restaurant-system

Generated by Doxygen 1.9.8

Chapter 1

README

Black Plate: Restaurant Management System using Maven Java

About

The profession of managing a restaurant. It includes the principles of OOP and using Java function of planning, organizing, staffing, directing, developing an attitude in food and beverage control systems, and efficiently and effectively planning menus at profitable prices, taking into consideration constraints and others.

More info on the wiki

https://github.com/alu0101132617/restaurant-system/wiki

Tools used:

- Java Development Kit (JDK)
- Maven
- JUnit (for testing)

Database:

- MySql

Functionality:

- Increases operational efficiency.
- Helps the restaurant manager to manage the restaurant more effectively and efficiently by computerizing Meal Ordering, Cart, and Restaurant Management Accounting.
- Avoids paperwork.
- Simple to learn and easy to use.

Clarification of important information:

The system is between the customer and waiter in the restaurant and it's not a virtual system. The waiter takes customer orders. Our system will facilitate the process of taking orders.

Application's GUI:

1. Logged in as an Admin: (Username: admin, Password: admin)

Update Prices choice:

Menu choice to see the previous updates:

2. Logged in as a Cashier: (Any username & password)

Menu choice:

Save Receipt:

All Receipt that has been saved, will be shown in the JavaApplication2 UPDATED file:

About us choice:

Database Schema:

Classes UML:

1.0.1 Configuración de la BBDD (MySQL / MariaDB)

Antes de ejecutar la aplicación, se debe tener instalado MySQL ó MariaDB y haber creado la base de datos correspondiente. 0. Asegurarse de que el gestor de base de datos está instalado y ejecutándose:

```
sudo systemctl status mysql/mariadb
sudo systemctl start mysql/mariadb
sudo systemctl enable mysql/mariadb
```

1. Iniciar sesión en el gestor de base de datos:
   ```
   sudo mysql -u root -p
   ```
2. Crear la base de datos del sistema:
   ```
   CREATE DATABASE project3;
   USE project3;
   ```
3. Importar el script de creación de tablas y datos iniciales (archivo database.sql incluido en el proyecto):
   ```
   SOURCE ruta/al/proyecto/database.sql;
   ```
4. Verificar que las tablas se hayan creado correctamente:
   ```
   SHOW TABLES;
   ```
5. Ejecutar el proyecto:
   ```
   mvn exec:java
   ```

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

Chapter 6

Namespace Documentation

6.1 Package es.ull.esit.app

Packages

- package middleware

Classes

- class AboutUs

  "About us" window displaying the restaurant's history and info.
- class AdminLogin

  Login window for authenticating administrators.
- class AdminProducts

  Administrative window for managing products and prices.
- class ApplicationLauncher

  Auxiliary entry point used to start the application's UI.
- class CashierLogin

  Login window for authenticating cashiers.
- class Login

  Login window for the Restaurant System.
- class Order

  Main menu window for taking customer orders.

6.2 Package es.ull.esit.app.middleware

Packages

- package model
- package service

Classes

- class ApiClient

  API client for interacting with the backend REST API.

6.3 Package es.ull.esit.app.middleware.model

Classes

- class Appetizer

  Client-side model representing an appetizer received from the backend API.
- class BillResult

  Data Transfer Object representing the result of a bill calculation.
- class Cashier

  Client-side model representing a cashier returned by the backend.
- class Drink

  Client-side model representing a drink returned by the backend API.
- class MainCourse

  Client-side model representing a main course returned by the backend.
- class User

  Client-side model representing an authenticated user.

6.4 Package es.ull.esit.app.middleware.service

Classes

- class AuthService

  Service responsible for handling authentication logic on the client side.
- class OrderService

  Service that handles order-related calculations and receipt generation.
- class ProductService

  Service handling business logic for menu products.
- class ReportService

  Service providing reporting and system status operations.

6.5 Package es.ull.esit.server

Packages

- package config
- package controller
- package repo

Classes

- class RestaurantApplication

  Main Spring Boot application class for the restaurant backend.

6.6 Package es.ull.esit.server.config

Classes

- class SecurityConfig

  Spring Security configuration for the backend.

6.7 Package es.ull.esit.server.controller

Classes

- class AppetizerController

  REST controller for managing appetizers.
- class AuthController

  Rest controller for handling authentication-related requests.
- class CashierController

  REST controller for managing cashiers.
- class DrinkController

  REST controller for managing drinks.
- class HealthController

  REST controller for health and database connectivity checks.
- class MainCourseController

  REST controller for managing main courses.
- class MenuController

  REST controller that exposes a consolidated restaurant menu.

6.8 Package es.ull.esit.server.middleware.model

Classes

- class Appetizer

  JPA entity that represents an appetizer in the menu.
- class Cashier

  JPA entity that represents a cashier in the system.
- class Drink

  JPA entity that represents a drink in the menu.
- class MainCourse

  JPA entity that represents a main course in the menu.
- class User

  JPA entity that represents a user in the system.

6.9 Package es.ull.esit.server.repo

Classes

- interface AppetizerRepository

  Repository interface for managing appetizers in the database.
- interface CashierRepository

  Repository interface for managing cashiers in the database.
- interface DrinkRepository

  Repository interface for managing drinks in the database.
- interface MainCourseRepository

  Repository interface for managing main courses in the database.
- interface UserRepository

  Repository interface for accessing users in the database.

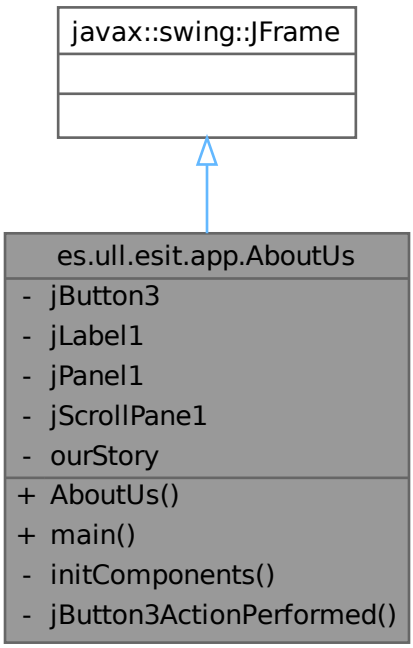
Chapter 7

Class Documentation

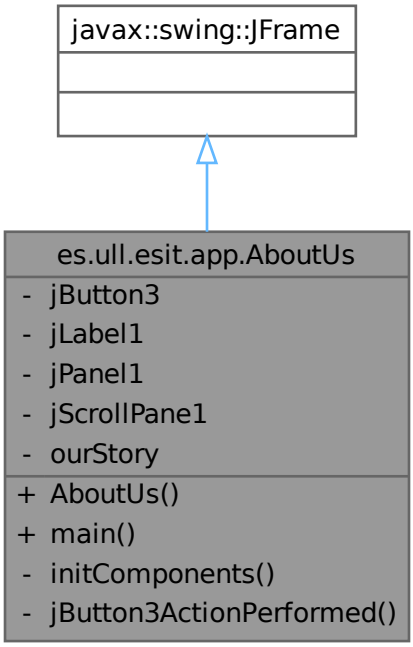
7.1 es.ull.esit.app.AboutUs Class Reference

"About us" window displaying the restaurant's history and info.

Inheritance diagram for es.ull.esit.app.AboutUs:



Collaboration diagram for es.ull.esit.app.AboutUs:



Public Member Functions

- AboutUs ()

  Default constructor.

Static Public Member Functions

- static void main (String[ ] args)

  Standalone entry point for testing the Info window.

Private Member Functions

- void initComponents ()

  Initializes and lays out Swing components.
- void jButton3ActionPerformed (java.awt.event.ActionEvent evt)

  Handler for the "Go Back" button.

Private Attributes

- javax.swing.JButton jButton3

  Button that navigates back to the cashier login window.
- javax.swing.JLabel jLabel1

  Logo or image displayed at the top of the Info window.
- javax.swing.JPanel jPanel1

  Main container panel for all components in the Info window.
- javax.swing.JScrollPane jScrollPane1

  Scroll pane that wraps the static "ourStory" text area.
- javax.swing.JTextArea ourStory

  Text area containing the restaurant history and description.

7.1.1 Detailed Description

"About us" window displaying the restaurant's history and info.

```
Simple Swing window that:
- shows a static text area with the project story ("Our Story").
- includes a navigation button to return to the previous screen
CashierLogin.

Does not perform any network or database operations:
all the content is embedded directly in the text area.
```

Definition at line 14 of file AboutUs.java.

7.1.2 Constructor & Destructor Documentation

7.1.2.1 AboutUs()

```
es.ull.esit.app.AboutUs.AboutUs ( )  [inline]
```

Default constructor.

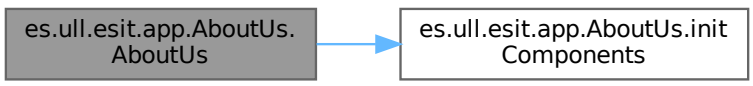
```
Creates an instance of the Info window and UI components.
```

Definition at line 21 of file AboutUs.java.

```
00021                     {
00022         initComponents();
00023     }
```

References es.ull.esit.app.AboutUs.initComponents().

Here is the call graph for this function:



7.1.3 Member Function Documentation

7.1.3.1 initComponents()

```
void es.ull.esit.app.AboutUs.initComponents ( )  [inline], [private]
```

Initializes and lays out Swing components.

```
Generates by the Form Editor: don't modify manually.

Sets up the JFrame, including the main panel, the static text area
with the
restaurant story, the logo/image label, and the "Go Back" button.
Also configure the layout and event handlers.
```

Definition at line 39 of file AboutUs.java.

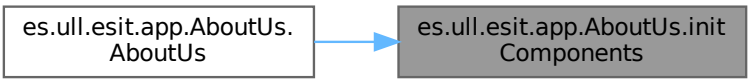
```
00039                                {
00040
00041     jPanel1 = new javax.swing.JPanel();
00042     jButton3 = new javax.swing.JButton();
00043     jScrollPane1 = new javax.swing.JScrollPane();
00044     ourStory = new javax.swing.JTextArea();
00045     jLabel1 = new javax.swing.JLabel();
00046
00047     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00048     setTitle("Info");
00049     setResizable(false);
00050
00051     jPanel1.setBackground(new java.awt.Color(248, 244, 230));
00052
00053     jButton3.setBackground(new java.awt.Color(255, 255, 255));
00054     jButton3.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00055     jButton3.setText("Go Back");
00056     jButton3.addActionListener(new java.awt.event.ActionListener() {
00057       public void actionPerformed(java.awt.event.ActionEvent evt) {
00058         jButton3ActionPerformed(evt);
00059       }
00060     });
00061
00062     ourStory.setEditable(false);
00063     ourStory.setBackground(new java.awt.Color(248, 244, 230));
00064     ourStory.setColumns(20);
00065     ourStory.setFont(new java.awt.Font("Yu Gothic UI Light", 0, 14)); // NOI18N
00066     ourStory.setRows(5);
00067     ourStory.setText(
00068         "  Our Story |\n" +
00069             "  Collecting flavors from all over the world to give everyone a taste of what they love.
    \n\n" +
00070             "  Black Plate came in with this vision in mind, to be more of a home rather than an
    establishment.\n\n" +
00071             "  2021 marked the beginning of the journey of Black Plate, starting from Al Khobar.\n\n"
    +
00072             "  Because your visit means a lot to us, we would love to hear your suggestions and
    concerns \n\n" +
00073             "  so we can improve!\n");
00074     ourStory.setBorder(javax.swing.BorderFactory.createLineBorder(new java.awt.Color(0, 0, 0)));
00075     jScrollPane1.setViewportView(ourStory);
00076
00077     // Updated path to match other UI resources
00078     jLabel1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/ui/screenshot.png")));
00079
00080     javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
00081     jPanel1.setLayout(jPanel1Layout);
00082     jPanel1Layout.setHorizontalGroup(
00083         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00084             .addGroup(jPanel1Layout.createSequentialGroup()
00085                 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00086                     .addGroup(jPanel1Layout.createSequentialGroup()
00087                         .addGap(55, 55, 55)
00088
    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00089                             .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 141,
00090                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00091                             .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 1004,
00092                                 javax.swing.GroupLayout.PREFERRED_SIZE)))
00093                     .addGroup(jPanel1Layout.createSequentialGroup()
00094                         .addGap(489, 489, 489)
00095                         .addComponent(jLabel1)))
```

```
00096                 .addContainerGap(159, Short.MAX_VALUE)));
00097     jPanel1Layout.setVerticalGroup(
00098         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00099             .addGroup(jPanel1Layout.createSequentialGroup()
00100                 .addContainerGap(71, Short.MAX_VALUE)
00101                 .addComponent(jLabel1)
00102                 .addGap(67, 67, 67)
00103                 .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 203,
00104                     javax.swing.GroupLayout.PREFERRED_SIZE)
00105                 .addGap(26, 26, 26)
00106                 .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00107                     javax.swing.GroupLayout.PREFERRED_SIZE)
00108                 .addGap(30, 30, 30)));
00109
00110     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00111     getContentPane().setLayout(layout);
00112     layout.setHorizontalGroup(
00113         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00114             .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00115                 Short.MAX_VALUE));
00116     layout.setVerticalGroup(
00117         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00118             .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00119                 Short.MAX_VALUE));
00120
00121     pack();
00122     setLocationRelativeTo(null);
00123   }// </editor-fold>//GEN-END:initComponents
```

References es.ull.esit.app.AboutUs.jButton3, es.ull.esit.app.AboutUs.jLabel1, es.ull.esit.app.AboutUs.jPanel1, es.ull.esit.app.AboutUs.jScrollPane1, and es.ull.esit.app.AboutUs.ourStory.

Referenced by es.ull.esit.app.AboutUs.AboutUs().

Here is the caller graph for this function:



7.1.3.2 jButton3ActionPerformed()

```
void es.ull.esit.app.AboutUs.jButton3ActionPerformed (
             java.awt.event.ActionEvent evt )  [inline], [private]
```

Handler for the "Go Back" button.

```
Closes the current Info window and opens the CashierLogin
window, returning the user to the cashier login screen.
```

Parameters

evt	Action event triggered by button click.

Definition at line 133 of file AboutUs.java.

```
00133                                                                   {//
      GEN-FIRST:event_jButton3ActionPerformed
00134     new CashierLogin().setVisible(true);
00135     this.dispose(); // Close current window
00136   }// GEN-LAST:event_jButton3ActionPerformed
```

7.1.3.3 main()

```
static void es.ull.esit.app.AboutUs.main (
             String[] args )  [inline], [static]
```

Standalone entry point for testing the Info window.

Sets the Nimbus look and feel if available and shows an instance of Info. In the normal application flow this window is opened from CashierLogin via the "About us" button.

Parameters

args	Command line arguments (not used).

Definition at line 148 of file AboutUs.java.

```
00148                                       {
00149     try {
00150       for (javax.swing.UIManager.LookAndFeelInfo info :
      javax.swing.UIManager.getInstalledLookAndFeels()) {
00151         if ("Nimbus".equals(info.getName())) {
00152           javax.swing.UIManager.setLookAndFeel(info.getClassName());
00153           break;
00154         }
00155       }
00156     } catch (ClassNotFoundException ex) {
00157       java.util.logging.Logger.getLogger(AboutUs.class.getName())
00158           .log(java.util.logging.Level.SEVERE, null, ex);
00159     } catch (InstantiationException ex) {
00160       java.util.logging.Logger.getLogger(AboutUs.class.getName())
00161           .log(java.util.logging.Level.SEVERE, null, ex);
00162     } catch (IllegalAccessException ex) {
00163       java.util.logging.Logger.getLogger(AboutUs.class.getName())
00164           .log(java.util.logging.Level.SEVERE, null, ex);
00165     } catch (javax.swing.UnsupportedLookAndFeelException ex) {
00166       java.util.logging.Logger.getLogger(AboutUs.class.getName())
00167           .log(java.util.logging.Level.SEVERE, null, ex);
00168     }
00169     // </editor-fold>
00170
00171     /* Create and display the form */
00172     java.awt.EventQueue.invokeLater(() -> new AboutUs().setVisible(true));
00173   }
```

7.1.4 Member Data Documentation

7.1.4.1 jButton3

```
javax.swing.JButton es.ull.esit.app.AboutUs.jButton3  [private]
```

Button that navigates back to the cashier login window.

Definition at line 177 of file AboutUs.java.

Referenced by es.ull.esit.app.AboutUs.initComponents().

7.1.4.2 jLabel1

```
javax.swing.JLabel es.ull.esit.app.AboutUs.jLabel1  [private]
```

Logo or image displayed at the top of the Info window.

Definition at line 179 of file AboutUs.java.

Referenced by es.ull.esit.app.AboutUs.initComponents().

7.1.4.3 jPanel1

```
javax.swing.JPanel es.ull.esit.app.AboutUs.jPanel1  [private]
```

Main container panel for all components in the Info window.

Definition at line 181 of file AboutUs.java.

Referenced by es.ull.esit.app.AboutUs.initComponents().

7.1.4.4 jScrollPane1

```
javax.swing.JScrollPane es.ull.esit.app.AboutUs.jScrollPane1  [private]
```

Scroll pane that wraps the static "ourStory" text area.

Definition at line 183 of file AboutUs.java.

Referenced by es.ull.esit.app.AboutUs.initComponents().

7.1.4.5 ourStory

```
javax.swing.JTextArea es.ull.esit.app.AboutUs.ourStory  [private]
```

Text area containing the restaurant history and description.

Definition at line 185 of file AboutUs.java.

Referenced by es.ull.esit.app.AboutUs.initComponents().

The documentation for this class was generated from the following file:

- AboutUs.java

7.2 es.ull.esit.app.AdminLogin Class Reference

Login window for authenticating administrators.

Inheritance diagram for es.ull.esit.app.AdminLogin:

Collaboration diagram for es.ull.esit.app.AdminLogin:

Public Member Functions

- AdminLogin ()

  Constructor.

Static Public Member Functions

- static void main (String[ ] args)

  Standalone entry point for testing the AdminLogin window.

Private Member Functions

- void initComponents ()

  Initializes GUI components.
- void jButton2ActionPerformed (java.awt.event.ActionEvent evt)

  Action handler for the "Update Prices" button.
- void jButton1ActionPerformed (java.awt.event.ActionEvent evt)

  Action handler for the "Menu" button.
- void jButton3ActionPerformed (java.awt.event.ActionEvent evt)

  Action handler for the "LogOut" button.

Private Attributes

- javax.swing.JButton jButton1

  Button that opens the main menu window.
- javax.swing.JButton jButton2

  Button that opens the price management window.
- javax.swing.JButton jButton3

  Button used to log out and return to the login screen.
- javax.swing.JLabel jLabel1

  Label that displays the application logo or image.
- javax.swing.JLabel jLabel3

  Label that displays the welcome text for the administrator.
- javax.swing.JPanel jPanel1

  Main container panel for all UI elements.

7.2.1 Detailed Description

Login window for authenticating administrators.

```
It is displayed after a successful login with role ADMIN.
Shows a Swing form that lets administrators:
- access to the product and price management window.
- access to the general menu window used to place orders.
- log out and return to the login window.
```

Definition at line 12 of file AdminLogin.java.

7.2.2 Constructor & Destructor Documentation

7.2.2.1 AdminLogin()

```
es.ull.esit.app.AdminLogin.AdminLogin ( )  [inline]
```

Constructor.

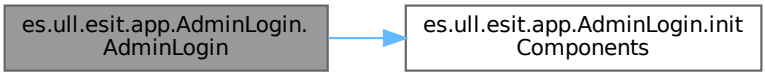
```
Creates the admin login window and initializes GUI components.
```

Definition at line 19 of file AdminLogin.java.

```
00019                       {
00020     initComponents();
00021   }
```

References es.ull.esit.app.AdminLogin.initComponents().

Here is the call graph for this function:



7.2.3 Member Function Documentation

7.2.3.1 initComponents()

```
void es.ull.esit.app.AdminLogin.initComponents ( )  [inline], [private]
```

Initializes GUI components.

```
Generated by the Form Editor: do not modify.
Sets up welcome label, logo image, buttons for "Update Prices",
"Menu", and "LogOut" and the layout and configuration of the window.
```

Definition at line 32 of file AdminLogin.java.

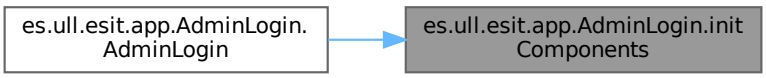
```
00032                             {
00033
00034     jPanel1 = new javax.swing.JPanel();
00035     jLabel3 = new javax.swing.JLabel();
00036     jLabel1 = new javax.swing.JLabel();
00037     jButton2 = new javax.swing.JButton();
00038     jButton1 = new javax.swing.JButton();
00039     jButton3 = new javax.swing.JButton();
00040
00041     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00042     setTitle("Admin ");
00043     setResizable(false);
00044
00045     jPanel1.setBackground(new java.awt.Color(248, 244, 230));
00046
00047     jLabel3.setFont(new java.awt.Font("Yu Gothic UI", 1, 24)); // NOI18N
00048     jLabel3.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00049     jLabel3.setText("Welcome Admin");
00050
00051     jLabel1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/ui/screenshot.png"))); //
      NOI18N
00052
00053     jButton2.setBackground(new java.awt.Color(153, 153, 153));
00054     jButton2.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00055     jButton2.setText("Update Prices");
00056     jButton2.addActionListener(new java.awt.event.ActionListener() {
00057
00058       public void actionPerformed(java.awt.event.ActionEvent evt) {
00059         jButton2ActionPerformed(evt);
00060       }
00061     });
00062
00063     jButton1.setBackground(new java.awt.Color(153, 153, 153));
00064     jButton1.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00065     jButton1.setText("Menu");
00066     jButton1.addActionListener(new java.awt.event.ActionListener() {
00067
00068       public void actionPerformed(java.awt.event.ActionEvent evt) {
00069         jButton1ActionPerformed(evt);
00070       }
00071     });
00072
00073     jButton3.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00074     jButton3.setText("LogOut");
00075     jButton3.addActionListener(new java.awt.event.ActionListener() {
00076
00077       public void actionPerformed(java.awt.event.ActionEvent evt) {
00078         jButton3ActionPerformed(evt);
00079       }
00080     });
00081
00082     javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
00083     jPanel1.setLayout(jPanel1Layout);
00084     jPanel1Layout.setHorizontalGroup(
00085         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00086             .addGroup(jPanel1Layout.createSequentialGroup()
00087                 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00088                     .addGroup(jPanel1Layout.createSequentialGroup()
00089                         .addGap(140, 140, 140)
00090                         .addComponent(jLabel1))
00091                     .addGroup(jPanel1Layout.createSequentialGroup()
00092                         .addGap(66, 66, 66)
00093
      .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00094                             .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 320,
```

```
00095                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00096                             .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 320,
00097                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00098                             .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 306,
00099                                 javax.swing.GroupLayout.PREFERRED_SIZE)))
00100                     .addGroup(jPanel1Layout.createSequentialGroup()
00101                         .addGap(152, 152, 152)
00102                         .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 141,
00103                             javax.swing.GroupLayout.PREFERRED_SIZE)))
00104                 .addContainerGap(69, Short.MAX_VALUE)));
00105     jPanel1Layout.setVerticalGroup(
00106         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00107             .addGroup(jPanel1Layout.createSequentialGroup()
00108                 .addGap(31, 31, 31)
00109                 .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
00110                     javax.swing.GroupLayout.PREFERRED_SIZE)
00111                 .addGap(18, 18, 18)
00112                 .addComponent(jLabel1)
00113                 .addGap(29, 29, 29)
00114                 .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00115                     javax.swing.GroupLayout.PREFERRED_SIZE)
00116                 .addGap(18, 18, 18)
00117                 .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00118                     javax.swing.GroupLayout.PREFERRED_SIZE)
00119                 .addGap(29, 29, 29)
00120                 .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00121                     javax.swing.GroupLayout.PREFERRED_SIZE)
00122                 .addContainerGap(115, Short.MAX_VALUE)));
00123 
00124     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00125     getContentPane().setLayout(layout);
00126     layout.setHorizontalGroup(
00127         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00128             .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00129                 javax.swing.GroupLayout.PREFERRED_SIZE));
00130     layout.setVerticalGroup(
00131         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00132             .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00133                 Short.MAX_VALUE));
00134 
00135     pack();
00136     setLocationRelativeTo(null);
00137   }// </editor-fold>//GEN-END:initComponents
```

References es.ull.esit.app.AdminLogin.jButton1, es.ull.esit.app.AdminLogin.jButton2, es.ull.esit.app.AdminLogin.jButton3, es.ull.esit.app.AdminLogin.jLabel1, es.ull.esit.app.AdminLogin.jLabel3, and es.ull.esit.app.AdminLogin.jPanel1.

Referenced by es.ull.esit.app.AdminLogin.AdminLogin().

Here is the caller graph for this function:



7.2.3.2 jButton1ActionPerformed()

```
void es.ull.esit.app.AdminLogin.jButton1ActionPerformed (
             java.awt.event.ActionEvent evt )  [inline], [private]
```

Action handler for the "Menu" button.

Actions steps:

- Opens the general menu window to place orders.

Parameters

evt	[java.awt.event.ActionEvent] Action event triggered by button click.

Definition at line 172 of file AdminLogin.java.

```
00172                                                                    {//
      GEN-FIRST:event_jButton1ActionPerformed
00173     try {
00174         new Order().setVisible(true);
00175         this.dispose();
00176     } catch (Exception ex) {
00177         ex.printStackTrace();
00178         javax.swing.JOptionPane.showMessageDialog(
00179             this,
00180             "Error opening menu window:\n" + ex.getMessage(),
00181             "Error",
00182             javax.swing.JOptionPane.ERROR_MESSAGE
00183         );
00184     }
00185   }// GEN-LAST:event_jButton1ActionPerformed
```

7.2.3.3 jButton2ActionPerformed()

```
void es.ull.esit.app.AdminLogin.jButton2ActionPerformed (
             java.awt.event.ActionEvent evt )  [inline], [private]
```

Action handler for the "Update Prices" button.

```
Actions steps:
- Opens the product administration window to modify prices and
products.
```

Parameters

evt	[java.awt.event.ActionEvent] Action event triggered by button click.

Definition at line 149 of file AdminLogin.java.

```
00149                                                                    {//
      GEN-FIRST:event_jButton2ActionPerformed
00150     try {
00151         new AdminProducts().setVisible(true);
00152         this.dispose();
00153     } catch (Exception ex) {
00154         ex.printStackTrace();
00155         javax.swing.JOptionPane.showMessageDialog(
00156             this,
00157             "Error opening product admin window:\n" + ex.getMessage(),
00158             "Error",
00159             javax.swing.JOptionPane.ERROR_MESSAGE
00160         );
00161     }
00162   }// GEN-LAST:event_jButton2ActionPerformed
```

7.2.3.4 jButton3ActionPerformed()

```
void es.ull.esit.app.AdminLogin.jButton3ActionPerformed (
             java.awt.event.ActionEvent evt )  [inline], [private]
```

Action handler for the "LogOut" button.

Actions steps:

- Logs out the admin and returns to the login window.

Parameters

evt	[java.awt.event.ActionEvent] Action event triggered by button click.

Definition at line 195 of file AdminLogin.java.

```
00195                                                                    {//
      GEN-FIRST:event_jButton3ActionPerformed
00196     new Login().setVisible(true);
00197     this.dispose();
00198   }// GEN-LAST:event_jButton3ActionPerformed
```

7.2.3.5 main()

```
static void es.ull.esit.app.AdminLogin.main (
             String[] args )  [inline], [static]
```

Standalone entry point for testing the AdminLogin window.

```
   Sets the Nimbus look and feel if available and shows an instance
   of AdminLogin. In the normal application flow this window is started
   from the Login class after a successful admin login.
```

Parameters

args	[String[]] Command line arguments (not used).

Definition at line 209 of file AdminLogin.java.

```
00209                                         {
00210     try {
00211       for (javax.swing.UIManager.LookAndFeelInfo info :
      javax.swing.UIManager.getInstalledLookAndFeels()) {
00212         if ("Nimbus".equals(info.getName())) {
00213           javax.swing.UIManager.setLookAndFeel(info.getClassName());
00214           break;
00215         }
00216       }
00217     } catch (ClassNotFoundException | InstantiationException | IllegalAccessException
00218         | javax.swing.UnsupportedLookAndFeelException ex) {
00219
      java.util.logging.Logger.getLogger(AdminLogin.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
00220     }
00221     // </editor-fold>
00222
00223     java.awt.EventQueue.invokeLater(new Runnable() {
00224       public void run() {
00225         new AdminLogin().setVisible(true);
00226       }
00227     });
00228   }
```

7.2.4 Member Data Documentation

7.2.4.1 jButton1

```
javax.swing.JButton es.ull.esit.app.AdminLogin.jButton1  [private]
```

Button that opens the main menu window.

Definition at line 232 of file AdminLogin.java.

Referenced by es.ull.esit.app.AdminLogin.initComponents().

7.2.4.2 jButton2

```
javax.swing.JButton es.ull.esit.app.AdminLogin.jButton2  [private]
```

Button that opens the price management window.

Definition at line 234 of file AdminLogin.java.

Referenced by es.ull.esit.app.AdminLogin.initComponents().

7.2.4.3 jButton3

```
javax.swing.JButton es.ull.esit.app.AdminLogin.jButton3  [private]
```

Button used to log out and return to the login screen.

Definition at line 236 of file AdminLogin.java.

Referenced by es.ull.esit.app.AdminLogin.initComponents().

7.2.4.4 jLabel1

```
javax.swing.JLabel es.ull.esit.app.AdminLogin.jLabel1  [private]
```

Label that displays the application logo or image.

Definition at line 238 of file AdminLogin.java.

Referenced by es.ull.esit.app.AdminLogin.initComponents().

7.2.4.5 jLabel3

```
javax.swing.JLabel es.ull.esit.app.AdminLogin.jLabel3  [private]
```

Label that displays the welcome text for the administrator.

Definition at line 240 of file AdminLogin.java.

Referenced by es.ull.esit.app.AdminLogin.initComponents().

7.2.4.6 jPanel1

```
javax.swing.JPanel es.ull.esit.app.AdminLogin.jPanel1  [private]
```

Main container panel for all UI elements.

Definition at line 242 of file AdminLogin.java.

Referenced by es.ull.esit.app.AdminLogin.initComponents().

The documentation for this class was generated from the following file:

- AdminLogin.java

7.3 es.ull.esit.app.AdminProducts Class Reference

Administrative window for managing products and prices.

Inheritance diagram for es.ull.esit.app.AdminProducts:

Collaboration diagram for es.ull.esit.app.AdminProducts:

Public Member Functions

- AdminProducts ()

  Constructor.

Static Public Member Functions

- static void main (String args[ ])

Standalone entry point for testing the AdminProducts window.

Private Member Functions

- void refreshAllTables ()

  Reloads all product tables.
- void initComponents ()

  Initializes GUI components.
- void jButton2ActionPerformed (java.awt.event.ActionEvent evt)

  Action handler for the "Add" button in the Drinks tab.
- void jButton3ActionPerformed (java.awt.event.ActionEvent evt)

  Action handler for the "Go Back" button.
- void jButton5ActionPerformed (java.awt.event.ActionEvent evt)

  Action handler for the "Add" button in the Appetizers tab.
- void jButton7ActionPerformed (java.awt.event.ActionEvent evt)

  Action handler for the "Add" button in the MainCourse tab.
- void jTable1KeyPressed (java.awt.event.KeyEvent evt)

  Key handler for the drinks table.
- void jTable1MouseClicked (java.awt.event.MouseEvent evt)

  Mouse handler for the drinks table.
- void jTable2MouseClicked (java.awt.event.MouseEvent evt)

  Mouse handler for the appetizers table.
- void jTable3MouseClicked (java.awt.event.MouseEvent evt)

  Mouse handler for the main course table.
- void jButton1ActionPerformed (java.awt.event.ActionEvent evt)

  Action handler for the "Update" button in the Drinks tab.
- void jButton4ActionPerformed (java.awt.event.ActionEvent evt)

  Action handler for the "Update" button in the Appetizers tab.
- void jButton6ActionPerformed (java.awt.event.ActionEvent evt)

  Action handler for the "Update" button in the MainCourse tab.

Private Attributes

- final ProductService productService

  Service used to call the REST API and apply product-related logic.
- javax.swing.JTextField itemname

  Text field for the drink name (Drinks tab).
- javax.swing.JTextField itemname1

  Text field for the appetizer name (Appetizers tab).
- javax.swing.JTextField itemname2

  Text field for the main course name (MainCourse tab).
- javax.swing.JTextField itemprice

  Text field for the drink price (Drinks tab).
- javax.swing.JTextField itemprice1

  Text field for the appetizer price (Appetizers tab).
- javax.swing.JTextField itemprice2

  Text field for the main course price (MainCourse tab).
- javax.swing.JButton jButton1

  Button to update an existing drink.
- javax.swing.JButton jButton2

Button to add a new drink.

- javax.swing.JButton jButton3

  Button to go back to the admin main window.
- javax.swing.JButton jButton4

  Button to update an existing appetizer.
- javax.swing.JButton jButton5

  Button to add a new appetizer.
- javax.swing.JButton jButton6

  Button to update an existing main course.
- javax.swing.JButton jButton7

  Button to add a new main course.
- javax.swing.JLabel jLabel1

  Main title label ("Items Prices Update Portal").
- javax.swing.JLabel jLabel10

  Helper label used in the MainCourse tab.
- javax.swing.JLabel jLabel2

  Section title label for the Drinks tab.
- javax.swing.JLabel jLabel3

  Helper label with hints for adding drinks.
- javax.swing.JLabel jLabel4

  Helper label with hints for updating drinks.
- javax.swing.JLabel jLabel5

  Helper label with hints for adding appetizers.
- javax.swing.JLabel jLabel6

  Section title label for the Appetizers tab.
- javax.swing.JLabel jLabel7

  Helper label with hints for updating appetizers.
- javax.swing.JLabel jLabel8

  Helper label with hints for adding main courses.
- javax.swing.JLabel jLabel9

  Section title label for the MainCourse tab.
- javax.swing.JPanel jPanel1

  Main container panel of the window.
- javax.swing.JPanel jPanel2

  Panel that contains the Drinks controls and table.
- javax.swing.JPanel jPanel3

  Panel that contains the Appetizers controls and table.
- javax.swing.JPanel jPanel4

  Panel that contains the MainCourse controls and table.
- javax.swing.JScrollPane jScrollPane1

  Scroll pane wrapping the drinks table.
- javax.swing.JScrollPane jScrollPane2

  Scroll pane wrapping the appetizers table.
- javax.swing.JScrollPane jScrollPane3

  Scroll pane wrapping the main course table.
- javax.swing.JTabbedPane jTabbedPane1

  Tabbed pane containing the three product categories.
- javax.swing.JTable jTable1

  Table displaying drink items.
- javax.swing.JTable jTable2

  Table displaying appetizer items.
- javax.swing.JTable jTable3

  Table displaying main course items.

7.3.1 Detailed Description

Administrative window for managing products and prices.

```
Swing window that allows administrators to:
- view drinks, appetizers and main courses loaded from the backend.
- add new items to each category.
- update the price and name of existing items.

All data is obtained and persisted through the ProductService, which
internally uses ApiClient to call the REST API.
```

Definition at line 24 of file AdminProducts.java.

7.3.2 Constructor & Destructor Documentation

7.3.2.1 AdminProducts()

```
es.ull.esit.app.AdminProducts.AdminProducts ( )  [inline]
```

Constructor.

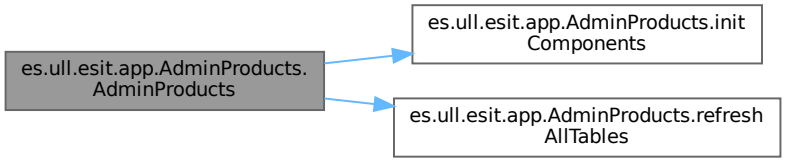
```
Creates the admin products window, initializes GUI components,
configures the table models and loads the initial data from the
backend service.
```

Definition at line 53 of file AdminProducts.java.

```
00053                           {
00054     initComponents();
00055
00056     // Initialize the Service Layer.
00057     ApiClient client = new ApiClient("http://localhost:8080");
00058     this.productService = new ProductService(client);
00059
00060     // Initialize table models and set shared headers.
00061     modelDrink = new DefaultTableModel();
00062     modelDrink.setColumnIdentifiers(columnNames);
00063     modelappetizers = new DefaultTableModel();
00064     modelappetizers.setColumnIdentifiers(columnNames);
00065     modelmaincourse = new DefaultTableModel();
00066     modelmaincourse.setColumnIdentifiers(columnNames);
00067
00068     // Link models to tables.
00069     jTable1.setModel(modelDrink);
00070     jTable2.setModel(modelappetizers);
00071     jTable3.setModel(modelmaincourse);
00072
00073     // Load initial data from backend.
00074     refreshAllTables();
00075   }
```

References es.ull.esit.app.AdminProducts.initComponents(), es.ull.esit.app.AdminProducts.jTable1, es.ull.esit.app.AdminProducts.jTa
es.ull.esit.app.AdminProducts.jTable3, and es.ull.esit.app.AdminProducts.refreshAllTables().

