除一个id的值 \*/* PreparedStatement pstmt = conn.prepareStatement(sql);  pstmt.setInt(1, id);  return pstmt.executeUpdate();  }  public int update(Order order)throws Exception{  Connection conn = JDBCconnection.*getConn*();  String sql = "update ordertable set id =?,orderNo=?,deskID=?,createtime=?,money=?,customerId=?,status=?,number=? where orderNo =?";  PreparedStatement pstmt = conn.prepareStatement(sql);  pstmt.setInt(1, order.getId());  pstmt.setString(2, order.getOrderNo());  pstmt.setString(3, order.getDeskId());  pstmt.setString(4, order.getCreatetime());  pstmt.setDouble(5, order.getMoney());  pstmt.setInt(6, order.getCustomerId());  pstmt.setString(7, order.getStatus());  pstmt.setInt(8, order.getNumber());  pstmt.setString(9, order.getOrderNo());  return pstmt.executeUpdate();  }  public int update2(Order order)throws Exception{  Connection conn = JDBCconnection.*getConn*();  String sql = "update ordertable set deskId=?,money=?,status=? where deskID =?";  PreparedStatement pstmt = conn.prepareStatement(sql);    pstmt.setString(1, order.getDeskId());  pstmt.setDouble(2, order.getMoney());  pstmt.setString(3, order.getStatus());  pstmt.setString(4, order.getDeskId());    return pstmt.executeUpdate();  }   public static void main(String[] args) {   }  }  付款界面：  package 界面类;  import java.awt.BorderLayout; import java.awt.EventQueue; 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 java.util.Vector;  import javax.swing.GroupLayout; import javax.swing.GroupLayout.Alignment; 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.JTextField; import javax.swing.LayoutStyle.ComponentPlacement; import javax.swing.border.EmptyBorder; import javax.swing.table.DefaultTableModel;  import 功能类.OrderDAO; import 实体类.Order; import 实体类.OrderItem; import 工具类.JDBCconnection;  public class orderdishminiFrame extends JFrame {   private JPanel contentPane;  private JTable orderdishtable;  private JTextField priceTxt;  private OrderDAO orderDao = new OrderDAO();   */\*\*  \* Launch the application.  \*/* public static void main(String[] args) {  EventQueue.*invokeLater*(new Runnable() {  public void run() {  try {  orderdishminiFrame frame = new orderdishminiFrame("");  frame.setVisible(true);  } catch (Exception e) {  e.printStackTrace();  }  }  });  }  */\*\*  \* Create the frame.  \*/* public orderdishminiFrame(final String deskno) {  *//setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);* setBounds(100, 100, 702, 476);  contentPane = new JPanel();  contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));  setContentPane(contentPane);  JScrollPane scrollPane = new JScrollPane();   JButton btnNewButton = new JButton("确认付款");  btnNewButton.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  MyOrderFrame of = new MyOrderFrame();  of.setVisible(true);  }  });    JButton btnNewButton\_1 = new JButton("删除");    JLabel lblNewLabel = new JLabel("总价格");    priceTxt = new JTextField();  priceTxt.setColumns(10);  GroupLayout gl\_contentPane = new GroupLayout(contentPane);  gl\_contentPane.setHorizontalGroup(  gl\_contentPane.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_contentPane.createSequentialGroup()  .addGap(74)  .addComponent(btnNewButton, GroupLayout.*PREFERRED\_SIZE*, 97, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*RELATED*, 275, Short.*MAX\_VALUE*)  .addComponent(btnNewButton\_1, GroupLayout.*PREFERRED\_SIZE*, 97, GroupLayout.*PREFERRED\_SIZE*)  .addGap(135))  .addGroup(gl\_contentPane.createSequentialGroup()  .addGap(25)  .addComponent(scrollPane, GroupLayout.*PREFERRED\_SIZE*, 629, GroupLayout.*PREFERRED\_SIZE*)  .addContainerGap(24, Short.*MAX\_VALUE*))  .addGroup(gl\_contentPane.createSequentialGroup()  .addGap(33)  .addComponent(lblNewLabel, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*RELATED*)  .addComponent(priceTxt, GroupLayout.*PREFERRED\_SIZE*, 156, GroupLayout.*PREFERRED\_SIZE*)  .addContainerGap(427, Short.*MAX\_VALUE*))  );  gl\_contentPane.setVerticalGroup(  gl\_contentPane.createParallelGroup(Alignment.*LEADING*)  .addGroup(Alignment.*TRAILING*, gl\_contentPane.createSequentialGroup()  .addGap(18)  .addComponent(scrollPane, GroupLayout.*PREFERRED\_SIZE*, 190, GroupLayout.*PREFERRED\_SIZE*)  .addGap(55)  .addGroup(gl\_contentPane.createParallelGroup(Alignment.*BASELINE*)  .addComponent(lblNewLabel)  .addComponent(priceTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addPreferredGap(ComponentPlacement.*RELATED*, 75, Short.*MAX\_VALUE*)  .addGroup(gl\_contentPane.createParallelGroup(Alignment.*BASELINE*)  .addComponent(btnNewButton)  .addComponent(btnNewButton\_1))  .addGap(53))  );   orderdishtable = new JTable();  orderdishtable.addMouseListener(new MouseAdapter() {  @Override   public void mousePressed(MouseEvent e) {  ordertableMousePressed(e);  }  });  orderdishtable.setModel(new DefaultTableModel(  new Object[][] {  },  new String[] {  "\u9910\u684C\u53F7", "\u83DC\u54C1\u540D", "\u4EF7\u683C"  }  ));  scrollPane.setViewportView(orderdishtable);  contentPane.setLayout(gl\_contentPane);    this.fillTable(new OrderItem(),deskno);  }  private void ordertableMousePressed(MouseEvent evt) {  *// TODO Auto-generated method stub* int row = orderdishtable.getSelectedRow();  String a = String.*valueOf*(orderdishtable.getValueAt(row, 0));   String b = String.*valueOf*(orderdishtable.getValueAt(row, 1));   String c = String.*valueOf*(orderdishtable.getValueAt(row, 2));   }  private void statuschangePerformed(ActionEvent evt) {  MyOrderFrame of = new MyOrderFrame();  of.setVisible(true);  }  *//初始化表格* private void fillTable(OrderItem orderdishmini,String deskno) {  DefaultTableModel dtm = (DefaultTableModel) orderdishtable.getModel();  dtm.setRowCount(0);*//清空* Connection con = (Connection) JDBCconnection.*getConn*();  ResultSet rs = null;   double sum=0.0;  PreparedStatement ps=null;    try {   ps=con.prepareStatement("select \* from orderitem ");    rs=ps.executeQuery();  while(rs.next()) {  if(rs.getString("deskno").equals(deskno)) {  Vector v = new Vector();  v.add(rs.getString("deskno"));  v.add(rs.getString("dishname"));  v.add(rs.getDouble("amount"));  sum+=rs.getDouble("amount");*//sum* dtm.addRow(v);  }  }  }catch(Exception e) {  e.printStackTrace();  }finally {  try {  JDBCconnection.*close*(con, ps, rs);  } catch (Exception e) {  *// TODO Auto-generated catch block* e.printStackTrace();  }  }  String ss=String.*valueOf*(sum);  priceTxt.setText(ss);  } }  订单明细功能：  package 功能类;  import java.sql.Connection; import java.sql.PreparedStatement; import java.sql.ResultSet; import java.sql.SQLException;  import 实体类.Order; import 实体类.OrderItem; import 工具类.JDBCconnection;  public class OrderItemDAO {  public ResultSet search(OrderItem orderitem) throws SQLException {  Connection conn = JDBCconnection.*getConn*();  String sql = "select \* from orderitem where deskno =?";  PreparedStatement pstmt = conn.prepareStatement(sql);    return pstmt.executeQuery();   }   public int add(OrderItem orderitem)throws Exception {  Connection conn = JDBCconnection.*getConn*();  String sql = "insert into orderitem values(?,?,?)";  PreparedStatement pstmt = conn.prepareStatement(sql);  pstmt.setString(1,orderitem.getDeskno());  pstmt.setString(2,orderitem.getDishname());  pstmt.setDouble(3,orderitem.getAmount());    return pstmt.executeUpdate();  }   */\*\*  \* 修改桌子  \* @param order  \* @return  \* @throws Exception  \*/* public int update(Order order)throws Exception{  Connection conn = JDBCconnection.*getConn*();  String sql = "update desk set money=?,status =? where dno =?";  PreparedStatement pstmt = conn.prepareStatement(sql);  pstmt.setDouble(1, order.getMoney());  pstmt.setString(2, order.getStatus());  pstmt.setString(3, order.getDeskId());   return pstmt.executeUpdate();  } }  餐品管理界面：  package 界面类;  import java.awt.EventQueue; import java.sql.Connection; import java.sql.ResultSet; import java.sql.ResultSetMetaData; import java.util.Vector;  import javax.swing.JInternalFrame; import javax.swing.GroupLayout; import javax.swing.GroupLayout.Alignment; import javax.swing.JScrollPane; import javax.swing.JTable; import javax.swing.LayoutStyle.ComponentPlacement; import javax.swing.table.DefaultTableModel;  import 功能类.CategoryDAO; import 功能类.DishDAO; import 实体类.Category; import 实体类.Dish; import 工具类.JDBCconnection;  import javax.swing.JPanel; import javax.swing.border.TitledBorder; import javax.swing.plaf.synth.SynthOptionPaneUI; import javax.swing.JLabel; import javax.swing.JOptionPane; import javax.swing.JTextField; import javax.swing.JComboBox; import javax.swing.JButton; import java.awt.event.ActionListener; import java.awt.event.ActionEvent; import javax.swing.JRadioButton; import javax.swing.ButtonGroup; import java.awt.event.MouseAdapter; import java.awt.event.MouseEvent;  public class DishManagerFrame extends JInternalFrame {  private JTable dishtable;  private JTextField s\_dishNameTxt;  private CategoryDAO categorydao = new CategoryDAO();  private DishDAO dishdao = new DishDAO();  private JComboBox dishJcb;  private JTextField idTxt;  private JTextField nameTxt;  private JTextField unitTxt;  private JTextField priceTxt;  private JRadioButton sellJrb = new JRadioButton();  private JRadioButton notsellJrb = new JRadioButton();  private JComboBox CategoryJcb;   private final ButtonGroup buttonGroup = new ButtonGroup();  private JTextField StatusTxt;  */\*\*  \* Launch the application.  \*/* public static void main(String[] args) {  EventQueue.*invokeLater*(new Runnable() {  public void run() {  try {  DishManagerFrame frame = new DishManagerFrame();  frame.setVisible(true);  } catch (Exception e) {  e.printStackTrace();  }  }  });  }   */\*\*  \* Create the frame.  \*/* public DishManagerFrame() {  setClosable(true);  setIconifiable(true);  setTitle("餐品管理");  setBounds(100, 100, 728, 694);    JScrollPane scrollPane = new JScrollPane();    JPanel panel = new JPanel();  panel.setBorder(new TitledBorder(null, "\u641C\u7D22\u6761\u4EF6", TitledBorder.*LEADING*, TitledBorder.*TOP*, null, null));    JPanel panel\_1 = new JPanel();  panel\_1.setBorder(new TitledBorder(null, "\u8868\u5355\u64CD\u4F5C", TitledBorder.*LEADING*, TitledBorder.*TOP*, null, null));  GroupLayout groupLayout = new GroupLayout(getContentPane());  groupLayout.setHorizontalGroup(  groupLayout.createParallelGroup(Alignment.*LEADING*)  .addGroup(Alignment.*TRAILING*, groupLayout.createSequentialGroup()  .addGap(24)  .addGroup(groupLayout.createParallelGroup(Alignment.*TRAILING*)  .addComponent(panel\_1, Alignment.*LEADING*, GroupLayout.*DEFAULT\_SIZE*, 644, Short.*MAX\_VALUE*)  .addComponent(panel, GroupLayout.*DEFAULT\_SIZE*, 644, Short.