Here is the call graph for this function:



7.3.3 Member Function Documentation

7.3.3.1 initComponents()

```
void es.ull.esit.app.AdminProducts.initComponents ( )  [inline], [private]
```

Initializes GUI components.

```
Generated by the Form Editor: do not modify manually.
Creates labels, buttons, tabbed panes and tables for the three
product categories (drinks, appetizers, main courses) and the
navigation button "Go Back".
```

Definition at line 181 of file AdminProducts.java.

```
00181                                {
00182
00183     jPanel1 = new javax.swing.JPanel();
00184     jLabel1 = new javax.swing.JLabel();
00185     jTabbedPane1 = new javax.swing.JTabbedPane();
00186     jPanel2 = new javax.swing.JPanel();
00187     itemname = new javax.swing.JTextField();
00188     itemprice = new javax.swing.JTextField();
00189     jScrollPane1 = new javax.swing.JScrollPane();
00190     jTable1 = new javax.swing.JTable();
00191     jButton1 = new javax.swing.JButton();
00192     jButton2 = new javax.swing.JButton();
00193     jLabel2 = new javax.swing.JLabel();
00194     jLabel3 = new javax.swing.JLabel();
00195     jLabel4 = new javax.swing.JLabel();
00196     jPanel3 = new javax.swing.JPanel();
00197     jLabel5 = new javax.swing.JLabel();
00198     jLabel6 = new javax.swing.JLabel();
00199     jLabel7 = new javax.swing.JLabel();
00200     jScrollPane2 = new javax.swing.JScrollPane();
00201     jTable2 = new javax.swing.JTable();
00202     itemname1 = new javax.swing.JTextField();
00203     itemprice1 = new javax.swing.JTextField();
00204     jButton4 = new javax.swing.JButton();
00205     jButton5 = new javax.swing.JButton();
00206     jPanel4 = new javax.swing.JPanel();
00207     jLabel8 = new javax.swing.JLabel();
00208     jLabel9 = new javax.swing.JLabel();
00209     jLabel10 = new javax.swing.JLabel();
00210     jScrollPane3 = new javax.swing.JScrollPane();
00211     jTable3 = new javax.swing.JTable();
00212     itemname2 = new javax.swing.JTextField();
00213     itemprice2 = new javax.swing.JTextField();
00214     jButton6 = new javax.swing.JButton();
00215     jButton7 = new javax.swing.JButton();
00216     jButton3 = new javax.swing.JButton();
00217
00218     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00219     setTitle("Admin");
00220     setResizable(false);
00221
00222     jPanel1.setBackground(new java.awt.Color(248, 244, 230));
00223
00224     jLabel1.setBackground(new java.awt.Color(255, 153, 0));
00225     jLabel1.setFont(new java.awt.Font("Yu Gothic UI", 1, 36)); // NOI18N
00226     jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00227     jLabel1.setText("Items Prices Update Portal");
00228
00229     jTabbedPane1.setBackground(new java.awt.Color(248, 244, 230));
00230     jTabbedPane1.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));
00231     jTabbedPane1.setCursor(new java.awt.Cursor(java.awt.Cursor.DEFAULT_CURSOR));
00232     jTabbedPane1.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00233
00234     jPanel2.setBackground(new java.awt.Color(248, 244, 230));
00235
00236     itemname.setFont(new java.awt.Font("Yu Gothic UI", 0, 18)); // NOI18N
00237     itemname.setHorizontalAlignment(javax.swing.JTextField.CENTER);
00238     itemname.setBorder(javax.swing.BorderFactory.createTitledBorder(
00239         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Item Name",
00240         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00241         new java.awt.Font("Yu Gothic UI", 0, 18))); // NOI18N
00242
00243     itemprice.setFont(new java.awt.Font("Yu Gothic UI", 0, 18)); // NOI18N
00244     itemprice.setHorizontalAlignment(javax.swing.JTextField.CENTER);
```

```
00245     itemprice.setBorder(javax.swing.BorderFactory.createTitledBorder(
00246         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Item Price",
00247         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00248         new java.awt.Font("Yu Gothic UI", 0, 18))); // NOI18N
00249 
00250     jTable1.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00251     jTable1.setModel(new javax.swing.table.DefaultTableModel(
00252         new Object[][] {
00253             {},
00254             {},
00255             {},
00256             {}
00257         },
00258         new String[] {
00259 
00260         }));
00261     jTable1.setInheritsPopupMenu(true);
00262     jTable1.getTableHeader().setResizingAllowed(false);
00263     jTable1.getTableHeader().setReorderingAllowed(false);
00264     jTable1.addMouseListener(new java.awt.event.MouseAdapter() {
00265       public void mouseClicked(java.awt.event.MouseEvent evt) {
00266         jTable1MouseClicked(evt);
00267       }
00268     });
00269     jTable1.addKeyListener(new java.awt.event.KeyAdapter() {
00270       public void keyPressed(java.awt.event.KeyEvent evt) {
00271         jTable1KeyPressed(evt);
00272       }
00273     });
00274     jScrollPane1.setViewportView(jTable1);
00275 
00276     jButton1.setBackground(new java.awt.Color(255, 255, 255));
00277     jButton1.setFont(new java.awt.Font("Thonburi", 0, 18)); // NOI18N
00278     jButton1.setText("Update");
00279     jButton1.addActionListener(new java.awt.event.ActionListener() {
00280       public void actionPerformed(java.awt.event.ActionEvent evt) {
00281         jButton1ActionPerformed(evt);
00282       }
00283     });
00284 
00285     jButton2.setBackground(new java.awt.Color(255, 255, 255));
00286     jButton2.setFont(new java.awt.Font("Thonburi", 0, 18)); // NOI18N
00287     jButton2.setText("Add");
00288     jButton2.addActionListener(new java.awt.event.ActionListener() {
00289       public void actionPerformed(java.awt.event.ActionEvent evt) {
00290         jButton2ActionPerformed(evt);
00291       }
00292     });
00293 
00294     jLabel2.setBackground(new java.awt.Color(255, 153, 0));
00295     jLabel2.setFont(new java.awt.Font("Yu Gothic UI", 1, 25)); // NOI18N
00296     jLabel2.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00297     jLabel2.setText("Available Drinks");
00298 
00299     jLabel3.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00300     jLabel3.setText("Enter new drink information carefully to add.");
00301 
00302     jLabel4.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00303     jLabel4.setText("Please select the drink to update the price.");
00304 
00305     javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);
00306     jPanel2.setLayout(jPanel2Layout);
00307     jPanel2Layout.setHorizontalGroup(
00308         jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00309             .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
      jPanel2Layout.createSequentialGroup()
00310                 .addGap(0, 27, Short.MAX_VALUE)
00311                 .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00312                     .addGroup(jPanel2Layout.createSequentialGroup()
00313                         .addGap(24, 24, 24)
00314 
      .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
00315                             .addGroup(jPanel2Layout.createSequentialGroup()
00316                                 .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 91,
00317                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00318                                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
      109,
00319                                     Short.MAX_VALUE)
00320                                 .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 103,
00321                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00322                                 .addGap(8, 8, 8))
00323                             .addComponent(itemprice)
00324                             .addComponent(itemname, javax.swing.GroupLayout.DEFAULT_SIZE, 311,
      Short.MAX_VALUE)
00325                             .addComponent(jLabel3, javax.swing.GroupLayout.DEFAULT_SIZE,
00326                                 javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
00327                         .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED,
```

```
00328                           javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
00329
    .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00330                           .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 586,
00331                               javax.swing.GroupLayout.PREFERRED_SIZE)
00332                           .addComponent(jLabel4, javax.swing.GroupLayout.PREFERRED_SIZE, 408,
00333                               javax.swing.GroupLayout.PREFERRED_SIZE))
00334                       .addGap(115, 115, 115))
00335                   .addGroup(jPanel2Layout.createSequentialGroup()
00336                       .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE, 1111,
00337                           javax.swing.GroupLayout.PREFERRED_SIZE)
00338                       .addGap(18, 18, 18)))));
00339     jPanel2Layout.setVerticalGroup(
00340         jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00341             .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
    jPanel2Layout.createSequentialGroup()
00342                 .addContainerGap()
00343                 .addComponent(jLabel2)
00344                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 21,
    Short.MAX_VALUE)
00345
    .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00346                     .addComponent(jLabel3)
00347                     .addComponent(jLabel4))
00348                 .addGap(18, 18, 18)
00349                 .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00350                     .addGroup(jPanel2Layout.createSequentialGroup()
00351                         .addComponent(itemname, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00352                             javax.swing.GroupLayout.PREFERRED_SIZE)
00353                         .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
00354                         .addComponent(itemprice, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00355                             javax.swing.GroupLayout.PREFERRED_SIZE)
00356                         .addGap(37, 37, 37)
00357
    .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00358                           .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00359                               javax.swing.GroupLayout.PREFERRED_SIZE)
00360                           .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00361                               javax.swing.GroupLayout.PREFERRED_SIZE)))
00362                     .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 203,
00363                         javax.swing.GroupLayout.PREFERRED_SIZE))
00364                 .addGap(48, 48, 48)));
00365
00366     jTabbedPane1.addTab("Drinks", jPanel2);
00367
00368     jPanel3.setBackground(new java.awt.Color(248, 244, 230));
00369
00370     jLabel5.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00371     jLabel5.setText("Enter new appetizer information carefully to add.");
00372
00373     jLabel6.setBackground(new java.awt.Color(255, 153, 0));
00374     jLabel6.setFont(new java.awt.Font("Yu Gothic UI", 1, 25)); // NOI18N
00375     jLabel6.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00376     jLabel6.setText("Available Appetizer");
00377
00378     jLabel7.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00379     jLabel7.setText("Please select the appetizer to update the price.");
00380
00381     jTable2.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00382     jTable2.setModel(new javax.swing.table.DefaultTableModel(
00383         new Object[][] {
00384             {},
00385             {},
00386             {},
00387             {}
00388         },
00389         new String[] {
00390
00391         }));
00392     jTable2.addMouseListener(new java.awt.event.MouseAdapter() {
00393       public void mouseClicked(java.awt.event.MouseEvent evt) {
00394         jTable2MouseClicked(evt);
00395       }
00396     });
00397     jScrollPane2.setViewportView(jTable2);
00398
00399     itemname1.setFont(new java.awt.Font("Yu Gothic UI", 0, 18)); // NOI18N
00400     itemname1.setHorizontalAlignment(javax.swing.JTextField.CENTER);
00401     itemname1.setBorder(javax.swing.BorderFactory.createTitledBorder(
00402         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Item Name",
00403         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00404         new java.awt.Font("Yu Gothic UI", 0, 18))); // NOI18N
00405
00406     itemprice1.setFont(new java.awt.Font("Yu Gothic UI", 0, 18)); // NOI18N
00407     itemprice1.setHorizontalAlignment(javax.swing.JTextField.CENTER);
00408     itemprice1.setBorder(javax.swing.BorderFactory.createTitledBorder(
00409         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Item Price",
```

```
00410         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00411         new java.awt.Font("Yu Gothic UI", 0, 18))); // NOI18N
00412
00413     jButton4.setBackground(new java.awt.Color(255, 255, 255));
00414     jButton4.setFont(new java.awt.Font("Thonburi", 0, 18)); // NOI18N
00415     jButton4.setText("Update");
00416     jButton4.addActionListener(new java.awt.event.ActionListener() {
00417       public void actionPerformed(java.awt.event.ActionEvent evt) {
00418         jButton4ActionPerformed(evt);
00419       }
00420     });
00421
00422     jButton5.setBackground(new java.awt.Color(255, 255, 255));
00423     jButton5.setFont(new java.awt.Font("Thonburi", 0, 18)); // NOI18N
00424     jButton5.setText("Add");
00425     jButton5.addActionListener(new java.awt.event.ActionListener() {
00426       public void actionPerformed(java.awt.event.ActionEvent evt) {
00427         jButton5ActionPerformed(evt);
00428       }
00429     });
00430
00431     javax.swing.GroupLayout jPanel3Layout = new javax.swing.GroupLayout(jPanel3);
00432     jPanel3.setLayout(jPanel3Layout);
00433     jPanel3Layout.setHorizontalGroup(
00434         jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00435             .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
      jPanel3Layout.createSequentialGroup()
00436                 .addGap(0, 27, Short.MAX_VALUE)
00437                 .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00438                     .addGroup(jPanel3Layout.createSequentialGroup()
00439                         .addGap(24, 24, 24)
00440
      .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
00441                             .addGroup(jPanel3Layout.createSequentialGroup()
00442                                 .addComponent(jButton5, javax.swing.GroupLayout.PREFERRED_SIZE, 91,
00443                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00444                                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
00445                                     javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
00446                                 .addComponent(jButton4, javax.swing.GroupLayout.PREFERRED_SIZE, 103,
00447                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00448                                 .addGap(8, 8, 8))
00449                             .addComponent(itemprice1)
00450                             .addComponent(itemname1)
00451                             .addComponent(jLabel5, javax.swing.GroupLayout.DEFAULT_SIZE,
00452                                 javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
00453                         .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED,
00454                             javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
00455
      .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00456                             .addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED_SIZE, 586,
00457                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00458                             .addComponent(jLabel7, javax.swing.GroupLayout.PREFERRED_SIZE, 408,
00459                                 javax.swing.GroupLayout.PREFERRED_SIZE))
00460                         .addGap(115, 115, 115))
00461                     .addGroup(jPanel3Layout.createSequentialGroup()
00462                         .addComponent(jLabel6, javax.swing.GroupLayout.PREFERRED_SIZE, 1111,
00463                             javax.swing.GroupLayout.PREFERRED_SIZE)
00464                         .addGap(18, 18, 18)))));
00465     jPanel3Layout.setVerticalGroup(
00466         jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00467             .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
      jPanel3Layout.createSequentialGroup()
00468                 .addContainerGap()
00469                 .addComponent(jLabel6)
00470                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 21,
      Short.MAX_VALUE)
00471
      .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00472                     .addComponent(jLabel5)
00473                     .addComponent(jLabel7))
00474                 .addGap(18, 18, 18)
00475                 .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00476                     .addGroup(jPanel3Layout.createSequentialGroup()
00477                         .addComponent(itemname1, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00478                             javax.swing.GroupLayout.PREFERRED_SIZE)
00479                         .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
00480                         .addComponent(itemprice1, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00481                             javax.swing.GroupLayout.PREFERRED_SIZE)
00482                         .addGap(37, 37, 37)
00483
      .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00484                             .addComponent(jButton4, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00485                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00486                             .addComponent(jButton5, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00487                                 javax.swing.GroupLayout.PREFERRED_SIZE)))
00488                     .addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED_SIZE, 203,
00489                         javax.swing.GroupLayout.PREFERRED_SIZE))
```

```
00490                 .addGap(48, 48, 48)));
00491
00492     jTabbedPane1.addTab("Appetizers", jPanel3);
00493
00494     jPanel4.setBackground(new java.awt.Color(248, 244, 230));
00495     jPanel4.setFont(new java.awt.Font("Yu Gothic UI", 0, 18)); // NOI18N
00496
00497     jLabel8.setBackground(new java.awt.Color(248, 244, 230));
00498     jLabel8.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00499     jLabel8.setText("Enter new item information carefully to add.");
00500
00501     jLabel9.setBackground(new java.awt.Color(255, 153, 0));
00502     jLabel9.setFont(new java.awt.Font("Yu Gothic UI", 1, 25)); // NOI18N
00503     jLabel9.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00504     jLabel9.setText("Avaliable MainCourse");
00505
00506     jLabel10.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00507     jLabel10.setText("Please select the item to update the price.");
00508
00509     jTable3.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00510     jTable3.setModel(new javax.swing.table.DefaultTableModel(
00511         new Object[][] {
00512             {},
00513             {},
00514             {},
00515             {}
00516         },
00517         new String[] {
00518
00519         }));
00520     jTable3.addMouseListener(new java.awt.event.MouseAdapter() {
00521       public void mouseClicked(java.awt.event.MouseEvent evt) {
00522         jTable3MouseClicked(evt);
00523       }
00524     });
00525     jScrollPane3.setViewportView(jTable3);
00526
00527     itemname2.setFont(new java.awt.Font("Yu Gothic UI", 0, 18)); // NOI18N
00528     itemname2.setHorizontalAlignment(javax.swing.JTextField.CENTER);
00529     itemname2.setBorder(javax.swing.BorderFactory.createTitledBorder(
00530         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Item Name",
00531         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00532         new java.awt.Font("Yu Gothic UI", 0, 18))); // NOI18N
00533
00534     itemprice2.setFont(new java.awt.Font("Yu Gothic UI", 0, 18)); // NOI18N
00535     itemprice2.setHorizontalAlignment(javax.swing.JTextField.CENTER);
00536     itemprice2.setBorder(javax.swing.BorderFactory.createTitledBorder(
00537         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Item Price",
00538         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00539         new java.awt.Font("Yu Gothic UI", 0, 18))); // NOI18N
00540
00541     jButton6.setBackground(new java.awt.Color(255, 255, 255));
00542     jButton6.setFont(new java.awt.Font("Thonburi", 0, 18)); // NOI18N
00543     jButton6.setText("Update");
00544     jButton6.addActionListener(new java.awt.event.ActionListener() {
00545       public void actionPerformed(java.awt.event.ActionEvent evt) {
00546         jButton6ActionPerformed(evt);
00547       }
00548     });
00549
00550     jButton7.setBackground(new java.awt.Color(255, 255, 255));
00551     jButton7.setFont(new java.awt.Font("Thonburi", 0, 18)); // NOI18N
00552     jButton7.setText("Add");
00553     jButton7.addActionListener(new java.awt.event.ActionListener() {
00554       public void actionPerformed(java.awt.event.ActionEvent evt) {
00555         jButton7ActionPerformed(evt);
00556       }
00557     });
00558
00559     javax.swing.GroupLayout jPanel4Layout = new javax.swing.GroupLayout(jPanel4);
00560     jPanel4.setLayout(jPanel4Layout);
00561     jPanel4Layout.setHorizontalGroup(
00562         jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00563             .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
      jPanel4Layout.createSequentialGroup()
00564                 .addGap(0, 27, Short.MAX_VALUE)
00565                 .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00566                     .addGroup(jPanel4Layout.createSequentialGroup()
00567                         .addGap(24, 24, 24)
00568
      .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
00569                             .addGroup(jPanel4Layout.createSequentialGroup()
00570                                 .addComponent(jButton7, javax.swing.GroupLayout.PREFERRED_SIZE, 91,
00571                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00572                                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
      109,
00573                                     Short.MAX_VALUE)
```

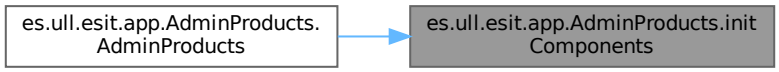
```
00574                                   .addComponent(jButton6, javax.swing.GroupLayout.PREFERRED_SIZE, 103,
00575                                       javax.swing.GroupLayout.PREFERRED_SIZE)
00576                                   .addGap(8, 8, 8))
00577                               .addComponent(itemprice2)
00578                               .addComponent(itemname2, javax.swing.GroupLayout.DEFAULT_SIZE, 311,
      Short.MAX_VALUE)
00579                               .addComponent(jLabel8, javax.swing.GroupLayout.DEFAULT_SIZE,
00580                                   javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
00581                           .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED,
00582                               javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
00583
      .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00584                               .addComponent(jScrollPane3, javax.swing.GroupLayout.PREFERRED_SIZE, 586,
00585                                   javax.swing.GroupLayout.PREFERRED_SIZE)
00586                               .addComponent(jLabel10, javax.swing.GroupLayout.PREFERRED_SIZE, 408,
00587                                   javax.swing.GroupLayout.PREFERRED_SIZE))
00588                           .addGap(115, 115, 115))
00589                       .addGroup(jPanel4Layout.createSequentialGroup()
00590                           .addComponent(jLabel9, javax.swing.GroupLayout.PREFERRED_SIZE, 1111,
00591                               javax.swing.GroupLayout.PREFERRED_SIZE)
00592                           .addGap(18, 18, 18)))));
00593       jPanel4Layout.setVerticalGroup(
00594           jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00595               .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
      jPanel4Layout.createSequentialGroup()
00596                   .addContainerGap()
00597                   .addComponent(jLabel9)
00598                   .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 21,
      Short.MAX_VALUE)
00599
      .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00600                       .addComponent(jLabel8)
00601                       .addComponent(jLabel10))
00602                   .addGap(18, 18, 18)
00603                   .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00604                       .addGroup(jPanel4Layout.createSequentialGroup()
00605                           .addComponent(itemname2, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00606                               javax.swing.GroupLayout.PREFERRED_SIZE)
00607                           .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
00608                           .addComponent(itemprice2, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00609                               javax.swing.GroupLayout.PREFERRED_SIZE)
00610                           .addGap(37, 37, 37)
00611
      .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00612                               .addComponent(jButton6, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00613                                   javax.swing.GroupLayout.PREFERRED_SIZE)
00614                               .addComponent(jButton7, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00615                                   javax.swing.GroupLayout.PREFERRED_SIZE)))
00616                       .addComponent(jScrollPane3, javax.swing.GroupLayout.PREFERRED_SIZE, 203,
00617                           javax.swing.GroupLayout.PREFERRED_SIZE))
00618                   .addGap(48, 48, 48)));
00619
00620       jTabbedPane1.addTab("MainCourse", jPanel4);
00621
00622       jButton3.setBackground(new java.awt.Color(255, 255, 255));
00623       jButton3.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00624       jButton3.setText("Go Back");
00625       jButton3.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));
00626       jButton3.addActionListener(new java.awt.event.ActionListener() {
00627         public void actionPerformed(java.awt.event.ActionEvent evt) {
00628           jButton3ActionPerformed(evt);
00629         }
00630       });
00631
00632       javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
00633       jPanel1.setLayout(jPanel1Layout);
00634       jPanel1Layout.setHorizontalGroup(
00635           jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00636               .addGroup(jPanel1Layout.createSequentialGroup()
00637                   .addGap(25, 25, 25)
00638                   .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00639                       .addGroup(jPanel1Layout.createSequentialGroup()
00640                           .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 122,
00641                               javax.swing.GroupLayout.PREFERRED_SIZE)
00642                           .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
00643                               javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
00644                           .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 689,
00645                               javax.swing.GroupLayout.PREFERRED_SIZE)
00646                           .addGap(218, 218, 218))
00647                       .addGroup(jPanel1Layout.createSequentialGroup()
00648                           .addComponent(jTabbedPane1, javax.swing.GroupLayout.PREFERRED_SIZE,
00649                               javax.swing.GroupLayout.DEFAULT_SIZE,
      javax.swing.GroupLayout.PREFERRED_SIZE)
00650                           .addContainerGap(29, Short.MAX_VALUE)))));
00651       jPanel1Layout.setVerticalGroup(
00652           jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00653               .addGroup(jPanel1Layout.createSequentialGroup()
```

```
00654                 .addContainerGap()
00655
      .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
00656                     .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 60,
00657                         javax.swing.GroupLayout.PREFERRED_SIZE)
00658                     .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 37,
00659                         javax.swing.GroupLayout.PREFERRED_SIZE))
00660                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED,
00661                     javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
00662                 .addComponent(jTabbedPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 392,
00663                     javax.swing.GroupLayout.PREFERRED_SIZE)
00664                 .addGap(29, 29, 29)));
00665
00666     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00667     getContentPane().setLayout(layout);
00668     layout.setHorizontalGroup(
00669         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00670             .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00671                 Short.MAX_VALUE));
00672     layout.setVerticalGroup(
00673         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00674             .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
00675                 .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00676                     javax.swing.GroupLayout.PREFERRED_SIZE)
00677                 .addGap(0, 0, Short.MAX_VALUE)));
00678
00679     pack();
00680     setLocationRelativeTo(null);
00681   }// </editor-fold>//GEN-END:initComponents
```

References es.ull.esit.app.AdminProducts.itemname, es.ull.esit.app.AdminProducts.itemname1, es.ull.esit.app.AdminProducts.itemna
es.ull.esit.app.AdminProducts.itemprice, es.ull.esit.app.AdminProducts.itemprice1, es.ull.esit.app.AdminProducts.itemprice2,
es.ull.esit.app.AdminProducts.jButton1, es.ull.esit.app.AdminProducts.jButton2, es.ull.esit.app.AdminProducts.jButton3,
es.ull.esit.app.AdminProducts.jButton4, es.ull.esit.app.AdminProducts.jButton5, es.ull.esit.app.AdminProducts.jButton6,
es.ull.esit.app.AdminProducts.jButton7, es.ull.esit.app.AdminProducts.jLabel1, es.ull.esit.app.AdminProducts.jLabel10,
es.ull.esit.app.AdminProducts.jLabel2, es.ull.esit.app.AdminProducts.jLabel3, es.ull.esit.app.AdminProducts.jLabel4,
es.ull.esit.app.AdminProducts.jLabel5, es.ull.esit.app.AdminProducts.jLabel6, es.ull.esit.app.AdminProducts.jLabel7,
es.ull.esit.app.AdminProducts.jLabel8, es.ull.esit.app.AdminProducts.jLabel9, es.ull.esit.app.AdminProducts.jPanel1,
es.ull.esit.app.AdminProducts.jPanel2, es.ull.esit.app.AdminProducts.jPanel3, es.ull.esit.app.AdminProducts.jPanel4,
es.ull.esit.app.AdminProducts.jScrollPane1, es.ull.esit.app.AdminProducts.jScrollPane2, es.ull.esit.app.AdminProducts.jScrollPane3,
es.ull.esit.app.AdminProducts.jTabbedPane1, es.ull.esit.app.AdminProducts.jTable1, es.ull.esit.app.AdminProducts.jTable2,
and es.ull.esit.app.AdminProducts.jTable3.

Referenced by es.ull.esit.app.AdminProducts.AdminProducts().

Here is the caller graph for this function:



7.3.3.2 jButton1ActionPerformed()

```
void es.ull.esit.app.AdminProducts.jButton1ActionPerformed (
             java.awt.event.ActionEvent evt )  [inline], [private]
```

Action handler for the "Update" button in the Drinks tab.

```
   Sends the modified drink data (name and price) to the backend,
   resets the selection and reloads the drinks table.
```

Parameters

evt	[java.awt.event.ActionEvent] Event triggered by the button click.

Definition at line 896 of file AdminProducts.java.

```
00896                                                                    {//
      GEN-FIRST:event_jButton1ActionPerformed
00897     String name = itemname.getText();
00898     String price = itemprice.getText();
00899
00900     new Thread(() -> {
00901       try {
00902         productService.updateDrink(selectedDrinkID, name, price);
00903         SwingUtilities.invokeLater(() -> {
00904           JOptionPane.showMessageDialog(this, "Drink updated successfully.");
00905           itemname.setText("");
00906           itemprice.setText("");
00907           selectedDrinkID = null;
00908           loadDrinks();
00909         });
00910       } catch (Exception ex) {
00911         SwingUtilities
00912             .invokeLater(() -> JOptionPane.showMessageDialog(this, "Error updating drink: " +
      ex.getMessage()));
00913       }
00914     }).start();
00915   }// GEN-LAST:event_jButton1ActionPerformed
```

7.3.3.3 jButton2ActionPerformed()

```
void es.ull.esit.app.AdminProducts.jButton2ActionPerformed (
             java.awt.event.ActionEvent evt )  [inline], [private]
```

Action handler for the "Add" button in the Drinks tab.

```
Steps:
- Reads the name and price from the text fields.
- Uses ProductService to create a new Drink in the backend.
- Shows a confirmation dialog and refreshes the drinks table.
```

Parameters

evt	[java.awt.event.ActionEvent] Event triggered by the button click.

Definition at line 693 of file AdminProducts.java.

```
00693                                                                    {//
      GEN-FIRST:event_jButton2ActionPerformed
00694     String name = itemname.getText();
00695     String price = itemprice.getText();
00696
00697     new Thread(() -> {
00698       try {
00699         productService.addDrink(name, price);
00700         SwingUtilities.invokeLater(() -> {
00701           JOptionPane.showMessageDialog(this, "Drink Added Successfully.");
00702           itemname.setText("");
00703           itemprice.setText("");
00704           loadDrinks();
00705         });
00706       } catch (Exception ex) {
00707         SwingUtilities.invokeLater(() -> JOptionPane.showMessageDialog(this, "Error adding drink: " +
      ex.getMessage()));
00708       }
00709     }).start();
00710   }// GEN-LAST:event_jButton2ActionPerformed
```

7.3.3.4 jButton3ActionPerformed()

```
void es.ull.esit.app.AdminProducts.jButton3ActionPerformed (
            java.awt.event.ActionEvent evt )  [inline], [private]
```

Action handler for the "Go Back" button.

```
Closes the current AdminProducts window and returns to the
AdminLogin window.
```

Parameters

evt	[java.awt.event.ActionEvent] Event triggered by the button click.

Definition at line 720 of file AdminProducts.java.

```
00720                                                                    {//
      GEN-FIRST:event_jButton3ActionPerformed
00721     new AdminLogin().setVisible(true);
00722     this.dispose();
00723   }// GEN-LAST:event_jButton3ActionPerformed
```

7.3.3.5 jButton4ActionPerformed()

```
void es.ull.esit.app.AdminProducts.jButton4ActionPerformed (
            java.awt.event.ActionEvent evt )  [inline], [private]
```

Action handler for the "Update" button in the Appetizers tab.

```
Updates the selected appetizer in the backend and refreshes the
appetizers table.
```

Parameters

evt	[java.awt.event.ActionEvent] Event triggered by the button click.

Definition at line 925 of file AdminProducts.java.

```
00925                                                                    {//
      GEN-FIRST:event_jButton4ActionPerformed
00926     String name = itemname1.getText();
00927     String price = itemprice1.getText();
00928
00929     new Thread(() -> {
00930       try {
00931         productService.updateAppetizer(selectedAppetizerID, name, price);
00932         SwingUtilities.invokeLater(() -> {
00933           JOptionPane.showMessageDialog(this, "Appetizer updated successfully.");
00934           itemname1.setText("");
00935           itemprice1.setText("");
00936           selectedAppetizerID = null;
00937           loadAppetizer();
00938         });
00939       } catch (Exception ex) {
00940         SwingUtilities
00941             .invokeLater(() -> JOptionPane.showMessageDialog(this, "Error updating appetizer: " +
      ex.getMessage()));
00942       }
00943     }).start();
00944   }// GEN-LAST:event_jButton4ActionPerformed
```

7.3.3.6 jButton5ActionPerformed()

```
void es.ull.esit.app.AdminProducts.jButton5ActionPerformed (
            java.awt.event.ActionEvent evt )  [inline], [private]
```

Action handler for the "Add" button in the Appetizers tab.

```
Creates a new appetizer in the backend using the data entered
by the administrator and reloads the appetizers table.
```

Parameters

evt	[java.awt.event.ActionEvent] Event triggered by the button click.

Definition at line 733 of file AdminProducts.java.

```
00733                                                                {//
      GEN-FIRST:event_jButton5ActionPerformed
00734     String name = itemname1.getText();
00735     String price = itemprice1.getText();
00736
00737     new Thread(() -> {
00738       try {
00739         productService.addAppetizer(name, price);
00740         SwingUtilities.invokeLater(() -> {
00741           JOptionPane.showMessageDialog(this, "Appetizer Added Successfully.");
00742           itemname1.setText("");
00743           itemprice1.setText("");
00744           loadAppetizer();
00745         });
00746       } catch (Exception ex) {
00747         SwingUtilities
00748             .invokeLater(() -> JOptionPane.showMessageDialog(this, "Error adding appetizer: " +
      ex.getMessage()));
00749       }
00750     }).start();
00751   }// GEN-LAST:event_jButton5ActionPerformed
```

7.3.3.7 jButton6ActionPerformed()

```
void es.ull.esit.app.AdminProducts.jButton6ActionPerformed (
            java.awt.event.ActionEvent evt )  [inline], [private]
```

Action handler for the "Update" button in the MainCourse tab.

```
Updates the selected main course data in the backend and reloads
the main course list.
```

Parameters

evt	[java.awt.event.ActionEvent] Event triggered by the button click.

Definition at line 954 of file AdminProducts.java.

```
00954                                                                {//
      GEN-FIRST:event_jButton6ActionPerformed
00955     String name = itemname2.getText();
00956     String price = itemprice2.getText();
00957
00958     new Thread(() -> {
00959       try {
00960         productService.updateMainCourse(selectedMainCourseID, name, price);
```

```
00961         SwingUtilities.invokeLater(() -> {
00962           JOptionPane.showMessageDialog(this, "Main course updated successfully.");
00963           itemname2.setText("");
00964           itemprice2.setText("");
00965           selectedMainCourseID = null;
00966           loadmainCourse();
00967         });
00968       } catch (Exception ex) {
00969         SwingUtilities
00970             .invokeLater(() -> JOptionPane.showMessageDialog(this, "Error updating main course: " +
      ex.getMessage()));
00971       }
00972     }).start();
00973   }// GEN-LAST:event_jButton6ActionPerformed
```

7.3.3.8 jButton7ActionPerformed()

```
void es.ull.esit.app.AdminProducts.jButton7ActionPerformed (
             java.awt.event.ActionEvent evt )  [inline], [private]
```

Action handler for the "Add" button in the MainCourse tab.

Sends a new main course to the backend and refreshes the corresponding table.

Parameters

evt	[java.awt.event.ActionEvent] Event triggered by the button click.

Definition at line 761 of file AdminProducts.java.

```
00761                                                                {//
      GEN-FIRST:event_jButton7ActionPerformed
00762     String name = itemname2.getText();
00763     String price = itemprice2.getText();
00764 
00765     new Thread(() -> {
00766       try {
00767         productService.addMainCourse(name, price);
00768         SwingUtilities.invokeLater(() -> {
00769           JOptionPane.showMessageDialog(this, "Main Course Added Successfully.");
00770           itemname2.setText("");
00771           itemprice2.setText("");
00772           loadmainCourse();
00773         });
00774       } catch (Exception ex) {
00775         SwingUtilities
00776             .invokeLater(() -> JOptionPane.showMessageDialog(this, "Error adding main course: " +
      ex.getMessage()));
00777       }
00778     }).start();
00779   }// GEN-LAST:event_jButton7ActionPerformed
```

7.3.3.9 jTable1KeyPressed()

```
void es.ull.esit.app.AdminProducts.jTable1KeyPressed (
             java.awt.event.KeyEvent evt )  [inline], [private]
```

Key handler for the drinks table.

Currently not used. Kept for potential future keyboard handling when navigating drink rows.

Parameters

evt	[java.awt.event.KeyEvent] Key event generated by the table.

Definition at line 789 of file AdminProducts.java.

```
00789                                                    {// GEN-FIRST:event_jTable1KeyPressed
00790     // No action needed for key press in this version.
00791   }// GEN-LAST:event_jTable1KeyPressed
```

7.3.3.10 jTable1MouseClicked()

```
void es.ull.esit.app.AdminProducts.jTable1MouseClicked (
            java.awt.event.MouseEvent evt )  [inline], [private]
```

Mouse handler for the drinks table.

```
When a row is clicked:
- Saves the selected drink ID.
- Requests detailed data from the backend.
- Fills the name and price fields so they can be edited.
```

Parameters

evt	[java.awt.event.MouseEvent] Mouse event generated by the table.

Definition at line 803 of file AdminProducts.java.

```
00803                                                    {//
      GEN-FIRST:event_jTable1MouseClicked
00804     int row = jTable1.getSelectedRow();
00805     if (row == -1)
00806       return;
00807
00808     String idStr = jTable1.getValueAt(row, 0).toString();
00809     selectedDrinkID = Long.valueOf(idStr);
00810     System.out.println("Selected Drink ID: " + selectedDrinkID);
00811
00812     new Thread(() -> {
00813       try {
00814         Drink drink = productService.getDrinkById(selectedDrinkID);
00815         SwingUtilities.invokeLater(() -> {
00816           itemname.setText(drink.getItemDrinks());
00817           itemprice.setText(String.valueOf(drink.getDrinksPrice()));
00818         });
00819       } catch (Exception ex) {
00820         SwingUtilities
00821             .invokeLater(() -> JOptionPane.showMessageDialog(null, "Error loading drink details: " +
      ex.getMessage()));
00822       }
00823     }).start();
00824   }// GEN-LAST:event_jTable1MouseClicked
```

7.3.3.11 jTable2MouseClicked()

```
void es.ull.esit.app.AdminProducts.jTable2MouseClicked (
            java.awt.event.MouseEvent evt )  [inline], [private]
```

Mouse handler for the appetizers table.

```
Loads the selected appetizer from the backend and populates
the corresponding text fields to allow editing.
```

Parameters

evt	[java.awt.event.MouseEvent] Mouse event generated by the table.

Definition at line 834 of file AdminProducts.java.

```
00834                                                    {//
      GEN-FIRST:event_jTable2MouseClicked
00835     int row = jTable2.getSelectedRow();
00836     if (row == -1)
00837       return;
00838
00839     String idStr = jTable2.getValueAt(row, 0).toString();
00840     selectedAppetizerID = Long.valueOf(idStr);
00841     System.out.println("Selected Appetizer ID: " + selectedAppetizerID);
00842
00843     new Thread(() -> {
00844       try {
00845         Appetizer appetizer = productService.getAppetizerById(selectedAppetizerID);
00846         SwingUtilities.invokeLater(() -> {
00847           itemname1.setText(appetizer.getItemAppetizers());
00848           itemprice1.setText(String.valueOf(appetizer.getAppetizersPrice()));
00849         });
00850       } catch (Exception ex) {
00851         SwingUtilities.invokeLater(
00852             () -> JOptionPane.showMessageDialog(null, "Error loading appetizer details: " +
      ex.getMessage()));
00853       }
00854     }).start();
00855   }// GEN-LAST:event_jTable2MouseClicked
```

7.3.3.12 jTable3MouseClicked()

```
void es.ull.esit.app.AdminProducts.jTable3MouseClicked (
             java.awt.event.MouseEvent evt )  [inline], [private]
```

Mouse handler for the main course table.

```
Loads the selected main course from the backend and puts its
data into the editable text fields.
```

Parameters

evt	[java.awt.event.MouseEvent] Mouse event generated by the table.

Definition at line 865 of file AdminProducts.java.

```
00865                                                    {//
      GEN-FIRST:event_jTable3MouseClicked
00866     int row = jTable3.getSelectedRow();
00867     if (row == -1)
00868       return;
00869
00870     String idStr = jTable3.getValueAt(row, 0).toString();
00871     selectedMainCourseID = Long.valueOf(idStr);
00872     System.out.println("Selected Main Course ID: " + selectedMainCourseID);
00873
00874     new Thread(() -> {
00875       try {
00876         MainCourse mainCourse = productService.getMainCourseById(selectedMainCourseID);
00877         SwingUtilities.invokeLater(() -> {
00878           itemname2.setText(mainCourse.getItemFood());
00879           itemprice2.setText(String.valueOf(mainCourse.getFoodPrice()));
00880         });
00881       } catch (Exception ex) {
00882         SwingUtilities.invokeLater(
00883             () -> JOptionPane.showMessageDialog(null, "Error loading main course details: " +
      ex.getMessage()));
00884       }
00885     }).start();
00886   }// GEN-LAST:event_jTable3MouseClicked
```

7.3.3.13 main()

```
static void es.ull.esit.app.AdminProducts.main (
            String args[] )  [inline], [static]
```

Standalone entry point for testing the AdminProducts window.

```
Sets the Nimbus look and feel if available and shows an instance
of AdminProducts. In the normal application flow, this window is
opened from AdminLogin after authentication.
```

Parameters

args	[String[]] Command line arguments (not used).

Definition at line 984 of file AdminProducts.java.

```
00984                                       {
00985     try {
00986       for (javax.swing.UIManager.LookAndFeelInfo info :
      javax.swing.UIManager.getInstalledLookAndFeels()) {
00987         if ("Nimbus".equals(info.getName())) {
00988           javax.swing.UIManager.setLookAndFeel(info.getClassName());
00989           break;
00990         }
00991       }
00992     } catch (ClassNotFoundException ex) {
00993
      java.util.logging.Logger.getLogger(AdminProducts.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
00994     } catch (InstantiationException ex) {
00995
      java.util.logging.Logger.getLogger(AdminProducts.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
00996     } catch (IllegalAccessException ex) {
00997
      java.util.logging.Logger.getLogger(AdminProducts.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
00998     } catch (javax.swing.UnsupportedLookAndFeelException ex) {
00999
      java.util.logging.Logger.getLogger(AdminProducts.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
01000     }
01001     // </editor-fold>
01002
01003     /* Create and display the form */
01004     java.awt.EventQueue.invokeLater(new Runnable() {
01005       public void run() {
01006         new AdminProducts().setVisible(true);
01007       }
01008     });
01009   }
```

7.3.3.14 refreshAllTables()

```
void es.ull.esit.app.AdminProducts.refreshAllTables ( )  [inline], [private]
```

Reloads all product tables.

```
Helper method that triggers asynchronous loading of drinks,
appetizers and main courses from the server.
```

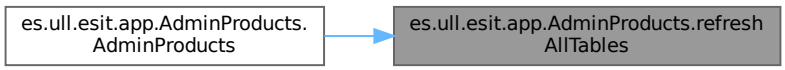
Definition at line 83 of file AdminProducts.java.

```
00083                                   {
00084     loadDrinks();
00085     loadAppetizer();
```

```
00086     loadmainCourse();
00087   }
```

Referenced by es.ull.esit.app.AdminProducts.AdminProducts().

Here is the caller graph for this function:



7.3.4 Member Data Documentation

7.3.4.1 itemname

```
javax.swing.JTextField es.ull.esit.app.AdminProducts.itemname  [private]
```

Text field for the drink name (Drinks tab).

Definition at line 1013 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.2 itemname1

```
javax.swing.JTextField es.ull.esit.app.AdminProducts.itemname1  [private]
```

Text field for the appetizer name (Appetizers tab).

Definition at line 1015 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.3 itemname2

```
javax.swing.JTextField es.ull.esit.app.AdminProducts.itemname2  [private]
```

Text field for the main course name (MainCourse tab).

Definition at line 1017 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.4 itemprice

```
javax.swing.JTextField es.ull.esit.app.AdminProducts.itemprice  [private]
```

Text field for the drink price (Drinks tab).

Definition at line 1019 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.5 itemprice1

```
javax.swing.JTextField es.ull.esit.app.AdminProducts.itemprice1  [private]
```

Text field for the appetizer price (Appetizers tab).

Definition at line 1021 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.6 itemprice2

```
javax.swing.JTextField es.ull.esit.app.AdminProducts.itemprice2  [private]
```

Text field for the main course price (MainCourse tab).

Definition at line 1023 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.7 jButton1

```
javax.swing.JButton es.ull.esit.app.AdminProducts.jButton1  [private]
```

Button to update an existing drink.

Definition at line 1025 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.8 jButton2

```
javax.swing.JButton es.ull.esit.app.AdminProducts.jButton2  [private]
```

Button to add a new drink.

Definition at line 1027 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.9 jButton3

```
javax.swing.JButton es.ull.esit.app.AdminProducts.jButton3  [private]
```

Button to go back to the admin main window.

Definition at line 1029 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.10 jButton4

```
javax.swing.JButton es.ull.esit.app.AdminProducts.jButton4  [private]
```

Button to update an existing appetizer.

Definition at line 1031 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.11 jButton5

```
javax.swing.JButton es.ull.esit.app.AdminProducts.jButton5  [private]
```

Button to add a new appetizer.

Definition at line 1033 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.12 jButton6

```
javax.swing.JButton es.ull.esit.app.AdminProducts.jButton6  [private]
```

Button to update an existing main course.

Definition at line 1035 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.13 jButton7

```
javax.swing.JButton es.ull.esit.app.AdminProducts.jButton7  [private]
```

Button to add a new main course.

Definition at line 1037 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.14 jLabel1

```
javax.swing.JLabel es.ull.esit.app.AdminProducts.jLabel1  [private]
```

Main title label ("Items Prices Update Portal").

Definition at line 1039 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.15 jLabel10

```
javax.swing.JLabel es.ull.esit.app.AdminProducts.jLabel10  [private]
```

Helper label used in the MainCourse tab.

Definition at line 1041 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.16 jLabel2

```
javax.swing.JLabel es.ull.esit.app.AdminProducts.jLabel2  [private]
```

Section title label for the Drinks tab.

Definition at line 1043 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.17 jLabel3

```
javax.swing.JLabel es.ull.esit.app.AdminProducts.jLabel3  [private]
```

Helper label with hints for adding drinks.

Definition at line 1045 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.18 jLabel4

```
javax.swing.JLabel es.ull.esit.app.AdminProducts.jLabel4  [private]
```

Helper label with hints for updating drinks.

Definition at line 1047 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.19 jLabel5

```
javax.swing.JLabel es.ull.esit.app.AdminProducts.jLabel5  [private]
```

Helper label with hints for adding appetizers.

Definition at line 1049 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.20 jLabel6

```
javax.swing.JLabel es.ull.esit.app.AdminProducts.jLabel6  [private]
```

Section title label for the Appetizers tab.

Definition at line 1051 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.21 jLabel7

```
javax.swing.JLabel es.ull.esit.app.AdminProducts.jLabel7  [private]
```

Helper label with hints for updating appetizers.

Definition at line 1053 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.22 jLabel8

```
javax.swing.JLabel es.ull.esit.app.AdminProducts.jLabel8  [private]
```

Helper label with hints for adding main courses.

Definition at line 1055 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.23 jLabel9

```
javax.swing.JLabel es.ull.esit.app.AdminProducts.jLabel9  [private]
```

Section title label for the MainCourse tab.

Definition at line 1057 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.24 jPanel1

```
javax.swing.JPanel es.ull.esit.app.AdminProducts.jPanel1  [private]
```

Main container panel of the window.

Definition at line 1059 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.25 jPanel2

```
javax.swing.JPanel es.ull.esit.app.AdminProducts.jPanel2  [private]
```

Panel that contains the Drinks controls and table.

Definition at line 1061 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.26 jPanel3

```
javax.swing.JPanel es.ull.esit.app.AdminProducts.jPanel3  [private]
```

Panel that contains the Appetizers controls and table.

Definition at line 1063 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.27 jPanel4

```
javax.swing.JPanel es.ull.esit.app.AdminProducts.jPanel4  [private]
```

Panel that contains the MainCourse controls and table.

Definition at line 1065 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.28 jScrollPane1

```
javax.swing.JScrollPane es.ull.esit.app.AdminProducts.jScrollPane1  [private]
```

Scroll pane wrapping the drinks table.

Definition at line 1067 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.29 jScrollPane2

```
javax.swing.JScrollPane es.ull.esit.app.AdminProducts.jScrollPane2  [private]
```

Scroll pane wrapping the appetizers table.

Definition at line 1069 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.30 jScrollPane3

```
javax.swing.JScrollPane es.ull.esit.app.AdminProducts.jScrollPane3  [private]
```

Scroll pane wrapping the main course table.

Definition at line 1071 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.31 jTabbedPane1

```
javax.swing.JTabbedPane es.ull.esit.app.AdminProducts.jTabbedPane1  [private]
```

Tabbed pane containing the three product categories.

Definition at line 1073 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.initComponents().

7.3.4.32 jTable1

```
javax.swing.JTable es.ull.esit.app.AdminProducts.jTable1  [private]
```

Table displaying drink items.

Definition at line 1075 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.AdminProducts(), and es.ull.esit.app.AdminProducts.initComponents().

7.3.4.33 jTable2

```
javax.swing.JTable es.ull.esit.app.AdminProducts.jTable2  [private]
```

Table displaying appetizer items.

Definition at line 1077 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.AdminProducts(), and es.ull.esit.app.AdminProducts.initComponents().

7.3.4.34 jTable3

```
javax.swing.JTable es.ull.esit.app.AdminProducts.jTable3  [private]
```

Table displaying main course items.

Definition at line 1079 of file AdminProducts.java.

Referenced by es.ull.esit.app.AdminProducts.AdminProducts(), and es.ull.esit.app.AdminProducts.initComponents().

7.3.4.35 productService

```
final ProductService es.ull.esit.app.AdminProducts.productService  [private]
```

Service used to call the REST API and apply product-related logic.

Definition at line 27 of file AdminProducts.java.

The documentation for this class was generated from the following file:

- AdminProducts.java

7.4 es.ull.esit.app.middleware.ApiClient Class Reference

API client for interacting with the backend REST API.

Collaboration diagram for es.ull.esit.app.middleware.ApiClient:

Public Member Functions

- ApiClient (String baseUrl)

  Constructs the ApiClient with the given base URL.
- List< Appetizer > getAllAppetizers () throws Exception

  GET all appetizers.
- Appetizer getAppetizerById (Long id) throws Exception

  GET appetizer by ID.
- Appetizer createAppetizer (Appetizer appetizer) throws Exception

  POST create new appetizer.
- Appetizer updateAppetizer (Long id, Appetizer appetizer) throws Exception

  PUT update appetizer by ID.
- void deleteAppetizer (Long id) throws Exception

  DELETE appetizer by ID.
- List< Cashier > getAllCashiers () throws Exception

  GET all cashiers.
- Cashier getCashierById (Long id) throws Exception

  GET cashier by ID.
- Cashier getCashierByName (String name) throws Exception

  GET cashier by name.
- Cashier updateCashier (Long id, Cashier cashier) throws Exception

  PUT update cashier by ID.
- List< Drink > getAllDrinks () throws Exception

  @brieef GET all drinks.
- Drink getDrinkById (Long id) throws Exception

  GET drink by ID.
- Drink createDrink (Drink drink) throws Exception

  POST create new drink.
- Drink updateDrink (Long id, Drink drink) throws Exception

  PUT update drink by ID.
- void deleteDrink (Long id) throws Exception

  DELETE drink by ID.
- List< MainCourse > getAllMainCourses () throws Exception

  GET all maincourses.
- MainCourse getMainCourseById (Long id) throws Exception

  GET maincourse by ID.
- MainCourse createMainCourse (MainCourse mainCourse) throws Exception

  POST create new maincourse.
- MainCourse updateMainCourse (Long id, MainCourse mainCourse) throws Exception

  PUT update maincourse by ID.
- void deleteMainCourse (Long id) throws Exception

  DELETE maincourse by ID.
- void login (String ignored) throws Exception

  Legacy login method kept for compatibility.
- User login (String username, String password) throws Exception

  Authenticates a user against the backend.

Private Attributes

- final HttpClient http

  Low-level HTTP client.
- final String baseUrl

  Base URL of the Rest API.
- final ObjectMapper mapper

  JSON object mapper for serialization/deserialization: converts between JSON and Java objects.

7.4.1 Detailed Description

API client for interacting with the backend REST API.

```
Wraps HTTPClient and provides methods to:
- performs CRUD operations on Appetizers, Cashiers, Drinks,
MainCourses.
- sends login requests to the backend.
```

Definition at line 25 of file ApiClient.java.

7.4.2 Constructor & Destructor Documentation

7.4.2.1 ApiClient()

```
es.ull.esit.app.middleware.ApiClient.ApiClient (
            String baseUrl )  [inline]
```

Constructs the ApiClient with the given base URL.

```
If the base URL ends with "/", the slash is removed to avoid double
slashes.
```

Parameters

baseUrl	[String] Base URL of the REST API, such as "http://localhost:8080".

Definition at line 46 of file ApiClient.java.

```
00046                                     {
00047     this.baseUrl = baseUrl.endsWith("/") ? baseUrl.substring(0, baseUrl.length() - 1) : baseUrl;
00048     this.http = HttpClient.newBuilder()
00049         .connectTimeout(Duration.ofSeconds(5))
00050         .build();
00051     this.mapper = new ObjectMapper();
00052   }
```

References es.ull.esit.app.middleware.ApiClient.baseUrl.

7.4.3 Member Function Documentation

7.4.3.1 createAppetizer()

```
Appetizer es.ull.esit.app.middleware.ApiClient.createAppetizer (
```

```
            Appetizer appetizer ) throws Exception  [inline]
```

POST create new appetizer.

Creates a new appetizer in the backend.

Parameters

appetizer	[Appetizer] Appetizer object to create.

Returns

[Appetizer] Created Appetizer object returned from the backend.

Exceptions

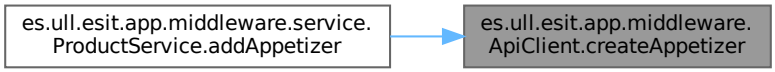
Exception	If an error occurs during the request.

Definition at line 205 of file ApiClient.java.

```
00205                                                                    {
00206     return post("/api/appetizers", appetizer, Appetizer.class);
00207   }
```

Referenced by es.ull.esit.app.middleware.service.ProductService.addAppetizer().

Here is the caller graph for this function:



7.4.3.2 createDrink()

```
Drink es.ull.esit.app.middleware.ApiClient.createDrink (
            Drink drink ) throws Exception  [inline]
```

POST create new drink.

```
  Creates a new drink in the backend.
```

Parameters

drink	[Drink] Drink object to create.

Returns

[Drink] Created Drink object returned from the backend.

Exceptions

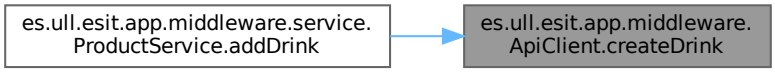
Exception	If an error occurs during the request.

Definition at line 356 of file ApiClient.java.

```
00356                                                   {
00357     return post("/api/drinks", drink, Drink.class);
00358   }
```

Referenced by es.ull.esit.app.middleware.service.ProductService.addDrink().

Here is the caller graph for this function:



7.4.3.3 createMainCourse()

```
MainCourse es.ull.esit.app.middleware.ApiClient.createMainCourse (
            MainCourse mainCourse ) throws Exception  [inline]
```

POST create new maincourse.

Creates a new maincourse in the backend.

Parameters

mainCourse	[MainCourse] MainCourse object to create.

Returns

[MainCourse] Created MainCourse object returned from the backend.

Exceptions

Exception	If an error occurs during the request.

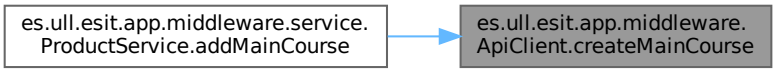
Definition at line 441 of file ApiClient.java.

```
00441                                                           {
00442     return post("/api/maincourses", mainCourse, MainCourse.class);
```

```
00443   }
```

Referenced by es.ull.esit.app.middleware.service.ProductService.addMainCourse().

Here is the caller graph for this function:



7.4.3.4 deleteAppetizer()

```
void es.ull.esit.app.middleware.ApiClient.deleteAppetizer (
            Long id ) throws Exception  [inline]
```

DELETE appetizer by ID.

Deletes an existing appetizer in the backend.

Parameters

id	[Long] ID of the appetizer to delete.

Exceptions

Exception	If an error occurs during the request.

Definition at line 244 of file ApiClient.java.

```
00244                                                {
00245     HttpRequest req = HttpRequest.newBuilder()
00246         .uri(URI.create(baseUrl + "/api/appetizers/" + id))
00247         .DELETE()
00248         .build();
00249     http.send(req, HttpResponse.BodyHandlers.ofString());
00250   }
```

References es.ull.esit.app.middleware.ApiClient.baseUrl, and es.ull.esit.app.middleware.ApiClient.http.

7.4.3.5 deleteDrink()

```
void es.ull.esit.app.middleware.ApiClient.deleteDrink (
            Long id ) throws Exception  [inline]
```

DELETE drink by ID.

```
   Deletes an existing drink in the backend.
```

Parameters

id	[Long] ID of the drink to delete.

Exceptions

Exception	If an error occurs during the request

Definition at line 397 of file ApiClient.java.

```
00397                                           {
00398     HttpRequest req = HttpRequest.newBuilder()
00399         .uri(URI.create(baseUrl + "/api/drinks/" + id))
00400         .DELETE()
00401         .build();
00402     http.send(req, HttpResponse.BodyHandlers.ofString());
00403   }
```

References es.ull.esit.app.middleware.ApiClient.baseUrl, and es.ull.esit.app.middleware.ApiClient.http.

7.4.3.6 deleteMainCourse()

```
void es.ull.esit.app.middleware.ApiClient.deleteMainCourse (
            Long id ) throws Exception  [inline]
```

DELETE maincourse by ID.

```
   Deletes an existing maincourse in the backend by its ID.
```

Parameters

id	[Long] ID of the maincourse to delete.

Exceptions

Exception	If an error occurs during the request.

Definition at line 482 of file ApiClient.java.

```
00482                                                   {
00483     HttpRequest req = HttpRequest.newBuilder()
00484         .uri(URI.create(baseUrl + "/api/maincourses/" + id))
00485         .DELETE()
00486         .build();
00487     http.send(req, HttpResponse.BodyHandlers.ofString());
00488   }
```

References es.ull.esit.app.middleware.ApiClient.baseUrl, and es.ull.esit.app.middleware.ApiClient.http.

7.4.3.7 getAllAppetizers()

```
List< Appetizer > es.ull.esit.app.middleware.ApiClient.getAllAppetizers ( ) throws Exception
[inline]
```

GET all appetizers.

Retrieves a list of all appetizers from the backend.

Returns

[List<Appetizer>] List of all appetizers.

Exceptions

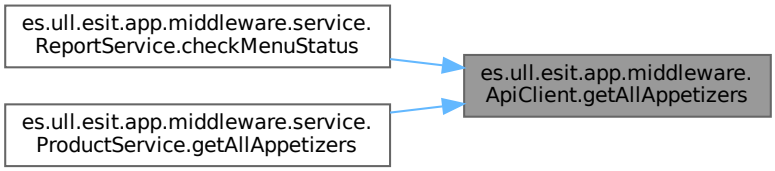
Exception	If an error occurs during the request.

Definition at line 179 of file ApiClient.java.

```
00179                                                    {
00180     return getList("/api/appetizers", new TypeReference<List<Appetizer>>() {
00181     });
00182   }
```

Referenced by es.ull.esit.app.middleware.service.ReportService.checkMenuStatus(), and es.ull.esit.app.middleware.service.ProductS

Here is the caller graph for this function:



7.4.3.8 getAllCashiers()

```
List< Cashier > es.ull.esit.app.middleware.ApiClient.getAllCashiers ( ) throws Exception
[inline]
```

GET all cashiers.

Retrieves a list of all cashiers from the backend.

Returns

[List<Cashier>] List of all cashiers.

Exceptions

Exception	If an error occurs during the request.

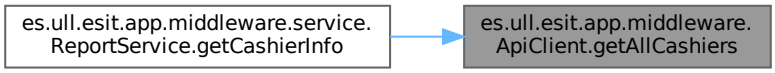
Definition at line 262 of file ApiClient.java.

```
00262                                                {
```

```
00263     return getList("/api/cashiers", new TypeReference<List<Cashier>>() {
00264     });
00265   }
```

Referenced by es.ull.esit.app.middleware.service.ReportService.getCashierInfo().

Here is the caller graph for this function:



7.4.3.9 getAllDrinks()

```
List< Drink > es.ull.esit.app.middleware.ApiClient.getAllDrinks ( ) throws Exception  [inline]
```

@brieef GET all drinks.

```
    Retrieves a list of all drinks from the backend.
```

Returns

[List<Drink>] List of all drinks.

Exceptions

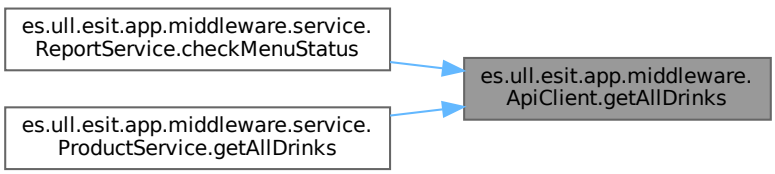
Exception	If an error occurs during the request.

Definition at line 329 of file ApiClient.java.

```
00329                                                 {
00330     return getList("/api/drinks", new TypeReference<List<Drink>>() {
00331     });
00332   }
```

Referenced by es.ull.esit.app.middleware.service.ReportService.checkMenuStatus(), and es.ull.esit.app.middleware.service.ProductS

Here is the caller graph for this function:



7.4.3.10 getAllMainCourses()

```
List< MainCourse > es.ull.esit.app.middleware.ApiClient.getAllMainCourses ( ) throws Exception
[inline]
```

GET all maincourses.

```
Retrieves a list of all maincourses from the backend.
```

Returns

[List<MainCourse>] List of all maincourses.

Exceptions

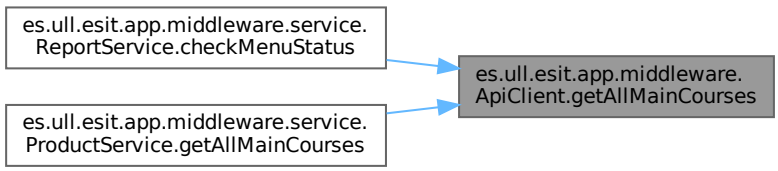
Exception	If an error occurs during the request.

Definition at line 414 of file ApiClient.java.

```
00414                                                                {
00415     return getList("/api/maincourses", new TypeReference<List<MainCourse»() {
00416     });
00417   }
```

Referenced by es.ull.esit.app.middleware.service.ReportService.checkMenuStatus(), and es.ull.esit.app.middleware.service.ProductS

Here is the caller graph for this function:



7.4.3.11 getAppetizerById()

```
Appetizer es.ull.esit.app.middleware.ApiClient.getAppetizerById (
            Long id ) throws Exception  [inline]
```

GET appetizer by ID.

```
Retrieves an appetizer by its ID from the backend.
```

Parameters

id	[Long] ID of the appetizer.

Returns

[Appetizer] Appetizer object.

Exceptions

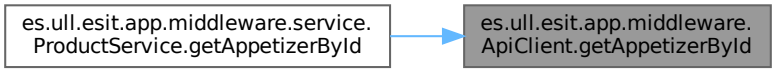
Exception	If an error occurs during the request.

Definition at line 193 of file ApiClient.java.

```
00193                                                   {
00194     return get("/api/appetizers/" + id, Appetizer.class);
00195   }
```

Referenced by es.ull.esit.app.middleware.service.ProductService.getAppetizerById().

Here is the caller graph for this function:



7.4.3.12 getCashierById()

```
Cashier es.ull.esit.app.middleware.ApiClient.getCashierById (
            Long id ) throws Exception  [inline]
```

GET cashier by ID.

Retrieves a cashier by its ID from the backend.

Parameters

id	[Long] ID of the cashier.

Returns

[Cashier] Cashier object.

Exceptions

Exception	If an error occurs during the request.

Definition at line 275 of file ApiClient.java.

```
00275                                                 {
00276     return get("/api/cashiers/" + id, Cashier.class);
00277   }
```

7.4.3.13 getCashierByName()

```
Cashier es.ull.esit.app.middleware.ApiClient.getCashierByName (
            String name ) throws Exception  [inline]
```

GET cashier by name.

Retrieves a cashier by its username from the backend.

Parameters

name	[String] username of the cashier.

Returns

[Cashier] Cashier object.

Exceptions

Exception	If an error occurs during the request.

Definition at line 287 of file ApiClient.java.

```
00287                                                                {
00288     return get("/api/cashiers/name/" + name, Cashier.class);
00289   }
```

7.4.3.14 getDrinkById()

```
Drink es.ull.esit.app.middleware.ApiClient.getDrinkById (
            Long id ) throws Exception  [inline]
```

GET drink by ID.

```
   Retrieves a drink by its ID from the backend.
```

Parameters

id	[Long] ID of the drink.

Returns

[Drink] Drink object.

Exceptions

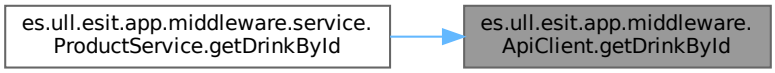
Exception	If an error occurs during the request.

Definition at line 343 of file ApiClient.java.

```
00343                                                   {
00344     return get("/api/drinks/" + id, Drink.class);
00345   }
```

Referenced by es.ull.esit.app.middleware.service.ProductService.getDrinkById().

Here is the caller graph for this function:



7.4.3.15 getMainCourseById()

```
MainCourse es.ull.esit.app.middleware.ApiClient.getMainCourseById (
            Long id ) throws Exception  [inline]
```

GET maincourse by ID.

```
   Retrieves a maincourse by its ID from the backend.
```

Parameters

id	[Long] ID of the maincourse.

Returns

[MainCourse] MainCourse object.

Exceptions

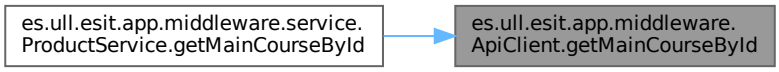
Exception	If an error occurs during the request.

Definition at line 428 of file ApiClient.java.

```
00428                                                       {
00429     return get("/api/maincourses/" + id, MainCourse.class);
00430   }
```

Referenced by es.ull.esit.app.middleware.service.ProductService.getMainCourseById().

Here is the caller graph for this function:



7.4.3.16 login() [1/2]

```
void es.ull.esit.app.middleware.ApiClient.login (
            String ignored ) throws Exception  [inline]
```

Legacy login method kept for compatibility.

```
   If authentication is added in the future, it can be implemented here.
```

Parameters

ignored	[String] Ignored parameter.

Exceptions

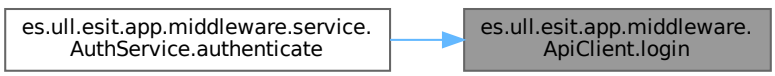
Exception	if login fails.

Definition at line 500 of file ApiClient.java.

```
00500                                                {
00501     // No-op: kept for compatibility.
00502   }
```

Referenced by es.ull.esit.app.middleware.service.AuthService.authenticate().

Here is the caller graph for this function:



7.4.3.17 login() [2/2]

```
User es.ull.esit.app.middleware.ApiClient.login (
            String username,
            String password ) throws Exception  [inline]
```

Authenticates a user against the backend.

```
Sends a POST request to "/api/login" with username and password.
If the response status is 200, it returns the User parsed from JSON.
For any other status code it throws a RuntimeException.
```

Parameters

username	[String] Username entered by the user.
password	[String] Password entered by the user.

Returns

[User] Authenticated User object.

Exceptions

Exception	If the request or JSON parsing fails.

Definition at line 516 of file ApiClient.java.

```
00516                                                                    {
00517
00518     String jsonBody = mapper.writeValueAsString(Map.of(
00519         "username", username,
00520         "password", password));
00521
00522     HttpRequest request = HttpRequest.newBuilder()
00523
00524         .uri(URI.create(baseUrl + "/api/login"))
00525         .header("Content-Type", "application/json")
00526         .POST(HttpRequest.BodyPublishers.ofString(jsonBody))
00527         .build();
00528
00529     HttpResponse<String> response = http.send(request, HttpResponse.BodyHandlers.ofString());
00530
00531     System.out.println("POST /api/login -> HTTP " + response.statusCode());
00532     System.out.println("Response body: '" + response.body() + "'");
00533
00534     if (response.statusCode() == 200) {
00535       return mapper.readValue(response.body(), User.class);
00536     } else {
00537       throw new RuntimeException("Login failed with status: " + response.statusCode());
00538     }
00539   }
```

References es.ull.esit.app.middleware.ApiClient.baseUrl, es.ull.esit.app.middleware.ApiClient.http, and es.ull.esit.app.middleware.Api

7.4.3.18 updateAppetizer()

```
Appetizer es.ull.esit.app.middleware.ApiClient.updateAppetizer (
            Long id,
            Appetizer appetizer ) throws Exception  [inline]
```

PUT update appetizer by ID.

Updates an existing appetizer in the backend.

Parameters

id	[Long] ID of the appetizer to update.
appetizer	[Appetizer] Appetizer object with updated data.

Returns

[Appetizer] Updated Appetizer object returned from the backend.

Exceptions

Exception	If an error occurs during the request.

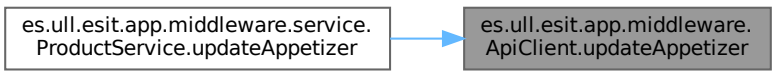
Definition at line 218 of file ApiClient.java.

```
00218                                                                                {
00219     String json = mapper.writeValueAsString(appetizer);
00220     HttpRequest req = HttpRequest.newBuilder()
00221         .uri(URI.create(baseUrl + "/api/appetizers/" + id))
00222         .PUT(HttpRequest.BodyPublishers.ofString(json))
00223         .header("Content-Type", "application/json")
00224         .build();
00225     HttpResponse<String> res = http.send(req, HttpResponse.BodyHandlers.ofString());
00226
00227     int status = res.statusCode();
00228     String body = res.body();
00229
00230     if (status < 200 || status >= 300) {
00231       throw new RuntimeException("PUT /api/appetizers/" + id + " failed with HTTP " + status + " body:
      " + body);
00232     }
00233
00234     return mapper.readValue(body, Appetizer.class);
00235   }
```

References es.ull.esit.app.middleware.ApiClient.baseUrl, es.ull.esit.app.middleware.ApiClient.http, and es.ull.esit.app.middleware.Api

Referenced by es.ull.esit.app.middleware.service.ProductService.updateAppetizer().

Here is the caller graph for this function:



7.4.3.19 updateCashier()

```
Cashier es.ull.esit.app.middleware.ApiClient.updateCashier (
            Long id,
            Cashier cashier ) throws Exception  [inline]
```

PUT update cashier by ID.

```
   Updates an existing cashier in the backend (name and/or salary).
```

Parameters

id	[Long] ID of the cashier to update.
cashier	[Cashier] Cashier object with updated data.

Returns

[Cashier] Updated Cashier object returned from the backend.

Exceptions

Exception	If an error occurs during the request.

Definition at line 301 of file ApiClient.java.

```
00301                                                                    {
00302     String json = mapper.writeValueAsString(cashier);
00303     HttpRequest req = HttpRequest.newBuilder()
00304         .uri(URI.create(baseUrl + "/api/cashiers/" + id))
00305         .PUT(HttpRequest.BodyPublishers.ofString(json))
00306         .header("Content-Type", "application/json")
00307         .build();
00308     HttpResponse<String> res = http.send(req, HttpResponse.BodyHandlers.ofString());
00309
00310     int status = res.statusCode();
00311     String body = res.body();
00312
00313     if (status < 200 || status >= 300) {
00314       throw new RuntimeException("PUT /api/cashiers/" + id + " failed with HTTP " + status + " body: "
      + body);
00315     }
00316
00317     return mapper.readValue(body, Cashier.class);
00318   }
```

References es.ull.esit.app.middleware.ApiClient.baseUrl, es.ull.esit.app.middleware.ApiClient.http, and es.ull.esit.app.middleware.Api

7.4.3.20 updateDrink()

```
Drink es.ull.esit.app.middleware.ApiClient.updateDrink (
            Long id,
            Drink drink ) throws Exception  [inline]
```

PUT update drink by ID.

Updates an existing drink in the backend by its ID.

Parameters

id	[Long] ID of the drink to update.
drink	[Drink] Drink object with updated data.

Returns

[Drink] Updated Drink object returned from the backend.

Exceptions

Exception	If an error occurs during the request.

Definition at line 370 of file ApiClient.java.

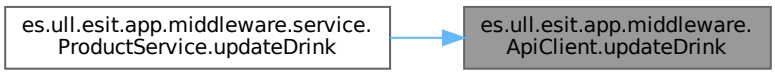
```
00370                                                                    {
```

```
00371     String json = mapper.writeValueAsString(drink);
00372     HttpRequest req = HttpRequest.newBuilder()
00373         .uri(URI.create(baseUrl + "/api/drinks/" + id))
00374         .PUT(HttpRequest.BodyPublishers.ofString(json))
00375         .header("Content-Type", "application/json")
00376         .build();
00377     HttpResponse<String> res = http.send(req, HttpResponse.BodyHandlers.ofString());
00378
00379     int status = res.statusCode();
00380     String body = res.body();
00381
00382     if (status < 200 || status >= 300) {
00383       throw new RuntimeException("PUT /api/drinks/" + id + " failed with HTTP " + status + " body: " +
      body);
00384     }
00385
00386     return mapper.readValue(body, Drink.class);
00387   }
```

References es.ull.esit.app.middleware.ApiClient.baseUrl, es.ull.esit.app.middleware.ApiClient.http, and es.ull.esit.app.middleware.Api

Referenced by es.ull.esit.app.middleware.service.ProductService.updateDrink().

Here is the caller graph for this function:



7.4.3.21 updateMainCourse()

```
MainCourse es.ull.esit.app.middleware.ApiClient.updateMainCourse (
            Long id,
            MainCourse mainCourse ) throws Exception  [inline]
```

PUT update maincourse by ID.

```
   Updates an existing maincourse in the backend.
```

Parameters

id	[Long] ID of the maincourse to update.
mainCourse	[MainCourse] MainCourse object with updated data.

Returns

[MainCourse] Updated MainCourse object returned from the backend.

Exceptions

Exception	If an error occurs during the request.

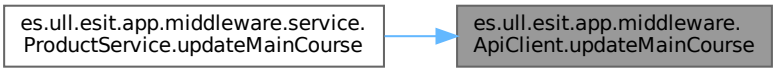
Definition at line 455 of file ApiClient.java.

```
00455                                                                                {
00456     String json = mapper.writeValueAsString(mainCourse);
00457     HttpRequest req = HttpRequest.newBuilder()
00458         .uri(URI.create(baseUrl + "/api/maincourses/" + id))
00459         .PUT(HttpRequest.BodyPublishers.ofString(json))
00460         .header("Content-Type", "application/json")
00461         .build();
00462     HttpResponse<String> res = http.send(req, HttpResponse.BodyHandlers.ofString());
00463
00464     int status = res.statusCode();
00465     String body = res.body();
00466
00467     if (status < 200 || status >= 300) {
00468       throw new RuntimeException("PUT /api/maincourses/" + id + " failed with HTTP " + status + "
      body: " + body);
00469     }
00470
00471     return mapper.readValue(body, MainCourse.class);
00472   }
```

References es.ull.esit.app.middleware.ApiClient.baseUrl, es.ull.esit.app.middleware.ApiClient.http, and es.ull.esit.app.middleware.Api

Referenced by es.ull.esit.app.middleware.service.ProductService.updateMainCourse().

Here is the caller graph for this function:



7.4.4 Member Data Documentation

7.4.4.1 baseUrl

```
final String es.ull.esit.app.middleware.ApiClient.baseUrl  [private]
```

Base URL of the Rest API.

Definition at line 30 of file ApiClient.java.

Referenced by es.ull.esit.app.middleware.ApiClient.ApiClient(), es.ull.esit.app.middleware.ApiClient.deleteAppetizer(), es.ull.esit.app.middleware.ApiClient.deleteDrink(), es.ull.esit.app.middleware.ApiClient.deleteMainCourse(), es.ull.esit.app.middleware.ApiClient.login(), es.ull.esit.app.middleware.ApiClient.updateAppetizer(), es.ull.esit.app.middleware.ApiClie es.ull.esit.app.middleware.ApiClient.updateDrink(), and es.ull.esit.app.middleware.ApiClient.updateMainCourse().

7.4.4.2 http

```
final HttpClient es.ull.esit.app.middleware.ApiClient.http  [private]
```

Low-level HTTP client.

to send requests.

Definition at line 27 of file ApiClient.java.

Referenced by es.ull.esit.app.middleware.ApiClient.deleteAppetizer(), es.ull.esit.app.middleware.ApiClient.deleteDrink(), es.ull.esit.app.middleware.ApiClient.deleteMainCourse(), es.ull.esit.app.middleware.ApiClient.login(), es.ull.esit.app.middleware.ApiCl es.ull.esit.app.middleware.ApiClient.updateCashier(), es.ull.esit.app.middleware.ApiClient.updateDrink(), and es.ull.esit.app.middleware.ApiClient.updateMainCourse().

7.4.4.3 mapper

```
final ObjectMapper es.ull.esit.app.middleware.ApiClient.mapper  [private]
```

JSON object mapper for serialization/deserialization: converts between JSON and Java objects.

Definition at line 36 of file ApiClient.java.

Referenced by es.ull.esit.app.middleware.ApiClient.login(), es.ull.esit.app.middleware.ApiClient.updateAppetizer(), es.ull.esit.app.middleware.ApiClient.updateCashier(), es.ull.esit.app.middleware.ApiClient.updateDrink(), and es.ull.esit.app.middleware.ApiClient.updateMainCourse().

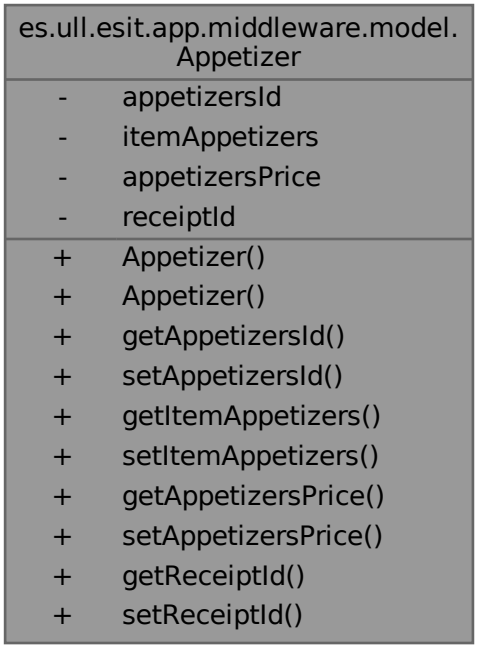
The documentation for this class was generated from the following file:

- ApiClient.java

7.5 es.ull.esit.app.middleware.model.Appetizer Class Reference

Client-side model representing an appetizer received from the backend API.

Collaboration diagram for es.ull.esit.app.middleware.model.Appetizer:



Public Member Functions

- Appetizer ()
  Default constructor required for JSON deserialization.
- Appetizer (Long appetizersId, String itemAppetizers, Integer appetizersPrice, Long receiptId)
  Constructs an appetizer with all fields.
- Long getAppetizersId ()
  Gets the appetizer identifier.
- void setAppetizersId (Long appetizersId)
  Sets the appetizer identifier.
- String getItemAppetizers ()
  Gets the appetizer item name.
- void setItemAppetizers (String itemAppetizers)
  Sets the appetizer item name.
- Integer getAppetizersPrice ()
  Gets the appetizer price.
- void setAppetizersPrice (Integer appetizersPrice)
  Sets the appetizer price.
- Long getReceiptId ()
  Gets the identifier of the related receipt.
- void setReceiptId (Long receiptId)
  Sets the identifier of the related receipt.

Private Attributes

- Long appetizersId
  Unique identifier of the appetizer (JSON property "appetizersId").
- String itemAppetizers
  Name of the appetizer item (JSON property "itemAppetizers").
- Integer appetizersPrice
  Price of the appetizer (JSON property "appetizersPrice").
- Long receiptId
  Identifier of the receipt this appetizer belongs to (JSON property "receiptId").

7.5.1 Detailed Description

Client-side model representing an appetizer received from the backend API.

```
It is used by the Swing application to deserialize JSON sent by
the REST server for the "/api/appetizers" endpoint.
```

Definition at line 11 of file src/main/java/es/ull/esit/app/middleware/model/Appetizer.java.

7.5.2 Constructor & Destructor Documentation

7.5.2.1 Appetizer() [1/2]

```
es.ull.esit.app.middleware.model.Appetizer.Appetizer ( )  [inline]
```

Default constructor required for JSON deserialization.

Definition at line 35 of file src/main/java/es/ull/esit/app/middleware/model/Appetizer.java.

```
00035                     {
00036     }
```

7.5.2.2 Appetizer() [2/2]

```
es.ull.esit.app.middleware.model.Appetizer.Appetizer (
            Long appetizersId,
            String itemAppetizers,
            Integer appetizersPrice,
            Long receiptId )  [inline]
```

Constructs an appetizer with all fields.

Parameters

appetizersId	[Long] Unique identifier of the appetizer.
itemAppetizers	[String] Name of the appetizer item.
appetizersPrice	[Integer] Price of the appetizer.
receiptId	[Long] Identifier of the related receipt (optional).

Definition at line 46 of file src/main/java/es/ull/esit/app/middleware/model/Appetizer.java.

```
00046
    {
00047     this.appetizersId = appetizersId;
00048     this.itemAppetizers = itemAppetizers;
00049     this.appetizersPrice = appetizersPrice;
00050     this.receiptId = receiptId;
00051   }
```

References es.ull.esit.app.middleware.model.Appetizer.appetizersId, es.ull.esit.app.middleware.model.Appetizer.appetizersPrice, es.ull.esit.app.middleware.model.Appetizer.itemAppetizers, and es.ull.esit.app.middleware.model.Appetizer.receiptId.

7.5.3 Member Function Documentation

7.5.3.1 getAppetizersId()

```
Long es.ull.esit.app.middleware.model.Appetizer.getAppetizersId ( )  [inline]
```

Gets the appetizer identifier.

Returns

[Long] Unique identifier of the appetizer.

Definition at line 58 of file src/main/java/es/ull/esit/app/middleware/model/Appetizer.java.

```
00058                                {
00059     return appetizersId;
00060   }
```

References es.ull.esit.app.middleware.model.Appetizer.appetizersId.

7.5.3.2 getAppetizersPrice()

```
Integer es.ull.esit.app.middleware.model.Appetizer.getAppetizersPrice ( )  [inline]
```

Gets the appetizer price.

Returns

[Integer] Price of the appetizer.

Definition at line 94 of file src/main/java/es/ull/esit/app/middleware/model/Appetizer.java.

```
00094                                    {
00095     return appetizersPrice;
00096   }
```

References es.ull.esit.app.middleware.model.Appetizer.appetizersPrice.

7.5.3.3 getItemAppetizers()

```
String es.ull.esit.app.middleware.model.Appetizer.getItemAppetizers ( )  [inline]
```

Gets the appetizer item name.

Returns

[String] Name of the appetizer item.

Definition at line 76 of file src/main/java/es/ull/esit/app/middleware/model/Appetizer.java.

```
00076                                    {
00077     return itemAppetizers;
00078   }
```

References es.ull.esit.app.middleware.model.Appetizer.itemAppetizers.

7.5.3.4 getReceiptId()

```
Long es.ull.esit.app.middleware.model.Appetizer.getReceiptId ( )  [inline]
```

Gets the identifier of the related receipt.

Returns

[Long] Identifier of the receipt or null if not set.

Definition at line 112 of file src/main/java/es/ull/esit/app/middleware/model/Appetizer.java.

```
00112                                  {
00113     return receiptId;
00114   }
```

References es.ull.esit.app.middleware.model.Appetizer.receiptId.

7.5.3.5 setAppetizersId()

```
void es.ull.esit.app.middleware.model.Appetizer.setAppetizersId (
            Long appetizersId )  [inline]
```

Sets the appetizer identifier.

Parameters

appetizersId	[Long] Unique identifier of the appetizer.

Definition at line 67 of file src/main/java/es/ull/esit/app/middleware/model/Appetizer.java.

```
00067                                          {
00068     this.appetizersId = appetizersId;
00069   }
```

References es.ull.esit.app.middleware.model.Appetizer.appetizersId.

7.5.3.6 setAppetizersPrice()

```
void es.ull.esit.app.middleware.model.Appetizer.setAppetizersPrice (
            Integer appetizersPrice )  [inline]
```

Sets the appetizer price.

Parameters

appetizersPrice	[Integer] Price of the appetizer.

Definition at line 103 of file src/main/java/es/ull/esit/app/middleware/model/Appetizer.java.

```
00103                                                    {
00104     this.appetizersPrice = appetizersPrice;
00105   }
```

References es.ull.esit.app.middleware.model.Appetizer.appetizersPrice.

7.5.3.7 setItemAppetizers()

```
void es.ull.esit.app.middleware.model.Appetizer.setItemAppetizers (
            String itemAppetizers )  [inline]
```

Sets the appetizer item name.

Parameters

itemAppetizers	[String] Name of the appetizer item.

Definition at line 85 of file src/main/java/es/ull/esit/app/middleware/model/Appetizer.java.

```
00085                                                {
00086     this.itemAppetizers = itemAppetizers;
00087   }
```

References es.ull.esit.app.middleware.model.Appetizer.itemAppetizers.

7.5.3.8 setReceiptId()

```
void es.ull.esit.app.middleware.model.Appetizer.setReceiptId (
            Long receiptId )  [inline]
```

Sets the identifier of the related receipt.

Parameters

receiptId	[Long] Identifier of the receipt.

Definition at line 121 of file src/main/java/es/ull/esit/app/middleware/model/Appetizer.java.

```
00121                                          {
00122     this.receiptId = receiptId;
00123   }
```

References es.ull.esit.app.middleware.model.Appetizer.receiptId.

7.5.4 Member Data Documentation

7.5.4.1 appetizersId

```
Long es.ull.esit.app.middleware.model.Appetizer.appetizersId  [private]
```

Unique identifier of the appetizer (JSON property "appetizersId").

Definition at line 15 of file src/main/java/es/ull/esit/app/middleware/model/Appetizer.java.

Referenced by es.ull.esit.app.middleware.model.Appetizer.Appetizer(), es.ull.esit.app.middleware.model.Appetizer.getAppetizersId(), and es.ull.esit.app.middleware.model.Appetizer.setAppetizersId().

7.5.4.2 appetizersPrice

```
Integer es.ull.esit.app.middleware.model.Appetizer.appetizersPrice  [private]
```

Price of the appetizer (JSON property "appetizersPrice").

Definition at line 23 of file src/main/java/es/ull/esit/app/middleware/model/Appetizer.java.