*MAX\_VALUE*)  .addComponent(scrollPane, GroupLayout.*DEFAULT\_SIZE*, 644, Short.*MAX\_VALUE*))  .addGap(48))  );  groupLayout.setVerticalGroup(  groupLayout.createParallelGroup(Alignment.*LEADING*)  .addGroup(groupLayout.createSequentialGroup()  .addContainerGap()  .addComponent(panel, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addGap(43)  .addComponent(scrollPane, GroupLayout.*PREFERRED\_SIZE*, 115, GroupLayout.*PREFERRED\_SIZE*)  .addGap(32)  .addComponent(panel\_1, GroupLayout.*PREFERRED\_SIZE*, 267, GroupLayout.*PREFERRED\_SIZE*)  .addContainerGap(115, Short.*MAX\_VALUE*))  );    JLabel lblNewLabel\_2 = new JLabel("编号:");    idTxt = new JTextField();  idTxt.setEditable(false);  idTxt.setColumns(10);    JLabel lblNewLabel\_3 = new JLabel("餐品名称:");    nameTxt = new JTextField();  nameTxt.setColumns(10);    JLabel lblNewLabel\_4 = new JLabel("餐品单位:");    unitTxt = new JTextField();  unitTxt.setColumns(10);    JLabel lblNewLabel\_5 = new JLabel("餐品价格:");    priceTxt = new JTextField();  priceTxt.setColumns(10);    JLabel lblNewLabel\_6 = new JLabel("餐品状态:");    JLabel lblNewLabel\_7 = new JLabel("餐品类别:");    CategoryJcb = new JComboBox();    JButton btnNewButton\_1 = new JButton("修改");  btnNewButton\_1.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent evt) {  DishUpdateActionPerformed(evt);  }  });    JButton btnNewButton\_2 = new JButton("删除");  btnNewButton\_2.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  DishDeleteActionPerformed(e);  }  });    StatusTxt = new JTextField();  StatusTxt.setColumns(10);  GroupLayout gl\_panel\_1 = new GroupLayout(panel\_1);  gl\_panel\_1.setHorizontalGroup(  gl\_panel\_1.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_panel\_1.createSequentialGroup()  .addContainerGap()  .addGroup(gl\_panel\_1.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_panel\_1.createSequentialGroup()  .addGroup(gl\_panel\_1.createParallelGroup(Alignment.*LEADING*, false)  .addGroup(gl\_panel\_1.createSequentialGroup()  .addComponent(lblNewLabel\_2, GroupLayout.*PREFERRED\_SIZE*, 38, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*RELATED*, GroupLayout.*DEFAULT\_SIZE*, Short.*MAX\_VALUE*)  .addComponent(idTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addGroup(gl\_panel\_1.createSequentialGroup()  .addComponent(lblNewLabel\_4, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*RELATED*)  .addComponent(unitTxt)))  .addGap(57)  .addGroup(gl\_panel\_1.createParallelGroup(Alignment.*LEADING*, false)  .addGroup(gl\_panel\_1.createSequentialGroup()  .addComponent(lblNewLabel\_3, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addGap(18)  .addComponent(nameTxt, GroupLayout.*PREFERRED\_SIZE*, 134, GroupLayout.*PREFERRED\_SIZE*))  .addGroup(gl\_panel\_1.createSequentialGroup()  .addComponent(lblNewLabel\_5, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addGap(18)  .addComponent(priceTxt))))  .addGroup(gl\_panel\_1.createSequentialGroup()  .addComponent(lblNewLabel\_6, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*RELATED*)  .addComponent(StatusTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addGap(186)  .addComponent(lblNewLabel\_7, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*UNRELATED*)  .addComponent(CategoryJcb, GroupLayout.*PREFERRED\_SIZE*, 114, GroupLayout.*PREFERRED\_SIZE*))  .addGroup(gl\_panel\_1.createSequentialGroup()  .addComponent(btnNewButton\_1, GroupLayout.*PREFERRED\_SIZE*, 97, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*RELATED*, 418, Short.*MAX\_VALUE*)  .addComponent(btnNewButton\_2, GroupLayout.*PREFERRED\_SIZE*, 97, GroupLayout.*PREFERRED\_SIZE*)))  .addContainerGap())  );  gl\_panel\_1.setVerticalGroup(  gl\_panel\_1.createParallelGroup(Alignment.*TRAILING*)  .addGroup(gl\_panel\_1.createSequentialGroup()  .addGap(35)  .addGroup(gl\_panel\_1.createParallelGroup(Alignment.*BASELINE*)  .addComponent(lblNewLabel\_2)  .addComponent(nameTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(lblNewLabel\_3)  .addComponent(idTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addGap(27)  .addGroup(gl\_panel\_1.createParallelGroup(Alignment.*BASELINE*)  .addComponent(lblNewLabel\_4)  .addComponent(unitTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(lblNewLabel\_5)  .addComponent(priceTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addGap(35)  .addGroup(gl\_panel\_1.createParallelGroup(Alignment.*BASELINE*)  .addComponent(lblNewLabel\_6)  .addComponent(lblNewLabel\_7)  .addComponent(CategoryJcb, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(StatusTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*))  .addPreferredGap(ComponentPlacement.*RELATED*, 30, Short.*MAX\_VALUE*)  .addComponent(btnNewButton\_1)  .addGap(29))  .addGroup(gl\_panel\_1.createSequentialGroup()  .addContainerGap(190, Short.*MAX\_VALUE*)  .addComponent(btnNewButton\_2)  .addGap(31))  );  panel\_1.setLayout(gl\_panel\_1);    JLabel lblNewLabel = new JLabel("餐品名称:");    s\_dishNameTxt = new JTextField();  s\_dishNameTxt.setColumns(10);    JLabel lblNewLabel\_1 = new JLabel("餐品类别:");    dishJcb = new JComboBox();    JButton btnNewButton = new JButton("查询");  btnNewButton.addActionListener(new ActionListener() {  public void actionPerformed(ActionEvent e) {  DishSearchActionPerformed(e);  }  });  GroupLayout gl\_panel = new GroupLayout(panel);  gl\_panel.setHorizontalGroup(  gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_panel.createSequentialGroup()  .addContainerGap()  .addComponent(lblNewLabel, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*RELATED*)  .addComponent(s\_dishNameTxt, GroupLayout.*PREFERRED\_SIZE*, 115, GroupLayout.*PREFERRED\_SIZE*)  .addGap(82)  .addComponent(lblNewLabel\_1, GroupLayout.*PREFERRED\_SIZE*, 58, GroupLayout.*PREFERRED\_SIZE*)  .addGap(18)  .addComponent(dishJcb, GroupLayout.*PREFERRED\_SIZE*, 80, GroupLayout.*PREFERRED\_SIZE*)  .addPreferredGap(ComponentPlacement.*RELATED*, 59, Short.*MAX\_VALUE*)  .addComponent(btnNewButton, GroupLayout.*PREFERRED\_SIZE*, 97, GroupLayout.*PREFERRED\_SIZE*)  .addGap(40))  );  gl\_panel.setVerticalGroup(  gl\_panel.createParallelGroup(Alignment.*LEADING*)  .addGroup(gl\_panel.createSequentialGroup()  .addGap(19)  .addGroup(gl\_panel.createParallelGroup(Alignment.*BASELINE*)  .addComponent(lblNewLabel)  .addComponent(s\_dishNameTxt, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(dishJcb, GroupLayout.*PREFERRED\_SIZE*, GroupLayout.*DEFAULT\_SIZE*, GroupLayout.*PREFERRED\_SIZE*)  .addComponent(btnNewButton)  .addComponent(lblNewLabel\_1))  .addContainerGap(18, Short.*MAX\_VALUE*))  );  panel.setLayout(gl\_panel);    dishtable = new JTable();  dishtable.addMouseListener(new MouseAdapter() {  @Override  public void mousePressed(MouseEvent me) {  DishTableMousePressed(me);  }  });  scrollPane.setViewportView(dishtable);  dishtable.setModel(new DefaultTableModel(  new Object[][] {  },  new String[] {  "\u7F16\u53F7", "\u9910\u54C1\u540D\u79F0", "\u9910\u54C1\u5355\u4F4D", "\u9910\u54C1\u4EF7\u683C", "\u9910\u54C1\u72B6\u6001", "\u9910\u54C1\u7C7B\u522B"  }  ));  getContentPane().setLayout(groupLayout);  this.fillCategory("search");  this.fillCategory("modify");  this.fillTable(new Dish());  }  */\*\*  \* 菜品删除事件处理;  \* @param e  \*/* private void DishDeleteActionPerformed(ActionEvent e) {  *// TODO Auto-generated method stub* String id = this.idTxt.getText();  if(id == null || id.length() == 0) {  JOptionPane.*showMessageDialog*(null, "请选择要删除的记录!");  return ;  }    int flag = JOptionPane.*showConfirmDialog*(null, "确定要删除该记录吗？");    if(flag == 0) {  try {  int isDelete = dishdao.delete(Integer.*parseInt*(id));  *//System.out.println(isDelete);* if(isDelete == 1) {  JOptionPane.*showMessageDialog*(null, "删除成功!");  resetValue();  this.fillTable(new Dish());  } else {  JOptionPane.*showMessageDialog*(null, "删除失败!");  }    } catch (NumberFormatException e1) {  *// TODO Auto-generated catch block* e1.printStackTrace();  } catch (Exception e1) {  *// TODO Auto-generated catch block* e1.printStackTrace();  }   }   }  */\*\*  \* 修改事件处理;  \* @param evt  \*/* private void DishUpdateActionPerformed(ActionEvent evt) {  *// TODO Auto-generated method stub* String id = this.idTxt.getText();  String name = this.nameTxt.getText();  String unit = this.unitTxt.getText();  String price = this.priceTxt.getText();  String status = this.StatusTxt.getText();    Category category = (Category)this.CategoryJcb.getSelectedItem();    int categoryId = category.getId();    if(id == null || id.length() == 0) {  JOptionPane.*showMessageDialog*(null, "请选择要修改的记录!");  return ;  }    if(name == null || name.length() == 0) {  JOptionPane.*showMessageDialog*(null, "修改后的菜品名字不能为空!");  return ;  }    if(status == null || status.length() == 0) {  JOptionPane.*showMessageDialog*(null, "菜品状态不能为空!");  return ;  }    Dish dish = new Dish(Integer.*parseInt*(id),name,categoryId, unit,Float.*parseFloat*(price), status);    try {  int flag = dishdao.update(dish);  if(flag == 1) {  JOptionPane.*showMessageDialog*(null, "菜品更新成功!");  resetValue();  this.fillTable(new Dish());  } else {  JOptionPane.*showMessageDialog*(null, "菜品更新失败");  }    } catch(Exception e2) {  e2.printStackTrace();  }   }   private void resetValue() {  *// TODO Auto-generated method stub* this.idTxt.setText("");  this.nameTxt.setText("");  this.priceTxt.setText("");  this.unitTxt.setText("");  this.StatusTxt.setText("");    if(this.CategoryJcb.getItemCount() > 0) {  this.CategoryJcb.setSelectedIndex(0);  }  }  */\*\*  \* 表格行点击事件处理;  \* @param me  \*/* private void DishTableMousePressed(MouseEvent me) {  *// TODO Auto-generated method stub* int row = this.dishtable.getSelectedRow();  this.idTxt.setText((String)dishtable.getValueAt(row, 0));  this.nameTxt.setText((String)dishtable.getValueAt(row, 1));  this.unitTxt.setText((String)dishtable.getValueAt(row, 2));  this.priceTxt.setText((String)dishtable.getValueAt(row, 3));  this.StatusTxt.setText((String)dishtable.getValueAt(row, 4));    String categoryName = (String)this.dishtable.getValueAt(row, 5);  int count = this.CategoryJcb.getItemCount();   for(int i = 0;i < count;i++) {  Category item = (Category)this.CategoryJcb.getItemAt(i);  if(item.getName().equals(categoryName)) {  this.CategoryJcb.setSelectedIndex(i);  }  }    }   */\*\*  \* 菜单查询事件方法处理  \* @param e  \*/* private void DishSearchActionPerformed(ActionEvent e) {  *// TODO Auto-generated method stub* String name = this.s\_dishNameTxt.getText();  Category c = (Category)this.dishJcb.getSelectedItem();  int categoryid = c.getId();    Dish dish = new Dish(name,categoryid);  this.fillTable(dish);  }   */\*\*  \* 初始化下拉框  \* @param type 下拉框类型  \*/* private void fillCategory(String type) {  Connection conn = JDBCconnection.*getConn*();  Category category = null;  try {  ResultSet rs = categorydao.*list*(new Category());  category = new Category();  category.setName("请选择");  category.setId(0);  if("search".equals(type)) {  this.dishJcb.addItem(category);  } else if("modify".equals(type)) {    }   while(rs.next()) {  category = new Category();  category.setName(rs.getString("name"));  category.setId(rs.getInt("id"));      if("search".equals(type)){  this.dishJcb.addItem(category);  } else if("modify".equals("modify")) {  this.CategoryJcb.addItem(category);  }  }    } catch(Exception e) {  e.printStackTrace();  }   }   */\*\*  \* 初始化表格数据;  \*/* private void fillTable(Dish dish) {  DefaultTableModel dtm = (DefaultTableModel)dishtable.getModel();  dtm.setRowCount(0); *//设置成0行 相当于清空表格* Connection conn = null;    try {  conn = (Connection )JDBCconnection.*getConn*();  try {  ResultSet rs = dishdao.*list*(dish);     while(rs.next()) {  Vector v = new Vector();  v.add(rs.getString("id"));  v.add(rs.getString("Dish\_name"));  v.add(rs.getString("unit"));  v.add(rs.getString("price"));  v.add(rs.getString("status"));  v.add(rs.getString("name"));  dtm.addRow(v);  }  } catch (Exception e) {  *// TODO Auto-generated catch block* e.printStackTrace();  }  } catch (Exception e) {  e.printStackTrace();  } finally {  JDBCconnection.*close*(conn, null, null);  }  } }  餐品功能实现：  package 功能类;  import java.sql.Connection; import java.sql.PreparedStatement; import java.sql.ResultSet; import java.sql.SQLException;  import 实体类.Customer; import 实体类.Dish; import 工具类.JDBCconnection;  public class DishDAO {  */\*\*  \* 餐品添加  \* @param d  \* @return  \* @throws Exception  \*/* public int add(Dish d) throws Exception{  String sql = "insert into dish(Dish\_name,categoryid,unit,price,status) values(?,?,?,?,?)";  Connection conn = JDBCconnection.*getConn*();   PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sql);  pstmt.setString(1,d.getName());  pstmt.setInt(2, d.getCategoryid());  pstmt.setString(3, d.getUnit());  pstmt.setDouble(4, d.getPrice());  pstmt.setString(5, d.getStatus());    return pstmt.executeUpdate();   }   */\*\*  \* 菜品信息查询  \* @param d  \* @return  \*/* public static ResultSet list(Dish d) throws Exception{ *//这里可能少了category* Connection conn = JDBCconnection.*getConn*();  StringBuffer sb = new StringBuffer("select \* from category c,dish d where c.id = d.categoryid");    if(d.getName() != null) {  if(d.getName().length() != 0)  sb.append(" and d.Dish\_name like '%"+d.getName()+"%'");  }    if(d.getCategoryid() != 0) { *//这里限定了在哪个菜单里面查询;* sb.append(" and d.categoryid="+d.getCategoryid());  }   *//System.out.println(d);* PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sb.toString());  return pstmt.executeQuery();   }    */\*\*  \* 菜单信息删除  \* @param id  \* @return  \* @throws Exception  \*/* public int delete(int id) throws Exception {  String sql = "delete from dish where id=?";  Connection conn = JDBCconnection.*getConn*();    PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sql);  pstmt.setInt(1, id);    return pstmt.executeUpdate();  }    */\*\*  \* 菜单信息修改  \* @param d  \* @return  \* @throws Exception  \*/* public int update(Dish d) throws Exception{  Connection conn = JDBCconnection.*getConn*();  String sql = "update dish set Dish\_name=?,categoryid=?,unit=?,price=?,status=? where id=? ";   PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sql);   pstmt.setString(1, d.getName());  pstmt.setInt(2, d.getCategoryid());  pstmt.setString(3, d.getUnit());  pstmt.setDouble(4, d.getPrice());  pstmt.setString(5, d.getStatus());  pstmt.setInt(6, d.getId());  return pstmt.executeUpdate();  }    */\*\*  \*指定菜单类别下是否存在菜品;  \* @throws SQLException   \*/* public boolean existDish(int id) throws SQLException {  String sql = "select \* from dish where categoryid=?";  Connection conn = (Connection)JDBCconnection.*getConn*();  PreparedStatement pstmt = (PreparedStatement) conn.prepareStatement(sql);  pstmt.setInt(1, id);  ResultSet rs = pstmt.executeQuery();  return rs.next();  }   }  **数据库：**  1686899412889  数据库代码：  /\*  SQLyog Professional v13.1.1 (32 bit)  MySQL - 8.0.26 : Database - canyin  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \*/  /\*!40101 SET NAMES utf8 \*/;  /\*!40101 SET SQL\_MODE=''\*/;  /\*!40014 SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0 \*/;  /\*!40014 SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0 \*/;  /\*!40101 SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='NO\_AUTO\_VALUE\_ON\_ZERO' \*/;  /\*!40111 SET @OLD\_SQL\_NOTES=@@SQL\_NOTES, SQL\_NOTES=0 \*/;  CREATE DATABASE /\*!32312 IF NOT EXISTS\*/`canyin` /\*!40100 DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4\_0900\_ai\_ci \*/ /\*!80016 DEFAULT ENCRYPTION='N' \*/;  USE `canyin`;  /\*Table structure for table `category` \*/  DROP TABLE IF EXISTS `category`;  CREATE TABLE `category` (  `id` int NOT NULL AUTO\_INCREMENT,  `name` varchar(20) NOT NULL,  `describ` varchar(20) DEFAULT NULL,  PRIMARY KEY (`id`)  ) ENGINE=InnoDB AUTO\_INCREMENT=7 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;  /\*Data for the table `category` \*/  insert into `category`(`id`,`name`,`describ`) values  (1,'冷盘助兴','凉菜冷盘'),  (2,'能量荤配','肉类晕菜'),  (3,'田园时蔬','菜类'),  (4,'粗食辅助','主食面类');  /\*Table structure for table `customer` \*/  DROP TABLE IF EXISTS `customer`;  CREATE TABLE `customer` (  `id` int NOT NULL AUTO\_INCREMENT,  `name` varchar(20) NOT NULL,  `sex` varchar(4) DEFAULT NULL,  `tel` varchar(20) DEFAULT NULL,  `cardID` varchar(10) NOT NULL,  `company` varchar(255) DEFAULT NULL,  PRIMARY KEY (`id`)  ) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;  /\*Data for the table `customer` \*/  insert into `customer`(`id`,`name`,`sex`,`tel`,`cardID`,`company`) values  (1,'张小红','女','1111111111111','01111',NULL),  (2,'李晓强','男','22222222222','01112',NULL),  (3,'孙笑笑','女','3333333333','01113',NULL),  (4,'赵明','男','44444444444','01114',NULL);  /\*Table structure for table `desk` \*/  DROP TABLE IF EXISTS `desk`;  CREATE TABLE `desk` (  `id` int NOT NULL AUTO\_INCREMENT,  `dno` varchar(8) NOT NULL,  `seating` int NOT NULL,  `status` varchar(10) NOT NULL,  PRIMARY KEY (`id`,`dno`)  ) ENGINE=InnoDB AUTO\_INCREMENT=9 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;  /\*Data for the table `desk` \*/  insert into `desk`(`id`,`dno`,`seating`,`status`) values  (1,'111',2,'已预定'),  (2,'222',4,'就餐中'),  (3,'333',6,'就餐中'),  (4,'666',8,'已预定'),  (5,'888',12,'已预定'),  (6,'555',2,'已结账'),  (7,'777',4,'已结账');  /\*Table structure for table `dish` \*/  DROP TABLE IF EXISTS `dish`;  CREATE TABLE `dish` (  `id` int NOT NULL AUTO\_INCREMENT,  `Dish\_name` varchar(255) DEFAULT NULL,  `categoryid` int DEFAULT NULL,  `unit` varchar(255) DEFAULT NULL,  `price` double DEFAULT NULL,  `status` varchar(255) DEFAULT NULL,  PRIMARY KEY (`id`)  ) ENGINE=InnoDB AUTO\_INCREMENT=4 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;  /\*Data for the table `dish` \*/  insert into `dish`(`id`,`Dish\_name`,`categoryid`,`unit`,`price`,`status`) values  (2,'土豆丝',1,'1',123,'出售'),  (3,'番茄炒蛋',3,'22',10,'出售');  /\*Table structure for table `employee` \*/  DROP TABLE IF EXISTS `employee`;  CREATE TABLE `employee` (  `id` int NOT NULL AUTO\_INCREMENT,  `name` varchar(20) NOT NULL,  `sex` varchar(2) DEFAULT NULL,  `birthday` datetime(6) DEFAULT NULL,  `identityID` varchar(18) DEFAULT NULL,  `address` varchar(40) DEFAULT NULL,  `tel` varchar(11) DEFAULT NULL,  `position` varchar(4) NOT NULL,  `freeze` varchar(255) DEFAULT NULL,  PRIMARY KEY (`id`)  ) ENGINE=InnoDB AUTO\_INCREMENT=2 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;  /\*Data for the table `employee` \*/  /\*Table structure for table `orderitem` \*/  DROP TABLE IF EXISTS `orderitem`;  CREATE TABLE `orderitem` (  `deskno` varchar(255) NOT NULL,  `dishname` varchar(255) NOT NULL,  `amount` double NOT NULL  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;  /\*Data for the table `orderitem` \*/  insert into `orderitem`(`deskno`,`dishname`,`amount`) values  ('333','土豆丝',123),  ('111','土豆丝',123),  /\*Table structure for table `ordertable` \*/  DROP TABLE IF EXISTS `ordertable`;  CREATE TABLE `ordertable` (  `id` int NOT NULL,  `orderNo` varchar(20) NOT NULL,  `deskID` int DEFAULT NULL,  `createtime` date DEFAULT NULL,  `money` double DEFAULT NULL,  `customerId` bigint DEFAULT NULL,  `status` varchar(4) DEFAULT NULL,  `number` int DEFAULT NULL,  PRIMARY KEY (`id`,`orderNo`)  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;  /\*Data for the table `ordertable` \*/  /\*Table structure for table `user` \*/  DROP TABLE IF EXISTS `user`;  CREATE TABLE `user` (  `id` int NOT NULL,  `username` varchar(20) NOT NULL,  `password` varchar(20) NOT NULL,  PRIMARY KEY (`id`)  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;  /\*Data for the table `user` \*/  insert into `user`(`id`,`username`,`password`) values  (1,'wyq','888888'),  (2,'1','1');  /\*!40101 SET SQL\_MODE=@OLD\_SQL\_MODE \*/;  /\*!40014 SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS \*/;  /\*!40014 SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS \*/;  /\*!40111 SET SQL\_NOTES=@OLD\_SQL\_NOTES \*/;  **系统测试：**  1686899598335  **主界面：**  IMG_256  增加员工：  [SOX1CVIMGD~8)IK)98]OYS  修改员工：  A_Q}F2@AM)HQX{YED~QV7RI  删除员工：  X44_644Q{5XE53_{19Z`VBG  查询员工：  L07WB2YL6SDBCPXBXROHLEF  增加客户：  MW0PFO6[8)V{J($`Z$4IWLR  修改客户：  FCP[LL{7@K%CTZ8F(6TP(P5  删除客户：  1E2EV678RZMVCEGEZ%(4LON  查询客户：  0M[T[3)XJW`FL1}Q5(VHI1Y  增加餐台：  ZQ`IYMG}KZTNSCK~G@~L1P9  修改餐台：  7I`%)(_ETS0[RV@SNL25AIC  删除餐台：  O3Y2J1UCUS88M`C7}%TV{65  查询餐台：  0EHP3L0M$}Y]F4~]EGMAE08  DGEBDY68}`(FQ(QJ53U43}P  订单增加：  DGEBDY68}`(FQ(QJ53U43}P  订单删除：  7M%@O`]79YCV7HXQLD]SGAP  菜品添加：  UQAO$$YJCA7~OT36)F]NYPL  菜品修改：  ])6]`Y9D]6TG{3(JEKQUCZD  菜品删除：  R23Z5AY@S@XBXNNXNVF4UKN  菜品查询：  F]0E}8TJDJ_ZB{6{~4@BF4B  菜品类别添加：  4@F9I8$U1BGRL5JDU)_4H@Q  菜品类别修改：  [0W~]W3_6OE2RG[D0JY1HSM  菜品类别查询：  BM`KZ)]YC_%846A7JLW{FDQ  菜品类别删除：  _LQ~{6`4RB[AOY]B8ZWO[_Y | | | | | |
| **实训成果**：  参见基于C/S架构的餐饮管理系统设计说明书。 | | | | | |
| **实训总结**：（包括心得体会、存在的问题和改进方向。）  本栏正文排版要求：宋体5号，行距18磅，段首缩进2字符。  本栏可跨多页。  在本次实训当中我要做的是一个餐饮管理系统，首先对该系统的数据进行了分析，然后在数据库当中构建了管理员表、员工表、客户表、餐桌表、菜品表、订单表。其次，对该些表进行关系模型的建立找出他们的关系及其建立的顺序。而后就是在Java当中写入代码，连接数据库进行数据的存储，并且实现图形化界面建立好基本的系统框架之后，开始细分每个模块的功能，并对其进行图形化界面。在写入过程当中我熟悉的运用了实现接口、分块写功能、科学命名法等将该系统的代码写的清晰透彻方便后期优化找bug。  写完代码之后，开始了运行调试，对每一个功能进行调试，考虑每一个情况，找出bug并进行优化完善。  在本此实训当中我运用到了上个学期和本学期学到的Java知识和数据库知识，让我对接口、Java的封装、JDBC连接数据库、Java泛型、Java序列化机制、Java多线程机制、数据库建表、数据库分析数据模型、数据库建立存储过程和触发器等知识模块有了更深的认识和了解。并且能更加熟练地运用并实现图形化界面，提高了命名规范、分块实现的意识，和对Java面对对象程序设计的理解。 | | | | | |