Referenced by es.ull.esit.app.middleware.model.Appetizer.Appetizer(), es.ull.esit.app.middleware.model.Appetizer.getAppetizersPrice and es.ull.esit.app.middleware.model.Appetizer.setAppetizersPrice().

7.5.4.3 itemAppetizers

```
String es.ull.esit.app.middleware.model.Appetizer.itemAppetizers  [private]
```

Name of the appetizer item (JSON property "itemAppetizers").

Definition at line 19 of file src/main/java/es/ull/esit/app/middleware/model/Appetizer.java.

Referenced by es.ull.esit.app.middleware.model.Appetizer.Appetizer(), es.ull.esit.app.middleware.model.Appetizer.getItemAppetizers(and es.ull.esit.app.middleware.model.Appetizer.setItemAppetizers().

7.5.4.4 receiptId

```
Long es.ull.esit.app.middleware.model.Appetizer.receiptId  [private]
```

Identifier of the receipt this appetizer belongs to (JSON property "receiptId").

Currently not populated by the backend.

Definition at line 30 of file src/main/java/es/ull/esit/app/middleware/model/Appetizer.java.

Referenced by es.ull.esit.app.middleware.model.Appetizer.Appetizer(), es.ull.esit.app.middleware.model.Appetizer.getReceiptId(), and es.ull.esit.app.middleware.model.Appetizer.setReceiptId().

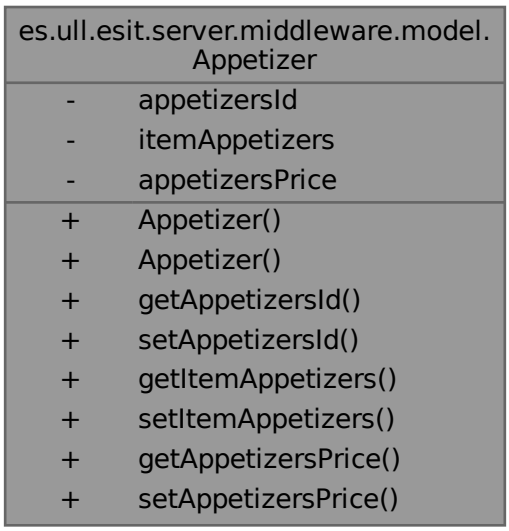
The documentation for this class was generated from the following file:

- src/main/java/es/ull/esit/app/middleware/model/Appetizer.java

7.6 es.ull.esit.server.middleware.model.Appetizer Class Reference

JPA entity that represents an appetizer in the menu.

Collaboration diagram for es.ull.esit.server.middleware.model.Appetizer:



Public Member Functions

- Appetizer ()

  Default constructor required by JPA.
- Appetizer (Long appetizersId, String itemAppetizers, Integer appetizersPrice)

  Full constructor.
- Long getAppetizersId ()

  Gets the appetizer ID.
- void setAppetizersId (Long appetizersId)

  Sets the appetizer ID.
- String getItemAppetizers ()

  Gets the appetizer name.
- void setItemAppetizers (String itemAppetizers)

  Sets the appetizer name.
- Integer getAppetizersPrice ()

  Gets the appetizer price.
- void setAppetizersPrice (Integer appetizersPrice)

  Sets the appetizer price.

Private Attributes

- Long appetizersId

  Primary key of the appetizer (column "id").
- String itemAppetizers

  Name of the appetizer (column "name").
- Integer appetizersPrice

  Price of the appetizer (column "price").

7.6.1 Detailed Description

JPA entity that represents an appetizer in the menu.

```
It is mapped to the "appetizers" table used by the REST API.
Each record has an auto-increment ID, a name and a price.
```

Definition at line 15 of file server/src/main/java/es/ull/esit/server/middleware/model/Appetizer.java.

7.6.2 Constructor & Destructor Documentation

7.6.2.1 Appetizer() [1/2]

```
es.ull.esit.server.middleware.model.Appetizer.Appetizer ( )  [inline]
```

Default constructor required by JPA.

Definition at line 34 of file server/src/main/java/es/ull/esit/server/middleware/model/Appetizer.java.

```
00034                       {
00035   }
```

7.6.2.2 Appetizer() [2/2]

```
es.ull.esit.server.middleware.model.Appetizer.Appetizer (
            Long appetizersId,
            String itemAppetizers,
            Integer appetizersPrice )  [inline]
```

Full constructor.

Parameters

appetizersId	[Long] Identifier of the appetizer.
itemAppetizers	[String] Name of the appetizer.
appetizersPrice	[Integer] Price of the appetizer.

Definition at line 44 of file server/src/main/java/es/ull/esit/server/middleware/model/Appetizer.java.

```
00044                                                                                   {
00045     this.appetizersId = appetizersId;
00046     this.itemAppetizers = itemAppetizers;
```

```
00047     this.appetizersPrice = appetizersPrice;
00048   }
```

7.6.3 Member Function Documentation

7.6.3.1 getAppetizersId()

```
Long es.ull.esit.server.middleware.model.Appetizer.getAppetizersId ( )  [inline]
```

Gets the appetizer ID.

Returns

[Long] Appetizer id.

Definition at line 55 of file server/src/main/java/es/ull/esit/server/middleware/model/Appetizer.java.

```
00055                                {
00056     return appetizersId;
00057   }
```

7.6.3.2 getAppetizersPrice()

```
Integer es.ull.esit.server.middleware.model.Appetizer.getAppetizersPrice ( )  [inline]
```

Gets the appetizer price.

Returns

[Integer] Appetizer price.

Definition at line 92 of file server/src/main/java/es/ull/esit/server/middleware/model/Appetizer.java.

```
00092                                    {
00093     return appetizersPrice;
00094   }
```

7.6.3.3 getItemAppetizers()

```
String es.ull.esit.server.middleware.model.Appetizer.getItemAppetizers ( )  [inline]
```

Gets the appetizer name.

Returns

[String] Appetizer name.

Definition at line 74 of file server/src/main/java/es/ull/esit/server/middleware/model/Appetizer.java.

```
00074                                  {
00075     return itemAppetizers;
00076   }
```

7.6.3.4 setAppetizersId()

```
void es.ull.esit.server.middleware.model.Appetizer.setAppetizersId (
            Long appetizersId )  [inline]
```

Sets the appetizer ID.

Generated by the database.

Parameters

appetizersId	[Long] Appetizer id.

Definition at line 65 of file server/src/main/java/es/ull/esit/server/middleware/model/Appetizer.java.

```
00065                                                {
00066     this.appetizersId = appetizersId;
00067   }
```

7.6.3.5 setAppetizersPrice()

```
void es.ull.esit.server.middleware.model.Appetizer.setAppetizersPrice (
            Integer appetizersPrice )  [inline]
```

Sets the appetizer price.

Parameters

appetizersPrice	[Integer] Appetizer price.

Definition at line 101 of file server/src/main/java/es/ull/esit/server/middleware/model/Appetizer.java.

```
00101                                                    {
00102     this.appetizersPrice = appetizersPrice;
00103   }
```

7.6.3.6 setItemAppetizers()

```
void es.ull.esit.server.middleware.model.Appetizer.setItemAppetizers (
            String itemAppetizers )  [inline]
```

Sets the appetizer name.

Parameters

itemAppetizers	[String] Appetizer name.

Definition at line 83 of file server/src/main/java/es/ull/esit/server/middleware/model/Appetizer.java.

```
00083                                                  {
00084     this.itemAppetizers = itemAppetizers;
00085   }
```

7.6.4 Member Data Documentation

7.6.4.1 appetizersId

```
Long es.ull.esit.server.middleware.model.Appetizer.appetizersId  [private]
```

Primary key of the appetizer (column "id").

Definition at line 21 of file server/src/main/java/es/ull/esit/server/middleware/model/Appetizer.java.

7.6.4.2 appetizersPrice

```
Integer es.ull.esit.server.middleware.model.Appetizer.appetizersPrice  [private]
```

Price of the appetizer (column "price").

Definition at line 29 of file server/src/main/java/es/ull/esit/server/middleware/model/Appetizer.java.

7.6.4.3 itemAppetizers

```
String es.ull.esit.server.middleware.model.Appetizer.itemAppetizers  [private]
```

Name of the appetizer (column "name").

Definition at line 25 of file server/src/main/java/es/ull/esit/server/middleware/model/Appetizer.java.

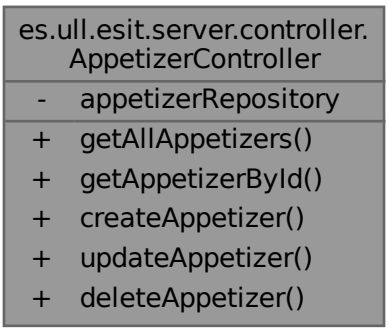
The documentation for this class was generated from the following file:

- server/src/main/java/es/ull/esit/server/middleware/model/Appetizer.java

7.7 es.ull.esit.server.controller.AppetizerController Class Reference

REST controller for managing appetizers.

Collaboration diagram for es.ull.esit.server.controller.AppetizerController:



Public Member Functions

- ResponseEntity< List< Appetizer > > getAllAppetizers ()

  Returns the complete list of appetizers.
- ResponseEntity< Appetizer > getAppetizerById (@PathVariable Long id)

  Returns a single appetizer by its id.
- ResponseEntity< Appetizer > createAppetizer (@RequestBody Appetizer appetizer)

  Creates a new appetizer.
- ResponseEntity< Appetizer > updateAppetizer (@PathVariable Long id, @RequestBody Appetizer appetizer)

  Updates an existing appetizer.
- ResponseEntity< Void > deleteAppetizer (@PathVariable Long id)

  Deletes an appetizer by its id.

Private Attributes

- AppetizerRepository appetizerRepository

  Repository used to perform database operations on appetizers.

7.7.1 Detailed Description

REST controller for managing appetizers.

Rovides CRUD operations for Appetizer entities using the path "/api/appetizers". These endpoints are used by the Swing client to load and modify appetizer information.

Definition at line 21 of file AppetizerController.java.

7.7.2 Member Function Documentation

7.7.2.1 createAppetizer()

```
ResponseEntity< Appetizer > es.ull.esit.server.controller.AppetizerController.createAppetizer
(
            @RequestBody Appetizer appetizer )  [inline]
```

Creates a new appetizer.

```
HTTP POST /api/appetizers
```

Parameters

appetizer	[Appetizer] Appetizer to create.

Returns

[ResponseEntity<Appetizer>] The created appetizer with status 201.

Definition at line 65 of file AppetizerController.java.

```
00065                                                                                {
00066     Appetizer saved = appetizerRepository.save(appetizer);
00067     return ResponseEntity.status(HttpStatus.CREATED).body(saved);
00068   }
```

7.7.2.2 deleteAppetizer()

```
ResponseEntity< Void > es.ull.esit.server.controller.AppetizerController.deleteAppetizer (
            @PathVariable Long id )  [inline]
```

Deletes an appetizer by its id.

```
HTTP DELETE /api/appetizers/{id}
```

Parameters

id	[Long] Identifier of the appetizer to delete.

Returns

[ResponseEntity<Void>] 204 if deleted or 404 if not found.

Definition at line 99 of file AppetizerController.java.

```
00099                                                                    {
00100     if (!appetizerRepository.existsById(id)) {
00101       return ResponseEntity.notFound().build();
00102     }
00103     appetizerRepository.deleteById(id);
00104     return ResponseEntity.noContent().build();
00105   }
```

7.7.2.3 getAllAppetizers()

```
ResponseEntity< List< Appetizer > > es.ull.esit.server.controller.AppetizerController.get↩
AllAppetizers ( )  [inline]
```

Returns the complete list of appetizers.

```
HTTP GET /api/appetizers
```

Returns

[ResponseEntity<List<Appetizer>>] List of appetizers with status 200.

Definition at line 35 of file AppetizerController.java.

```
00035                                                          {
00036     List<Appetizer> appetizers = appetizerRepository.findAll();
00037     return ResponseEntity.ok(appetizers);
00038   }
```

7.7.2.4 getAppetizerById()

```
ResponseEntity< Appetizer > es.ull.esit.server.controller.AppetizerController.getAppetizerById
(
            @PathVariable Long id )  [inline]
```

Returns a single appetizer by its id.

```
HTTP GET /api/appetizers/{id}
```

Parameters

id	[Long] Identifier of the appetizer.

Returns

[ResponseEntity<Appetizer>] The appetizer if found or 404 otherwise.

Definition at line 49 of file AppetizerController.java.

```
00049                                                                    {
00050     Optional<Appetizer> appetizer = appetizerRepository.findById(id);
00051     return appetizer
00052         .map(ResponseEntity::ok)
00053         .orElseGet(() -> ResponseEntity.notFound().build());
00054   }
```

7.7.2.5 updateAppetizer()

```
ResponseEntity< Appetizer > es.ull.esit.server.controller.AppetizerController.updateAppetizer
(
            @PathVariable Long id,
            @RequestBody Appetizer appetizer )  [inline]
```

Updates an existing appetizer.

```
HTTP PUT /api/appetizers/{id}
```

Parameters

id	[Long] Identifier of the appetizer to update.
appetizer	[Appetizer] Updated appetizer data.

Returns

[ResponseEntity<Appetizer>] The updated appetizer or 404 if not found.

Definition at line 80 of file AppetizerController.java.

```
00081                                                                                {
00082     if (!appetizerRepository.existsById(id)) {
00083       return ResponseEntity.notFound().build();
00084     }
00085     appetizer.setAppetizersId(id);
00086     Appetizer updated = appetizerRepository.save(appetizer);
00087     return ResponseEntity.ok(updated);
00088   }
```

7.7.3 Member Data Documentation

7.7.3.1 appetizerRepository

```
AppetizerRepository es.ull.esit.server.controller.AppetizerController.appetizerRepository
[private]
```

Repository used to perform database operations on appetizers.

Definition at line 25 of file AppetizerController.java.

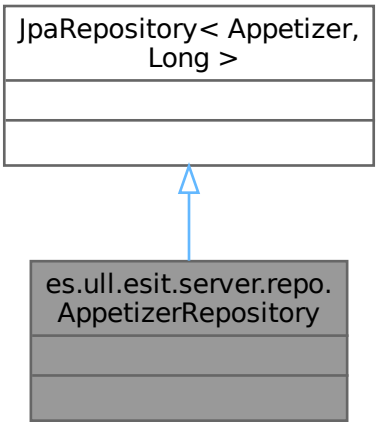
The documentation for this class was generated from the following file:

- AppetizerController.java

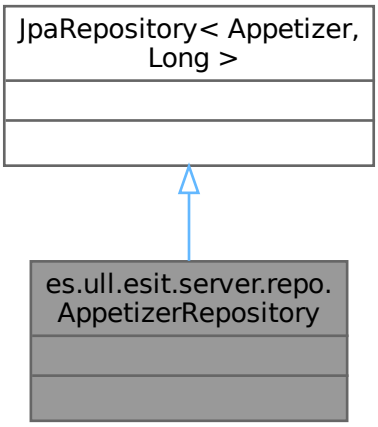
7.8 es.ull.esit.server.repo.AppetizerRepository Interface Reference

Repository interface for managing appetizers in the database.

Inheritance diagram for es.ull.esit.server.repo.AppetizerRepository:



Collaboration diagram for es.ull.esit.server.repo.AppetizerRepository:



7.8.1 Detailed Description

Repository interface for managing appetizers in the database.

```
Extends JpaRepository to provide basic CRUD operations on the "appetizers"
table, such as findAll, findById, save and delete.
```

Definition at line 12 of file AppetizerRepository.java.

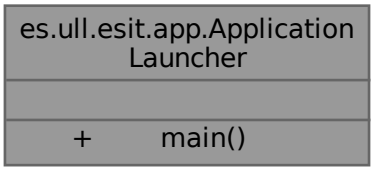
The documentation for this interface was generated from the following file:

- AppetizerRepository.java

7.9 es.ull.esit.app.ApplicationLauncher Class Reference

Auxiliary entry point used to start the application's UI.

Collaboration diagram for es.ull.esit.app.ApplicationLauncher:



Static Public Member Functions

- static void main (String[ ] args)

  Alternate entry point used to launch the Login window.

7.9.1 Detailed Description

Auxiliary entry point used to start the application's UI.

```
Serves as a simple launcher responsible for opening
the Login window when the program is executed directly.
```

Definition at line 10 of file ApplicationLauncher.java.

7.9.2 Member Function Documentation

7.9.2.1 main()

```
static void es.ull.esit.app.ApplicationLauncher.main (
            String[] args )  [inline], [static]
```

Alternate entry point used to launch the Login window.

```
Creates and displays the application’s Login screen.
```

Parameters

args	[String[]] Command-line arguments (unused).

Definition at line 19 of file ApplicationLauncher.java.

```
00019                                       {
00020     new Login().setVisible(true);
00021   }
```

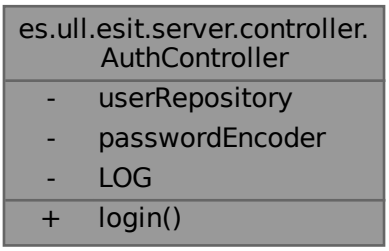
The documentation for this class was generated from the following file:

- ApplicationLauncher.java

7.10 es.ull.esit.server.controller.AuthController Class Reference

Rest controller for handling authentication-related requests.

Collaboration diagram for es.ull.esit.server.controller.AuthController:



Classes

- class LoginRequest

  Simple DTO (Data Transfer Object) for login requests payload.

Public Member Functions

- ResponseEntity<?> login (@RequestBody LoginRequest request)

  Endpoint for handling user login requests:

Private Attributes

- UserRepository userRepository

  Repository for accessing user data.
- PasswordEncoder passwordEncoder

  Component used to verify raw passwords against stored hashes.

Static Private Attributes

- static final Logger LOG = LoggerFactory.getLogger(AuthController.class)

  Logger for logging authentication events.

7.10.1 Detailed Description

Rest controller for handling authentication-related requests.

```
Exposes an HTTP endpoint that allows clients to log in by sending
a username and password. The credentials are validated against the
users stored in the database.
```

Definition at line 23 of file AuthController.java.

7.10.2 Member Function Documentation

7.10.2.1 login()

```
ResponseEntity<?> es.ull.esit.server.controller.AuthController.login (
            @RequestBody LoginRequest request )  [inline]
```

Endpoint for handling user login requests:

- receives a username and password in the request body.
- searches for the user in the database.
- compares the provided password with the stored password hash.
- return 200 OK with user data if credentials are valid.
- return 401 Unauthorized if credentials are invalid.

Parameters

request	[LoginRequest] Login request payload containing username and password.

Returns

[ResponseEntity] HTTP response indicating success or failure of authentication.

Definition at line 61 of file AuthController.java.

```
00061                                                                {
00062
00063     LOG.info("Login attempt for user: {}", request.username);
00064
00065     Optional<User> userOpt = userRepository.findByUsername(request.username);
00066
00067     if (!userOpt.isPresent()) {
00068       LOG.warn("User not found in DB: {}", request.username);
00069       return ResponseEntity.status(401).body("Invalid credentials");
00070     }
```

```
00071
00072     User user = userOpt.get();
00073     LOG.debug("User found. Stored hash = {}", user.getPasswordHash()); // Mejor debug, no info
00074
00075     if (!passwordEncoder.matches(request.password, user.getPasswordHash())) {
00076       LOG.warn("Password mismatch for user: {}", request.username);
00077       return ResponseEntity.status(401).body("Invalid credentials");
00078     }
00079
00080     LOG.info("Login OK for user: {}", request.username);
00081
00082     return ResponseEntity.ok(user);
00083
00084   }
```

7.10.3 Member Data Documentation

7.10.3.1 LOG

```
final Logger es.ull.esit.server.controller.AuthController.LOG = LoggerFactory.getLogger(Auth↩
Controller.class)  [static], [private]
```

Logger for logging authentication events.

Definition at line 26 of file AuthController.java.

7.10.3.2 passwordEncoder

```
PasswordEncoder es.ull.esit.server.controller.AuthController.passwordEncoder  [private]
```

Component used to verify raw passwords against stored hashes.

Definition at line 34 of file AuthController.java.

7.10.3.3 userRepository

```
UserRepository es.ull.esit.server.controller.AuthController.userRepository  [private]
```

Repository for accessing user data.

Definition at line 30 of file AuthController.java.

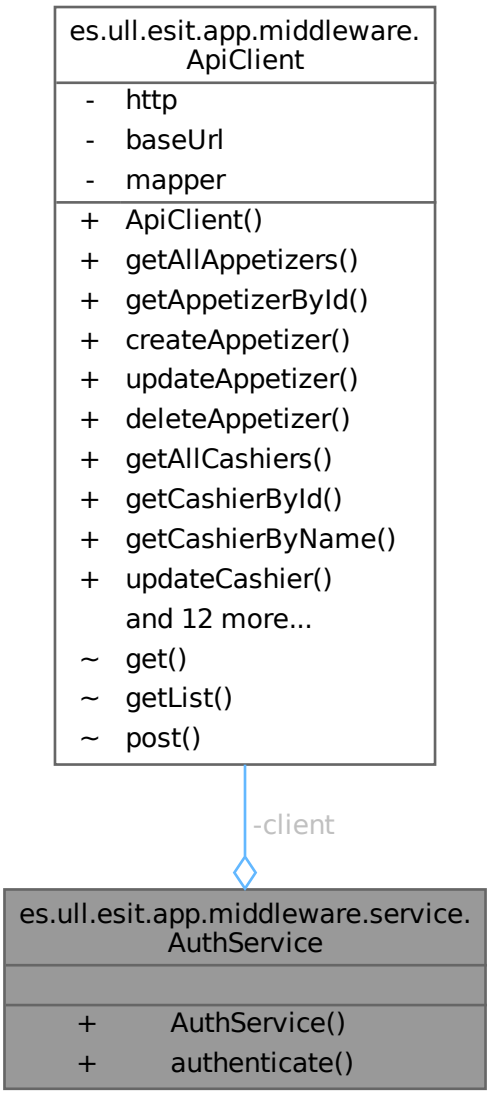
The documentation for this class was generated from the following file:

- AuthController.java

7.11 es.ull.esit.app.middleware.service.AuthService Class Reference

Service responsible for handling authentication logic on the client side.

Collaboration diagram for es.ull.esit.app.middleware.service.AuthService:



Public Member Functions

- AuthService (ApiClient client)

  Constructs an AuthService with the given API client.
- User authenticate (String username, String password) throws Exception

  Attempts to authenticate a user against the backend.

Private Attributes

- final ApiClient client

  REST API client used to communicate with the backend login endpoint.

7.11.1 Detailed Description

Service responsible for handling authentication logic on the client side.

```
Performs local validation of credentials and delegates the actual login
process to the ApiClient.
```

Definition at line 12 of file AuthService.java.

7.11.2 Constructor & Destructor Documentation

7.11.2.1 AuthService()

```
es.ull.esit.app.middleware.service.AuthService.AuthService (
            ApiClient client )  [inline]
```

Constructs an AuthService with the given API client.

Parameters

client	[ApiClient] Client used to perform HTTP requests to the backend.

Definition at line 22 of file AuthService.java.

```
00022                                       {
00023     this.client = client;
00024   }
```

References es.ull.esit.app.middleware.service.AuthService.client.

7.11.3 Member Function Documentation

7.11.3.1 authenticate()

```
User es.ull.esit.app.middleware.service.AuthService.authenticate (
            String username,
            String password ) throws Exception  [inline]
```

Attempts to authenticate a user against the backend.

```
Validates username and password locally, then calls
ApiClient.login(username, password). If validation fails or the
server rejects the credentials, an exception is thrown.
```

Parameters

username	[String] Username entered by the user.
password	[String] Password entered by the user.

Returns

[User] Authenticated user object returned by the backend.

Exceptions

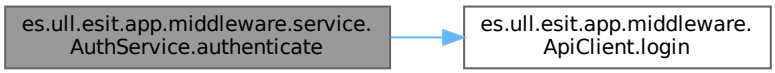
Exception	If validation fails or the backend returns an error.

Definition at line 38 of file AuthService.java.

```
00038                                                                        {
00039     // 1. Local Validation.
00040     if (username == null || username.trim().isEmpty()) {
00041       throw new IllegalArgumentException("Username cannot be empty.");
00042     }
00043     if (password == null || password.isEmpty()) {
00044       throw new IllegalArgumentException("Password cannot be empty.");
00045     }
00046
00047     // 2. Call API.
00048     return client.login(username, password);
00049   }
```

References es.ull.esit.app.middleware.service.AuthService.client, and es.ull.esit.app.middleware.ApiClient.login().

Here is the call graph for this function:



7.11.4 Member Data Documentation

7.11.4.1 client

```
final ApiClient es.ull.esit.app.middleware.service.AuthService.client  [private]
```

REST API client used to communicate with the backend login endpoint.

Definition at line 15 of file AuthService.java.

Referenced by es.ull.esit.app.middleware.service.AuthService.authenticate(), and es.ull.esit.app.middleware.service.AuthService.Auth

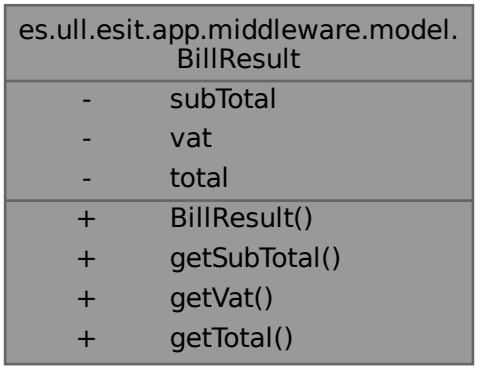
The documentation for this class was generated from the following file:

- AuthService.java

7.12 es.ull.esit.app.middleware.model.BillResult Class Reference

Data Transfer Object representing the result of a bill calculation.

Collaboration diagram for es.ull.esit.app.middleware.model.BillResult:



Public Member Functions

- BillResult (double subTotal, double vat, double total)

  Constructs a bill result with the given amounts.
- double getSubTotal ()

  Gets the subtotal amount.
- double getVat ()

  Gets the VAT amount.
- double getTotal ()

  Gets the total amount.

Private Attributes

- final double subTotal

  Calculated subtotal amount before VAT.
- final double vat

  Calculated VAT amount.
- final double total

  Final total amount including VAT.

7.12.1 Detailed Description

Data Transfer Object representing the result of a bill calculation.

```
It stores subtotal, VAT and total and is used exclusively on the client
side by the OrderService.
```

Definition at line 9 of file BillResult.java.

7.12.2 Constructor & Destructor Documentation

7.12.2.1 BillResult()

```
es.ull.esit.app.middleware.model.BillResult.BillResult (
            double subTotal,
            double vat,
            double total )  [inline]
```

Constructs a bill result with the given amounts.

Parameters

subTotal	[double] Subtotal amount before VAT.
vat	[double] VAT amount.
total	[double] Final total amount including VAT.

Definition at line 27 of file BillResult.java.

```
00027                                                       {
00028     this.subTotal = subTotal;
00029     this.vat = vat;
00030     this.total = total;
00031   }
```

References es.ull.esit.app.middleware.model.BillResult.subTotal, es.ull.esit.app.middleware.model.BillResult.total, and es.ull.esit.app.middleware.model.BillResult.vat.

7.12.3 Member Function Documentation

7.12.3.1 getSubTotal()

```
double es.ull.esit.app.middleware.model.BillResult.getSubTotal ( )  [inline]
```

Gets the subtotal amount.

Returns

[double] Subtotal amount before VAT.

Definition at line 38 of file BillResult.java.

```
00038                                {
00039     return subTotal;
00040   }
```

References es.ull.esit.app.middleware.model.BillResult.subTotal.

7.12.3.2 getTotal()

```
double es.ull.esit.app.middleware.model.BillResult.getTotal ( )  [inline]
```

Gets the total amount.

Returns

[double] Final total amount including VAT.

Definition at line 56 of file BillResult.java.

```
00056                               {
00057     return total;
00058   }
```

References es.ull.esit.app.middleware.model.BillResult.total.

7.12.3.3 getVat()

```
double es.ull.esit.app.middleware.model.BillResult.getVat ( )  [inline]
```

Gets the VAT amount.

Returns

[double] VAT amount.

Definition at line 47 of file BillResult.java.

```
00047                       {
00048     return vat;
00049   }
```

References es.ull.esit.app.middleware.model.BillResult.vat.

7.12.4 Member Data Documentation

7.12.4.1 subTotal

```
final double es.ull.esit.app.middleware.model.BillResult.subTotal  [private]
```

Calculated subtotal amount before VAT.

Definition at line 12 of file BillResult.java.

Referenced by es.ull.esit.app.middleware.model.BillResult.BillResult(), and es.ull.esit.app.middleware.model.BillResult.getSubTotal().

7.12.4.2 total

```
final double es.ull.esit.app.middleware.model.BillResult.total  [private]
```

Final total amount including VAT.

Definition at line 18 of file BillResult.java.

Referenced by es.ull.esit.app.middleware.model.BillResult.BillResult(), and es.ull.esit.app.middleware.model.BillResult.getTotal().

7.12.4.3 vat

```
final double es.ull.esit.app.middleware.model.BillResult.vat  [private]
```

Calculated VAT amount.

Definition at line 15 of file BillResult.java.

Referenced by es.ull.esit.app.middleware.model.BillResult.BillResult(), and es.ull.esit.app.middleware.model.BillResult.getVat().

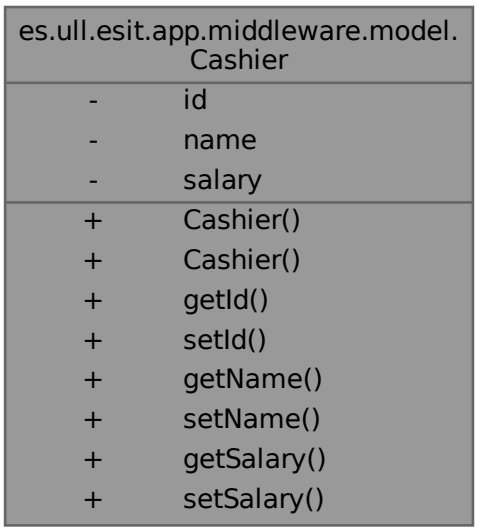
The documentation for this class was generated from the following file:

- BillResult.java

7.13 es.ull.esit.app.middleware.model.Cashier Class Reference

Client-side model representing a cashier returned by the backend.

Collaboration diagram for es.ull.esit.app.middleware.model.Cashier:



Public Member Functions

- Cashier ()

  Default constructor required for JSON deserialization.
- Cashier (Long id, String name, Integer salary)

  Constructs a cashier with all fields.
- Long getId ()

  Gets the cashier identifier.
- void setId (Long id)

  Sets the cashier identifier.
- String getName ()

  Gets the cashier name.
- void setName (String name)

  Sets the cashier name.
- Integer getSalary ()

  Gets the cashier salary.
- void setSalary (Integer salary)

  Sets the cashier salary.

Private Attributes

- Long id

  Unique identifier of the cashier (JSON property "id").
- String name

  Cashier name (JSON property "name").
- Integer salary

  Cashier salary (JSON property "salary").

7.13.1 Detailed Description

Client-side model representing a cashier returned by the backend.

```
It is used by the Swing application to display and manage cashier
information obtained from the "/api/cashiers" endpoint.
```

Definition at line 11 of file src/main/java/es/ull/esit/app/middleware/model/Cashier.java.

7.13.2 Constructor & Destructor Documentation

7.13.2.1 Cashier() [1/2]

```
es.ull.esit.app.middleware.model.Cashier.Cashier ( )  [inline]
```

Default constructor required for JSON deserialization.

Definition at line 28 of file src/main/java/es/ull/esit/app/middleware/model/Cashier.java.

```
00028                     {
00029     }
```

7.13.2.2 Cashier() [2/2]

```
es.ull.esit.app.middleware.model.Cashier.Cashier (
            Long id,
            String name,
            Integer salary )  [inline]
```

Constructs a cashier with all fields.

Parameters

id	[Long] Unique identifier of the cashier.
name	[String] Name of the cashier.
salary	[Integer] Salary of the cashier.

Definition at line 38 of file src/main/java/es/ull/esit/app/middleware/model/Cashier.java.

```
00038                                                     {
00039         this.id = id;
00040         this.name = name;
```

```
00041     this.salary = salary;
00042   }
```

References es.ull.esit.app.middleware.model.Cashier.id, es.ull.esit.app.middleware.model.Cashier.name, and es.ull.esit.app.middleware.model.Cashier.salary.

7.13.3 Member Function Documentation

7.13.3.1 getId()

```
Long es.ull.esit.app.middleware.model.Cashier.getId ( )  [inline]
```

Gets the cashier identifier.

Returns

[Long] Unique identifier of the cashier.

Definition at line 49 of file src/main/java/es/ull/esit/app/middleware/model/Cashier.java.

```
00049                     {
00050     return id;
00051   }
```

References es.ull.esit.app.middleware.model.Cashier.id.

7.13.3.2 getName()

```
String es.ull.esit.app.middleware.model.Cashier.getName ( )  [inline]
```

Gets the cashier name.

Returns

[String] Name of the cashier.

Definition at line 67 of file src/main/java/es/ull/esit/app/middleware/model/Cashier.java.

```
00067                       {
00068     return name;
00069   }
```

References es.ull.esit.app.middleware.model.Cashier.name.

7.13.3.3 getSalary()

```
Integer es.ull.esit.app.middleware.model.Cashier.getSalary ( )  [inline]
```

Gets the cashier salary.

Returns

[Integer] Salary of the cashier.

Definition at line 85 of file src/main/java/es/ull/esit/app/middleware/model/Cashier.java.

```
00085                         {
00086     return salary;
00087   }
```

References es.ull.esit.app.middleware.model.Cashier.salary.

7.13.3.4 setId()

```
void es.ull.esit.app.middleware.model.Cashier.setId (
            Long id )  [inline]
```

Sets the cashier identifier.

Parameters

id	[Long] Unique identifier of the cashier.

Definition at line 58 of file src/main/java/es/ull/esit/app/middleware/model/Cashier.java.

```
00058                                {
00059     this.id = id;
00060   }
```

References es.ull.esit.app.middleware.model.Cashier.id.

7.13.3.5 setName()

```
void es.ull.esit.app.middleware.model.Cashier.setName (
             String name )  [inline]
```

Sets the cashier name.

Parameters

name	[String] Name of the cashier.

Definition at line 76 of file src/main/java/es/ull/esit/app/middleware/model/Cashier.java.

```
00076                                    {
00077     this.name = name;
00078   }
```

References es.ull.esit.app.middleware.model.Cashier.name.

7.13.3.6 setSalary()

```
void es.ull.esit.app.middleware.model.Cashier.setSalary (
             Integer salary )  [inline]
```

Sets the cashier salary.

Parameters

salary	[Integer] Salary of the cashier.

Definition at line 94 of file src/main/java/es/ull/esit/app/middleware/model/Cashier.java.

```
00094                                      {
00095     this.salary = salary;
00096   }
```

References es.ull.esit.app.middleware.model.Cashier.salary.

7.13.4 Member Data Documentation

7.13.4.1 id

```
Long es.ull.esit.app.middleware.model.Cashier.id  [private]
```

Unique identifier of the cashier (JSON property "id").

Definition at line 15 of file src/main/java/es/ull/esit/app/middleware/model/Cashier.java.

Referenced by es.ull.esit.app.middleware.model.Cashier.Cashier(), es.ull.esit.app.middleware.model.Cashier.getId(), and es.ull.esit.app.middleware.model.Cashier.setId().

7.13.4.2 name

```
String es.ull.esit.app.middleware.model.Cashier.name  [private]
```

Cashier name (JSON property "name").

Definition at line 19 of file src/main/java/es/ull/esit/app/middleware/model/Cashier.java.

Referenced by es.ull.esit.app.middleware.model.Cashier.Cashier(), es.ull.esit.app.middleware.model.Cashier.getName(), and es.ull.esit.app.middleware.model.Cashier.setName().

7.13.4.3 salary

```
Integer es.ull.esit.app.middleware.model.Cashier.salary  [private]
```

Cashier salary (JSON property "salary").

Definition at line 23 of file src/main/java/es/ull/esit/app/middleware/model/Cashier.java.

Referenced by es.ull.esit.app.middleware.model.Cashier.Cashier(), es.ull.esit.app.middleware.model.Cashier.getSalary(), and es.ull.esit.app.middleware.model.Cashier.setSalary().

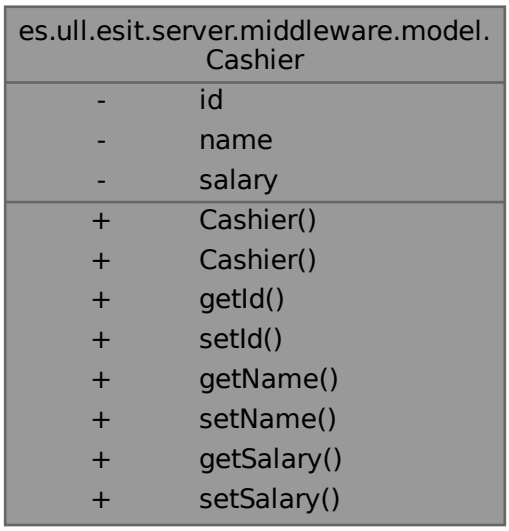
The documentation for this class was generated from the following file:

- src/main/java/es/ull/esit/app/middleware/model/Cashier.java

7.14 es.ull.esit.server.middleware.model.Cashier Class Reference

JPA entity that represents a cashier in the system.

Collaboration diagram for es.ull.esit.server.middleware.model.Cashier:



Public Member Functions

- Cashier ()
  Default constructor required by JPA.
- Cashier (Long id, String name, Integer salary)
  Full constructor.
- Long getId ()
  Gets the cashier identifier.
- void setId (Long id)
  Sets the cashier identifier.
- String getName ()
  Gets the cashier name (username).
- void setName (String name)
  Sets the cashier name (username).
- Integer getSalary ()
  Gets the cashier salary.
- void setSalary (Integer salary)
  Sets the cashier salary.

Private Attributes

- Long id
  Primary key (unique identifier) of the cashier (column "cashier_id").
- String name
  Cashier username (column "cashier_name").
- Integer salary
  Cashier salary (column "cashier_salary").

7.14.1 Detailed Description

JPA entity that represents a cashier in the system.

```
It is mapped to the "cashier" table used by the REST API.
Each row contains a unique identifier, name (username), and salary.
```

Definition at line 15 of file server/src/main/java/es/ull/esit/server/middleware/model/Cashier.java.

7.14.2 Constructor & Destructor Documentation

7.14.2.1 Cashier() [1/2]

```
es.ull.esit.server.middleware.model.Cashier.Cashier ( )  [inline]
```

Default constructor required by JPA.

Definition at line 33 of file server/src/main/java/es/ull/esit/server/middleware/model/Cashier.java.

```
00033                     {
00034     }
```

7.14.2.2 Cashier() [2/2]

```
es.ull.esit.server.middleware.model.Cashier.Cashier (
            Long id,
            String name,
            Integer salary )  [inline]
```

Full constructor.

Parameters

id	[Long] Identifier of the cashier.
name	[String] Username of the cashier.
salary	[Integer] Salary of the cashier.

Definition at line 43 of file server/src/main/java/es/ull/esit/server/middleware/model/Cashier.java.

```
00043                                                 {
00044     this.id = id;
00045     this.name = name;
00046     this.salary = salary;
00047   }
```

7.14.3 Member Function Documentation

7.14.3.1 getId()

```
Long es.ull.esit.server.middleware.model.Cashier.getId ( )  [inline]
```

Gets the cashier identifier.

Returns

[Long] Cashier id.

Definition at line 54 of file server/src/main/java/es/ull/esit/server/middleware/model/Cashier.java.

```
00054                       {
00055     return id;
00056   }
```

7.14.3.2 getName()

```
String es.ull.esit.server.middleware.model.Cashier.getName ( )  [inline]
```

Gets the cashier name (username).

Returns

[String] Cashier name (username).

Definition at line 73 of file server/src/main/java/es/ull/esit/server/middleware/model/Cashier.java.

```
00073                           {
00074     return name;
00075   }
```

7.14.3.3 getSalary()

```
Integer es.ull.esit.server.middleware.model.Cashier.getSalary ( )  [inline]
```

Gets the cashier salary.

Returns

[Integer] cashier salary.

Definition at line 91 of file server/src/main/java/es/ull/esit/server/middleware/model/Cashier.java.

```
00091                               {
00092     return salary;
00093   }
```

7.14.3.4 setId()

```
void es.ull.esit.server.middleware.model.Cashier.setId (
            Long id )  [inline]
```

Sets the cashier identifier.

Must be unique.

Parameters

id	[Long] New cashier id.

Definition at line 64 of file server/src/main/java/es/ull/esit/server/middleware/model/Cashier.java.

```
00064                                {
00065     this.id = id;
00066   }
```

7.14.3.5 setName()

```
void es.ull.esit.server.middleware.model.Cashier.setName (
            String name )  [inline]
```

Sets the cashier name (username).

Parameters

name	[String] New cashier name (username).