基于C/S架构的餐饮管理系统

设计说明书

作者：王雅晴B 20210304131

计算机科学与工程学院

2023年 06 月 15日

**摘要**

摘要应概括反映出课程设计的内容、方法、成果和结论。结果和结论性字句是摘要的重点，在文字论述上要多些，以加深读者的印象；用精炼、概括的语言来表达，每项内容不宜展开论证或说明，要客观陈述，不宜加主观评价；要独立成文，选词用语要避免与全文尤其是前言和结论部分雷同；摘要中不宜使用公式、图表，不标注引用文献编号。避免将摘要写成目录式的内容介绍。

关键词是供检索用的主题词条，应采用能覆盖全文主要内容的通用技术词条(参照教材附录A的名词索引或其它技术术语标准)。关键词一般列3~5个，按词条的外延层次排列（外延大的在前面）。

**关键词：**java ，C/S架构，餐饮管理系统

目录

[1 引言 1](#_Toc11882504)

[1.1 编写目的 1](#_Toc11882505)

[1.2 参考资料 1](#_Toc11882506)

[2 需求规约 2](#_Toc11882507)

[2.1 功能需求 2](#_Toc11882508)

[2.2 界面需求 3](#_Toc11882509)

[2.3 数据需求 3](#_Toc11882510)

[3 系统设计 4](#_Toc11882511)

[3.1 运行环境 4](#_Toc11882512)

[3.2 系统静态结构设计 4](#_Toc11882513)

[3.3 人机接口设计 5](#_Toc11882514)

[3.4 XXX模块设计说明 7](#_Toc11882515)

[3.5 XXX模块设计说明 7](#_Toc11882516)

[4 数据库设计 8](#_Toc11882517)

[4.1 数据库环境说明 8](#_Toc11882518)

[4.2 数据库的命名规则 8](#_Toc11882519)

[4.3 逻辑结构设计 8](#_Toc11882520)

[4.4 物理结构设计 9](#_Toc11882521)

[4.5 安全性设计 10](#_Toc11882522)

[5 测试用例设计 12](#_Toc11882523)

[附录 14](#_Toc11882524)

[附录1 程序运行结果 14](#_Toc11882525)

[附录2 程序源代码 15](#_Toc11882526)

# 1 引言

## 1.1 编写目的

本文档是“王雅晴”的软件设计说明书。编写目的是：明确软件需求、概要设计、详细设计和功能测试用例，为软件后续开发和维护提供指导。本文档的读者对象是需求分析人员、系统设计人员、系统开发人员、系统测试人员。

## 1.2 参考资料

列出有关资料的名称、作者、文件编号或版本等。

[1] 文东. 数据库系统开发[M]. 北京: 北京科海电子出版社，2011．

[2]

参考资料格式要求：

[1] 袁庆龙，候文义．Ni-P合金镀层组织形貌及显微硬度研究［Ｊ］．太原理工大学学报，2001，32(1)：51-53.（连续出版物：［序号］ 主要责任者．文献题名［J］．刊名，出版年份，卷号(期号)：起止页码）

[2] 刘国钧，王连成．图书馆史研究［Ｍ］．北京：高等教育出版社，1979．（专著：［序号］ 主要责任者．文献题名［Ｍ］．出版地：出版者，出版年．）

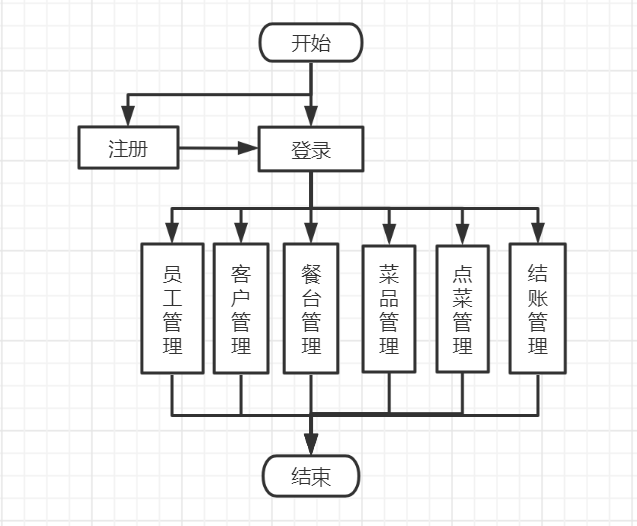
[3] 孙品一．高校学报编辑工作现代化特征［Ｃ］．中国高等学校自然科学学报研究会．科技编辑学论文集(2)．北京：北京师范大学出版社，1998：10-22．（论文集：[序号］ 主要责任者．文献题名[Ｃ]∥主编．论文集名．出版地：出版者，出版年：起止页码．）

[4] 张和生．地质力学系统理论［Ｄ］．太原：太原理工大学，1998．(学位论文:［序号］ 主要责任．文献题名［Ｄ］．保存地：保存单位，年份．)

[5] 冯西桥．核反应堆压力容器的LBB分析［Ｒ］.北京：清华大学核能技术设计研究院，1997．(报告:［序号］ 主要责任．文献题名［Ｒ］．报告地：报告会主办单位，年份．)

# 2 需求规约

## 2.1 功能需求



（**图名： 仿宋，小四。**

**图内文字：宋体，五号。）**

图2.1 图书管理系统功能模块图

2.1.1 登录用例说明

用例名称：管理员登陆。

用例描述：登录已有管理员账号，若无管理员账号可输入老板密匙进行管理员账号密码的录入，管理员登陆后可进行餐饮管理系统的操作。

前置条件：登录已有管理员账号，若无可输入老板密匙注册。

后置条件：管理员取得使用餐饮管理系统的权限。

活动步骤：

1. 管理员输入账号密码。
2. 系统检查输入信息的正确性。
3. 系统检查数据库中是否匹配该账号密码。
4. 系统显示登录结果是否成功。
5. 管理员获得使用系统权限。

扩展点：

1. 若无现有管理员账号可注册管理员账号。

异常处理：

1. 如果用户输入为空，则用例终止。
2. 如果用户输入的账号不存在，则用例终止。
3. 如果用户输入密码与账号不匹配。则用例终止。

2.1.2 员工管理用例说明

用例名称：员工管理。

用例描述：在该模块可进行员工信息的增删改查和显示。

前置条件：登录管理员账号后进入相应模块输入员工信息进行增删改查和显示。

后置条件：员工信息进行更新。

活动步骤：

1. 输入员工相应信息进行增加。
2. 系统在数据库中增加该员工信息。
3. 输入员工编号进行删除。
4. 系统查询该编号是否存在，若不存在则显示该编号不存在，若存在则删除该员工信息。
5. 输入员工编号进行修改。
6. 系统查询该编号是否存在，若不存在则显示该编号不存在，若存在则重新录入该员工信息。
7. 输入员工编号进行查询员工信息。
8. 系统查询该编号是否存在，若不存在则显示该编号不存在，若存在则显示该员工信息。
9. 显示所有员工信息。
10. 系统查询所有员工信息进行显示。

扩展点：

1. 增加了显示所有员工信息操作。

异常处理：

1. 如果用户输入的信息为空，则用例终止。
2. 如果输入的员工电话号码不为11位数字，身份证号码不为18位数字，则用例终止。
3. 如果员工输入的员工编号不存在，则用例终止。

2.1.3 客户管理用例说明

用例名称：客户管理。

用例描述：在该模块可进行客户信息的增删改查和显示。

前置条件：登录管理员账号后进入相应模块输入客户信息进行增删改查和显示及余额充值等操作。

后置条件：客户信息进行更新。

活动步骤：

1. 输入客户相应信息进行增加。
2. 系统在数据库当中进行员工信息的增加。
3. 输入客户电话号码进行删除。
4. 系统查询该客户电话号码是否存在，若不存在则显示该电话号码不存在，若存在则删除该客户信息。
5. 输入客户电话号码和修改后的信息进行客户信息修改。
6. 系统查询该客户电话号码是否存在，若不存在则显示该客户电话号码不存在，若存在则重新录入该员工信息。
7. 输入客户电话号码进行客户信息查询。
8. 系统查询该客户电话号码是否存在，若不存在则显示该客户电话号码不存在，若存在则显示该客户信息。
9. 进行客户余额的充值。
10. 系统在数据库中进行用户信息的更新。
11. 显示所有客户信息。
12. 系统查询所有客户信息进行显示。

扩展点：

1. 增加了显示所有客户信息操作，充值客户余额的操作。

异常处理：

1.如果客户输入的信息为空，则用例终止。

2.如果输入的客户电话号码不为11位数字，则用例终止。

3.如果客户输入的客户编号不存在，则用例终止。

2.1.4 餐台管理用例说明

用例名称：餐台管理。

用例描述：在该模块可进行餐台信息的增删改和显示。

前置条件：登录管理员账号后进入相应模块输入餐台信息进行增删改和显示。

后置条件：餐台信息进行更新。

活动步骤：

1. 输入餐台相应信息进行增加。
2. 系统在数据库中进行餐台信息的增加。
3. 输入餐台编号进行删除。
4. 系统查询是否存在该餐桌编号，若存在则删除，若不存在则显示该餐桌编号不存在。
5. 输入餐台编号和修改后的信息进行餐台信息修改。

系统查询是否存在该餐桌编号，若存在则重新录入该餐台信息，若不存在则显示该餐桌编号不存在

1. 显示所有餐台信息。
2. 系统查询所有餐台信息进行显示。

扩展点：

1. 增加了显示所有餐台信息操作。

异常处理：

1.如果餐台输入的信息为空，则用例终止。

2.如果餐台输入的餐台编号不存在，则用例终止。

2.1.5 菜品管理用例说明

用例名称：菜品管理。

用例描述：在该模块可进行菜品信息的增删改查和显示及菜品分类。

前置条件：登录管理员账号后进入相应模块输入菜品信息进行增删改查和显示及菜品分类等操作。

后置条件：菜品信息进行更新。

活动步骤：

1. 输入菜品相应信息进行增加。
2. 系统进行菜品的信息录入。
3. 输入菜品编号进行删除。
4. 系统查询该菜品编号是否存在，若无该菜品编号则显示不存在该编号，若存在则重新录入该菜品信息。
5. 输入菜品编号进行菜品信息查询。
6. 系统系统查询该菜品编号是否存在，若无该菜品编号则显示不存在该编号，若存在则显示该菜品信息。
7. 进行菜品的分类管理。
8. 点击不同菜品类别系统查询该类别菜品进行显示。
9. 显示所有菜品信息。
10. 系统查询所有菜品信息进行显示。