Definition at line 82 of file server/src/main/java/es/ull/esit/server/middleware/model/Cashier.java.

```
00082                                    {
00083     this.name = name;
00084   }
```

7.14.3.6 setSalary()

```
void es.ull.esit.server.middleware.model.Cashier.setSalary (
            Integer salary )  [inline]
```

Sets the cashier salary.

Parameters

salary	[Integer] New cashier salary.

Definition at line 100 of file server/src/main/java/es/ull/esit/server/middleware/model/Cashier.java.

```
00100                                      {
00101     this.salary = salary;
00102   }
```

7.14.4 Member Data Documentation

7.14.4.1 id

```
Long es.ull.esit.server.middleware.model.Cashier.id  [private]
```

Primary key (unique identifier) of the cashier (column "cashier_id").

Definition at line 20 of file server/src/main/java/es/ull/esit/server/middleware/model/Cashier.java.

7.14.4.2 name

```
String es.ull.esit.server.middleware.model.Cashier.name  [private]
```

Cashier username (column "cashier_name").

Definition at line 24 of file server/src/main/java/es/ull/esit/server/middleware/model/Cashier.java.

7.14.4.3 salary

```
Integer es.ull.esit.server.middleware.model.Cashier.salary  [private]
```

Cashier salary (column "cashier_salary").

Definition at line 28 of file server/src/main/java/es/ull/esit/server/middleware/model/Cashier.java.

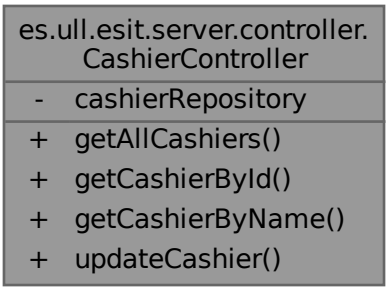
The documentation for this class was generated from the following file:

- server/src/main/java/es/ull/esit/server/middleware/model/Cashier.java

7.15 es.ull.esit.server.controller.CashierController Class Reference

REST controller for managing cashiers.

Collaboration diagram for es.ull.esit.server.controller.CashierController:



Public Member Functions

- ResponseEntity< List< Cashier > > getAllCashiers ()

  Returns all cashiers.
- ResponseEntity< Cashier > getCashierById (@PathVariable Long id)

  Returns a single cashier by ID.
- ResponseEntity< Cashier > getCashierByName (@PathVariable String name)

  Returns a single cashier by name (username).
- ResponseEntity< Cashier > updateCashier (@PathVariable Long id, @RequestBody Cashier cashier)

  Updates an existing cashier, but only the name and salary fields.

Private Attributes

- CashierRepository cashierRepository

7.15.1 Detailed Description

REST controller for managing cashiers.

```
Provides CRUD operations for Cashier entities using the path
"/api/cashiers".

Cashiers are created automatically when users with 'CASHIER' role are
added to the table 'users'.
This controller is mainly READ-ONLY and allows updating salary/name.
```

Definition at line 25 of file CashierController.java.

7.15.2 Member Function Documentation

7.15.2.1 getAllCashiers()

```
ResponseEntity< List< Cashier > > es.ull.esit.server.controller.CashierController.getAll↩
Cashiers ( )  [inline]
```

Returns all cashiers.

```
HTTP GET /api/cashiers
```

Returns

[ResponseEntity<List<Cashier>>] List of cashiers with HTTP 200 status.

Definition at line 40 of file CashierController.java.

```
00040                                                    {
00041     List<Cashier> cashiers = cashierRepository.findAll();
00042     return ResponseEntity.ok(cashiers);
00043   }
```

7.15.2.2 getCashierById()

```
ResponseEntity< Cashier > es.ull.esit.server.controller.CashierController.getCashierById (
            @PathVariable Long id )  [inline]
```

Returns a single cashier by ID.

```
HTTP GET /api/cashiers/{id}
```

Parameters

id	[Long] Identifier of the cashier.

Returns

[ResponseEntity<Cashier>] The cashier if found with HTTP 200 status, or HTTP 404 if not found.

Definition at line 55 of file CashierController.java.

```
00055                                                                   {
00056     Optional<Cashier> cashier = cashierRepository.findById(id);
00057     return cashier
00058         .map(ResponseEntity::ok)
00059         .orElseGet(() -> ResponseEntity.notFound().build());
00060   }
```

7.15.2.3 getCashierByName()

```
ResponseEntity< Cashier > es.ull.esit.server.controller.CashierController.getCashierByName (
            @PathVariable String name )  [inline]
```

Returns a single cashier by name (username).

```
HTTP GET /api/cashiers/name/{name}
```

Parameters

name	[String] Name of the cashier.

Returns

[ResponseEntity<Cashier>] The cashier if found with HTTP 200 status, or HTTP 404 if not found.

Definition at line 72 of file CashierController.java.

```
00072                                                                       {
00073     Optional<Cashier> cashier = cashierRepository.findByName(name);
00074     return cashier
00075         .map(ResponseEntity::ok)
00076         .orElseGet(() -> ResponseEntity.notFound().build());
00077   }
```

7.15.2.4 updateCashier()

```
ResponseEntity< Cashier > es.ull.esit.server.controller.CashierController.updateCashier (
            @PathVariable Long id,
            @RequestBody Cashier cashier )  [inline]
```

Updates an existing cashier, but only the name and salary fields.

```
HTTP PUT /api/cashiers/{id}
```

Parameters

id	[Long] Identifier of the cashier to update.
cashier	[Cashier] New data for the cashier.

Returns

[ResponseEntity<Cashier>] The updated cashier with HTTP 200 status, or HTTP 404 if not found.

Definition at line 90 of file CashierController.java.

```
00091                                       {
00092     Optional<Cashier> existingOpt = cashierRepository.findById(id);
00093     if (!existingOpt.isPresent()) {
00094       return ResponseEntity.notFound().build();
00095     }
00096
00097     Cashier existing = existingOpt.get();
00098     existing.setName(cashier.getName());
00099     existing.setSalary(cashier.getSalary());
00100
00101     Cashier updated = cashierRepository.save(existing);
00102     return ResponseEntity.ok(updated);
00103   }
```

7.15.3 Member Data Documentation

7.15.3.1 cashierRepository

```
CashierRepository es.ull.esit.server.controller.CashierController.cashierRepository  [private]
```

Definition at line 29 of file CashierController.java.

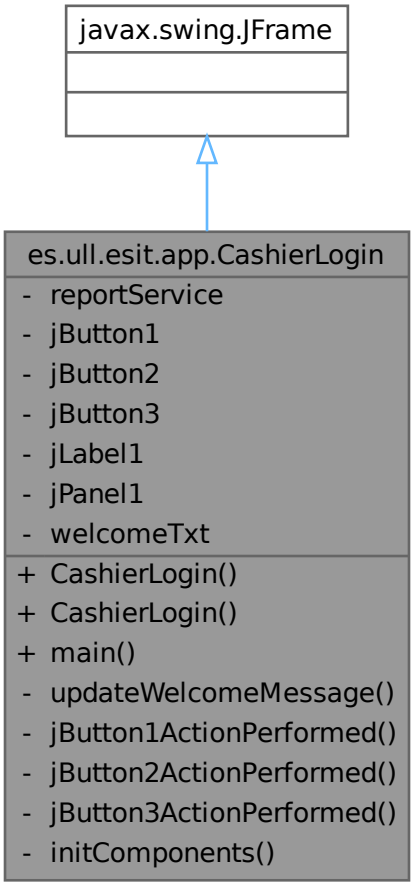
The documentation for this class was generated from the following file:

- CashierController.java

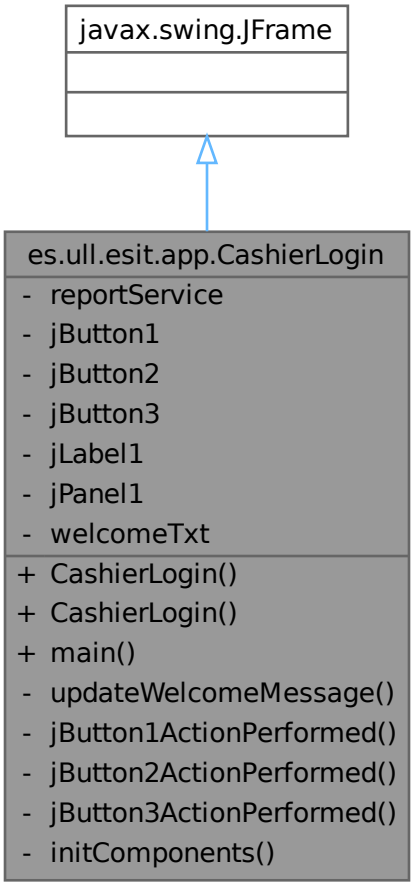
7.16 es.ull.esit.app.CashierLogin Class Reference

Login window for authenticating cashiers.

Inheritance diagram for es.ull.esit.app.CashierLogin:



Collaboration diagram for es.ull.esit.app.CashierLogin:



Public Member Functions

- CashierLogin ()

  Default constructor.
- CashierLogin (String name)

  Constructor with cashier name.

Static Public Member Functions

- static void main (String args[ ])

  Standalone entry point for testing the Cashier window.

Private Member Functions

- void updateWelcomeMessage (String name)

  Updates the welcome message label.
- void jButton1ActionPerformed (java.awt.event.ActionEvent evt)

  Action handler for the "About us" button.
- void jButton2ActionPerformed (java.awt.event.ActionEvent evt)

  Action handler for the "Menu" button.
- void jButton3ActionPerformed (java.awt.event.ActionEvent evt)

  Action handler for the "LogOut" button.
- void initComponents ()

  Initializes GUI components.

Private Attributes

- final ReportService reportService

  Service used to retrieve reports and basic status from the backend.
- javax.swing.JButton jButton1

  Button that opens the "About us" information window.
- javax.swing.JButton jButton2

  Button that opens the general menu window (Frame1).
- javax.swing.JButton jButton3

  Button used to log out and return to the login screen.
- javax.swing.JLabel jLabel1

  Label that displays the logo or application image.
- javax.swing.JPanel jPanel1

  Main container panel for all UI elements.
- javax.swing.JLabel welcomeTxt

  Label that displays the welcome message for the cashier.

7.16.1 Detailed Description

Login window for authenticating cashiers.

```
It is displayed after a successful login with role CASHIER.
Shows a Swing form that lets cashiers:
- access to the general menu window used to register orders.
- access to the "About us" information window.
- log out and return to the login window.

Additionally, some actions perform background calls to the backend
using ReportService to:
- retrieve cashier information.
- check the menu or system status.
```

Definition at line 23 of file CashierLogin.java.

7.16.2 Constructor & Destructor Documentation

7.16.2.1 CashierLogin() [1/2]

```
es.ull.esit.app.CashierLogin.CashierLogin ( )  [inline]
```

Default constructor.

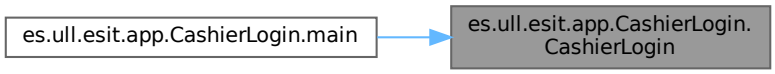
```
Creates a cashier window without a specific name in the welcome
message.
Internally delegates to the constructor that accepts a name.
```

Definition at line 35 of file CashierLogin.java.

```
00035                         {
00036     this((String) null);
00037   }
```

Referenced by es.ull.esit.app.CashierLogin.main().

Here is the caller graph for this function:



7.16.2.2 CashierLogin() [2/2]

```
es.ull.esit.app.CashierLogin.CashierLogin (
             String name )  [inline]
```

Constructor with cashier name.

```
Creates the cashier window, initializes the GUI, builds an
ApiClient pointing to the backend, creates a ReportService and
updates the welcome message using the cashier name.
```

Parameters

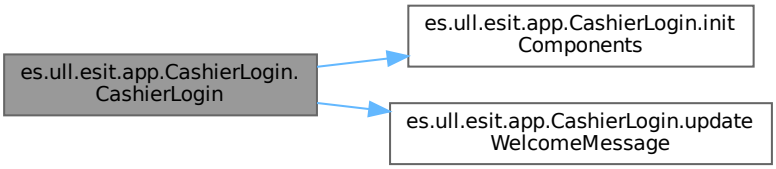
name	[String] Name of the cashier returned by the backend (or null for a generic welcome message).

Definition at line 49 of file CashierLogin.java.

```
00049                                 {
00050     initComponents();
00051
00052     ApiClient client = new ApiClient("http://localhost:8080");
00053     this.reportService = new ReportService(client);
00054
00055     updateWelcomeMessage(name);
00056   }
```

References es.ull.esit.app.CashierLogin.initComponents(), and es.ull.esit.app.CashierLogin.updateWelcomeMessage().

Here is the call graph for this function:



7.16.3 Member Function Documentation

7.16.3.1 initComponents()

```
void es.ull.esit.app.CashierLogin.initComponents ( )  [inline], [private]
```

Initializes GUI components.

```
Generated by the Form Editor: do not modify.
Sets up the welcome label, logo image, buttons for "About us", "Menu",
"LogOut",
and the layout and configuration of the window.
```

Definition at line 195 of file CashierLogin.java.

```
00195                                {
00196
00197     jPanel1 = new javax.swing.JPanel();
00198     welcomeTxt = new javax.swing.JLabel();
00199     jButton1 = new javax.swing.JButton();
00200     jButton2 = new javax.swing.JButton();
00201     jLabel1 = new javax.swing.JLabel();
00202     jButton3 = new javax.swing.JButton();
00203
00204     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00205     setTitle("Cashier");
00206     setResizable(false);
00207
00208     jPanel1.setBackground(new java.awt.Color(248, 244, 230));
00209
00210     welcomeTxt.setFont(new java.awt.Font("Yu Gothic UI", 1, 24)); // NOI18N
00211     welcomeTxt.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00212     welcomeTxt.setText("Welcome Cashier");
00213
00214     jButton1.setBackground(new java.awt.Color(153, 153, 153));
00215     jButton1.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00216     jButton1.setText("About us");
00217     jButton1.addActionListener(new java.awt.event.ActionListener() {
00218       public void actionPerformed(java.awt.event.ActionEvent evt) {
00219         jButton1ActionPerformed(evt);
00220       }
00221     });
00222
00223     jButton2.setBackground(new java.awt.Color(153, 153, 153));
00224     jButton2.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00225     jButton2.setText("Menu");
00226     jButton2.addActionListener(new java.awt.event.ActionListener() {
00227       public void actionPerformed(java.awt.event.ActionEvent evt) {
00228         jButton2ActionPerformed(evt);
00229       }
00230     });
00231
```

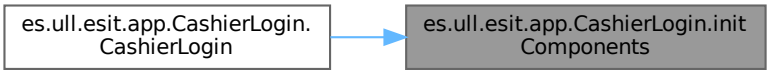
```
00232     jLabel1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/ui/screenshot.png"))); //
      NOI18N
00233
00234     jButton3.setBackground(new java.awt.Color(255, 255, 255));
00235     jButton3.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00236     jButton3.setText("LogOut");
00237     jButton3.addActionListener(new java.awt.event.ActionListener() {
00238       public void actionPerformed(java.awt.event.ActionEvent evt) {
00239         jButton3ActionPerformed(evt);
00240       }
00241     });
00242
00243     javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
00244     jPanel1.setLayout(jPanel1Layout);
00245     jPanel1Layout.setHorizontalGroup(
00246         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00247             .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
      jPanel1Layout.createSequentialGroup()
00248                 .addContainerGap(109, Short.MAX_VALUE)
00249                 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00250                     .addComponent(welcomeTxt, javax.swing.GroupLayout.PREFERRED_SIZE, 299,
00251                         javax.swing.GroupLayout.PREFERRED_SIZE)
00252
      .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00253                         .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
      jPanel1Layout.createSequentialGroup()
00254                             .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 320,
00255                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00256                             .addGap(98, 98, 98))
00257                         .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
      jPanel1Layout.createSequentialGroup()
00258                             .addComponent(jLabel1)
00259                             .addGap(183, 183, 183))
00260                         .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
      jPanel1Layout.createSequentialGroup()
00261                             .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 141,
00262                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00263                             .addGap(190, 190, 190)))))
00264             .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00265                 .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
      jPanel1Layout.createSequentialGroup()
00266                     .addContainerGap(108, Short.MAX_VALUE)
00267                     .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 320,
00268                         javax.swing.GroupLayout.PREFERRED_SIZE)
00269                     .addGap(99, 99, 99))));
00270     jPanel1Layout.setVerticalGroup(
00271         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00272             .addGroup(jPanel1Layout.createSequentialGroup()
00273                 .addGap(45, 45, 45)
00274                 .addComponent(welcomeTxt, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
00275                     javax.swing.GroupLayout.PREFERRED_SIZE)
00276                 .addGap(18, 18, 18)
00277                 .addComponent(jLabel1)
00278                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 99,
      Short.MAX_VALUE)
00279                 .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00280                     javax.swing.GroupLayout.PREFERRED_SIZE)
00281                 .addGap(57, 57, 57)
00282                 .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00283                     javax.swing.GroupLayout.PREFERRED_SIZE)
00284                 .addGap(68, 68, 68))
00285             .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00286                 .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
      jPanel1Layout.createSequentialGroup()
00287                     .addContainerGap(290, Short.MAX_VALUE)
00288                     .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00289                         javax.swing.GroupLayout.PREFERRED_SIZE)
00290                     .addGap(242, 242, 242))));
00291
00292     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00293     getContentPane().setLayout(layout);
00294     layout.setHorizontalGroup(
00295         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00296             .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00297                 Short.MAX_VALUE));
00298     layout.setVerticalGroup(
00299         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00300             .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00301                 Short.MAX_VALUE));
00302
00303     pack();
00304     setLocationRelativeTo(null);
00305   }// </editor-fold>//GEN-END:initComponents
```

References es.ull.esit.app.CashierLogin.jButton1, es.ull.esit.app.CashierLogin.jButton2, es.ull.esit.app.CashierLogin.jButton3,

es.ull.esit.app.CashierLogin.jLabel1, es.ull.esit.app.CashierLogin.jPanel1, and es.ull.esit.app.CashierLogin.welcomeTxt.

Referenced by es.ull.esit.app.CashierLogin.CashierLogin().

Here is the caller graph for this function:



7.16.3.2 jButton1ActionPerformed()

```
void es.ull.esit.app.CashierLogin.jButton1ActionPerformed (
            java.awt.event.ActionEvent evt )  [inline], [private]
```

Action handler for the "About us" button.

```
Action steps:
- Starts a background thread.
- Uses ReportService.getCashierInfo() to retrieve cashier data from
the backend and logs the number of cashiers found.
- Opens the Info window in the Event Dispatch Thread.
- Closes the current Cashier window.

Any error retrieving data is logged to the standard error stream
but does not prevent opening the Info window.
```

Parameters

evt	[java.awt.event.ActionEvent] Action event triggered by button click.

Definition at line 90 of file CashierLogin.java.

```
00090                                                                   {
00091     new Thread(() -> {
00092       try {
00093         List<es.ull.esit.app.middleware.model.Cashier> cashiers = reportService.getCashierInfo();
00094         System.out.println("Found " + cashiers.size() + " cashiers in DB.");
00095
00096       } catch (Exception ex) {
00097         System.err.println("Warning: Could not fetch cashier stats: " + ex.getMessage());
00098       }
00099       SwingUtilities.invokeLater(() -> {
00100         new AboutUs().setVisible(true);
00101         dispose();
00102       });
00103     }).start();
00104   }
```

References es.ull.esit.app.CashierLogin.reportService.

7.16.3.3 jButton2ActionPerformed()

```
void es.ull.esit.app.CashierLogin.jButton2ActionPerformed (
            java.awt.event.ActionEvent evt )  [inline], [private]
```

Action handler for the "Menu" button.

```
Action steps:
- Starts a background thread.
- Uses ReportService.checkMenuStatus() to verify communication
with the backend and logs the returned status.
- Opens the general menu window (Frame1) in the Event Dispatch
Thread.
- Closes the current Cashier window.

Any error while checking the status is logged to the standard
error stream but does not prevent opening the menu window.
```

Parameters

evt	[java.awt.event.ActionEvent] Action event triggered by button click.

Definition at line 123 of file CashierLogin.java.

```
00123                                                                {
00124     new Thread(() -> {
00125       try {
00126         String status = reportService.checkMenuStatus();
00127         System.out.println(status);
00128
00129       } catch (Exception ex) {
00130         System.err.println("Error checking menu status: " + ex.getMessage());
00131       }
00132
00133       SwingUtilities.invokeLater(() -> {
00134         new Order().setVisible(true);
00135         dispose();
00136       });
00137     }).start();
00138   }
```

References es.ull.esit.app.CashierLogin.reportService.

7.16.3.4 jButton3ActionPerformed()

```
void es.ull.esit.app.CashierLogin.jButton3ActionPerformed (
             java.awt.event.ActionEvent evt )  [inline], [private]
```

Action handler for the "LogOut" button.

```
Action steps:
- Opens the Login window.
- Closes the current Cashier window.

Any server-side logout or token invalidation, if needed, should be
handled outside this class.
```

Parameters

evt	[java.awt.event.ActionEvent] Action event triggered by button click.

Definition at line 153 of file CashierLogin.java.

```
00153                                                                {
00154     new Login().setVisible(true);
00155     dispose();
00156   }
```

7.16.3.5 main()

```
static void es.ull.esit.app.CashierLogin.main (
            String args[] )  [inline], [static]
```

Standalone entry point for testing the Cashier window.

```
Sets the Nimbus look and feel if available and shows an instance
of Cashier without a specific cashier name. In the normal
application flow this window is started from the Login class after
a successful cashier login.
```

Parameters

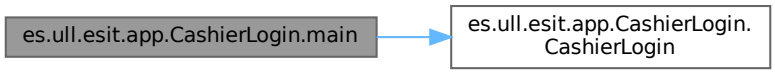
args	[String[]] Command line arguments (not used).

Definition at line 168 of file CashierLogin.java.

```
00168                                       {
00169     try {
00170       for (javax.swing.UIManager.LookAndFeelInfo info :
      javax.swing.UIManager.getInstalledLookAndFeels()) {
00171         if ("Nimbus".equals(info.getName())) {
00172           javax.swing.UIManager.setLookAndFeel(info.getClassName());
00173           break;
00174         }
00175       }
00176     } catch (ClassNotFoundException | InstantiationException | IllegalAccessException
00177         | javax.swing.UnsupportedLookAndFeelException ex) {
00178
      java.util.logging.Logger.getLogger(CashierLogin.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
00179     }
00180
00181     java.awt.EventQueue.invokeLater(() -> new CashierLogin().setVisible(true));
00182   }
```

References es.ull.esit.app.CashierLogin.CashierLogin().

Here is the call graph for this function:



7.16.3.6 updateWelcomeMessage()

```
void es.ull.esit.app.CashierLogin.updateWelcomeMessage (
            String name )  [inline], [private]
```

Updates the welcome message label.

```
If the name is null or empty, the label shows "Welcome Cashier".
Otherwise the label shows "Welcome " followed by the given name.
```

Parameters

name	[String] Cashier name or null.

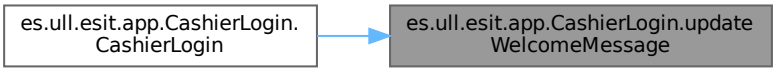
Definition at line 66 of file CashierLogin.java.

```
00066                                                {
00067     if (name == null || name.trim().isEmpty()) {
00068       welcomeTxt.setText("Welcome Cashier");
00069     } else {
00070       welcomeTxt.setText("Welcome " + name);
00071     }
00072   }
```

References es.ull.esit.app.CashierLogin.welcomeTxt.

Referenced by es.ull.esit.app.CashierLogin.CashierLogin().

Here is the caller graph for this function:



7.16.4 Member Data Documentation

7.16.4.1 jButton1

```
javax.swing.JButton es.ull.esit.app.CashierLogin.jButton1  [private]
```

Button that opens the "About us" information window.

Definition at line 309 of file CashierLogin.java.

Referenced by es.ull.esit.app.CashierLogin.initComponents().

7.16.4.2 jButton2

```
javax.swing.JButton es.ull.esit.app.CashierLogin.jButton2  [private]
```

Button that opens the general menu window (Frame1).

Definition at line 311 of file CashierLogin.java.

Referenced by es.ull.esit.app.CashierLogin.initComponents().

7.16.4.3 jButton3

```
javax.swing.JButton es.ull.esit.app.CashierLogin.jButton3  [private]
```

Button used to log out and return to the login screen.

Definition at line 313 of file CashierLogin.java.

Referenced by es.ull.esit.app.CashierLogin.initComponents().

7.16.4.4 jLabel1

```
javax.swing.JLabel es.ull.esit.app.CashierLogin.jLabel1  [private]
```

Label that displays the logo or application image.

Definition at line 315 of file CashierLogin.java.

Referenced by es.ull.esit.app.CashierLogin.initComponents().

7.16.4.5 jPanel1

```
javax.swing.JPanel es.ull.esit.app.CashierLogin.jPanel1  [private]
```

Main container panel for all UI elements.

Definition at line 317 of file CashierLogin.java.

Referenced by es.ull.esit.app.CashierLogin.initComponents().

7.16.4.6 reportService

```
final ReportService es.ull.esit.app.CashierLogin.reportService  [private]
```

Service used to retrieve reports and basic status from the backend.

Definition at line 26 of file CashierLogin.java.

Referenced by es.ull.esit.app.CashierLogin.jButton1ActionPerformed(), and es.ull.esit.app.CashierLogin.jButton2ActionPerformed().

7.16.4.7 welcomeTxt

```
javax.swing.JLabel es.ull.esit.app.CashierLogin.welcomeTxt  [private]
```

Label that displays the welcome message for the cashier.

Definition at line 319 of file CashierLogin.java.

Referenced by es.ull.esit.app.CashierLogin.initComponents(), and es.ull.esit.app.CashierLogin.updateWelcomeMessage().

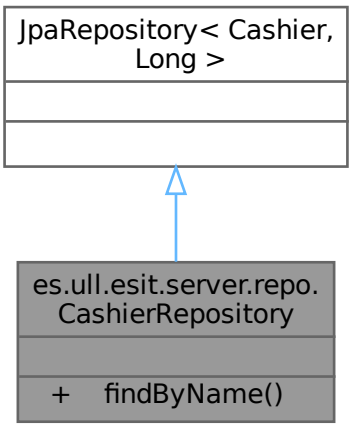
The documentation for this class was generated from the following file:

- CashierLogin.java

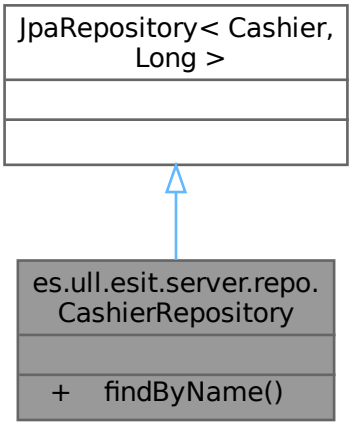
7.17 es.ull.esit.server.repo.CashierRepository Interface Reference

Repository interface for managing cashiers in the database.

Inheritance diagram for es.ull.esit.server.repo.CashierRepository:



Collaboration diagram for es.ull.esit.server.repo.CashierRepository:



Public Member Functions

- Optional< Cashier > findByName (String name)

  Finds a cashier by their name.

7.17.1 Detailed Description

Repository interface for managing cashiers in the database.

```
Extends JpaRepository to provide basic CRUD operations on the
"Cashier" table, such as findAll, findById, save and delete.
```

Definition at line 14 of file CashierRepository.java.

7.17.2 Member Function Documentation

7.17.2.1 findByName()

```
Optional< Cashier > es.ull.esit.server.repo.CashierRepository.findByName (
            String name )
```

Finds a cashier by their name.

Parameters

name	[String] Name of the cashier.

Returns

[Optional<Cashier>] The cashier if found, or empty if not found.

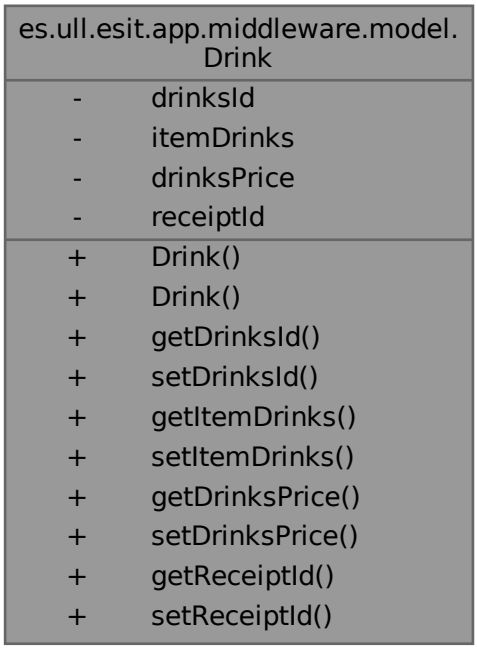
The documentation for this interface was generated from the following file:

- CashierRepository.java

7.18 es.ull.esit.app.middleware.model.Drink Class Reference

Client-side model representing a drink returned by the backend API.

Collaboration diagram for es.ull.esit.app.middleware.model.Drink:



Public Member Functions

- Drink ()

  Default constructor required for JSON deserialization.
- Drink (Long drinksId, String itemDrinks, Integer drinksPrice, Long receiptId)

  Constructs a drink with all fields.
- Long getDrinksId ()

  Gets the drink identifier.
- void setDrinksId (Long drinksId)

  Sets the drink identifier.
- String getItemDrinks ()

  Gets the drink item name.
- void setItemDrinks (String itemDrinks)

  Sets the drink item name.
- Integer getDrinksPrice ()

  Gets the drink price.
- void setDrinksPrice (Integer drinksPrice)

  Sets the drink price.
- Long getReceiptId ()

  Gets the identifier of the related receipt.
- void setReceiptId (Long receiptId)

  Sets the identifier of the related receipt.

Private Attributes

- Long drinksId

  Unique identifier of the drink (JSON property "drinksId").
- String itemDrinks

  Name of the drink item (JSON property "itemDrinks").
- Integer drinksPrice

  Price of the drink (JSON property "drinksPrice").
- Long receiptId

  Identifier of the receipt this drink belongs to (JSON property "receiptId").

7.18.1 Detailed Description

Client-side model representing a drink returned by the backend API.

```
This class is used to deserialize and manipulate drink data obtained
from the "/api/drinks" endpoint.
```

Definition at line 11 of file src/main/java/es/ull/esit/app/middleware/model/Drink.java.

7.18.2 Constructor & Destructor Documentation

7.18.2.1 Drink() [1/2]

```
es.ull.esit.app.middleware.model.Drink.Drink ( )  [inline]
```

Default constructor required for JSON deserialization.

Definition at line 35 of file src/main/java/es/ull/esit/app/middleware/model/Drink.java.

```
00035                 {
00036     }
```

7.18.2.2 Drink() [2/2]

```
es.ull.esit.app.middleware.model.Drink.Drink (
            Long drinksId,
            String itemDrinks,
            Integer drinksPrice,
            Long receiptId )  [inline]
```

Constructs a drink with all fields.

Parameters

drinksId	[Long] Unique identifier of the drink.
itemDrinks	[String] Name of the drink item.
drinksPrice	[Integer] Price of the drink.
receiptId	[Long] Identifier of the related receipt (optional).

Definition at line 46 of file src/main/java/es/ull/esit/app/middleware/model/Drink.java.

```
00046                                                                                {
00047     this.drinksId = drinksId;
00048     this.itemDrinks = itemDrinks;
00049     this.drinksPrice = drinksPrice;
00050     this.receiptId = receiptId;
00051   }
```

References es.ull.esit.app.middleware.model.Drink.drinksId, es.ull.esit.app.middleware.model.Drink.drinksPrice, es.ull.esit.app.middleware.model.Drink.itemDrinks, and es.ull.esit.app.middleware.model.Drink.receiptId.

7.18.3 Member Function Documentation

7.18.3.1 getDrinksId()

```
Long es.ull.esit.app.middleware.model.Drink.getDrinksId ( )  [inline]
```

Gets the drink identifier.

Returns

[Long] Unique identifier of the drink.

Definition at line 58 of file src/main/java/es/ull/esit/app/middleware/model/Drink.java.

```
00058                                {
00059     return drinksId;
00060   }
```

References es.ull.esit.app.middleware.model.Drink.drinksId.

7.18.3.2 getDrinksPrice()

```
Integer es.ull.esit.app.middleware.model.Drink.getDrinksPrice ( )  [inline]
```

Gets the drink price.

Returns

[Integer] Price of the drink.

Definition at line 94 of file src/main/java/es/ull/esit/app/middleware/model/Drink.java.

```
00094                                   {
00095     return drinksPrice;
00096   }
```

References es.ull.esit.app.middleware.model.Drink.drinksPrice.

7.18.3.3 getItemDrinks()

```
String es.ull.esit.app.middleware.model.Drink.getItemDrinks ( )  [inline]
```

Gets the drink item name.

Returns

[String] Name of the drink item.

Definition at line 76 of file src/main/java/es/ull/esit/app/middleware/model/Drink.java.

```
00076                                  {
00077     return itemDrinks;
00078   }
```

References es.ull.esit.app.middleware.model.Drink.itemDrinks.

7.18.3.4 getReceiptId()

```
Long es.ull.esit.app.middleware.model.Drink.getReceiptId ( )  [inline]
```

Gets the identifier of the related receipt.

Returns

[Long] Identifier of the receipt or null if not set.

Definition at line 112 of file src/main/java/es/ull/esit/app/middleware/model/Drink.java.

```
00112                               {
00113     return receiptId;
00114   }
```

References es.ull.esit.app.middleware.model.Drink.receiptId.

7.18.3.5 setDrinksId()

```
void es.ull.esit.app.middleware.model.Drink.setDrinksId (
            Long drinksId )  [inline]
```

Sets the drink identifier.

Parameters

drinksId	[Long] Unique identifier of the drink.

Definition at line 67 of file src/main/java/es/ull/esit/app/middleware/model/Drink.java.

```
00067                                       {
00068     this.drinksId = drinksId;
00069   }
```

References es.ull.esit.app.middleware.model.Drink.drinksId.

7.18.3.6 setDrinksPrice()

```
void es.ull.esit.app.middleware.model.Drink.setDrinksPrice (
            Integer drinksPrice )  [inline]
```

Sets the drink price.

Parameters

drinksPrice	[Integer] Price of the drink.

Definition at line 103 of file src/main/java/es/ull/esit/app/middleware/model/Drink.java.

```
00103                                           {
00104     this.drinksPrice = drinksPrice;
00105   }
```

References es.ull.esit.app.middleware.model.Drink.drinksPrice.

7.18.3.7 setItemDrinks()

```
void es.ull.esit.app.middleware.model.Drink.setItemDrinks (
            String itemDrinks )  [inline]
```

Sets the drink item name.

Parameters

itemDrinks	[String] Name of the drink item.

Definition at line 85 of file src/main/java/es/ull/esit/app/middleware/model/Drink.java.

```
00085                                           {
00086     this.itemDrinks = itemDrinks;
00087   }
```

References es.ull.esit.app.middleware.model.Drink.itemDrinks.

7.18.3.8 setReceiptId()

```
void es.ull.esit.app.middleware.model.Drink.setReceiptId (
            Long receiptId )  [inline]
```

Sets the identifier of the related receipt.

Parameters

receipt↩ Id	[Long] Identifier of the receipt.

Definition at line 121 of file src/main/java/es/ull/esit/app/middleware/model/Drink.java.

```
00121                                           {
00122     this.receiptId = receiptId;
00123   }
```

References es.ull.esit.app.middleware.model.Drink.receiptId.

7.18.4 Member Data Documentation

7.18.4.1 drinksId

```
Long es.ull.esit.app.middleware.model.Drink.drinksId  [private]
```

Unique identifier of the drink (JSON property "drinksId").

Definition at line 15 of file src/main/java/es/ull/esit/app/middleware/model/Drink.java.

Referenced by es.ull.esit.app.middleware.model.Drink.Drink(), es.ull.esit.app.middleware.model.Drink.getDrinksId(), and es.ull.esit.app.middleware.model.Drink.setDrinksId().

7.18.4.2 drinksPrice

```
Integer es.ull.esit.app.middleware.model.Drink.drinksPrice  [private]
```

Price of the drink (JSON property "drinksPrice").

Definition at line 23 of file src/main/java/es/ull/esit/app/middleware/model/Drink.java.

Referenced by es.ull.esit.app.middleware.model.Drink.Drink(), es.ull.esit.app.middleware.model.Drink.getDrinksPrice(), and es.ull.esit.app.middleware.model.Drink.setDrinksPrice().

7.18.4.3 itemDrinks

```
String es.ull.esit.app.middleware.model.Drink.itemDrinks  [private]
```

Name of the drink item (JSON property "itemDrinks").

Definition at line 19 of file src/main/java/es/ull/esit/app/middleware/model/Drink.java.

Referenced by es.ull.esit.app.middleware.model.Drink.Drink(), es.ull.esit.app.middleware.model.Drink.getItemDrinks(), and es.ull.esit.app.middleware.model.Drink.setItemDrinks().

7.18.4.4 receiptId

```
Long es.ull.esit.app.middleware.model.Drink.receiptId  [private]
```

Identifier of the receipt this drink belongs to (JSON property "receiptId").

Currently not provided by the backend.

Definition at line 30 of file src/main/java/es/ull/esit/app/middleware/model/Drink.java.

Referenced by es.ull.esit.app.middleware.model.Drink.Drink(), es.ull.esit.app.middleware.model.Drink.getReceiptId(), and es.ull.esit.app.middleware.model.Drink.setReceiptId().

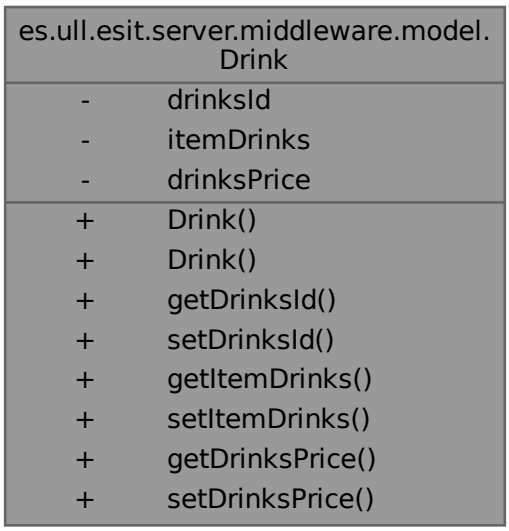
The documentation for this class was generated from the following file:

- src/main/java/es/ull/esit/app/middleware/model/Drink.java

7.19 es.ull.esit.server.middleware.model.Drink Class Reference

JPA entity that represents a drink in the menu.

Collaboration diagram for es.ull.esit.server.middleware.model.Drink:



Public Member Functions

- Drink ()

  Default constructor required by JPA.
- Drink (Long drinksId, String itemDrinks, Integer drinksPrice)

  Full constructor.
- Long getDrinksId ()

  Gets the drink identifier.
- void setDrinksId (Long drinksId)

  Sets the drink identifier.
- String getItemDrinks ()

  Gets the drink name.
- void setItemDrinks (String itemDrinks)

  Sets the drink name.
- Integer getDrinksPrice ()

  Gets the drink price.
- void setDrinksPrice (Integer drinksPrice)

  Sets the drink price.

Private Attributes

- Long drinksId

  Primary key of the drink (column "id").
- String itemDrinks

  Name of the drink (column "name").
- Integer drinksPrice

  Price of the drink in integer units (column "price").

7.19.1 Detailed Description

JPA entity that represents a drink in the menu.

```
It is mapped to the "drinks" table used by the REST API.
Each row contains an auto generated identifier, a name and a price.
```

Definition at line 15 of file server/src/main/java/es/ull/esit/server/middleware/model/Drink.java.

7.19.2 Constructor & Destructor Documentation

7.19.2.1 Drink() [1/2]

```
es.ull.esit.server.middleware.model.Drink.Drink ( )  [inline]
```

Default constructor required by JPA.

Definition at line 34 of file server/src/main/java/es/ull/esit/server/middleware/model/Drink.java.

```
00034                 {
00035   }
```

7.19.2.2 Drink() [2/2]

```
es.ull.esit.server.middleware.model.Drink.Drink (
            Long drinksId,
            String itemDrinks,
            Integer drinksPrice )  [inline]
```

Full constructor.

Parameters

drinksId	[Long] Identifier of the drink.
itemDrinks	[String] Name of the drink.
drinksPrice	[Integer] Price of the drink.

Definition at line 44 of file server/src/main/java/es/ull/esit/server/middleware/model/Drink.java.

```
00044                                                                   {
00045     this.drinksId = drinksId;
00046     this.itemDrinks = itemDrinks;
```

```
00047     this.drinksPrice = drinksPrice;
00048   }
```

7.19.3 Member Function Documentation

7.19.3.1 getDrinksId()

```
Long es.ull.esit.server.middleware.model.Drink.getDrinksId ( )  [inline]
```

Gets the drink identifier.