扩展点：

1. 增加了显示所有菜品信息操作，菜品管理的操作。

异常处理：

1.如果输入的菜品信息为空，则用例终止。

2.如果输入的菜品编号不存在，则用例终止。

2.1.6 点菜管理用例说明

用例名称：点菜管理。

用例描述：在该模块可进行点菜操作。

前置条件：登录管理员账号后进入点菜模块选择菜品类别，菜品信息，菜品数量，桌号，客户进行点菜下单，还可以查看菜品推荐进行点单。

后置条件：将点菜的菜单下单至后台。

活动步骤：

1. 查看菜品推荐。
2. 系统将菜品推荐界面显示。
3. 选择菜品类别进行菜品选择。
4. 系统由该类别查询该类别所有菜品进行显示在下拉选择当中。
5. 选择桌号，客户姓名进行下单。
6. 系统确定信息输入后台订单当中。

扩展点：

1. 增加了菜品推荐的操作，当桌号为忙碌时不可在该桌号点单。

异常处理：

1.如果还未点菜就下单，则用例终止。

2.如果无点菜进行删除操作，则用例终止。

2.1.7 结账管理用例说明

用例名称：结账管理。

用例描述：在该模块可进行结账操作。

前置条件：点菜后将菜单信息传至后台可进行结账操作。

后置条件：将点菜的菜单进行结算。

活动步骤：

1.选择结账桌号。

2.点击结账进行账单结算。

3.结账后，系统请求该客户信息查询该客户余额是否能够结算。

4.若余额不足则显示余额不足，无法结账，需先充值余额。

5.若余额足则结账成功。

扩展点：

1. 增加了流水查询的操作，将下单的桌号传至后台进行下拉选择。

异常处理：

1.如果余额不足，则用例终止。

## 2.2 界面需求

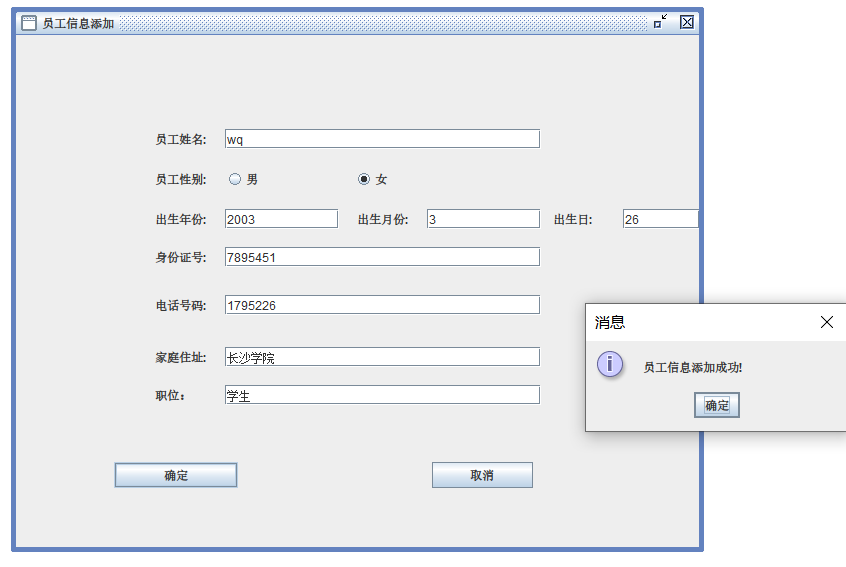
**系统测试：**



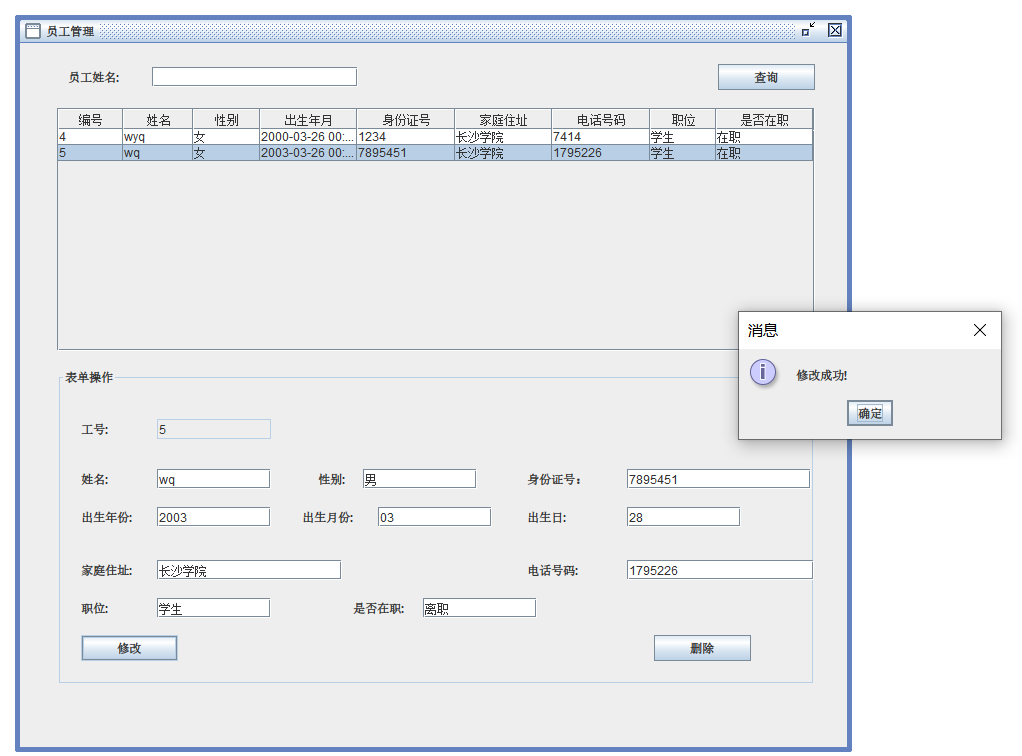
**主界面：**

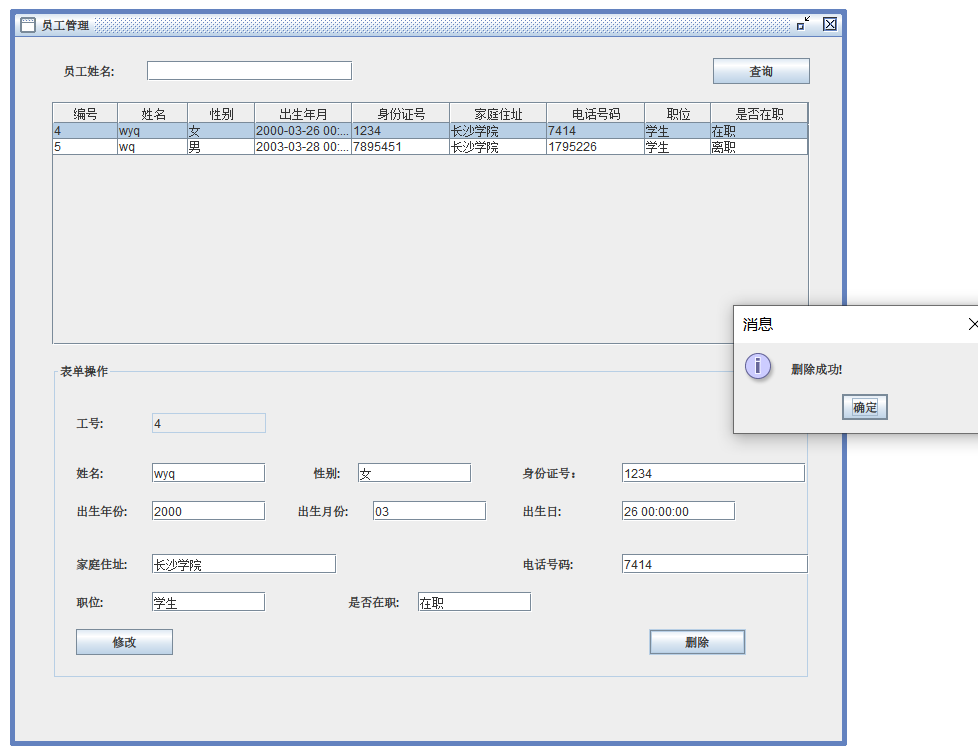


增加员工：



修改员工：

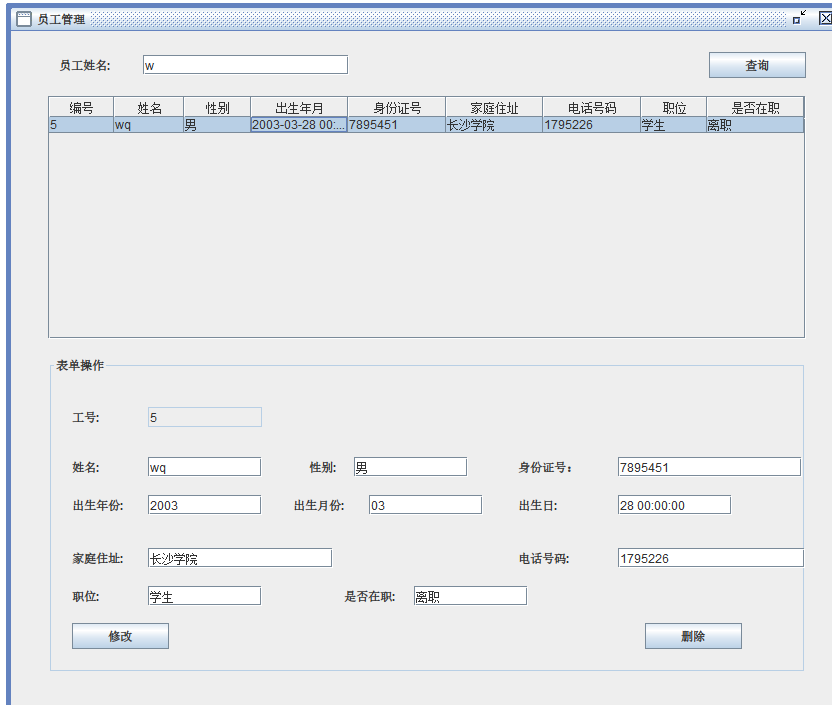


删除员工：

查询员工：

查询员工

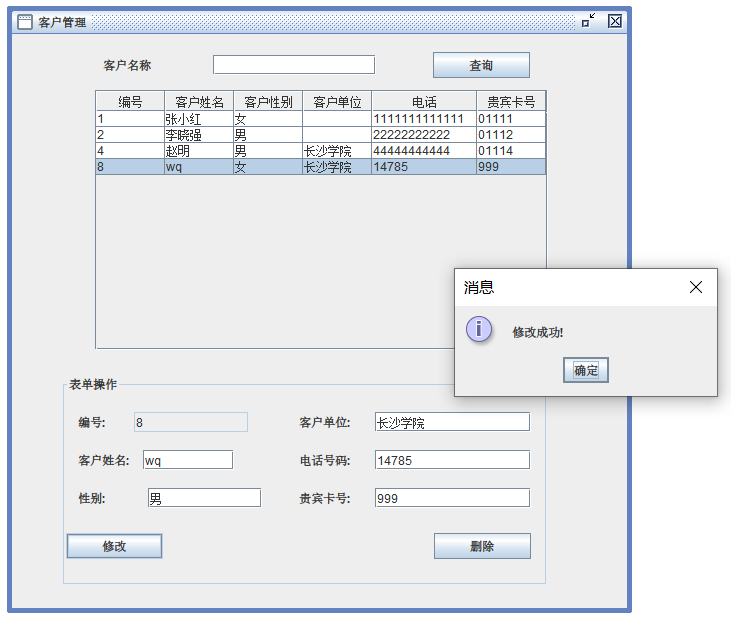
查询员工：



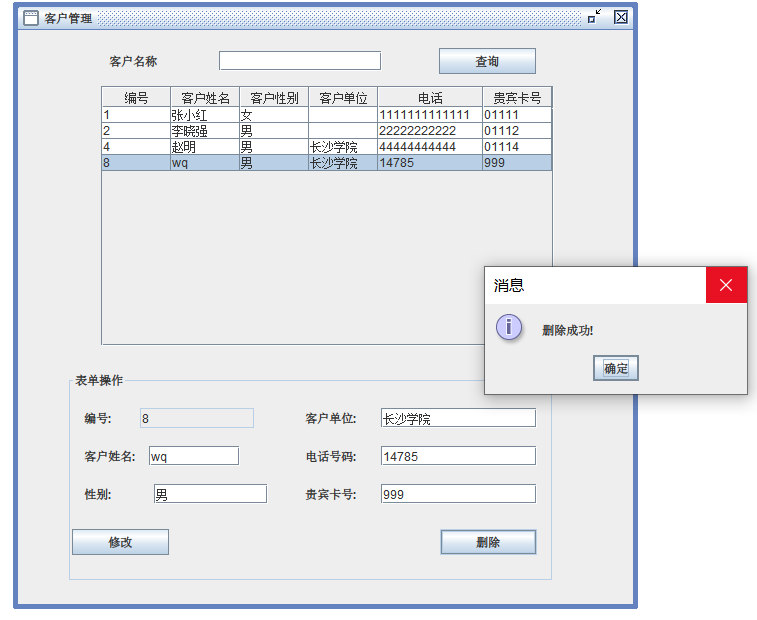
增加客户：



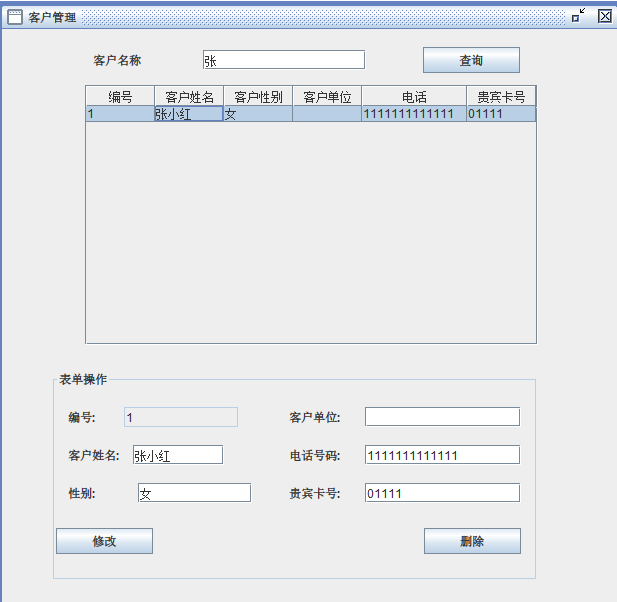
修改客户：



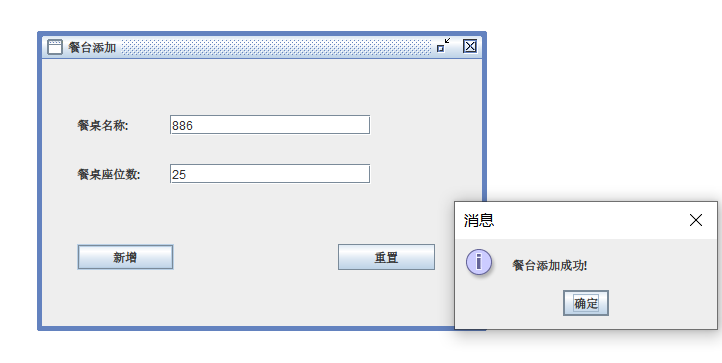
删除客户：



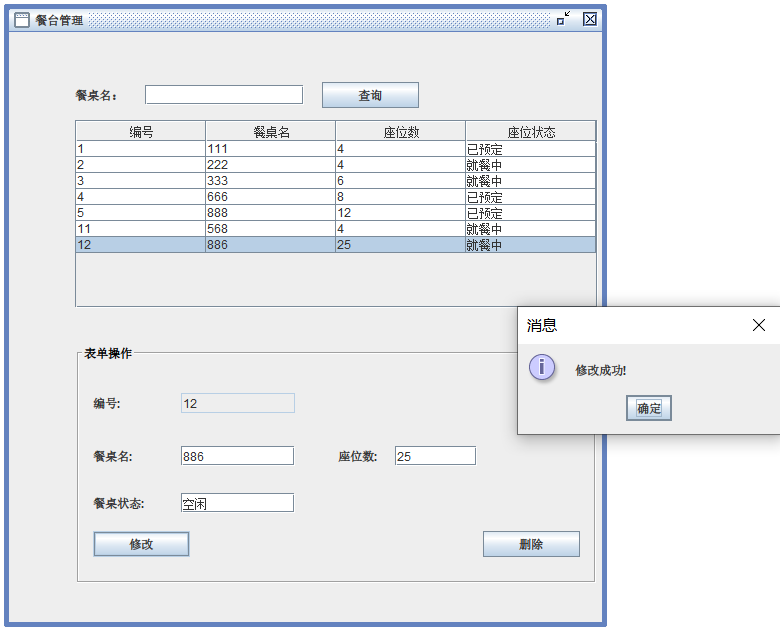
查询客户：



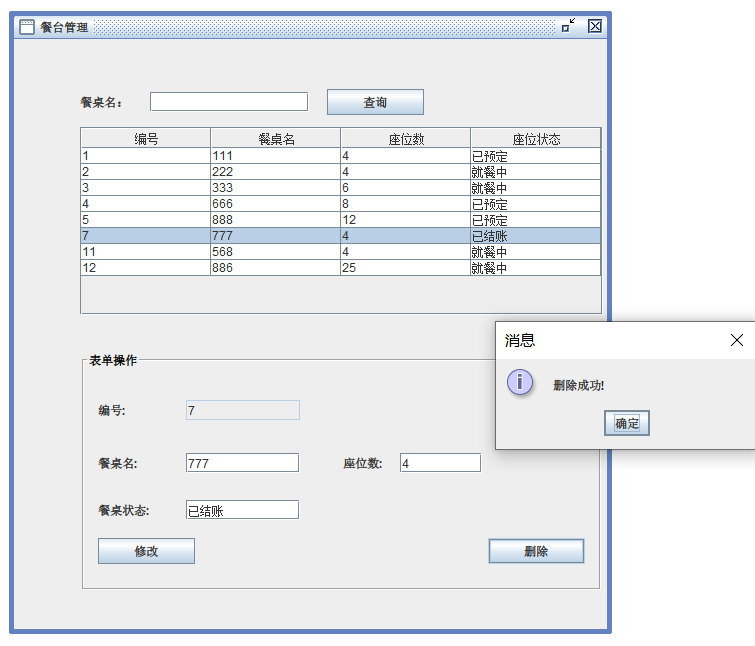
增加餐台：



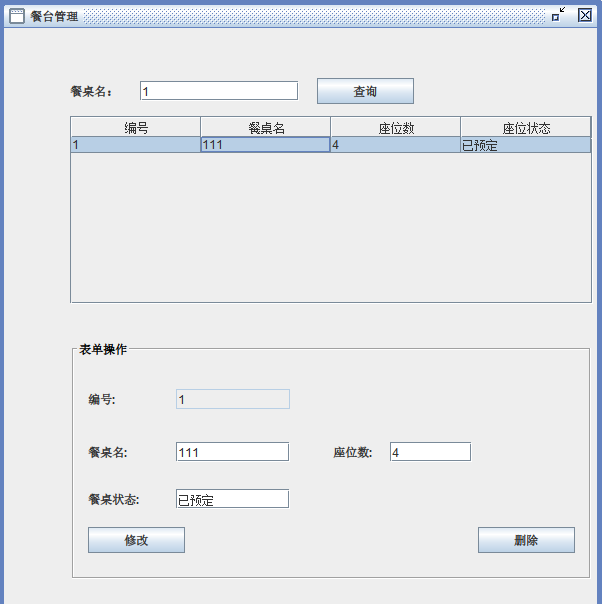
修改餐台：



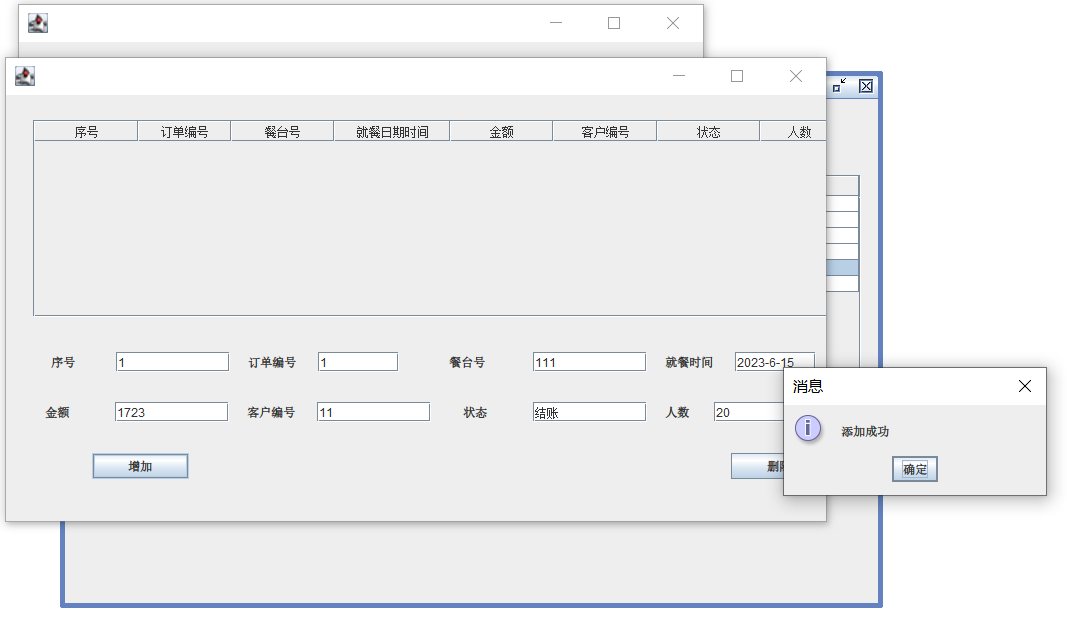
删除餐台：



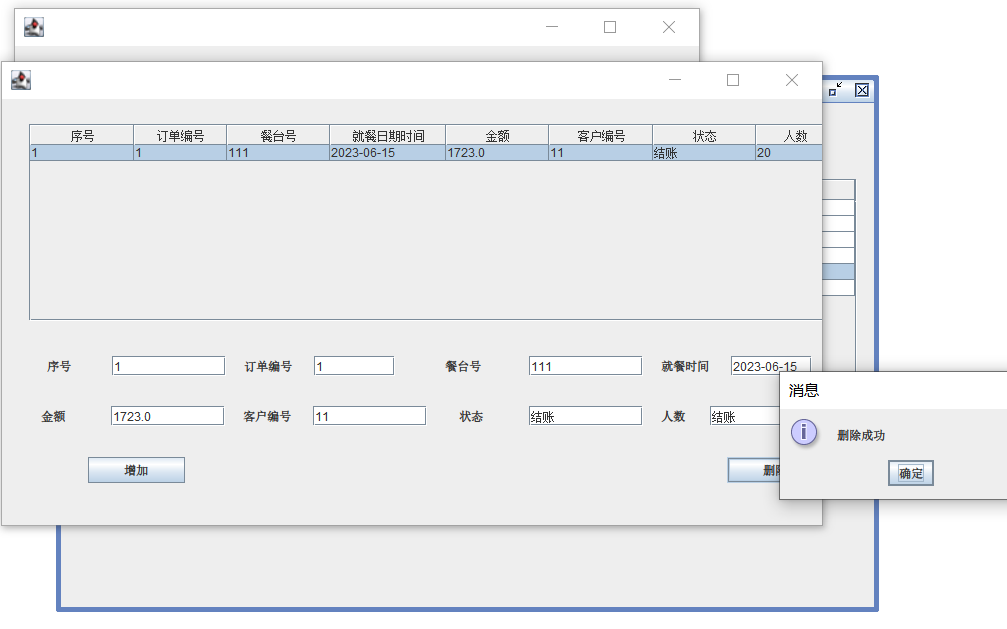
查询餐台：



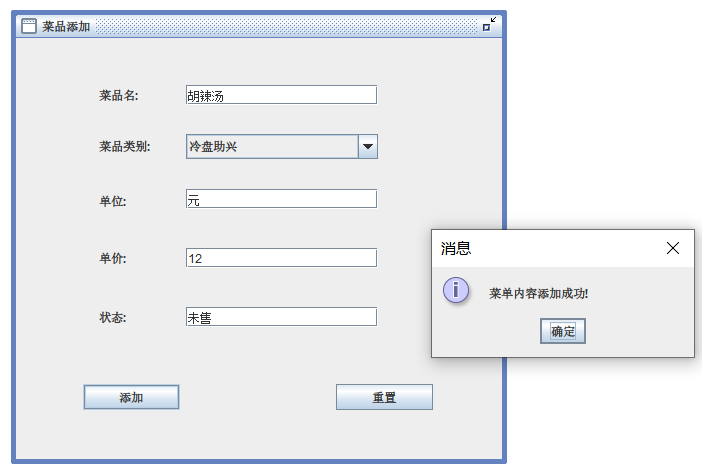
订单增加：



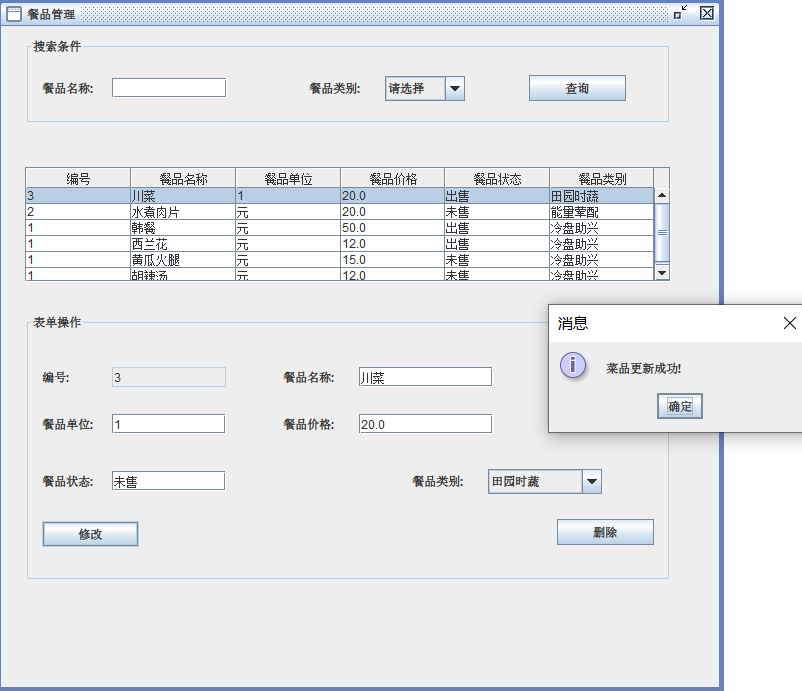
订单删除：



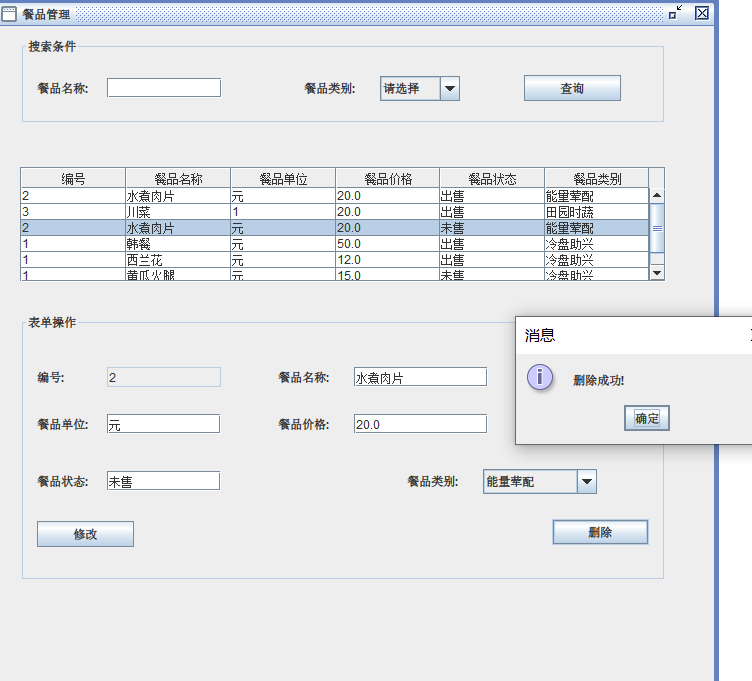
菜品添加：



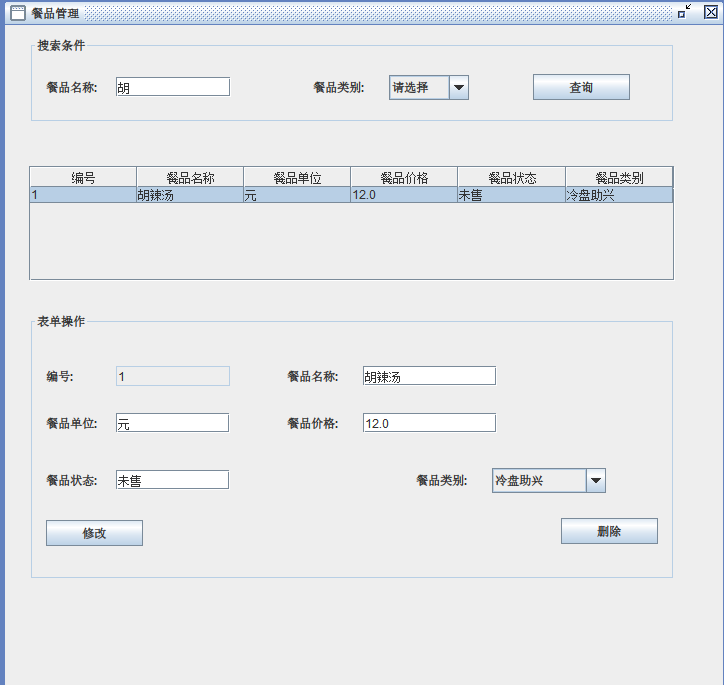
菜品修改：



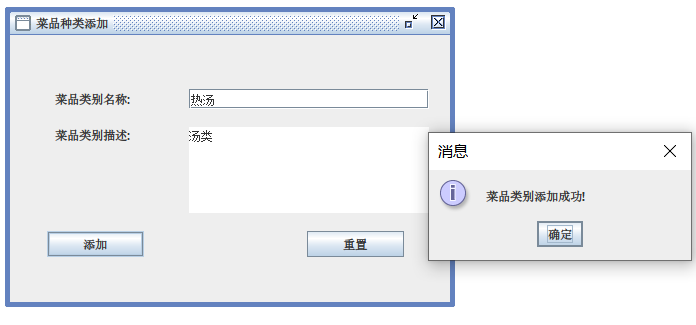
菜品删除：



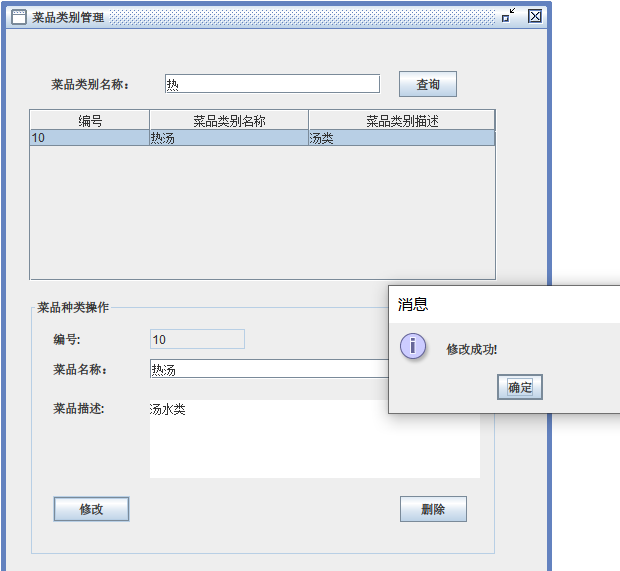
菜品查询：



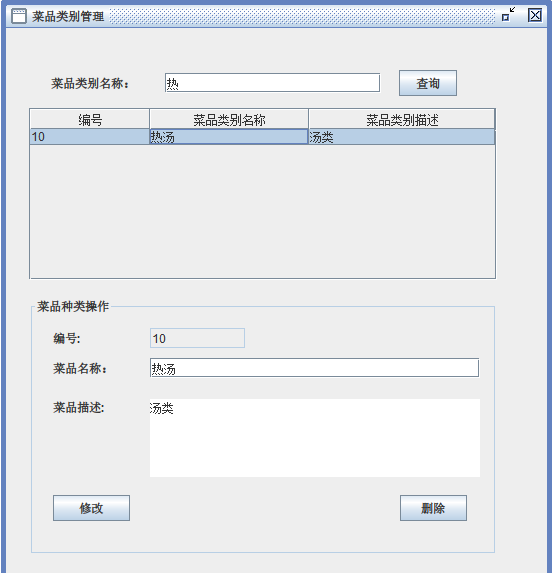
菜品类别添加：



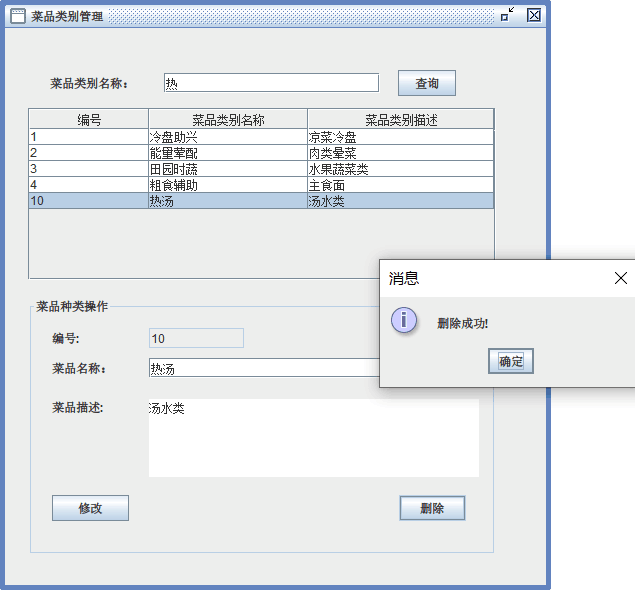
菜品类别修改：



菜品类别查询：



菜品类别删除：



## 2.3 数据需求

提示：说明程序需处理的数据实体及其属性，数据名称，数据格式，数据结构，数据取值范围，数据存储要求，数据处理过程。例如，用户实体

学生实体属性有学号，姓名,…。表格说明范例如表2.1所示。

表2.1 管理员实体

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| id | 字符串 | 10 | 否 | 是 | 管理员编号 |
| username | 字符串 | 20 | 否 | 否 | 管理账号 |
| password | 字符串 | 10 | 否 | 否 | 管理员密码 |

表2.2 员工实体

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| id | 字符串 | 40 | 否 | 是 | 员工编号 |
| name | 字符串 | 258 | 否 | 否 | 员工姓名 |
| sex | 字符串 | 258 | 否 | 否 | 员工性别 |
| birthday | 整型 | 20 | 否 | 否 | 员工年龄 |
| address | 字符串 | 258 | 否 | 否 | 员工地址 |
| identityID | 字符串 | 258 |  | 否 | 员工身份证号 |