Returns

[Long] Drink id.

Definition at line 55 of file server/src/main/java/es/ull/esit/server/middleware/model/Drink.java.

```
00055                           {
00056     return drinksId;
00057   }
```

7.19.3.2 getDrinksPrice()

```
Integer es.ull.esit.server.middleware.model.Drink.getDrinksPrice ( )  [inline]
```

Gets the drink price.

Returns

[Integer] Drink price.

Definition at line 93 of file server/src/main/java/es/ull/esit/server/middleware/model/Drink.java.

```
00093                                {
00094     return drinksPrice;
00095   }
```

7.19.3.3 getItemDrinks()

```
String es.ull.esit.server.middleware.model.Drink.getItemDrinks ( )  [inline]
```

Gets the drink name.

Returns

[String] Drink name.

Definition at line 75 of file server/src/main/java/es/ull/esit/server/middleware/model/Drink.java.

```
00075                              {
00076     return itemDrinks;
00077   }
```

7.19.3.4 setDrinksId()

```
void es.ull.esit.server.middleware.model.Drink.setDrinksId (
             Long drinksId )  [inline]
```

Sets the drink identifier.

Generated by the database.

Parameters

drinksId	[Long] Drink id.

Definition at line 66 of file server/src/main/java/es/ull/esit/server/middleware/model/Drink.java.

```
00066                                          {
00067     this.drinksId = drinksId;
00068   }
```

7.19.3.5 setDrinksPrice()

```
void es.ull.esit.server.middleware.model.Drink.setDrinksPrice (
            Integer drinksPrice )  [inline]
```

Sets the drink price.

Parameters

drinksPrice	[Integer] New price of the drink.

Definition at line 102 of file server/src/main/java/es/ull/esit/server/middleware/model/Drink.java.

```
00102                                              {
00103     this.drinksPrice = drinksPrice;
00104   }
```

7.19.3.6 setItemDrinks()

```
void es.ull.esit.server.middleware.model.Drink.setItemDrinks (
            String itemDrinks )  [inline]
```

Sets the drink name.

Parameters

itemDrinks	[String] New name of the drink.

Definition at line 84 of file server/src/main/java/es/ull/esit/server/middleware/model/Drink.java.

```
00084                                           {
00085     this.itemDrinks = itemDrinks;
00086   }
```

7.19.4 Member Data Documentation

7.19.4.1 drinksId

```
Long es.ull.esit.server.middleware.model.Drink.drinksId  [private]
```

Primary key of the drink (column "id").

Definition at line 21 of file server/src/main/java/es/ull/esit/server/middleware/model/Drink.java.

7.19.4.2 drinksPrice

```
Integer es.ull.esit.server.middleware.model.Drink.drinksPrice  [private]
```

Price of the drink in integer units (column "price").

Definition at line 29 of file server/src/main/java/es/ull/esit/server/middleware/model/Drink.java.

7.19.4.3 itemDrinks

```
String es.ull.esit.server.middleware.model.Drink.itemDrinks  [private]
```

Name of the drink (column "name").

Definition at line 25 of file server/src/main/java/es/ull/esit/server/middleware/model/Drink.java.

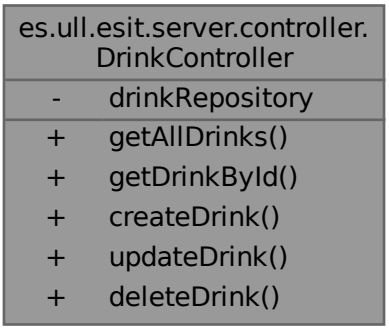
The documentation for this class was generated from the following file:

- server/src/main/java/es/ull/esit/server/middleware/model/Drink.java

7.20 es.ull.esit.server.controller.DrinkController Class Reference

REST controller for managing drinks.

Collaboration diagram for es.ull.esit.server.controller.DrinkController:



Public Member Functions

- ResponseEntity< List< Drink > > getAllDrinks ()

  Returns the complete list of drinks.
- ResponseEntity< Drink > getDrinkById (@PathVariable Long id)

  Returns a single drink by its id.
- ResponseEntity< Drink > createDrink (@RequestBody Drink drink)

  Creates a new drink.
- ResponseEntity< Drink > updateDrink (@PathVariable Long id, @RequestBody Drink drink)

  Updates an existing drink.
- ResponseEntity< Void > deleteDrink (@PathVariable Long id)

  Deletes a drink by its id.

Private Attributes

- DrinkRepository drinkRepository

  Repository used to access the "drinks" table.

7.20.1 Detailed Description

REST controller for managing drinks.

```
Provides CRUD operations for Drink entities using the path
"/api/drinks". These endpoints are used by the Swing client to load
and modify drink information.
```

Definition at line 22 of file DrinkController.java.

7.20.2 Member Function Documentation

7.20.2.1 createDrink()

```
ResponseEntity< Drink > es.ull.esit.server.controller.DrinkController.createDrink (
            @RequestBody Drink drink )  [inline]
```

Creates a new drink.

```
HTTP POST /api/drinks
```

Parameters

drink	[Drink] Drink data sent in the request body.

Returns

[ResponseEntity<Drink>] The created drink with status 201.

Definition at line 65 of file DrinkController.java.

```
00065                                                                   {
00066     Drink saved = drinkRepository.save(drink);
00067     return ResponseEntity.status(HttpStatus.CREATED).body(saved);
00068   }
```

7.20.2.2 deleteDrink()

```
ResponseEntity< Void > es.ull.esit.server.controller.DrinkController.deleteDrink (
            @PathVariable Long id )  [inline]
```

Deletes a drink by its id.

```
HTTP DELETE /api/drinks/{id}
```

Parameters

id	[Long] Identifier of the drink to delete.

Returns

[ResponseEntity<Void>] 204 if deleted or 404 if not found.

Definition at line 99 of file DrinkController.java.

```
00099                                                                    {
00100     if (!drinkRepository.existsById(id)) {
00101       return ResponseEntity.notFound().build();
00102     }
00103     drinkRepository.deleteById(id);
00104     return ResponseEntity.noContent().build();
00105   }
```

7.20.2.3 getAllDrinks()

```
ResponseEntity< List< Drink > > es.ull.esit.server.controller.DrinkController.getAllDrinks (
) [inline]
```

Returns the complete list of drinks.

```
HTTP GET /api/drinks
```

Returns

[ResponseEntity<List<Drink>>] List of drinks with status 200.

Definition at line 36 of file DrinkController.java.

```
00036                                                {
00037     List<Drink> drinks = drinkRepository.findAll();
00038     return ResponseEntity.ok(drinks);
00039   }
```

7.20.2.4 getDrinkById()

```
ResponseEntity< Drink > es.ull.esit.server.controller.DrinkController.getDrinkById (
            @PathVariable Long id ) [inline]
```

Returns a single drink by its id.

```
HTTP GET /api/drinks/{id}
```

Parameters

id	[Long] Identifier of the drink.

Returns

[ResponseEntity<Drink>] The drink if found or 404 otherwise.

Definition at line 50 of file DrinkController.java.

```
00050                                                                    {
00051     Optional<Drink> drink = drinkRepository.findById(id);
00052     return drink.map(ResponseEntity::ok)
00053         .orElseGet(() -> ResponseEntity.notFound().build());
00054   }
```

7.20.2.5 updateDrink()

```
ResponseEntity< Drink > es.ull.esit.server.controller.DrinkController.updateDrink (
            @PathVariable Long id,
            @RequestBody Drink drink )  [inline]
```

Updates an existing drink.

```
HTTP PUT /api/drinks/{id}
```

Parameters

id	[Long] Identifier of the drink to update.
drink	[Drink] New data for the drink.

Returns

[ResponseEntity<Drink>] The updated drink or 404 if not found.

Definition at line 80 of file DrinkController.java.

```
00081                                {
00082     if (!drinkRepository.existsById(id)) {
00083       return ResponseEntity.notFound().build();
00084     }
00085     drink.setDrinksId(id);
00086     Drink updated = drinkRepository.save(drink);
00087     return ResponseEntity.ok(updated);
00088   }
```

7.20.3 Member Data Documentation

7.20.3.1 drinkRepository

```
DrinkRepository es.ull.esit.server.controller.DrinkController.drinkRepository  [private]
```

Repository used to access the "drinks" table.

Definition at line 26 of file DrinkController.java.

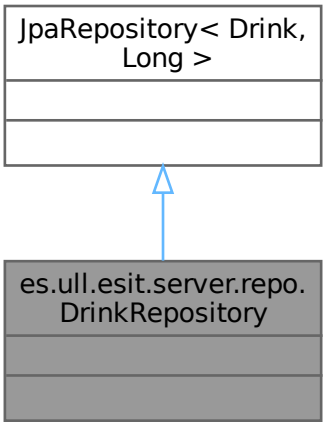
The documentation for this class was generated from the following file:

- DrinkController.java

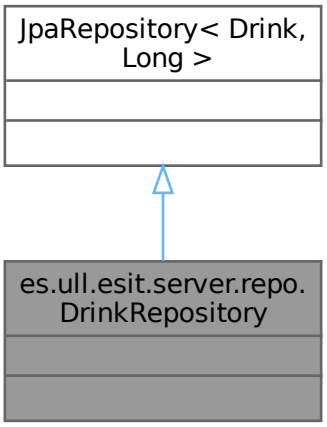
7.21 es.ull.esit.server.repo.DrinkRepository Interface Reference

Repository interface for managing drinks in the database.

Inheritance diagram for es.ull.esit.server.repo.DrinkRepository:



Collaboration diagram for es.ull.esit.server.repo.DrinkRepository:



7.21.1 Detailed Description

Repository interface for managing drinks in the database.

```
Extends JpaRepository to provide basic CRUD operations on the "drinks"
table, such as findAll, findById, save and delete.
```

Definition at line 12 of file DrinkRepository.java.

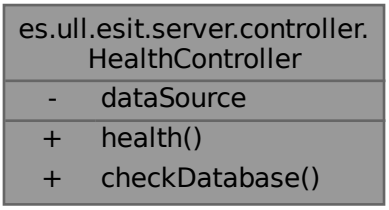
The documentation for this interface was generated from the following file:

- DrinkRepository.java

7.22 es.ull.esit.server.controller.HealthController Class Reference

REST controller for health and database connectivity checks.

Collaboration diagram for es.ull.esit.server.controller.HealthController:



Public Member Functions

- ResponseEntity< Map< String, Object > > health ()

  Basic health endpoint.
- ResponseEntity< Map< String, Object > > checkDatabase ()

  Database connectivity check.

Private Attributes

- DataSource dataSource

  DataSource used to verify connectivity with the database.

7.22.1 Detailed Description

REST controller for health and database connectivity checks.

```
Exposes simple diagnostic endpoints to:
- receive basic service liveness information.
- check connectivity with the underlying database.
Not required by the Swing frontend but are very
useful for debugging, monitoring or for external tools that need to
verify that the service and the database are up and running.
```

Definition at line 27 of file HealthController.java.

7.22.2 Member Function Documentation

7.22.2.1 checkDatabase()

```
ResponseEntity< Map< String, Object > > es.ull.esit.server.controller.HealthController.↩
checkDatabase ( )  [inline]
```

Database connectivity check.

```
Endpoint: GET /api/db-check

Tries to obtain a JDBC connection from the configured DataSource
and verifies that it is valid. If the connection is valid, information
about the database catalog and URL is included in the response.

On success:
- HTTP 200 with JSON fields "status" = "UP" and "database" =
"Connected".

On failure:
- HTTP 503 (SERVICE_UNAVAILABLE) with "status" = "DOWN" and an
additional "error" message describing the issue.
```

Returns

[ResponseEntity<Map<String, Object>>] Database connectivity status: HTTP 200 if connected, HTTP 503 otherwise.

Definition at line 73 of file HealthController.java.

```
00073                                                          {
00074     Map<String, Object> response = new HashMap<>();
00075
00076     try (Connection conn = dataSource.getConnection()) {
00077       boolean isValid = conn.isValid(2);
00078
00079       if (isValid) {
00080         response.put("status", "UP");
00081         response.put("database", "Connected");
00082         response.put("catalog", conn.getCatalog());
00083         response.put("url", conn.getMetaData().getURL());
00084         response.put("timestamp", System.currentTimeMillis());
00085         return ResponseEntity.ok(response);
00086       } else {
00087         response.put("status", "DOWN");
00088         response.put("database", "Connection not valid");
00089         return ResponseEntity.status(HttpStatus.SERVICE_UNAVAILABLE).body(response);
00090       }
00091     } catch (Exception e) {
00092       response.put("status", "DOWN");
00093       response.put("database", "Connection failed");
00094       response.put("error", e.getMessage());
00095       response.put("timestamp", System.currentTimeMillis());
00096       return ResponseEntity.status(HttpStatus.SERVICE_UNAVAILABLE).body(response);
00097     }
00098   }
```

7.22.2.2 health()

```
ResponseEntity< Map< String, Object > > es.ull.esit.server.controller.HealthController.health
( )  [inline]
```

Basic health endpoint.

```
Endpoint: GET /api/health

Returns a small JSON payload indicating that the service is up,
together with a timestamp and a human-readable service name.
```

Returns

[ResponseEntity<Map<String, Object>>] Health status with HTTP 200.

Definition at line 44 of file HealthController.java.

```
00044                                                {
00045     Map<String, Object> response = new HashMap<>();
00046     response.put("status", "UP");
00047     response.put("timestamp", System.currentTimeMillis());
00048     response.put("service", "Restaurant Server");
00049     return ResponseEntity.ok(response);
00050   }
```

7.22.3 Member Data Documentation

7.22.3.1 dataSource

```
DataSource es.ull.esit.server.controller.HealthController.dataSource  [private]
```

DataSource used to verify connectivity with the database.

Definition at line 31 of file HealthController.java.

The documentation for this class was generated from the following file:

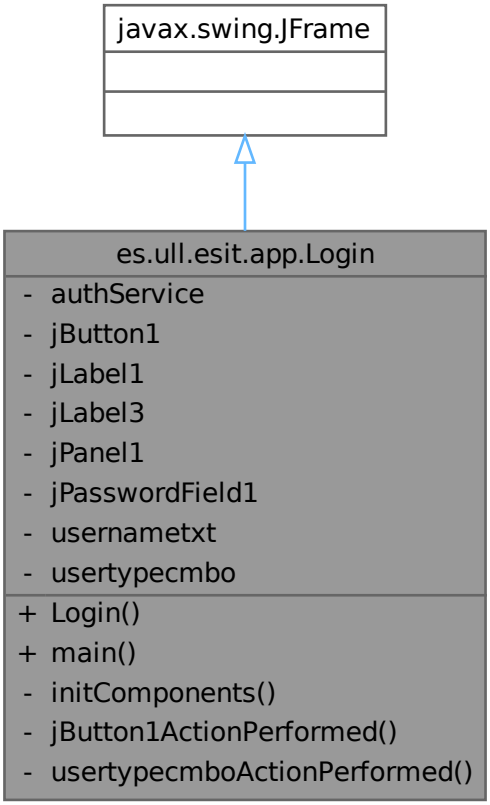
- HealthController.java

7.23 es.ull.esit.app.Login Class Reference

Login window for the Restaurant System.

Inheritance diagram for es.ull.esit.app.Login:

Collaboration diagram for es.ull.esit.app.Login:



Public Member Functions

- Login ()

  Constructor.

Static Public Member Functions

- static void main (String[ ] args)

  Main entry point to start the login window application.

Private Member Functions

- void initComponents ()

  Initializes GUI components.
- void jButton1ActionPerformed (java.awt.event.ActionEvent evt)

  Action handler for the "Log In" button.
- void usertypecmboActionPerformed (java.awt.event.ActionEvent evt)

  Action handler for the user type combo box.

Private Attributes

- final AuthService authService

  Service to handle authentication logic.
- javax.swing.JButton jButton1

  Button to perform login ("Log In").
- javax.swing.JLabel jLabel1

  Label for the image/logo.
- javax.swing.JLabel jLabel3

  Label for the title ("User Login").
- javax.swing.JPanel jPanel1

  Main panel container.
- javax.swing.JPasswordField jPasswordField1

  Input field for the password.
- javax.swing.JTextField usernametxt

  Input field for the username.
- javax.swing.JComboBox< String > usertypecmbo

  Combo box for user type (admin/cashier).

7.23.1 Detailed Description

Login window for the Restaurant System.

```
Shows a Swing form that lets the user:
- enter his/her/their username.
- enter his/her/their password.
- select user type (admin or cashier) from a combo box.
- click the "Log In" button to authenticate.

The credentials are sent to the backend through AuthService and
ApiClient.
According to the role returned by the server (ADMIN or CASHIER), the
application opens the corresponding window: AdminLogin or Cashier.
```

Definition at line 23 of file Login.java.

7.23.2 Constructor & Destructor Documentation

7.23.2.1 Login()

```
es.ull.esit.app.Login.Login ( )  [inline]
```

Constructor.

Creates the login window, initializes the AuthService and GUI components.

The ApiClient is configured to connect to the backend at "http://localhost:8080", which must match the URL of the Spring Boot backend server.

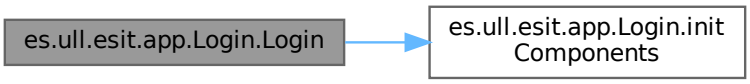
Definition at line 37 of file Login.java.

```
00037                 {
00038     initComponents();
00039     ApiClient client = new ApiClient("http://localhost:8080");
00040     this.authService = new AuthService(client);
```

```
00041    }
```

References es.ull.esit.app.Login.initComponents().

Here is the call graph for this function:



7.23.3 Member Function Documentation

7.23.3.1 initComponents()

```
void es.ull.esit.app.Login.initComponents ( )  [inline], [private]
```

Initializes GUI components.

```
    Generated by the Form Editor: do not modify.
    Sets up logo image, main label, input fields for username and password, combo box for user type, and buttor
```

Definition at line 51 of file Login.java.

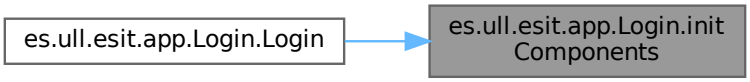
```
00051                                {
00052
00053     jPanel1 = new javax.swing.JPanel();
00054     jLabel1 = new javax.swing.JLabel();
00055     jLabel3 = new javax.swing.JLabel();
00056     usernametxt = new javax.swing.JTextField();
00057     usertypecmbo = new javax.swing.JComboBox<>();
00058     jButton1 = new javax.swing.JButton();
00059     jPasswordField1 = new javax.swing.JPasswordField();
00060
00061     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00062     setTitle("Login");
00063     setResizable(false);
00064
00065     jPanel1.setBackground(new java.awt.Color(248, 244, 230));
00066
00067     jLabel1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/ui/screenshot.png"))); //
      NOI18N
00068
00069     jLabel3.setFont(new java.awt.Font("Yu Gothic UI", 1, 24)); // NOI18N
00070     jLabel3.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00071     jLabel3.setText("User Login ");
00072
00073     usernametxt.setFont(new java.awt.Font("Yu Gothic UI", 0, 18)); // NOI18N
00074     usernametxt.setHorizontalAlignment(javax.swing.JTextField.CENTER);
00075     usernametxt.setBorder(javax.swing.BorderFactory.createTitledBorder(
00076         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Username",
00077         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00078         new java.awt.Font("Yu Gothic UI", 0, 18))); // NOI18N
00079
00080     usertypecmbo.setEditable(true);
00081     usertypecmbo.setFont(new java.awt.Font("Yu Gothic UI", 0, 18)); // NOI18N
00082     usertypecmbo.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] { "admin", "cashier"
      }));
00083     usertypecmbo.addActionListener(new java.awt.event.ActionListener() {
00084       public void actionPerformed(java.awt.event.ActionEvent evt) {
00085         usertypecmboActionPerformed(evt);
00086       }
00087     });
00088
```

```
00089     jButton1.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00090     jButton1.setText("Log In");
00091     jButton1.addActionListener(new java.awt.event.ActionListener() {
00092       public void actionPerformed(java.awt.event.ActionEvent evt) {
00093         jButton1ActionPerformed(evt);
00094       }
00095     });
00096
00097     jPasswordField1.setFont(new java.awt.Font("Thonburi", 0, 18)); // NOI18N
00098     jPasswordField1.setHorizontalAlignment(javax.swing.JTextField.CENTER);
00099     jPasswordField1.setBorder(javax.swing.BorderFactory.createTitledBorder(
00100         javax.swing.BorderFactory.createLineBorder(new java.awt.Color(0, 0, 0)), "Password",
00101         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00102         new java.awt.Font("Lucida Grande", 0, 18))); // NOI18N
00103
00104     javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
00105     jPanel1.setLayout(jPanel1Layout);
00106     jPanel1Layout.setHorizontalGroup(
00107         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00108             .addGroup(jPanel1Layout.createSequentialGroup()
00109                 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00110                     .addGroup(jPanel1Layout.createSequentialGroup()
00111                         .addGap(132, 132, 132)
00112                         .addComponent(jLabel1))
00113                     .addGroup(jPanel1Layout.createSequentialGroup()
00114                         .addGap(61, 61, 61)
00115                         .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 309,
00116                             javax.swing.GroupLayout.PREFERRED_SIZE))
00117                     .addGroup(jPanel1Layout.createSequentialGroup()
00118                         .addGap(74, 74, 74)
00119
      .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00120                             .addComponent(jPasswordField1, javax.swing.GroupLayout.PREFERRED_SIZE,
      311,
00121                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00122                             .addComponent(usernametxt, javax.swing.GroupLayout.PREFERRED_SIZE, 311,
00123                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00124                             .addComponent(usertypecmbo, javax.swing.GroupLayout.PREFERRED_SIZE, 311,
00125                                 javax.swing.GroupLayout.PREFERRED_SIZE)))
00126                     .addGroup(jPanel1Layout.createSequentialGroup()
00127                         .addGap(146, 146, 146)
00128                         .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 141,
00129                             javax.swing.GroupLayout.PREFERRED_SIZE)))
00130                 .addContainerGap(86, Short.MAX_VALUE)));
00131     jPanel1Layout.setVerticalGroup(
00132         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00133             .addGroup(jPanel1Layout.createSequentialGroup()
00134                 .addGap(39, 39, 39)
00135                 .addComponent(jLabel1)
00136                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
00137                 .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
00138                     javax.swing.GroupLayout.PREFERRED_SIZE)
00139                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
00140                 .addComponent(usernametxt, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00141                     javax.swing.GroupLayout.PREFERRED_SIZE)
00142                 .addGap(18, 18, 18)
00143                 .addComponent(jPasswordField1, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00144                     javax.swing.GroupLayout.PREFERRED_SIZE)
00145                 .addGap(18, 18, 18)
00146                 .addComponent(usertypecmbo, javax.swing.GroupLayout.PREFERRED_SIZE, 46,
00147                     javax.swing.GroupLayout.PREFERRED_SIZE)
00148                 .addGap(18, 18, 18)
00149                 .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00150                     javax.swing.GroupLayout.PREFERRED_SIZE)
00151                 .addContainerGap(39, Short.MAX_VALUE)));
00152
00153     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00154     getContentPane().setLayout(layout);
00155     layout.setHorizontalGroup(
00156         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00157             .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00158                 javax.swing.GroupLayout.PREFERRED_SIZE));
00159     layout.setVerticalGroup(
00160         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00161             .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00162                 Short.MAX_VALUE));
00163
00164     pack();
00165     setLocationRelativeTo(null);
00166   }// </editor-fold>//GEN-END:initComponents
```

References es.ull.esit.app.Login.jButton1, es.ull.esit.app.Login.jLabel1, es.ull.esit.app.Login.jLabel3, es.ull.esit.app.Login.jPanel1, es.ull.esit.app.Login.jPasswordField1, es.ull.esit.app.Login.usernametxt, and es.ull.esit.app.Login.usertypecmbo.

Referenced by es.ull.esit.app.Login.Login().

Here is the caller graph for this function:



7.23.3.2 jButton1ActionPerformed()

```
void es.ull.esit.app.Login.jButton1ActionPerformed (
            java.awt.event.ActionEvent evt )  [inline], [private]
```

Action handler for the "Log In" button.

```
Action steps:
1. Read username and password from the fields.
2. Disable the button and show "Loading..." to avoid double clicks.
3. Run authentication in a background thread.
4. On success, open the admin or cashier window depending on the role.
5. On error, show a dialog and re-enable the button.
```

Parameters

evt	[java.awt.event.ActionEvent] Action event triggered by button click.

Definition at line 181 of file Login.java.

```
00181                                                                {//
      GEN-FIRST:event_jButton1ActionPerformed
00182     // Get username and password from input fields.
00183     String usernameInput = usernametxt.getText();
00184     String passwordInput = new String(jPasswordField1.getPassword());
00185
00186     jButton1.setEnabled(false);
00187     jButton1.setText("Loading...");
00188
00189     new Thread(() -> {
00190       try {
00191         // Authenticate user through AuthService.
00192         User loggedInUser = authService.authenticate(usernameInput, passwordInput);
00193
00194         SwingUtilities.invokeLater(() -> {
00195
00196           // Open the corresponding window based on user role.
00197           if ("ADMIN".equalsIgnoreCase(loggedInUser.getRole())) {
00198             // Open the admin window if the role user is ADMIN.
00199             new AdminLogin().setVisible(true);
00200           } else if ("CASHIER".equalsIgnoreCase(loggedInUser.getRole())) {
00201             // Open the cashier window if the role user is CASHIER.
00202             new CashierLogin(loggedInUser.getUsername()).setVisible(true);
00203           } else {
00204             JOptionPane.showMessageDialog(this, "Unknown Role: " + loggedInUser.getRole());
00205             jButton1.setEnabled(true);
00206             jButton1.setText("Log In");
00207             return;
00208           }
00209
00210           this.dispose();
```

```
00211         });
00212
00213       } catch (Exception e) {
00214         SwingUtilities.invokeLater(() -> {
00215           JOptionPane.showMessageDialog(this,
00216               "Login failed: " + e.getMessage(),
00217               "Login Error",
00218               JOptionPane.ERROR_MESSAGE);
00219
00220           jButton1.setEnabled(true);
00221           jButton1.setText("Log In");
00222         });
00223       }
00224     }).start();
00225   }// GEN-LAST:event_jButton1ActionPerformed
```

7.23.3.3 main()

```
static void es.ull.esit.app.Login.main (
             String[] args )  [inline], [static]
```

Main entry point to start the login window application.

Parameters

args	[String[]] Command line arguments (not used).

Definition at line 244 of file Login.java.

```
00244                                         {
00245     try {
00246       for (javax.swing.UIManager.LookAndFeelInfo info :
      javax.swing.UIManager.getInstalledLookAndFeels()) {
00247         if ("Nimbus".equals(info.getName())) {
00248           javax.swing.UIManager.setLookAndFeel(info.getClassName());
00249           break;
00250         }
00251       }
00252     } catch (ClassNotFoundException | InstantiationException | IllegalAccessException
00253         | javax.swing.UnsupportedLookAndFeelException ex) {
00254       java.util.logging.Logger.getLogger(Login.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
00255     }
00256
00257     java.awt.EventQueue.invokeLater(() -> new Login().setVisible(true));
00258   }
```

7.23.3.4 usertypecmboActionPerformed()

```
void es.ull.esit.app.Login.usertypecmboActionPerformed (
             java.awt.event.ActionEvent evt )  [inline], [private]
```

Action handler for the user type combo box.

```
The selection is not used in this implementation, as the role is
determined by the server.
```

Parameters

evt	[java.awt.event.ActionEvent] Action event triggered by combo box selection.

Definition at line 235 of file Login.java.

```
00235                                                                    {//
      GEN-FIRST:event_usertypecmboActionPerformed
00236     // No action needed here.
00237   }// GEN-LAST:event_usertypecmboActionPerformed
```

7.23.4 Member Data Documentation

7.23.4.1 authService

```
final AuthService es.ull.esit.app.Login.authService  [private]
```

Service to handle authentication logic.

Definition at line 26 of file Login.java.

7.23.4.2 jButton1

```
javax.swing.JButton es.ull.esit.app.Login.jButton1  [private]
```

Button to perform login ("Log In").

Definition at line 262 of file Login.java.

Referenced by es.ull.esit.app.Login.initComponents().

7.23.4.3 jLabel1

```
javax.swing.JLabel es.ull.esit.app.Login.jLabel1  [private]
```

Label for the image/logo.

Definition at line 264 of file Login.java.

Referenced by es.ull.esit.app.Login.initComponents().

7.23.4.4 jLabel3

```
javax.swing.JLabel es.ull.esit.app.Login.jLabel3  [private]
```

Label for the title ("User Login").

Definition at line 266 of file Login.java.

Referenced by es.ull.esit.app.Login.initComponents().

7.23.4.5 jPanel1

```
javax.swing.JPanel es.ull.esit.app.Login.jPanel1  [private]
```

Main panel container.

Definition at line 268 of file Login.java.

Referenced by es.ull.esit.app.Login.initComponents().

7.23.4.6 jPasswordField1

```
javax.swing.JPasswordField es.ull.esit.app.Login.jPasswordField1  [private]
```

Input field for the password.

Definition at line 270 of file Login.java.

Referenced by es.ull.esit.app.Login.initComponents().

7.23.4.7 usernametxt

```
javax.swing.JTextField es.ull.esit.app.Login.usernametxt  [private]
```

Input field for the username.

Definition at line 272 of file Login.java.

Referenced by es.ull.esit.app.Login.initComponents().

7.23.4.8 usertypecmbo

```
javax.swing.JComboBox<String> es.ull.esit.app.Login.usertypecmbo  [private]
```

Combo box for user type (admin/cashier).

Currently not trusted for security decisions.

Definition at line 277 of file Login.java.

Referenced by es.ull.esit.app.Login.initComponents().

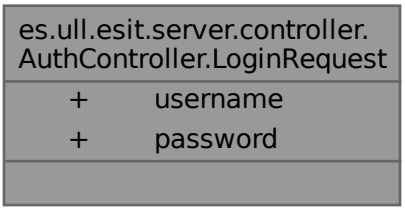
The documentation for this class was generated from the following file:

- Login.java

7.24 es.ull.esit.server.controller.AuthController.LoginRequest Class Reference

Simple DTO (Data Transfer Object) for login requests payload.

Collaboration diagram for es.ull.esit.server.controller.AuthController.LoginRequest:



Public Attributes

- String username
- String password

7.24.1 Detailed Description

Simple DTO (Data Transfer Object) for login requests payload.

```
It is populated automatically from the JSON request body of the HTTP
request.
```

Definition at line 42 of file AuthController.java.

7.24.2 Member Data Documentation

7.24.2.1 password

```
String es.ull.esit.server.controller.AuthController.LoginRequest.password
```

Definition at line 44 of file AuthController.java.

7.24.2.2 username

```
String es.ull.esit.server.controller.AuthController.LoginRequest.username
```

Definition at line 43 of file AuthController.java.

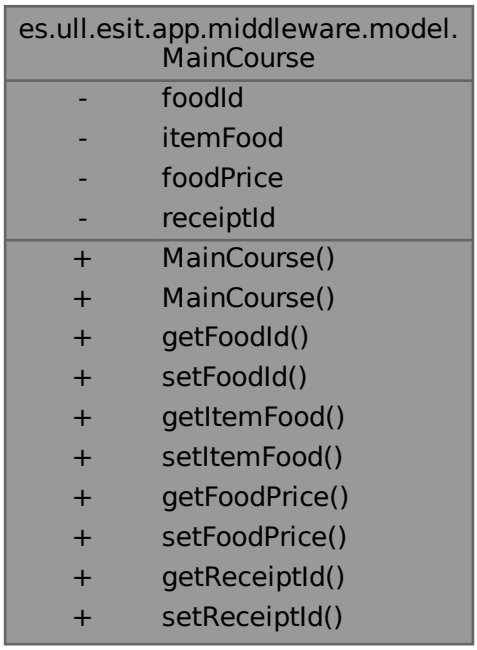
The documentation for this class was generated from the following file:

- AuthController.java

7.25 es.ull.esit.app.middleware.model.MainCourse Class Reference

Client-side model representing a main course returned by the backend.

Collaboration diagram for es.ull.esit.app.middleware.model.MainCourse:



Public Member Functions

- MainCourse ()
  Default constructor required for JSON deserialization.
- MainCourse (Long foodId, String itemFood, Integer foodPrice, Long receiptId)
  Constructs a main course with all fields.
- Long getFoodId ()
  Gets the dish identifier.
- void setFoodId (Long foodId)
  Sets the dish identifier.
- String getItemFood ()
  Gets the name of the dish.
- void setItemFood (String itemFood)
  Sets the name of the dish.
- Integer getFoodPrice ()
  Gets the price of the dish.
- void setFoodPrice (Integer foodPrice)
  Sets the price of the dish.
- Long getReceiptId ()
  Gets the identifier of the related receipt.
- void setReceiptId (Long receiptId)
  Sets the identifier of the related receipt.

Private Attributes

- Long foodId

  Unique identifier of the dish (JSON property "foodId").
- String itemFood

  Name of the dish (JSON property "itemFood").
- Integer foodPrice

  Price of the dish (JSON property "foodPrice").
- Long receiptId

  Identifier of the receipt this dish belongs to (JSON property "receiptId").

7.25.1 Detailed Description

Client-side model representing a main course returned by the backend.

```
Describes a single dish in the main course category as exposed by the
"/api/maincourses" endpoint.
```

Definition at line 11 of file src/main/java/es/ull/esit/app/middleware/model/MainCourse.java.

7.25.2 Constructor & Destructor Documentation

7.25.2.1 MainCourse() [1/2]

```
es.ull.esit.app.middleware.model.MainCourse.MainCourse ( )  [inline]
```

Default constructor required for JSON deserialization.

Definition at line 35 of file src/main/java/es/ull/esit/app/middleware/model/MainCourse.java.

```
00035                       {
00036   }
```

7.25.2.2 MainCourse() [2/2]

```
es.ull.esit.app.middleware.model.MainCourse.MainCourse (
            Long foodId,
            String itemFood,
            Integer foodPrice,
            Long receiptId )  [inline]
```

Constructs a main course with all fields.

Parameters

foodId	[Long] Unique identifier of the dish.
itemFood	[String] Name of the dish.
foodPrice	[Integer] Price of the dish.
receiptId	[Long] Identifier of the related receipt (optional).

Definition at line 46 of file src/main/java/es/ull/esit/app/middleware/model/MainCourse.java.

```
00046                                                                                {
00047     this.foodId = foodId;
00048     this.itemFood = itemFood;
00049     this.foodPrice = foodPrice;
00050     this.receiptId = receiptId;
00051   }
```

References es.ull.esit.app.middleware.model.MainCourse.foodId, es.ull.esit.app.middleware.model.MainCourse.foodPrice, es.ull.esit.app.middleware.model.MainCourse.itemFood, and es.ull.esit.app.middleware.model.MainCourse.receiptId.

7.25.3 Member Function Documentation

7.25.3.1 getFoodId()

```
Long es.ull.esit.app.middleware.model.MainCourse.getFoodId ( )  [inline]
```

Gets the dish identifier.

Returns

[Long] Unique identifier of the dish.

Definition at line 58 of file src/main/java/es/ull/esit/app/middleware/model/MainCourse.java.

```
00058                           {
00059     return foodId;
00060   }
```

References es.ull.esit.app.middleware.model.MainCourse.foodId.

7.25.3.2 getFoodPrice()

```
Integer es.ull.esit.app.middleware.model.MainCourse.getFoodPrice ( )  [inline]
```

Gets the price of the dish.

Returns

[Integer] Price of the dish.

Definition at line 94 of file src/main/java/es/ull/esit/app/middleware/model/MainCourse.java.

```
00094                               {
00095     return foodPrice;
00096   }
```

References es.ull.esit.app.middleware.model.MainCourse.foodPrice.

7.25.3.3 getItemFood()

```
String es.ull.esit.app.middleware.model.MainCourse.getItemFood ( )  [inline]
```

Gets the name of the dish.

Returns

[String] Name of the dish.

Definition at line 76 of file src/main/java/es/ull/esit/app/middleware/model/MainCourse.java.

```
00076                               {
00077     return itemFood;
00078   }
```

References es.ull.esit.app.middleware.model.MainCourse.itemFood.

7.25.3.4 getReceiptId()

```
Long es.ull.esit.app.middleware.model.MainCourse.getReceiptId ( )  [inline]
```

Gets the identifier of the related receipt.

Returns

[Long] Identifier of the receipt or null if not set.

Definition at line 112 of file src/main/java/es/ull/esit/app/middleware/model/MainCourse.java.

```
00112                                {
00113     return receiptId;
00114   }
```

References es.ull.esit.app.middleware.model.MainCourse.receiptId.

7.25.3.5 setFoodId()

```
void es.ull.esit.app.middleware.model.MainCourse.setFoodId (
            Long foodId )  [inline]
```

Sets the dish identifier.

Parameters

foodId	[Long] Unique identifier of the dish.

Definition at line 67 of file src/main/java/es/ull/esit/app/middleware/model/MainCourse.java.

```
00067                                    {
00068     this.foodId = foodId;
00069   }
```

References es.ull.esit.app.middleware.model.MainCourse.foodId.

7.25.3.6 setFoodPrice()

```
void es.ull.esit.app.middleware.model.MainCourse.setFoodPrice (
            Integer foodPrice )  [inline]
```

Sets the price of the dish.

Parameters

foodPrice	[Integer] Price of the dish.

Definition at line 103 of file src/main/java/es/ull/esit/app/middleware/model/MainCourse.java.

```
00103                                      {
00104     this.foodPrice = foodPrice;
00105   }
```

References es.ull.esit.app.middleware.model.MainCourse.foodPrice.

7.25.3.7 setItemFood()

```
void es.ull.esit.app.middleware.model.MainCourse.setItemFood (
            String itemFood )  [inline]
```

Sets the name of the dish.

Parameters

itemFood	[String] Name of the dish.

Definition at line 85 of file src/main/java/es/ull/esit/app/middleware/model/MainCourse.java.

```
00085                                           {
00086     this.itemFood = itemFood;
00087   }
```

References es.ull.esit.app.middleware.model.MainCourse.itemFood.

7.25.3.8 setReceiptId()

```
void es.ull.esit.app.middleware.model.MainCourse.setReceiptId (
            Long receiptId )  [inline]
```

Sets the identifier of the related receipt.

Parameters

receipt↩ Id	[Long] Identifier of the receipt.

Definition at line 121 of file src/main/java/es/ull/esit/app/middleware/model/MainCourse.java.

```
00121                                           {
00122     this.receiptId = receiptId;
00123   }
```

References es.ull.esit.app.middleware.model.MainCourse.receiptId.

7.25.4 Member Data Documentation

7.25.4.1 foodId

```
Long es.ull.esit.app.middleware.model.MainCourse.foodId  [private]
```

Unique identifier of the dish (JSON property "foodId").

Definition at line 15 of file src/main/java/es/ull/esit/app/middleware/model/MainCourse.java.

Referenced by es.ull.esit.app.middleware.model.MainCourse.getFoodId(), es.ull.esit.app.middleware.model.MainCourse.MainCourse(
and es.ull.esit.app.middleware.model.MainCourse.setFoodId().

7.25.4.2 foodPrice

```
Integer es.ull.esit.app.middleware.model.MainCourse.foodPrice  [private]
```

Price of the dish (JSON property "foodPrice").

Definition at line 23 of file src/main/java/es/ull/esit/app/middleware/model/MainCourse.java.

Referenced by es.ull.esit.app.middleware.model.MainCourse.getFoodPrice(), es.ull.esit.app.middleware.model.MainCourse.MainCou
and es.ull.esit.app.middleware.model.MainCourse.setFoodPrice().

7.25.4.3 itemFood

```
String es.ull.esit.app.middleware.model.MainCourse.itemFood  [private]
```

Name of the dish (JSON property "itemFood").

Definition at line 19 of file src/main/java/es/ull/esit/app/middleware/model/MainCourse.java.

Referenced by es.ull.esit.app.middleware.model.MainCourse.getItemFood(), es.ull.esit.app.middleware.model.MainCourse.MainCours
and es.ull.esit.app.middleware.model.MainCourse.setItemFood().

7.25.4.4 receiptId

```
Long es.ull.esit.app.middleware.model.MainCourse.receiptId  [private]
```

Identifier of the receipt this dish belongs to (JSON property "receiptId").

Currently not provided by the backend.

Definition at line 30 of file src/main/java/es/ull/esit/app/middleware/model/MainCourse.java.

Referenced by es.ull.esit.app.middleware.model.MainCourse.getReceiptId(), es.ull.esit.app.middleware.model.MainCourse.MainCours
and es.ull.esit.app.middleware.model.MainCourse.setReceiptId().