| tel | 字符串 | 258 |  | 否 | 员工电话号码 |
| position | 字符串 | 258 |  | 否 | 员工职位 |
| freeze | 字符串 | 258 |  | 否 | 员工是否在职 |

表2.3 客户实体

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| id | 字符串 | 400 | 否 | 是 | 客户编号 |
| name | 字符串 | 200 | 否 | 否 | 客户姓名 |
| sex | 整型 | 20 | 否 | 否 | 客户性别 |
| tel | 字符串 | 200 | 否 | 否 | 客户电话 |
| CardID | 字符串 | 258 |  | 否 | 贵宾卡号 |
| company | 字符串 | 258 |  | 否 | 客户住址 |

表2.4 餐桌实体

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| dno | 字符串 | 20 | 否 | 是 | 餐桌编号 |
| seating | 整型 | 10 | 否 | 否 | 餐桌人数 |
| status | 字符串 | 20 | 否 | 否 | 餐桌状态 |

表2.5 菜品实体

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| id | 字符串 | 40 | 否 | 是 | 菜品编号 |
| Dish\_name | 字符串 | 20 | 否 | 否 | 菜品名 |
| categoryid | 字符串 | 10 | 否 | 否 | 类别编号 |
| unit | 浮点型 | 20 | 否 | 否 | 单位 |
| price | 整形 | 60 |  | 否 | 菜品价格 |
| status | 字符串 | 258 |  | 否 | 菜品状态 |

表2.6 订单实体

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| id | 字符串 | 10 | 否 | 是 | 序号 |
| orderNo | 字符串 | 20 | 否 | 否 | 订单编号 |
| deskId | 字符串 | 20 | 否 | 否 | 餐台号 |
| createtime | 字符串 | 20 | 否 | 否 | 就餐时间 |
| money | 浮点型 | 20 | 否 | 否 | 金额 |
| customerId | 整型 | 20 | 否 | 否 | 客户编号 |
| status | 浮点型 | 20 | 否 | 否 | 状态 |
| number | 整形 | 10 | 否 | 否 | 就餐人数 |