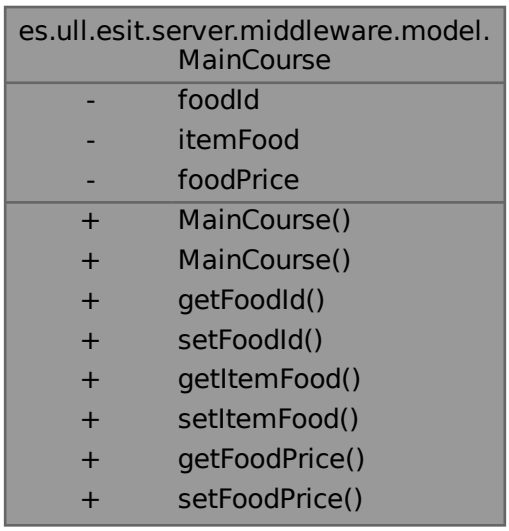
The documentation for this class was generated from the following file:

- src/main/java/es/ull/esit/app/middleware/model/MainCourse.java

7.26 es.ull.esit.server.middleware.model.MainCourse Class Reference

JPA entity that represents a main course in the menu.

Collaboration diagram for es.ull.esit.server.middleware.model.MainCourse:



Public Member Functions

- MainCourse ()

  Default constructor required by JPA.
- MainCourse (Long foodId, String itemFood, Integer foodPrice)

  Full constructor.
- Long getFoodId ()

  Gets the main course ID.
- void setFoodId (Long foodId)

  Sets the main course ID.
- String getItemFood ()

  Gets the main course name.
- void setItemFood (String itemFood)

  Sets the main course name.
- Integer getFoodPrice ()

  Gets the main course price.
- void setFoodPrice (Integer foodPrice)

  Sets the main course price.

Private Attributes

- Long foodId

  Primary key of the main course (column "id").
- String itemFood

  Name of the main course (column "name").
- Integer foodPrice

  Price of the main course (column "price").

7.26.1 Detailed Description

JPA entity that represents a main course in the menu.

```
It is mapped to the "maincourse" table used by the REST API.
Each record has an auto-increment ID, a name and a price.
```

Definition at line 15 of file server/src/main/java/es/ull/esit/server/middleware/model/MainCourse.java.

7.26.2 Constructor & Destructor Documentation

7.26.2.1 MainCourse() [1/2]

```
es.ull.esit.server.middleware.model.MainCourse.MainCourse ( )  [inline]
```

Default constructor required by JPA.

Definition at line 34 of file server/src/main/java/es/ull/esit/server/middleware/model/MainCourse.java.

```
00034                       {
00035   }
```

7.26.2.2 MainCourse() [2/2]

```
es.ull.esit.server.middleware.model.MainCourse.MainCourse (
            Long foodId,
            String itemFood,
            Integer foodPrice )  [inline]
```

Full constructor.

Parameters

foodId	[Long] Identifier of the main course.
itemFood	[String] Name of the main course.
foodPrice	[Integer] Price of the main course.

Definition at line 44 of file server/src/main/java/es/ull/esit/server/middleware/model/MainCourse.java.

```
00044                                                                   {
00045     this.foodId = foodId;
00046     this.itemFood = itemFood;
```

```
00047     this.foodPrice = foodPrice;
00048   }
```

7.26.3 Member Function Documentation

7.26.3.1 getFoodId()

```
Long es.ull.esit.server.middleware.model.MainCourse.getFoodId ( )  [inline]
```

Gets the main course ID.

```
   * @return [Long] Main course id.
```

Definition at line 55 of file server/src/main/java/es/ull/esit/server/middleware/model/MainCourse.java.

```
00055                           {
00056     return foodId;
00057   }
```

7.26.3.2 getFoodPrice()

```
Integer es.ull.esit.server.middleware.model.MainCourse.getFoodPrice ( )  [inline]
```

Gets the main course price.

Returns

[Integer] Main course price.

Definition at line 93 of file server/src/main/java/es/ull/esit/server/middleware/model/MainCourse.java.

```
00093                                {
00094     return foodPrice;
00095   }
```

7.26.3.3 getItemFood()

```
String es.ull.esit.server.middleware.model.MainCourse.getItemFood ( )  [inline]
```

Gets the main course name.

Returns

[String] Main course name.

Definition at line 75 of file server/src/main/java/es/ull/esit/server/middleware/model/MainCourse.java.

```
00075                             {
00076     return itemFood;
00077   }
```

7.26.3.4 setFoodId()

```
void es.ull.esit.server.middleware.model.MainCourse.setFoodId (
             Long foodId )  [inline]
```

Sets the main course ID.

```
   Generated by the database.
```

Parameters

foodId	[Long] Main course id.

Definition at line 66 of file server/src/main/java/es/ull/esit/server/middleware/model/MainCourse.java.

```
00066                                       {
00067     this.foodId = foodId;
00068   }
```

7.26.3.5 setFoodPrice()

```
void es.ull.esit.server.middleware.model.MainCourse.setFoodPrice (
            Integer foodPrice )  [inline]
```

Sets the main course price.

Parameters

foodPrice	[Integer] Main course price.

Definition at line 102 of file server/src/main/java/es/ull/esit/server/middleware/model/MainCourse.java.

```
00102                                            {
00103     this.foodPrice = foodPrice;
00104   }
```

7.26.3.6 setItemFood()

```
void es.ull.esit.server.middleware.model.MainCourse.setItemFood (
            String itemFood )  [inline]
```

Sets the main course name.

Parameters

itemFood	[String] Main course name.

Definition at line 84 of file server/src/main/java/es/ull/esit/server/middleware/model/MainCourse.java.

```
00084                                          {
00085     this.itemFood = itemFood;
00086   }
```

7.26.4 Member Data Documentation

7.26.4.1 foodId

```
Long es.ull.esit.server.middleware.model.MainCourse.foodId  [private]
```

Primary key of the main course (column "id").

Definition at line 21 of file server/src/main/java/es/ull/esit/server/middleware/model/MainCourse.java.

7.26.4.2 foodPrice

```
Integer es.ull.esit.server.middleware.model.MainCourse.foodPrice  [private]
```

Price of the main course (column "price").

Definition at line 29 of file server/src/main/java/es/ull/esit/server/middleware/model/MainCourse.java.

7.26.4.3 itemFood

```
String es.ull.esit.server.middleware.model.MainCourse.itemFood  [private]
```

Name of the main course (column "name").

Definition at line 25 of file server/src/main/java/es/ull/esit/server/middleware/model/MainCourse.java.

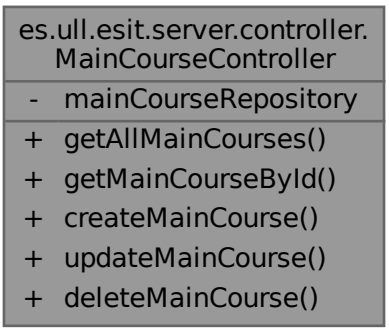
The documentation for this class was generated from the following file:

- server/src/main/java/es/ull/esit/server/middleware/model/MainCourse.java

7.27 es.ull.esit.server.controller.MainCourseController Class Reference

REST controller for managing main courses.

Collaboration diagram for es.ull.esit.server.controller.MainCourseController:



Public Member Functions

- ResponseEntity< List< MainCourse > > getAllMainCourses ()

  Returns all available main course items.
- ResponseEntity< MainCourse > getMainCourseById (@PathVariable Long id)

  Returns a specific main course by its ID.
- ResponseEntity< MainCourse > createMainCourse (@RequestBody MainCourse mainCourse)

  Creates a new main course entry.
- ResponseEntity< MainCourse > updateMainCourse (@PathVariable Long id, @RequestBody MainCourse mainCourse)

  Updates an existing main course entry.
- ResponseEntity< Void > deleteMainCourse (@PathVariable Long id)

  Deletes a main course by its ID.

Private Attributes

- MainCourseRepository mainCourseRepository

  Repository used to access the "main_courses" table.

7.27.1 Detailed Description

REST controller for managing main courses.

```
Provides CRUD operations for MainCourse entities using the path
/api/maincourses.
These endpoints are used by the Swing client to load and modify main
course information.
```

Definition at line 22 of file MainCourseController.java.

7.27.2 Member Function Documentation

7.27.2.1 createMainCourse()

```
ResponseEntity< MainCourse > es.ull.esit.server.controller.MainCourseController.createMain↩
Course (
            @RequestBody MainCourse mainCourse )  [inline]
```

Creates a new main course entry.

```
Endpoint: POST /api/maincourses
```

Parameters

mainCourse	[MainCourse] Main course data from the request body.

Returns

[ResponseEntity<MainCourse>] The created main course with status 201

Definition at line 68 of file MainCourseController.java.

```
00068                                                                                {
00069     MainCourse saved = mainCourseRepository.save(mainCourse);
00070     return ResponseEntity.status(HttpStatus.CREATED).body(saved);
00071   }
```

7.27.2.2 deleteMainCourse()

```
ResponseEntity< Void > es.ull.esit.server.controller.MainCourseController.deleteMainCourse (
            @PathVariable Long id )  [inline]
```

Deletes a main course by its ID.

```
Endpoint: DELETE /api/maincourses/{id}
```

Parameters

id	[Long] Identifier of the main course to delete.

Returns

[ResponseEntity<Void>] Status 204 if deleted or 404 if not found.

Definition at line 105 of file MainCourseController.java.

```
00105                                                                    {
00106     if (!mainCourseRepository.existsById(id)) {
00107       return ResponseEntity.notFound().build();
00108     }
00109
00110     mainCourseRepository.deleteById(id);
00111     return ResponseEntity.noContent().build();
00112   }
```

7.27.2.3 getAllMainCourses()

```
ResponseEntity< List< MainCourse > > es.ull.esit.server.controller.MainCourseController.get↩
AllMainCourses ( )  [inline]
```

Returns all available main course items.

```
Endpoint: GET /api/maincourses
```

Returns

[ResponseEntity<List<MainCourse>>] List of main courses with status 200.

Definition at line 37 of file MainCourseController.java.

```
00037                                                          {
00038     List<MainCourse> courses = mainCourseRepository.findAll();
00039     return ResponseEntity.ok(courses);
00040   }
```

7.27.2.4 getMainCourseById()

```
ResponseEntity< MainCourse > es.ull.esit.server.controller.MainCourseController.getMain↩
CourseById (
            @PathVariable Long id )  [inline]
```

Returns a specific main course by its ID.

```
Endpoint: GET /api/maincourses/{id}
```

Parameters

id	[Long] Identifier of the main course.

Returns

[ResponseEntity<MainCourse>] The main course if found or 404 otherwise.

Definition at line 52 of file MainCourseController.java.

```
00052                                                                    {
00053     Optional<MainCourse> course = mainCourseRepository.findById(id);
00054     return course
00055         .map(ResponseEntity::ok)
00056         .orElseGet(() -> ResponseEntity.notFound().build());
00057   }
```

7.27.2.5 updateMainCourse()

```
ResponseEntity< MainCourse > es.ull.esit.server.controller.MainCourseController.updateMain↩
Course (
             @PathVariable Long id,
             @RequestBody MainCourse mainCourse )  [inline]
```

Updates an existing main course entry.

```
   Endpoint: PUT /api/maincourses/{id}
```

Parameters

id	[Long] Identifier of the main course to update.
mainCourse	[MainCourse] Updated main course data from the request body.

Returns

[ResponseEntity<MainCourse>] The updated main course if found or 404 otherwise.

Definition at line 85 of file MainCourseController.java.

```
00085
     {
00086
00087     if (!mainCourseRepository.existsById(id)) {
00088       return ResponseEntity.notFound().build();
00089     }
00090
00091     mainCourse.setFoodId(id);
00092     MainCourse updated = mainCourseRepository.save(mainCourse);
00093     return ResponseEntity.ok(updated);
00094   }
```

7.27.3 Member Data Documentation

7.27.3.1 mainCourseRepository

```
MainCourseRepository es.ull.esit.server.controller.MainCourseController.mainCourseRepository
[private]
```

Repository used to access the "main_courses" table.

Definition at line 26 of file MainCourseController.java.

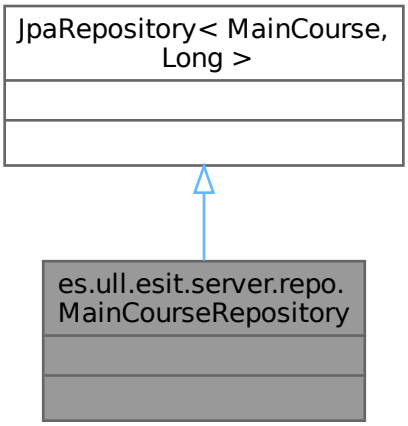
The documentation for this class was generated from the following file:

- MainCourseController.java

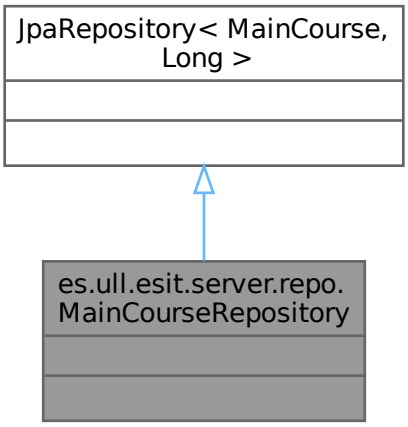
7.28 es.ull.esit.server.repo.MainCourseRepository Interface Reference

Repository interface for managing main courses in the database.

Inheritance diagram for es.ull.esit.server.repo.MainCourseRepository:



Collaboration diagram for es.ull.esit.server.repo.MainCourseRepository:



7.28.1 Detailed Description

Repository interface for managing main courses in the database.

```
Extends JpaRepository to provide basic CRUD operations on the
"main_courses"
table, such as findAll, findById, save and delete.
```

Definition at line 13 of file MainCourseRepository.java.

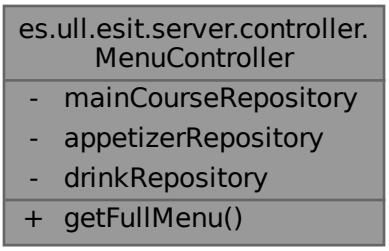
The documentation for this interface was generated from the following file:

- MainCourseRepository.java

7.29 es.ull.esit.server.controller.MenuController Class Reference

REST controller that exposes a consolidated restaurant menu.

Collaboration diagram for es.ull.esit.server.controller.MenuController:



Public Member Functions

- ResponseEntity< Map< String, Object > > getFullMenu ()

  Returns the full menu (main courses, appetizers and drinks).

Private Attributes

- MainCourseRepository mainCourseRepository

  Repository for main course entities.
- AppetizerRepository appetizerRepository

  Repository for appetizer entities.
- DrinkRepository drinkRepository

  Repository for drink entities.

7.29.1 Detailed Description

REST controller that exposes a consolidated restaurant menu.

```
Provides CRUD operation endpoints to retrieve the full menu in a
single
HTTP request. It aggregates main courses, appetizers and drinks into a
single JSON response.
```

Definition at line 29 of file MenuController.java.

7.29.2 Member Function Documentation

7.29.2.1 getFullMenu()

```
ResponseEntity< Map< String, Object > > es.ull.esit.server.controller.MenuController.get↩
FullMenu ( )  [inline]
```

Returns the full menu (main courses, appetizers and drinks).

```
Endpoint: GET /api/menu

The response body is a JSON object with the following keys:
- "mainCourses": list of MainCourse
- "appetizers": list of Appetizer
- "drinks": list of Drink
- "totalItems": integer with the total count of menu items
```

Returns

ResponseEntity containing the aggregated menu and HTTP 200.

Definition at line 57 of file MenuController.java.

```
00057                                                    {
00058     Map<String, Object> menu = new HashMap<>();
00059
00060     List<MainCourse> mainCourses = mainCourseRepository.findAll();
00061     List<Appetizer> appetizers = appetizerRepository.findAll();
00062     List<Drink> drinks = drinkRepository.findAll();
00063
00064     menu.put("mainCourses", mainCourses);
00065     menu.put("appetizers", appetizers);
00066     menu.put("drinks", drinks);
00067     menu.put("totalItems", mainCourses.size() + appetizers.size() + drinks.size());
00068
00069     return ResponseEntity.ok(menu);
00070   }
```

7.29.3 Member Data Documentation

7.29.3.1 appetizerRepository

```
AppetizerRepository es.ull.esit.server.controller.MenuController.appetizerRepository  [private]
```

Repository for appetizer entities.

Definition at line 37 of file MenuController.java.

7.29.3.2 drinkRepository

```
DrinkRepository es.ull.esit.server.controller.MenuController.drinkRepository  [private]
```

Repository for drink entities.

Definition at line 41 of file MenuController.java.

7.29.3.3 mainCourseRepository

```
MainCourseRepository es.ull.esit.server.controller.MenuController.mainCourseRepository  [private]
```

Repository for main course entities.

Definition at line 33 of file MenuController.java.

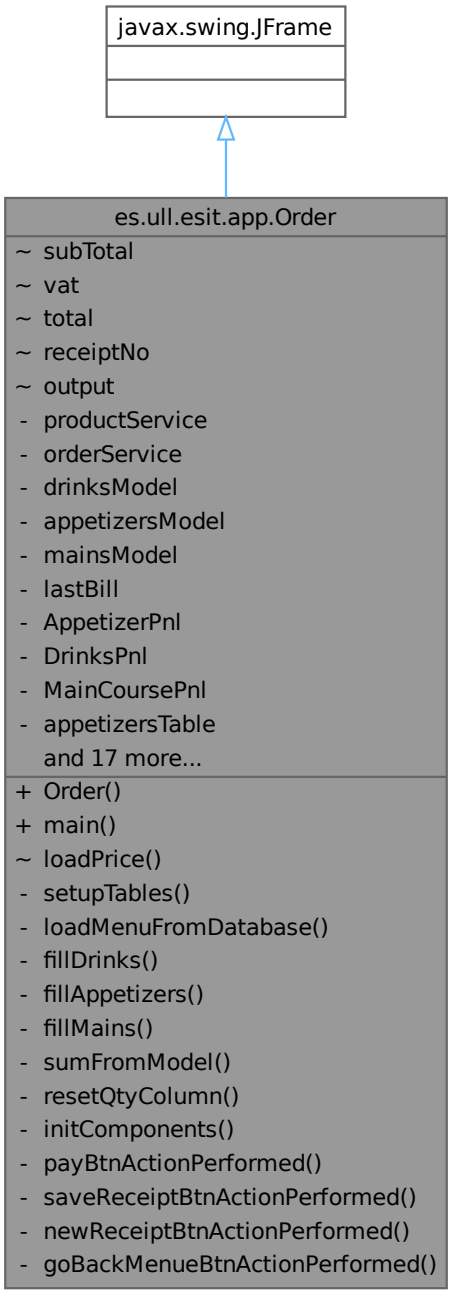
The documentation for this class was generated from the following file:

- MenuController.java

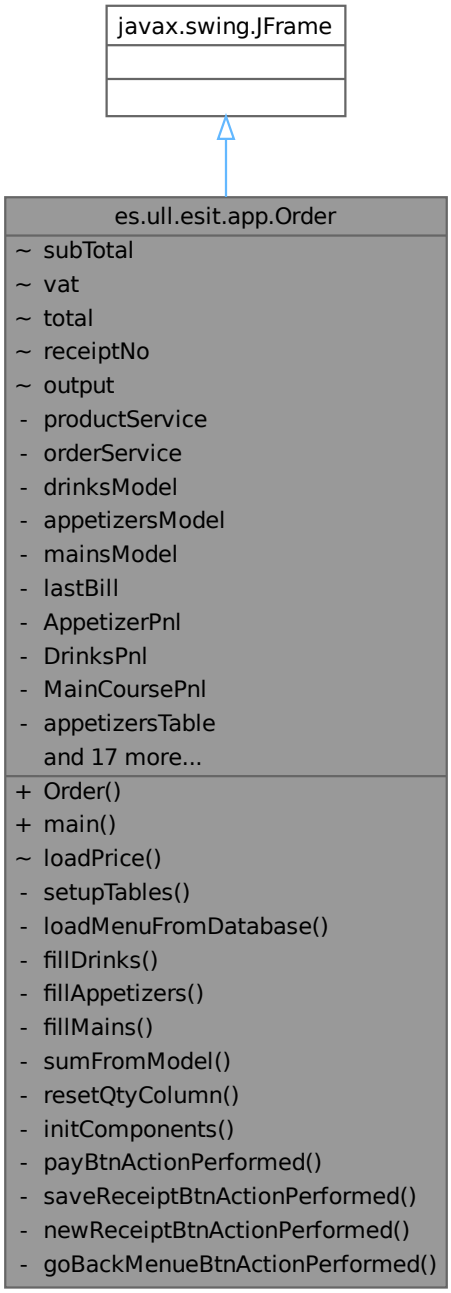
7.30 es.ull.esit.app.Order Class Reference

Main menu window for taking customer orders.

Inheritance diagram for es.ull.esit.app.Order:



Collaboration diagram for es.ull.esit.app.Order:



Public Member Functions

- Order ()

  Constructor.

Static Public Member Functions

- static void main (String args[ ])

Standalone entry point for testing the Order window.

Private Member Functions

- void setupTables ()

  Configures the three tables (Drinks, Appetizers, Main Course).
- void loadMenuFromDatabase ()

  Loads menu data (drinks, appetizers, main courses) from the backend in a background thread.
- void fillDrinks (java.util.List< Drink > drinks)

  Fills the drinks table model with data from the backend.
- void fillAppetizers (java.util.List< Appetizer > appetizers)

  Fills the appetizers table model with data from the backend.
- void fillMains (java.util.List< MainCourse > mains)

  Fills the main courses table model with data from the backend.
- double sumFromModel (DefaultTableModel model)

  Sums the total price for a given table model.
- void resetQtyColumn (DefaultTableModel model)

  Sets the Qty column of the given model to 0 in all rows.
- void initComponents ()

  Initializes GUI components.
- void payBtnActionPerformed (java.awt.event.ActionEvent evt)

  Calculates subtotal, VAT and total and shows a confirmation dialog.
- void saveReceiptBtnActionPerformed (java.awt.event.ActionEvent evt)

  Saves the current bill information to a text file through OrderService.
- void newReceiptBtnActionPerformed (java.awt.event.ActionEvent evt)

  Starts a new receipt and resets the form to its initial state.
- void goBackMenueBtnActionPerformed (java.awt.event.ActionEvent evt)

  Returns to the Login window.

Private Attributes

- final ProductService productService

  Service used to load products from the backend.
- final OrderService orderService = new OrderService()

  Service used to calculate bill totals and generate receipt files.
- DefaultTableModel drinksModel

  Table models for the three product categories.
- DefaultTableModel appetizersModel
- DefaultTableModel mainsModel
- BillResult lastBill

  Last calculated bill (after pressing Pay).
- javax.swing.JPanel AppetizerPnl

  Panel that groups all appetizer items and controls.
- javax.swing.JPanel DrinksPnl

  Panel that groups all drink items and controls.
- javax.swing.JPanel MainCoursePnl

  Panel that groups all main course items and controls.
- javax.swing.JTable appetizersTable

  Table for appetizers items.
- javax.swing.JScrollPane appetizersScroll

Scroll pane for appetizers table.

- javax.swing.JTable drinksTable

  Table for drinks items.
- javax.swing.JScrollPane drinksScroll

  Scroll pane for drinks table.
- javax.swing.JButton goBackMenueBtn

  Button to go back to the Login window.
- javax.swing.JLabel jLabel1

  Title label "BLACK PLATE MENU".
- javax.swing.JLabel jLabel2

  Logo / screenshot label on the top left.
- javax.swing.JPanel jPanel1

  Panel that shows subtotal, VAT, total and receipt number.
- javax.swing.JPanel jPanel2

  Main container panel of the Order window.
- javax.swing.JTable mainsTable

  Table for main course items.
- javax.swing.JScrollPane mainsScroll

  Scroll pane for main course table.
- javax.swing.JButton newReceiptBtn

  Button to start a new receipt.
- javax.swing.JButton payBtn

  Button to calculate and confirm payment.
- javax.swing.JLabel receiptNoLbl

  Label that shows the current receipt number.
- javax.swing.JButton saveReceiptBtn

  Button to save the current receipt to a file.
- javax.swing.JLabel subTotalLbl

  Label that displays the subtotal amount.
- javax.swing.JLabel totalLbl

  Label that displays the total amount.
- javax.swing.JLabel vatLbl

  Label that displays the VAT amount.

7.30.1 Detailed Description

Main menu window for taking customer orders.

```
Swing window that allows the cashier to:
- select quantities of drinks, appetizers and main courses.
- see the items and prices loaded dynamically from the database.
- calculate subtotal, VAT and total.
- save a simple text receipt to disk.

All menu items and prices are now loaded from the backend
using ProductService (instead of being hardcoded).
```

Definition at line 28 of file Order.java.

7.30.2 Constructor & Destructor Documentation

7.30.2.1 Order()

```
es.ull.esit.app.Order.Order ( )  [inline]
```

Constructor.

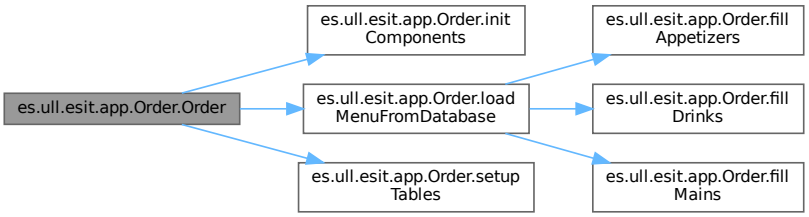
```
Creates the order menu window, initializes all Swing components,
configures the table models and loads the menu from the backend.
```

Definition at line 66 of file Order.java.

```
00066                 {
00067     initComponents();
00068
00069     // Initialize the Service Layer.
00070     ApiClient client = new ApiClient("http://localhost:8080");
00071     this.productService = new ProductService(client);
00072
00073     // Initialize receipt number label.
00074     receiptNoLbl.setText("Receipt No. : " + receiptNo);
00075
00076     // Configure tables and models.
00077     setupTables();
00078
00079     // Load menu items from database via REST API.
00080     loadMenuFromDatabase();
00081   }
```

References es.ull.esit.app.Order.initComponents(), es.ull.esit.app.Order.loadMenuFromDatabase(), es.ull.esit.app.Order.receiptNoLbl and es.ull.esit.app.Order.setupTables().

Here is the call graph for this function:



7.30.3 Member Function Documentation

7.30.3.1 fillAppetizers()

```
void es.ull.esit.app.Order.fillAppetizers (
            java.util.List< Appetizer > appetizers )  [inline], [private]
```

Fills the appetizers table model with data from the backend.

Parameters

appetizers	[java.util.List<Appetizer>] List of appetizers retrieved from the backend.

Definition at line 202 of file Order.java.

```
00202                                                                    {
00203     appetizersModel.setRowCount(0);
00204     for (Appetizer a : appetizers) {
00205       appetizersModel.addRow(new Object[] {
00206           a.getAppetizersId(),
00207           a.getItemAppetizers(),
00208           a.getAppetizersPrice(),
00209           0
00210       });
00211     }
00212   }
```

References es.ull.esit.app.Order.appetizersModel.

Referenced by es.ull.esit.app.Order.loadMenuFromDatabase().

Here is the caller graph for this function:

7.30.3.2 fillDrinks()

```
void es.ull.esit.app.Order.fillDrinks (
            java.util.List< Drink > drinks )  [inline], [private]
```

Fills the drinks table model with data from the backend.

Parameters

drinks	[java.util.List<Drink>] List of drinks retrieved from the backend.

Definition at line 185 of file Order.java.

```
00185                                                           {
00186     drinksModel.setRowCount(0);
00187     for (Drink d : drinks) {
00188       drinksModel.addRow(new Object[] {
00189           d.getDrinksId(),
00190           d.getItemDrinks(),
00191           d.getDrinksPrice(),
00192           0 // initial quantity
00193       });
00194     }
00195   }
```

References es.ull.esit.app.Order.drinksModel.

Referenced by es.ull.esit.app.Order.loadMenuFromDatabase().

Here is the caller graph for this function:

7.30.3.3 fillMains()

```
void es.ull.esit.app.Order.fillMains (
            java.util.List< MainCourse > mains )  [inline], [private]
```

Fills the main courses table model with data from the backend.

Parameters

mains	[java.util.List<MainCourse>] List of main courses retrieved from the backend.

Definition at line 219 of file Order.java.

```
00219                                                {
00220     mainsModel.setRowCount(0);
00221     for (MainCourse m : mains) {
00222       mainsModel.addRow(new Object[] {
00223           m.getFoodId(),
00224           m.getItemFood(),
00225           m.getFoodPrice(),
00226           0
00227       });
00228     }
00229   }
```

References es.ull.esit.app.Order.mainsModel.

Referenced by es.ull.esit.app.Order.loadMenuFromDatabase().

Here is the caller graph for this function:

7.30.3.4 goBackMenueBtnActionPerformed()

```
void es.ull.esit.app.Order.goBackMenueBtnActionPerformed (
            java.awt.event.ActionEvent evt )  [inline], [private]
```

Returns to the Login window.

```
   Closes the current Order window and opens a new instance
   of the Login form.
```

Parameters

evt	[java.awt.event.ActionEvent] Event triggered by clicking the "Go Back" button.

Definition at line 724 of file Order.java.

```
00724                                                    {//
      GEN-FIRST:event_goBackMenueBtnActionPerformed
```

```
00725     this.dispose();
00726     new Login().setVisible(true);
00727   }// GEN-LAST:event_goBackMenueBtnActionPerformed
```

7.30.3.5 initComponents()

```
void es.ull.esit.app.Order.initComponents ( )  [inline], [private]
```

Initializes GUI components.

```
   Generated by the Form Editor: do not modify manually.
   Creates and arranges the panels for Drinks, Appetizers,
   Main courses and the Receipt summary, as well as all labels,
   tables and buttons.
```

Definition at line 298 of file Order.java.

```
00298                                {
00299
00300     jPanel2 = new javax.swing.JPanel();
00301     DrinksPnl = new javax.swing.JPanel();
00302     drinksScroll = new javax.swing.JScrollPane();
00303     drinksTable = new javax.swing.JTable();
00304     AppetizerPnl = new javax.swing.JPanel();
00305     appetizersScroll = new javax.swing.JScrollPane();
00306     appetizersTable = new javax.swing.JTable();
00307     MainCoursePnl = new javax.swing.JPanel();
00308     mainsScroll = new javax.swing.JScrollPane();
00309     mainsTable = new javax.swing.JTable();
00310     jPanel1 = new javax.swing.JPanel();
00311     subTotalLbl = new javax.swing.JLabel();
00312     vatLbl = new javax.swing.JLabel();
00313     totalLbl = new javax.swing.JLabel();
00314     receiptNoLbl = new javax.swing.JLabel();
00315     payBtn = new javax.swing.JButton();
00316     newReceiptBtn = new javax.swing.JButton();
00317     saveReceiptBtn = new javax.swing.JButton();
00318     jLabel1 = new javax.swing.JLabel();
00319     goBackMenueBtn = new javax.swing.JButton();
00320     jLabel2 = new javax.swing.JLabel();
00321
00322     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00323     setTitle("Menu");
00324     setBackground(new java.awt.Color(204, 204, 204));
00325     setResizable(false);
00326
00327     jPanel2.setBackground(new java.awt.Color(248, 244, 230));
00328
00329     DrinksPnl.setBackground(new java.awt.Color(248, 244, 230));
00330     DrinksPnl.setBorder(javax.swing.BorderFactory.createTitledBorder(
00331         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Drinks",
00332         javax.swing.border.TitledBorder.CENTER, javax.swing.border.TitledBorder.TOP,
00333         new java.awt.Font("Yu Gothic UI", 0, 18))); // NOI18N
00334     DrinksPnl.setPreferredSize(new java.awt.Dimension(260, 278));
00335
00336     drinksTable.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00337     drinksTable.setModel(new javax.swing.table.DefaultTableModel(
00338         new Object[][] {
00339
00340         },
00341         new String[] {
00342
00343         }));
00344     drinksTable.getTableHeader().setReorderingAllowed(false);
00345     drinksScroll.setViewportView(drinksTable);
00346
00347     javax.swing.GroupLayout DrinksPnlLayout = new javax.swing.GroupLayout(DrinksPnl);
00348     DrinksPnl.setLayout(DrinksPnlLayout);
00349     DrinksPnlLayout.setHorizontalGroup(
00350         DrinksPnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00351             .addGroup(DrinksPnlLayout.createSequentialGroup()
00352                 .addContainerGap()
00353                 .addComponent(drinksScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 281,
      Short.MAX_VALUE)
00354                 .addContainerGap()));
00355     DrinksPnlLayout.setVerticalGroup(
00356         DrinksPnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00357             .addGroup(DrinksPnlLayout.createSequentialGroup()
```

```
00358                 .addContainerGap()
00359                 .addComponent(drinksScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 186,
    Short.MAX_VALUE)
00360                 .addContainerGap()));
00361
00362     AppetizerPnl.setBackground(new java.awt.Color(248, 244, 230));
00363     AppetizerPnl.setBorder(javax.swing.BorderFactory.createTitledBorder(
00364         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Appetizers",
00365         javax.swing.border.TitledBorder.CENTER, javax.swing.border.TitledBorder.TOP,
00366         new java.awt.Font("Yu Gothic UI", 0, 18))); // NOI18N
00367     AppetizerPnl.setPreferredSize(new java.awt.Dimension(260, 278));
00368
00369     appetizersTable.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00370     appetizersTable.setModel(new javax.swing.table.DefaultTableModel(
00371         new Object[][] {
00372
00373         },
00374         new String[] {
00375
00376         }));
00377     appetizersTable.getTableHeader().setReorderingAllowed(false);
00378     appetizersScroll.setViewportView(appetizersTable);
00379
00380     javax.swing.GroupLayout AppetizerPnlLayout = new javax.swing.GroupLayout(AppetizerPnl);
00381     AppetizerPnl.setLayout(AppetizerPnlLayout);
00382     AppetizerPnlLayout.setHorizontalGroup(
00383         AppetizerPnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00384             .addGroup(AppetizerPnlLayout.createSequentialGroup()
00385                 .addContainerGap()
00386                 .addComponent(appetizersScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 368,
    Short.MAX_VALUE)
00387                 .addContainerGap()));
00388     AppetizerPnlLayout.setVerticalGroup(
00389         AppetizerPnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00390             .addGroup(AppetizerPnlLayout.createSequentialGroup()
00391                 .addContainerGap()
00392                 .addComponent(appetizersScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 186,
    Short.MAX_VALUE)
00393                 .addContainerGap()));
00394
00395     MainCoursePnl.setBackground(new java.awt.Color(248, 244, 230));
00396     MainCoursePnl.setBorder(javax.swing.BorderFactory.createTitledBorder(
00397         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Main Course",
00398         javax.swing.border.TitledBorder.CENTER, javax.swing.border.TitledBorder.TOP,
00399         new java.awt.Font("Yu Gothic UI", 0, 18))); // NOI18N
00400     MainCoursePnl.setPreferredSize(new java.awt.Dimension(260, 278));
00401
00402     mainsTable.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00403     mainsTable.setModel(new javax.swing.table.DefaultTableModel(
00404         new Object[][] {
00405
00406         },
00407         new String[] {
00408
00409         }));
00410     mainsTable.getTableHeader().setReorderingAllowed(false);
00411     mainsScroll.setViewportView(mainsTable);
00412
00413     javax.swing.GroupLayout MainCoursePnlLayout = new javax.swing.GroupLayout(MainCoursePnl);
00414     MainCoursePnl.setLayout(MainCoursePnlLayout);
00415     MainCoursePnlLayout.setHorizontalGroup(
00416         MainCoursePnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00417             .addGroup(MainCoursePnlLayout.createSequentialGroup()
00418                 .addContainerGap()
00419                 .addComponent(mainsScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 427, Short.MAX_VALUE)
00420                 .addContainerGap()));
00421     MainCoursePnlLayout.setVerticalGroup(
00422         MainCoursePnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00423             .addGroup(MainCoursePnlLayout.createSequentialGroup()
00424                 .addContainerGap()
00425                 .addComponent(mainsScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 186, Short.MAX_VALUE)
00426                 .addContainerGap()));
00427
00428     jPanel1.setBackground(new java.awt.Color(248, 244, 230));
00429     jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(
00430         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Receipt",
00431         javax.swing.border.TitledBorder.CENTER, javax.swing.border.TitledBorder.TOP,
00432         new java.awt.Font("Yu Gothic UI", 0, 14))); // NOI18N
00433
00434     subTotalLbl.setFont(new java.awt.Font("Yu Gothic UI Light", 0, 18)); // NOI18N
00435     subTotalLbl.setText("SubTotal: 0.0 SR");
00436
00437     vatLbl.setFont(new java.awt.Font("Yu Gothic UI Light", 0, 18)); // NOI18N
00438     vatLbl.setText("VAT included: 0.0 SR");
00439
00440     totalLbl.setFont(new java.awt.Font("Yu Gothic UI Light", 0, 18)); // NOI18N
00441     totalLbl.setText("Total: 0.0 SR");
```

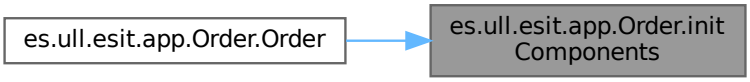
```
00442
00443     receiptNoLbl.setFont(new java.awt.Font("Yu Gothic UI Light", 0, 14)); // NOI18N
00444     receiptNoLbl.setText("Receipt No. : 0");
00445
00446     javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
00447     jPanel1.setLayout(jPanel1Layout);
00448     jPanel1Layout.setHorizontalGroup(
00449         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00450             .addGroup(jPanel1Layout.createSequentialGroup()
00451                 .addContainerGap()
00452                 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00453                     .addComponent(subTotalLbl)
00454                     .addComponent(vatLbl)
00455                     .addComponent(totalLbl)
00456                     .addComponent(receiptNoLbl))
00457                 .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)));
00458     jPanel1Layout.setVerticalGroup(
00459         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00460             .addGroup(jPanel1Layout.createSequentialGroup()
00461                 .addContainerGap()
00462                 .addComponent(subTotalLbl)
00463                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
00464                 .addComponent(vatLbl)
00465                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
00466                 .addComponent(totalLbl)
00467                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 10,
    Short.MAX_VALUE)
00468                 .addComponent(receiptNoLbl)));
00469
00470     payBtn.setFont(new java.awt.Font("Lucida Grande", 1, 18)); // NOI18N
00471     payBtn.setText("Pay");
00472     payBtn.addActionListener(new java.awt.event.ActionListener() {
00473       public void actionPerformed(java.awt.event.ActionEvent evt) {
00474         payBtnActionPerformed(evt);
00475       }
00476     });
00477
00478     newReceiptBtn.setFont(new java.awt.Font("Yu Gothic UI", 1, 14)); // NOI18N
00479     newReceiptBtn.setText("New Receipt");
00480     newReceiptBtn.addActionListener(new java.awt.event.ActionListener() {
00481       public void actionPerformed(java.awt.event.ActionEvent evt) {
00482         newReceiptBtnActionPerformed(evt);
00483       }
00484     });
00485
00486     saveReceiptBtn.setFont(new java.awt.Font("Yu Gothic UI", 1, 14)); // NOI18N
00487     saveReceiptBtn.setText("Save Receipt");
00488     saveReceiptBtn.addActionListener(new java.awt.event.ActionListener() {
00489       public void actionPerformed(java.awt.event.ActionEvent evt) {
00490         saveReceiptBtnActionPerformed(evt);
00491       }
00492     });
00493
00494     jLabel1.setBackground(new java.awt.Color(255, 153, 0));
00495     jLabel1.setFont(new java.awt.Font("Yu Gothic UI", 1, 36)); // NOI18N
00496     jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00497     jLabel1.setText("BLACK PLATE MENU");
00498
00499     goBackMenueBtn.setFont(new java.awt.Font("Yu Gothic UI", 1, 14)); // NOI18N
00500     goBackMenueBtn.setText("Go Back");
00501     goBackMenueBtn.addActionListener(new java.awt.event.ActionListener() {
00502       public void actionPerformed(java.awt.event.ActionEvent evt) {
00503         goBackMenueBtnActionPerformed(evt);
00504       }
00505     });
00506
00507     jLabel2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/ui/screenshot.png"))); //
    NOI18N
00508
00509     javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);
00510     jPanel2.setLayout(jPanel2Layout);
00511     jPanel2Layout.setHorizontalGroup(
00512         jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00513             .addGroup(jPanel2Layout.createSequentialGroup()
00514                 .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00515                     .addGroup(jPanel2Layout.createSequentialGroup()
00516                         .addGap(40, 40, 40)
00517                         .addComponent(DrinksPnl, javax.swing.GroupLayout.PREFERRED_SIZE, 305,
00518                             javax.swing.GroupLayout.PREFERRED_SIZE)
00519                         .addGap(18, 18, 18)
00520                         .addComponent(AppetizerPnl, javax.swing.GroupLayout.PREFERRED_SIZE, 392,
00521                             javax.swing.GroupLayout.PREFERRED_SIZE)
00522                         .addGap(18, 18, 18)
00523                         .addComponent(MainCoursePnl, javax.swing.GroupLayout.PREFERRED_SIZE, 451,
00524                             javax.swing.GroupLayout.PREFERRED_SIZE))
00525                     .addGroup(jPanel2Layout.createSequentialGroup()
00526                         .addGap(251, 251, 251)
```

```
00527                           .addComponent(jLabel2)
00528                           .addGap(18, 18, 18)
00529                           .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 433,
00530                               javax.swing.GroupLayout.PREFERRED_SIZE))
00531                       .addGroup(jPanel2Layout.createSequentialGroup()
00532                           .addGap(16, 16, 16)
00533                           .addComponent(goBackMenueBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 130,
00534                               javax.swing.GroupLayout.PREFERRED_SIZE)
00535                           .addGap(147, 147, 147)
00536                           .addComponent(payBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 180,
00537                               javax.swing.GroupLayout.PREFERRED_SIZE)
00538                           .addGap(18, 18, 18)
00539
    .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00540                               .addComponent(newReceiptBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 220,
00541                                   javax.swing.GroupLayout.PREFERRED_SIZE)
00542                               .addComponent(saveReceiptBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 220,
00543                                   javax.swing.GroupLayout.PREFERRED_SIZE))
00544                           .addGap(28, 28, 28)
00545                           .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
00546                               javax.swing.GroupLayout.DEFAULT_SIZE,
    javax.swing.GroupLayout.PREFERRED_SIZE)))
00547                   .addContainerGap(30, Short.MAX_VALUE)));
00548       jPanel2Layout.setVerticalGroup(
00549           jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00550               .addGroup(jPanel2Layout.createSequentialGroup()
00551                   .addContainerGap(16, Short.MAX_VALUE)
00552                   .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00553                       .addComponent(jLabel2, javax.swing.GroupLayout.Alignment.TRAILING)
00554                       .addComponent(jLabel1, javax.swing.GroupLayout.Alignment.TRAILING,
00555                           javax.swing.GroupLayout.PREFERRED_SIZE, 60,
    javax.swing.GroupLayout.PREFERRED_SIZE))
00556                   .addGap(40, 40, 40)
00557                   .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING,
    false)
00558                       .addComponent(AppetizerPnl, javax.swing.GroupLayout.DEFAULT_SIZE, 230,
    Short.MAX_VALUE)
00559                       .addComponent(MainCoursePnl, javax.swing.GroupLayout.DEFAULT_SIZE, 230,
    Short.MAX_VALUE)
00560                       .addComponent(DrinksPnl, javax.swing.GroupLayout.DEFAULT_SIZE, 230,
    Short.MAX_VALUE))
00561                   .addGap(18, 18, 18)
00562                   .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00563                       .addComponent(goBackMenueBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 40,
00564                           javax.swing.GroupLayout.PREFERRED_SIZE)
00565                       .addGroup(jPanel2Layout.createSequentialGroup()
00566                           .addComponent(newReceiptBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 30,
00567                               javax.swing.GroupLayout.PREFERRED_SIZE)
00568                           .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
00569                           .addComponent(saveReceiptBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 29,
00570                               javax.swing.GroupLayout.PREFERRED_SIZE))
00571                       .addComponent(payBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 70,
00572                           javax.swing.GroupLayout.PREFERRED_SIZE)
00573                       .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
    javax.swing.GroupLayout.DEFAULT_SIZE,
00574                           javax.swing.GroupLayout.PREFERRED_SIZE))
00575                   .addGap(30, 30, 30)));
00576
00577       javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00578       getContentPane().setLayout(layout);
00579       layout.setHorizontalGroup(
00580           layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00581               .addComponent(jPanel2, javax.swing.GroupLayout.DEFAULT_SIZE,
    javax.swing.GroupLayout.DEFAULT_SIZE,
00582                   Short.MAX_VALUE));
00583       layout.setVerticalGroup(
00584           layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00585               .addGroup(layout.createSequentialGroup()
00586                   .addComponent(jPanel2, javax.swing.GroupLayout.DEFAULT_SIZE,
    javax.swing.GroupLayout.DEFAULT_SIZE,
00587                       Short.MAX_VALUE)
00588                   .addContainerGap()));
00589
00590       pack();
00591       setLocationRelativeTo(null);
00592     }// </editor-fold>//GEN-END:initComponents
```

References es.ull.esit.app.Order.AppetizerPnl, es.ull.esit.app.Order.appetizersScroll, es.ull.esit.app.Order.appetizersTable, es.ull.esit.app.Order.DrinksPnl, es.ull.esit.app.Order.drinksScroll, es.ull.esit.app.Order.drinksTable, es.ull.esit.app.Order.goBackMenue es.ull.esit.app.Order.jLabel1, es.ull.esit.app.Order.jLabel2, es.ull.esit.app.Order.jPanel1, es.ull.esit.app.Order.jPanel2, es.ull.esit.app.Order.MainCoursePnl, es.ull.esit.app.Order.mainsScroll, es.ull.esit.app.Order.mainsTable, es.ull.esit.app.Order.newRec es.ull.esit.app.Order.payBtn, es.ull.esit.app.Order.receiptNoLbl, es.ull.esit.app.Order.saveReceiptBtn, es.ull.esit.app.Order.subTotalLb es.ull.esit.app.Order.totalLbl, and es.ull.esit.app.Order.vatLbl.