# 3 系统设计

先设置整体框架，写出登录界面、主界面、员工管理界面、客户管理界面、餐桌管理界面、菜品管理界面。然后再在登录界面中添加增加管理员功能，在员工管理当中加入增删改查和显示功能，在客户管理当中加入增删改查、显示功能，在餐桌管理当中加入增删改和显示功能，在菜品管理当中加入增删改查、显示和菜品分类功能。最后设计点菜界面和结账界面。

登录界面包括了管理员的账号密码的输入，要想增加管理员肯定是餐厅内部的操作，所以需要获得老板密匙才能进行增加。

主界面包括了员工管理界面、客户管理界面、餐桌管理界面、菜品管理界面、点菜界面、结账界面，所以设计了这些个按钮进行点击操作。

员工管理界面通过输入或者选择相应信息进行增删改查操作，所以设计了文本框输入所需信息来进行相应操作。

客户管理、餐桌管理、菜品管理设计思路同员工管理类似。菜品管理则多了菜品分类的功能。

点菜界面设计了可以选择已有客户和已有并且空闲的餐桌，当客户选择不同的菜品类型会出现不同的菜品进行选择，从而使客户可以更加方便的点单。还设计了菜品推荐，不熟悉该餐厅的客户可以点击查看菜品推荐来选择菜品。当客户选择菜品和菜品数量后，可以进行增加操作，点击增加操作则会加入菜单当中显示客户已点菜品，最后点击下单按钮进行下单至后台，即点单完成。

## 3.1 运行环境

IDEA,mySQL

## 3.2 系统静态结构设计

采用类图方式给出系统静态结构设计，说明本系统的各软件类的划分。类图要求详细描述，必须有属性和操作，不能仅仅是类名。如果一张图画不下，可以分模块叙述，也可以先画类的简图，在简图之后逐个画出每个类的类图。



系统静态结构说明

针对3.1中绘制的类图说明在类设计是如何体现功能需求和非功能需求的。

## 3.3 人机接口设计

管理员->员工管理界面->员工管理->员工->员工信息

管理员->客户管理界面->客户管理->客户->客户信息

管理员->餐桌管理界面->餐桌管理->餐桌->餐桌信息

管理员->菜品管理界面->菜品管理->菜品->菜品信息

管理员->点菜管理界面->点菜管理->点菜->点菜信息

管理员->结账管理界面->结账管理->结账->流水信息

3.3.2 用户接口1（具体标识符）设计

该界面是管理员登录界面，包括了两个label和两个text组件，点击注册按钮将跳入输入老板密匙界面，点击登录将登录餐饮管理系统。本用户接口实现管理员登录功能，判断账号密码是否存在并是否匹配来验证管理员身份。

3.3.3 用户接口4（主界面）设计



该界面包括了一个label界面和六个button组件。当点击不同的按钮将进入相应的模块。

## 3.5 XXX模块设计说明

用类似第3.4小节的方式，说明第2个模块乃至第N个模块的设计考虑。

如果有更多模块，

往后继续添加。

# 4 数据库设计

## 4.1 数据库环境说明

编译工具： SQLyog - 32 bit。

## 4.2 数据库的命名规则

命名规则:用英文单词组或首字节表示。

## 4.3 逻辑结构设计

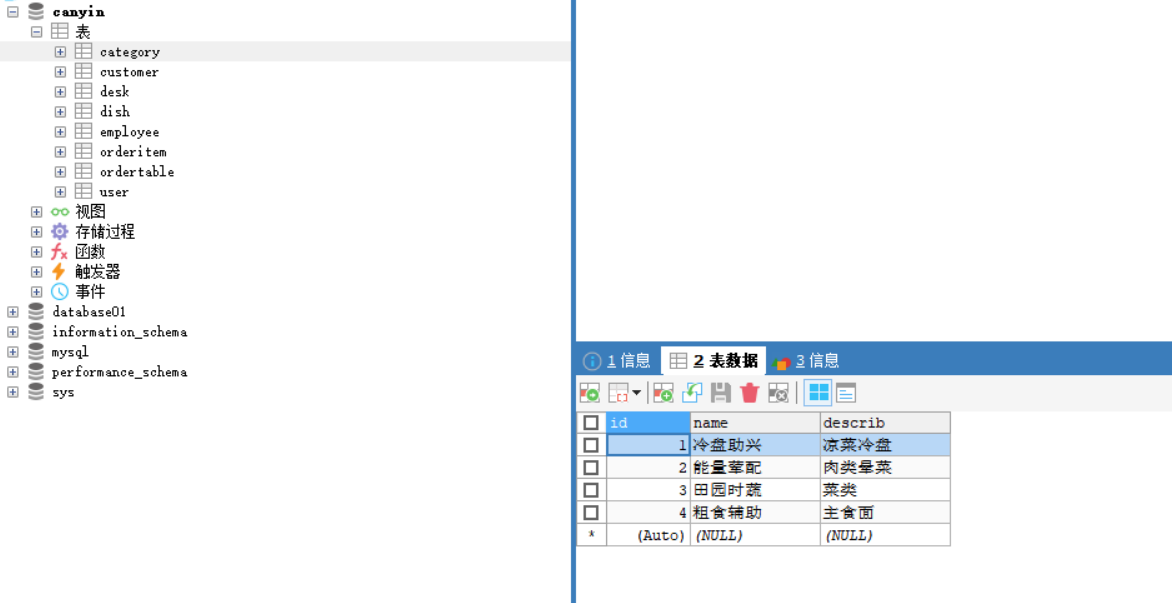
提示：设计逻辑模型。说明把上述原始数据进行分解、合并后重新组织起来的数据库全局逻辑结构，包括所确定的关键字和属性、重新确定的记录结构和文卷结构、所建立的各个文卷之间的相互关系，形成本数据库的数据库管理员视图。

E-R图

关系模型

范式分析

逻辑模型图



## 

## 4.4 物理结构设计

提示：建立系统程序员视图，包括：数据在内存中的安排，包括对索引区、缓冲区的设计；所使用的外存设备及外存空间的组织，包括索引区、数据块的组织与划分；访问数据的方式方法。下列必选项必须说明：

4.4.1 数据库表

说明数据库的表的设计。

表4.1 表汇总

|  |  |
| --- | --- |
| 表 名 | 功 能 说 明 |
| user | 管理员表 |
| employee | 员工表 |
| customer | 客户表 |
| Desk | 餐桌表 |
| dish | 菜品表 |
| Ordertable | 订单表 |
| orderitem | 流水表 |
| category | 菜品类别表 |

表4.2 XX表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 表 名 | | Canyin | | | | | |
| 数据库用户 | | root | | | | | |
| 主 键 | | id | | | | | |
| 其他排序字段 | |  | | | | | |
| 索引字段 | |  | | | | | |
| 序号 | 字段名称 | 数据类型（精度范围） | 允许为空Y/N | 唯一Y/N | 区别度 | 默认值 | 约束条件/说明 |
| 1 | id | 10 | N | Y |  | NULL |  |
| 2 | username | 20 | N | N |  | NULL |  |
| 3 | userpassword | 10 | N | N |  | NULL |  |
|  |  |  |  |  |  |  |  |
| sql脚本 | | CREATE TABLE users  (  id char(10) PRIMARY KEY,  username char(20) ,  userpassword char(10)  ); | | | | | |
| 备注 | |  | | | | | |

4.4.2 视图的设计（如果有的话）

4.4.3存储过程、函数及触发器的设计（如果有的话）

## 4.5 安全性设计

该系统软件信息较安全。

4.5.1 防止用户直接操作数据库的方法

（1）用户标识和鉴定

（2）存取控制

（3）数据加密

4.5.2 用户帐号密码的加密方法

防止用户直接操作数据库的方法。用户只能用帐号登录到应用软件，通过应用软件访问数据库，而没有其他途径操作数据库。（3）用户帐号密码的加密方法。对用户帐号的密码进行加密处理，确保在任何地方都不会出现密码的明文。

4.5.1 角色与权限

可以考虑：（1）角色与权限。确定每个角色对数据库表的操作权限，如创建、检索、更新、删除等。每个角色拥有刚好能够完成任务的权限，不多也不少。在应用时再为用户分配角色，则每个用户的权限等于他所兼角色的权限之和。如表4.3所示，表格仅供参考。

表4.3 角色与权限

|  |  |  |
| --- | --- | --- |
| 角色 | 可以访问的表与列 | 操作权限 |
| 角色A |  |  |
|  |  |
|  |  |
| 角色B |  |  |
|  |  |
|  |  |

列出角色与权限管理SQL脚本代码。

# 5 测试用例设计

对照需求中功能，写出功能的测试用例和测试结果。应考虑边界测试、默认测试、空测试、非法值测试。每个用例应说明下列内容：

(1)测试用例名称

(2)测试项：规定并简要说明本测试用例所要涉及的项和特性、对于每一项、可考虑引用需求说明书和设计说明书。

(3)输入说明：规定执行测试用例所需的各个输入。有些输入可以用值（允许适当的误差）来规定。而另一些输入，如常数表或事务文件可以用名来规定。规定所有合适的数据库、文件、终端信息、内存常驻区域和由操作系统传送的值。规定各输入间所需的所有关系（如时序关系等）。

(4)输出说明：规定测试项的所有输出和特性（如：响应时间）。提供各个输出或特性的正确值（在适当的误差范围内）。

(5)环境要求：规定执行本测试用例所需的硬件特征和配置、系统软件和应用软件。

(6)规程说明：描述对执行本测试用例的测试规程的一切特殊限制。这些限制可以包括特定的准备、操作人员干预、确定特殊的输出和清除过程。

(7)用例间的依赖关系：列出必须在本测试用例之前执行的测试用例名称，归纳依赖性质。建议采用表格描述每个用例，示例见表5.1。

表5.1 登录-注册测试用例

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 功能 | 用例编号 | 操作步骤  （输入条件） | 测试数据 | 预期结果 | 实际结果/判定 |
| 登录 | yl-001 | 输入‘用户名’和‘密码’ | admin，123456 | 登录成功 | 通过 |
| yl-002 | admin，1234567 | 登录失败 | 通过 |
| yl-003 | admin1，123456 | 登录失败 | 通过 |
| yl-004 | admin,空 | 登录失败 | 通过 |
| yl-005 | 空,123456 | 登录失败 | 通过 |
| yl-006 | admin,一个空格 | 登录失败 | 通过 |
| yl-007 | 一个空格，123456 | 登录失败 | 通过 |
| yl-008 | admin，密码长度超过32 | 登录失败 | 通过 |
| yl-009 | 用户名超过36，123456 | 登录失败 | 通过 |
| 注册 | yl-010 | 用户名，密码，确认密码，真是姓名，性别，出生日期，地址，邮政编码，电话，email，密保问题，密保答案，验证码 | Tony,123456,123456,test1,男，1991-10-10，长沙，410014，85577424，505@qq.com，我是谁，test1. | 创建成功 | 通过 |
|  |  |  |  |  |  |

# 附录

## 附录1 程序运行结果

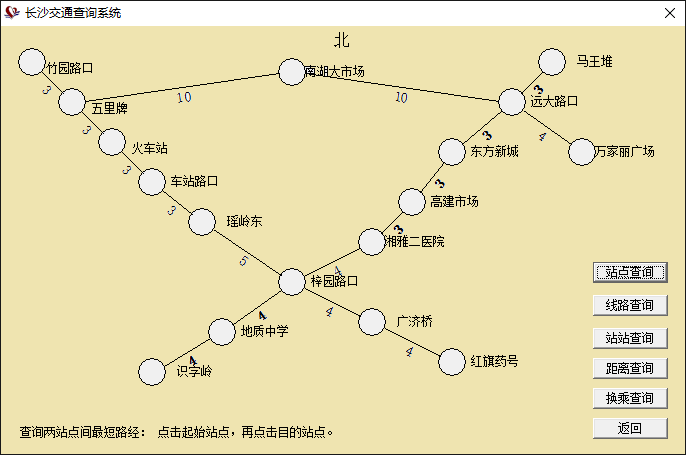


图1 查询界面

## 附录2 程序源代码

列出自己设计的主要代码