Referenced by es.ull.esit.app.Order.Order().

Here is the caller graph for this function:



7.30.3.6 loadMenuFromDatabase()

```
void es.ull.esit.app.Order.loadMenuFromDatabase ( )  [inline], [private]
```

Loads menu data (drinks, appetizers, main courses) from the backend in a background thread.

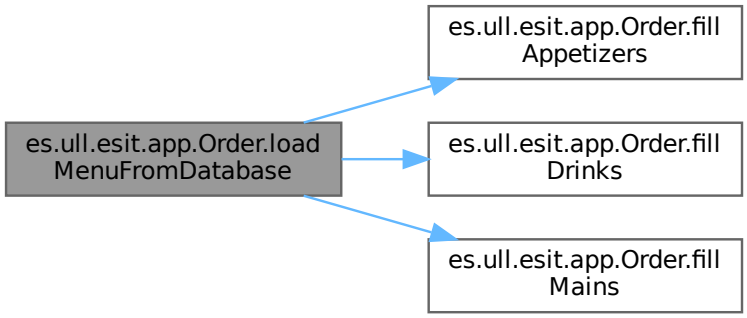
Definition at line 156 of file Order.java.

```
00156                                    {
00157     new Thread(() -> {
00158       try {
00159         java.util.List<Drink> drinks = productService.getAllDrinks();
00160         java.util.List<Appetizer> appetizers = productService.getAllAppetizers();
00161         java.util.List<MainCourse> mains = productService.getAllMainCourses();
00162 
00163         SwingUtilities.invokeLater(() -> {
00164           fillDrinks(drinks);
00165           fillAppetizers(appetizers);
00166           fillMains(mains);
00167         });
00168 
00169       } catch (Exception ex) {
00170         ex.printStackTrace();
00171         SwingUtilities.invokeLater(() -> JOptionPane.showMessageDialog(
00172             this,
00173             "Error loading menu from backend:\n" + ex.getMessage(),
00174             "Menu loading error",
00175             JOptionPane.ERROR_MESSAGE));
00176       }
00177     }).start();
00178   }
```

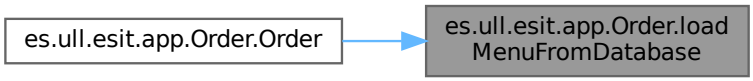
References es.ull.esit.app.Order.fillAppetizers(), es.ull.esit.app.Order.fillDrinks(), es.ull.esit.app.Order.fillMains(), and es.ull.esit.app.Order.productService.

Referenced by es.ull.esit.app.Order.Order().

Here is the call graph for this function:



Here is the caller graph for this function:



7.30.3.7 main()

```
static void es.ull.esit.app.Order.main (
            String args[] )  [inline], [static]
```

Standalone entry point for testing the Order window.

```
Sets the Nimbus look and feel if available and displays a new
instance of Order. In the normal application flow, this window
is opened after a successful login.
```

Parameters

args	[String[]] Command line arguments (not used).

Definition at line 738 of file Order.java.

```
00738                                     {
00739     /* Set the Nimbus look and feel */
00740     // <editor-fold defaultstate="collapsed" desc=" Look and feel setting code
00741     // (optional) ">
00742     /*
00743      * If Nimbus (introduced in Java SE 6) is not available, stay with the default
00744      * look and feel.
00745      * For details see
00746      * http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
00747      */
00748     try {
00749       for (javax.swing.UIManager.LookAndFeelInfo info :
      javax.swing.UIManager.getInstalledLookAndFeels()) {
00750         if ("Nimbus".equals(info.getName())) {
00751           javax.swing.UIManager.setLookAndFeel(info.getClassName());
00752           break;
00753         }
00754       }
00755     } catch (ClassNotFoundException ex) {
00756       java.util.logging.Logger.getLogger(Order.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
00757     } catch (InstantiationException ex) {
00758       java.util.logging.Logger.getLogger(Order.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
00759     } catch (IllegalAccessException ex) {
00760       java.util.logging.Logger.getLogger(Order.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
00761     } catch (javax.swing.UnsupportedLookAndFeelException ex) {
00762       java.util.logging.Logger.getLogger(Order.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
00763     }
00764     // </editor-fold>
00765
00766     /* Create and display the form */
00767     java.awt.EventQueue.invokeLater(new Runnable() {
00768       public void run() {
00769         new Order().setVisible(true);
```

```
00770       }
00771     });
00772   }
```

7.30.3.8 newReceiptBtnActionPerformed()

```
void es.ull.esit.app.Order.newReceiptBtnActionPerformed (
            java.awt.event.ActionEvent evt )  [inline], [private]
```

Starts a new receipt and resets the form to its initial state.

```
Only works when the current total is not zero. If so, it:
- resets Qty column to 0 in all tables,
- resets subtotal, VAT and total labels,
- clears internal subtotal, VAT and total variables,
- clears lastBill,
- increments the receipt number and updates its label.
```

Parameters

evt	[java.awt.event.ActionEvent] Event triggered by clicking the "New Receipt" button.

Definition at line 692 of file Order.java.

```
00692                                                                    {//
      GEN-FIRST:event_newReceiptBtnActionPerformed
00693     if (total == 0.0) {
00694       // Nothing to reset
00695       return;
00696     }
00697
00698     resetQtyColumn(drinksModel);
00699     resetQtyColumn(appetizersModel);
00700     resetQtyColumn(mainsModel);
00701
00702     subTotal = 0.0;
00703     vat = 0.0;
00704     total = 0.0;
00705     lastBill = null;
00706
00707     subTotalLbl.setText("SubTotal: 0.0 SR");
00708     vatLbl.setText("VAT included: 0.0 SR");
00709     totalLbl.setText("Total: 0.0 SR");
00710
00711     receiptNo++;
00712     receiptNoLbl.setText("Receipt No. : " + receiptNo);
00713   }// GEN-LAST:event_newReceiptBtnActionPerformed
```

7.30.3.9 payBtnActionPerformed()

```
void es.ull.esit.app.Order.payBtnActionPerformed (
            java.awt.event.ActionEvent evt )  [inline], [private]
```

Calculates subtotal, VAT and total and shows a confirmation dialog.

```
Steps:
- Sums all line prices from the three tables.
- Uses OrderService to compute BillResult (subtotal, VAT, total).
- Updates the labels in the Receipt panel.
- If no items were selected (sum == 0), shows an information dialog.
- Otherwise, shows a "Paid successfully" dialog.
```

Parameters

evt	[java.awt.event.ActionEvent] Event triggered by clicking the "Pay" button.

Definition at line 607 of file Order.java.

```
00607                                                                    {//
      GEN-FIRST:event_payBtnActionPerformed
00608     double itemsTotal = 0.0;
00609 
00610     itemsTotal += sumFromModel(drinksModel);
00611     itemsTotal += sumFromModel(appetizersModel);
00612     itemsTotal += sumFromModel(mainsModel);
00613 
00614     if (itemsTotal == 0.0) {
00615       JOptionPane.showMessageDialog(
00616           this,
00617           "Please select at least one item (Qty > 0).",
00618           "No items selected",
00619           JOptionPane.INFORMATION_MESSAGE);
00620       return;
00621     }
00622 
00623     // Calculate bill using the dedicated service
00624     lastBill = orderService.calculateBill(itemsTotal);
00625 
00626     subTotal = lastBill.getSubTotal();
00627     vat = lastBill.getVat();
00628     total = lastBill.getTotal();
00629 
00630     subTotalLbl.setText("SubTotal: " + subTotal + " SR");
00631     vatLbl.setText("VAT included: " + vat + " SR");
00632     totalLbl.setText("Total: " + total + " SR");
00633 
00634     JOptionPane.showMessageDialog(this, "Paid successfully");
00635   }// GEN-LAST:event_payBtnActionPerformed
```

7.30.3.10 resetQtyColumn()

```
void es.ull.esit.app.Order.resetQtyColumn (
             DefaultTableModel model )  [inline], [private]
```

Sets the Qty column of the given model to 0 in all rows.

Definition at line 271 of file Order.java.

```
00271                                                    {
00272     for (int row = 0; row < model.getRowCount(); row++) {
00273       model.setValueAt(0, row, 3); // column 3 is Qty
00274     }
00275   }
```

7.30.3.11 saveReceiptBtnActionPerformed()

```
void es.ull.esit.app.Order.saveReceiptBtnActionPerformed (
             java.awt.event.ActionEvent evt )  [inline], [private]
```

Saves the current bill information to a text file through OrderService.

The file is created under the "receipts" directory with the name:

- "billNo.<receiptNo>.txt"

If there is no calculated bill (lastBill == null or total == 0), an information dialog is shown and no file is created.

Parameters

evt	[java.awt.event.ActionEvent] Event triggered by clicking the "Save Receipt" button.

Definition at line 650 of file Order.java.

```
00650                                                                                {//
      GEN-FIRST:event_saveReceiptBtnActionPerformed
00651     try {
00652       if (lastBill == null || lastBill.getTotal() == 0.0) {
00653         JOptionPane.showMessageDialog(
00654             this,
00655             "There is no paid bill to save.\nPlease press Pay first.",
00656             "No bill",
00657             JOptionPane.INFORMATION_MESSAGE);
00658         return;
00659       }
00660
00661       orderService.generateReceiptFile(receiptNo, lastBill);
00662
00663       JOptionPane.showMessageDialog(
00664           this,
00665           "Receipt number: " + receiptNo + " has been saved successfully.",
00666           "Receipt saved",
00667           JOptionPane.INFORMATION_MESSAGE);
00668
00669     } catch (Exception ex) {
00670       ex.printStackTrace();
00671       JOptionPane.showMessageDialog(
00672           this,
00673           "Error saving receipt:\n" + ex.getMessage(),
00674           "Error",
00675           JOptionPane.ERROR_MESSAGE);
00676     }
00677   }// GEN-LAST:event_saveReceiptBtnActionPerformed
```

7.30.3.12 setupTables()

```
void es.ull.esit.app.Order.setupTables ( )  [inline], [private]
```

Configures the three tables (Drinks, Appetizers, Main Course).

```
Each table has four columns:
- ID (not editable)
- Item (not editable)
- Price (not editable)
- Qty (editable, quantity selected by the cashier)
```

Definition at line 92 of file Order.java.

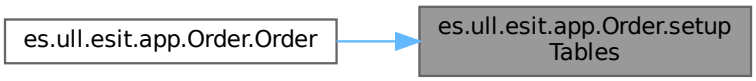
```
00092                             {
00093     // DRINKS
00094     drinksModel = new DefaultTableModel(
00095         new Object[] { "ID", "Item", "Price (SR)", "Qty" }, 0) {
00096       @Override
00097       public boolean isCellEditable(int row, int column) {
00098         // Only Qty column is editable
00099         return column == 3;
00100       }
00101
00102       @Override
00103       public Class<?> getColumnClass(int columnIndex) {
00104         return switch (columnIndex) {
00105           case 0 -> Long.class; // ID
00106           case 2, 3 -> Integer.class; // Price, Qty
00107           default -> String.class;
00108         };
00109       }
00110     };
00111     drinksTable.setModel(drinksModel);
00112
00113     // APPETIZERS
00114     appetizersModel = new DefaultTableModel(
00115         new Object[] { "ID", "Item", "Price (SR)", "Qty" }, 0) {
00116       @Override
```

```
00117       public boolean isCellEditable(int row, int column) {
00118         return column == 3;
00119       }
00120
00121       @Override
00122       public Class<?> getColumnClass(int columnIndex) {
00123         return switch (columnIndex) {
00124           case 0 -> Long.class;
00125           case 2, 3 -> Integer.class;
00126           default -> String.class;
00127         };
00128       }
00129     };
00130     appetizersTable.setModel(appetizersModel);
00131
00132     // MAIN COURSES
00133     mainsModel = new DefaultTableModel(
00134         new Object[] { "ID", "Item", "Price (SR)", "Qty" }, 0) {
00135       @Override
00136       public boolean isCellEditable(int row, int column) {
00137         return column == 3;
00138       }
00139
00140       @Override
00141       public Class<?> getColumnClass(int columnIndex) {
00142         return switch (columnIndex) {
00143           case 0 -> Long.class;
00144           case 2, 3 -> Integer.class;
00145           default -> String.class;
00146         };
00147       }
00148     };
00149     mainsTable.setModel(mainsModel);
00150   }
```

References es.ull.esit.app.Order.appetizersModel, es.ull.esit.app.Order.appetizersTable, es.ull.esit.app.Order.drinksModel, es.ull.esit.app.Order.drinksTable, es.ull.esit.app.Order.mainsModel, and es.ull.esit.app.Order.mainsTable.

Referenced by es.ull.esit.app.Order.Order().

Here is the caller graph for this function:



7.30.3.13 sumFromModel()

```
double es.ull.esit.app.Order.sumFromModel (
            DefaultTableModel model )  [inline], [private]
```

Sums the total price for a given table model.

```
For each row:
- reads Price (column 2) and Qty (column 3),
- if Qty > 0, accumulates (Qty * Price).
```

Parameters

model	[javax.swing.table.DefaultTableModel] Table model (drinksModel, appetizersModel or mainsModel).

Returns

[double] Sum of the line totals in that model.

Definition at line 241 of file Order.java.

```
00241                                                    {
00242     double sum = 0.0;
00243     for (int row = 0; row < model.getRowCount(); row++) {
00244       Object qtyObj = model.getValueAt(row, 3); // Qty
00245       Object priceObj = model.getValueAt(row, 2); // Price
00246 
00247       if (qtyObj == null || priceObj == null)
00248         continue;
00249 
00250       int qty;
00251       int price;
00252       try {
00253         qty = ((Number) qtyObj).intValue();
00254         price = ((Number) priceObj).intValue();
00255       } catch (ClassCastException e) {
00256         // Fallback in case something comes as String
00257         qty = Integer.parseInt(qtyObj.toString());
00258         price = Integer.parseInt(priceObj.toString());
00259       }
00260 
00261       if (qty > 0) {
00262         sum += qty * price;
00263       }
00264     }
00265     return sum;
00266   }
```

7.30.4 Member Data Documentation

7.30.4.1 AppetizerPnl

```
javax.swing.JPanel es.ull.esit.app.Order.AppetizerPnl  [private]
```

Panel that groups all appetizer items and controls.

Definition at line 776 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents().

7.30.4.2 appetizersModel

```
DefaultTableModel es.ull.esit.app.Order.appetizersModel  [private]
```

Definition at line 38 of file Order.java.

Referenced by es.ull.esit.app.Order.fillAppetizers(), and es.ull.esit.app.Order.setupTables().

7.30.4.3 appetizersScroll

```
javax.swing.JScrollPane es.ull.esit.app.Order.appetizersScroll  [private]
```

Scroll pane for appetizers table.

Definition at line 784 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents().

7.30.4.4 appetizersTable

```
javax.swing.JTable es.ull.esit.app.Order.appetizersTable  [private]
```

Table for appetizers items.

Definition at line 782 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents(), and es.ull.esit.app.Order.setupTables().

7.30.4.5 drinksModel

```
DefaultTableModel es.ull.esit.app.Order.drinksModel  [private]
```

Table models for the three product categories.

Definition at line 37 of file Order.java.

Referenced by es.ull.esit.app.Order.fillDrinks(), and es.ull.esit.app.Order.setupTables().

7.30.4.6 DrinksPnl

```
javax.swing.JPanel es.ull.esit.app.Order.DrinksPnl  [private]
```

Panel that groups all drink items and controls.

Definition at line 778 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents().

7.30.4.7 drinksScroll

```
javax.swing.JScrollPane es.ull.esit.app.Order.drinksScroll  [private]
```

Scroll pane for drinks table.

Definition at line 788 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents().

7.30.4.8 drinksTable

```
javax.swing.JTable es.ull.esit.app.Order.drinksTable  [private]
```

Table for drinks items.

Definition at line 786 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents(), and es.ull.esit.app.Order.setupTables().

7.30.4.9 goBackMenueBtn

```
javax.swing.JButton es.ull.esit.app.Order.goBackMenueBtn  [private]
```

Button to go back to the Login window.

Definition at line 790 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents().

7.30.4.10 jLabel1

```
javax.swing.JLabel es.ull.esit.app.Order.jLabel1  [private]
```

Title label "BLACK PLATE MENU".

Definition at line 792 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents().

7.30.4.11 jLabel2

```
javax.swing.JLabel es.ull.esit.app.Order.jLabel2  [private]
```

Logo / screenshot label on the top left.

Definition at line 794 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents().

7.30.4.12 jPanel1

```
javax.swing.JPanel es.ull.esit.app.Order.jPanel1  [private]
```

Panel that shows subtotal, VAT, total and receipt number.

Definition at line 796 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents().

7.30.4.13 jPanel2

```
javax.swing.JPanel es.ull.esit.app.Order.jPanel2  [private]
```

Main container panel of the Order window.

Definition at line 798 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents().

7.30.4.14 lastBill

```
BillResult es.ull.esit.app.Order.lastBill  [private]
```

Last calculated bill (after pressing Pay).

Definition at line 42 of file Order.java.

7.30.4.15 MainCoursePnl

```
javax.swing.JPanel es.ull.esit.app.Order.MainCoursePnl  [private]
```

Panel that groups all main course items and controls.

Definition at line 780 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents().

7.30.4.16 mainsModel

```
DefaultTableModel es.ull.esit.app.Order.mainsModel  [private]
```

Definition at line 39 of file Order.java.

Referenced by es.ull.esit.app.Order.fillMains(), and es.ull.esit.app.Order.setupTables().

7.30.4.17 mainsScroll

```
javax.swing.JScrollPane es.ull.esit.app.Order.mainsScroll  [private]
```

Scroll pane for main course table.

Definition at line 802 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents().

7.30.4.18 mainsTable

```
javax.swing.JTable es.ull.esit.app.Order.mainsTable  [private]
```

Table for main course items.

Definition at line 800 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents(), and es.ull.esit.app.Order.setupTables().

7.30.4.19 newReceiptBtn

```
javax.swing.JButton es.ull.esit.app.Order.newReceiptBtn  [private]
```

Button to start a new receipt.

Definition at line 804 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents().

7.30.4.20 orderService

```
final OrderService es.ull.esit.app.Order.orderService = new OrderService()  [private]
```

Service used to calculate bill totals and generate receipt files.

Definition at line 34 of file Order.java.

7.30.4.21 payBtn

```
javax.swing.JButton es.ull.esit.app.Order.payBtn  [private]
```

Button to calculate and confirm payment.

Definition at line 806 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents().

7.30.4.22 productService

```
final ProductService es.ull.esit.app.Order.productService  [private]
```

Service used to load products from the backend.

Definition at line 31 of file Order.java.

Referenced by es.ull.esit.app.Order.loadMenuFromDatabase().

7.30.4.23 receiptNoLbl

```
javax.swing.JLabel es.ull.esit.app.Order.receiptNoLbl  [private]
```

Label that shows the current receipt number.

Definition at line 808 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents(), and es.ull.esit.app.Order.Order().

7.30.4.24 saveReceiptBtn

```
javax.swing.JButton es.ull.esit.app.Order.saveReceiptBtn  [private]
```

Button to save the current receipt to a file.

Definition at line 810 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents().

7.30.4.25 subTotalLbl

```
javax.swing.JLabel es.ull.esit.app.Order.subTotalLbl  [private]
```

Label that displays the subtotal amount.

Definition at line 812 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents().

7.30.4.26 totalLbl

```
javax.swing.JLabel es.ull.esit.app.Order.totalLbl  [private]
```

Label that displays the total amount.

Definition at line 814 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents().

7.30.4.27 vatLbl

```
javax.swing.JLabel es.ull.esit.app.Order.vatLbl  [private]
```

Label that displays the VAT amount.

Definition at line 816 of file Order.java.

Referenced by es.ull.esit.app.Order.initComponents().

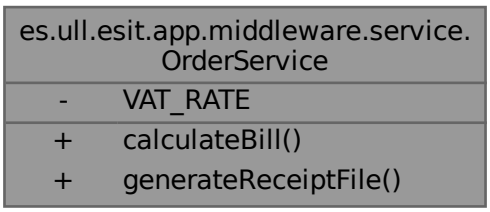
The documentation for this class was generated from the following file:

- Order.java

7.31 es.ull.esit.app.middleware.service.OrderService Class Reference

Service that handles order-related calculations and receipt generation.

Collaboration diagram for es.ull.esit.app.middleware.service.OrderService:



Public Member Functions

- BillResult calculateBill (double itemsTotalSum)

  Calculates subtotal, VAT and total for a given items sum.
- void generateReceiptFile (int receiptNo, BillResult bill) throws Exception

  Generates a local text file representing a receipt.

Static Private Attributes

- static final double VAT_RATE = 0.15

  VAT rate applied to the subtotal (15%).

7.31.1 Detailed Description

Service that handles order-related calculations and receipt generation.

```
Provides methods to compute totals (including VAT) and to generate
local text files representing printed receipts.
```

Definition at line 11 of file OrderService.java.

7.31.2 Member Function Documentation

7.31.2.1 calculateBill()

```
BillResult es.ull.esit.app.middleware.service.OrderService.calculateBill (
            double itemsTotalSum )  [inline]
```

Calculates subtotal, VAT and total for a given items sum.

```
The values are rounded to two decimal places.
```

Parameters

itemsTotalSum	[double] Sum of item prices before VAT.

Returns

[BillResult] Container holding subtotal, VAT and total amounts.

Definition at line 24 of file OrderService.java.

```
00024                                                    {
00025     double vat = itemsTotalSum * VAT_RATE;
00026     double total = itemsTotalSum + vat;
00027
00028     return new BillResult(
00029         Math.round(itemsTotalSum * 100.0) / 100.0,
00030         Math.round(vat * 100.0) / 100.0,
00031         Math.round(total * 100.0) / 100.0);
00032   }
```

References es.ull.esit.app.middleware.service.OrderService.VAT_RATE.

7.31.2.2 generateReceiptFile()

```
void es.ull.esit.app.middleware.service.OrderService.generateReceiptFile (
             int receiptNo,
             BillResult bill ) throws Exception  [inline]
```

Generates a local text file representing a receipt.

```
The file is stored under a "receipts" directory created in the
working folder. File name format is "billNo.<receiptNo>.txt".
```

Parameters

receiptNo	[int] Numeric identifier of the receipt.
bill	[BillResult] Calculated bill result to print.

Exceptions

Exception	If an error occurs while creating or writing the file.

Definition at line 44 of file OrderService.java.

```
00044                                                                                    {
00045     // Ensure the directory exists.
00046     new java.io.File("receipts").mkdirs();
00047
00048     try (java.io.PrintWriter output = new java.io.PrintWriter("receipts/billNo." + receiptNo +
      ".txt")) {
00049       output.println(" Bill number is: " + receiptNo);
00050       output.println("==============");
00051       output.println("--------------");
00052       output.println("Subtotal is: " + bill.getSubTotal() + " SR");
00053       output.println("vat: " + bill.getVat() + " SR");
00054       output.println("Total is: " + bill.getTotal() + " SR");
00055       output.println();
00056       output.println("THANK YOU FOR ORDERING");
00057     }
00058   }
```

7.31.3 Member Data Documentation

7.31.3.1 VAT_RATE

```
final double es.ull.esit.app.middleware.service.OrderService.VAT_RATE = 0.15  [static], [private]
```

VAT rate applied to the subtotal (15%).

Definition at line 14 of file OrderService.java.

Referenced by es.ull.esit.app.middleware.service.OrderService.calculateBill().

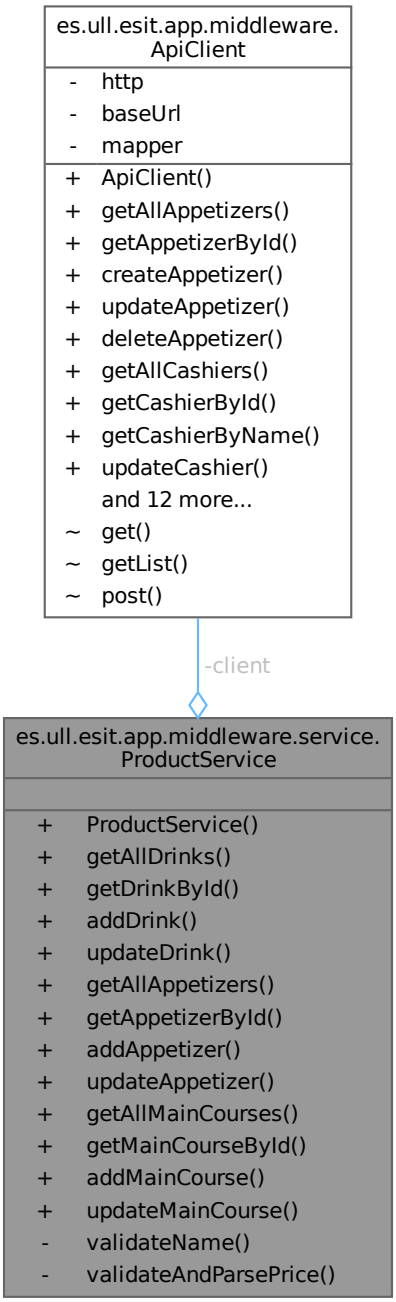
The documentation for this class was generated from the following file:

- OrderService.java

7.32 es.ull.esit.app.middleware.service.ProductService Class Reference

Service handling business logic for menu products.

Collaboration diagram for es.ull.esit.app.middleware.service.ProductService:



Public Member Functions

- ProductService (ApiClient client)

  Constructs a ProductService with the given API client.
- List< Drink > getAllDrinks () throws Exception

  Retrieves all drinks from the backend.
- Drink getDrinkById (Long id) throws Exception

Retrieves a single drink by its identifier.

- void addDrink (String name, String priceStr) throws Exception

  Adds a new drink to the backend menu.
- void updateDrink (Long id, String name, String priceStr) throws Exception

  Updates an existing drink in the backend menu.
- List< Appetizer > getAllAppetizers () throws Exception

  Retrieves all appetizers from the backend.
- Appetizer getAppetizerById (Long id) throws Exception

  Retrieves a single appetizer by its identifier.
- void addAppetizer (String name, String priceStr) throws Exception

  Adds a new appetizer to the backend menu.
- void updateAppetizer (Long id, String name, String priceStr) throws Exception

  Updates an existing appetizer in the backend menu.
- List< MainCourse > getAllMainCourses () throws Exception

  Retrieves all main courses from the backend.
- MainCourse getMainCourseById (Long id) throws Exception

  Retrieves a single main course by its identifier.
- void addMainCourse (String name, String priceStr) throws Exception

  Adds a new main course to the backend menu.
- void updateMainCourse (Long id, String name, String priceStr) throws Exception

  Updates an existing main course in the backend menu.

Private Member Functions

- void validateName (String name)

  Validates that the item name is not null or empty.
- int validateAndParsePrice (String priceStr)

  Validates and parses the price from string to integer.

Private Attributes

- final ApiClient client

  REST API client used to communicate with the backend menu endpoints.

7.32.1 Detailed Description

Service handling business logic for menu products.

```
Decouples Swing UI components from direct REST API calls. It centralizes
validation and orchestrates requests for drinks, appetizers and main
courses.
```

Definition at line 16 of file ProductService.java.

7.32.2 Constructor & Destructor Documentation

7.32.2.1 ProductService()

```
es.ull.esit.app.middleware.service.ProductService.ProductService (
            ApiClient client )  [inline]
```

Constructs a ProductService with the given API client.

Parameters

client	[ApiClient] REST client used to perform HTTP requests.

Definition at line 26 of file ProductService.java.

```
00026                                       {
00027     this.client = client;
00028   }
```

References es.ull.esit.app.middleware.service.ProductService.client.

7.32.3 Member Function Documentation

7.32.3.1 addAppetizer()

```
void es.ull.esit.app.middleware.service.ProductService.addAppetizer (
            String name,
            String priceStr ) throws Exception  [inline]
```

Adds a new appetizer to the backend menu.

```
   Validates user input and sends a POST request using the ApiClient.
```

Parameters

name	[String] Name of the new appetizer.
priceStr	[String] Price entered as a string.

Exceptions

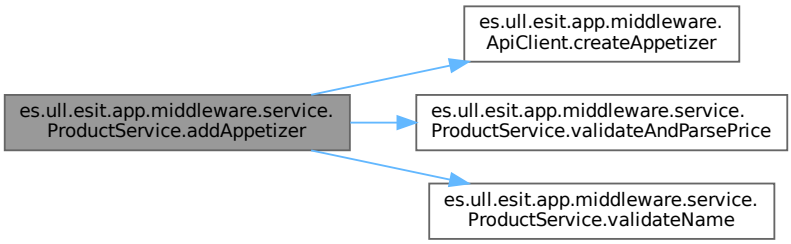
Exception	If validation fails or the backend returns an error.

Definition at line 123 of file ProductService.java.

```
00123                                                                   {
00124     int price = validateAndParsePrice(priceStr);
00125     validateName(name);
00126
00127     Appetizer newAppetizer = new Appetizer(null, name, price, null);
00128     client.createAppetizer(newAppetizer);
00129   }
```

References es.ull.esit.app.middleware.service.ProductService.client, es.ull.esit.app.middleware.ApiClient.createAppetizer(), es.ull.esit.app.middleware.service.ProductService.validateAndParsePrice(), and es.ull.esit.app.middleware.service.ProductService.va

Here is the call graph for this function:



7.32.3.2 addDrink()

```
void es.ull.esit.app.middleware.service.ProductService.addDrink (
            String name,
            String priceStr ) throws Exception  [inline]
```

Adds a new drink to the backend menu.

```
   Validates the user input and sends a POST request using the ApiClient.
```

Parameters

name	[String] Name of the new drink.
priceStr	[String] Price entered as a string.

Exceptions

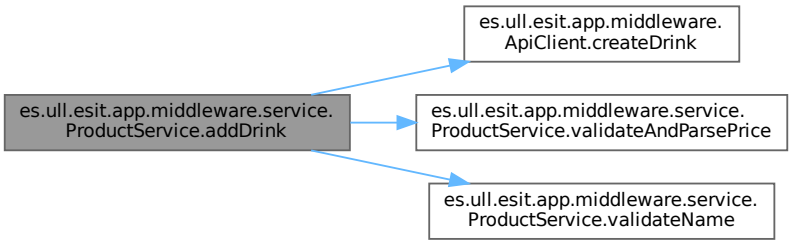
Exception	If validation fails or the backend returns an error.

Definition at line 62 of file ProductService.java.

```
00062                                                                    {
00063     int price = validateAndParsePrice(priceStr);
00064     validateName(name);
00065
00066     Drink newDrink = new Drink(null, name, price, null);
00067     client.createDrink(newDrink);
00068   }
```

References es.ull.esit.app.middleware.service.ProductService.client, es.ull.esit.app.middleware.ApiClient.createDrink(), es.ull.esit.app.middleware.service.ProductService.validateAndParsePrice(), and es.ull.esit.app.middleware.service.ProductService.va

Here is the call graph for this function:



7.32.3.3 addMainCourse()

```
void es.ull.esit.app.middleware.service.ProductService.addMainCourse (
            String name,
            String priceStr ) throws Exception  [inline]
```

Adds a new main course to the backend menu.

```
   Validates user input and sends a POST request using the ApiClient.
```

Parameters

name	[String] Name of the new main course.
priceStr	[String] Price entered as a string.

Exceptions

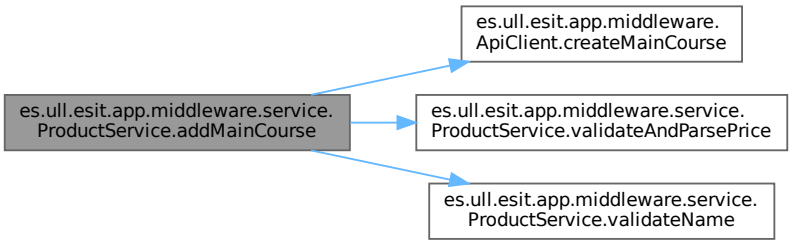
Exception	If validation fails or the backend returns an error.

Definition at line 184 of file ProductService.java.

```
00184                                                                    {
00185     int price = validateAndParsePrice(priceStr);
00186     validateName(name);
00187
00188     MainCourse newMainCourse = new MainCourse(null, name, price, null);
00189     client.createMainCourse(newMainCourse);
00190   }
```

References es.ull.esit.app.middleware.service.ProductService.client, es.ull.esit.app.middleware.ApiClient.createMainCourse(), es.ull.esit.app.middleware.service.ProductService.validateAndParsePrice(), and es.ull.esit.app.middleware.service.ProductService.va

Here is the call graph for this function:



7.32.3.4 getAllAppetizers()

```
List< Appetizer > es.ull.esit.app.middleware.service.ProductService.getAllAppetizers ( )
throws Exception  [inline]
```

Retrieves all appetizers from the backend.

Returns

[List<Appetizer>] List of all appetizers in the menu.

Exceptions

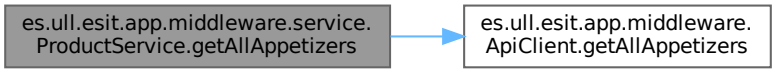
Exception	If an error occurs during the request.

Definition at line 99 of file ProductService.java.

```
00099                                                   {
00100     return client.getAllAppetizers();
00101   }
```

References es.ull.esit.app.middleware.service.ProductService.client, and es.ull.esit.app.middleware.ApiClient.getAllAppetizers().

Here is the call graph for this function:



7.32.3.5 getAllDrinks()

```
List< Drink > es.ull.esit.app.middleware.service.ProductService.getAllDrinks ( ) throws Exception
[inline]
```

Retrieves all drinks from the backend.

Returns

[List<Drink>] List of all drinks available in the menu.

Exceptions

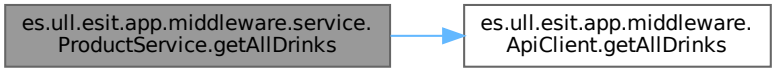
Exception	If an error occurs during the request.

Definition at line 38 of file ProductService.java.

```
00038                                                {
00039     return client.getAllDrinks();
00040   }
```

References es.ull.esit.app.middleware.service.ProductService.client, and es.ull.esit.app.middleware.ApiClient.getAllDrinks().

Here is the call graph for this function:



7.32.3.6 getAllMainCourses()

```
List< MainCourse > es.ull.esit.app.middleware.service.ProductService.getAllMainCourses ( )
throws Exception  [inline]
```

Retrieves all main courses from the backend.

Returns

[List<MainCourse>] List of all main courses in the menu.

Exceptions

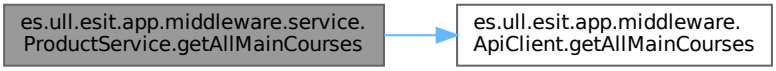
Exception	If an error occurs during the request.

Definition at line 160 of file ProductService.java.

```
00160                                                {
00161     return client.getAllMainCourses();
00162   }
```

References es.ull.esit.app.middleware.service.ProductService.client, and es.ull.esit.app.middleware.ApiClient.getAllMainCourses().

Here is the call graph for this function:



7.32.3.7 getAppetizerById()

```
Appetizer es.ull.esit.app.middleware.service.ProductService.getAppetizerById (
            Long id ) throws Exception  [inline]
```

Retrieves a single appetizer by its identifier.

Parameters

id	[Long] Unique identifier of the appetizer.

Returns

[Appetizer] Appetizer object corresponding to the given id.

Exceptions

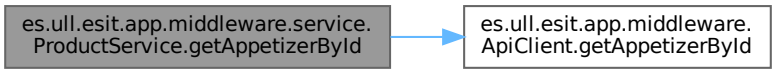
Exception	If an error occurs during the request.

Definition at line 110 of file ProductService.java.

```
00110                                                   {
00111     return client.getAppetizerById(id);
00112   }
```

References es.ull.esit.app.middleware.service.ProductService.client, and es.ull.esit.app.middleware.ApiClient.getAppetizerById().

Here is the call graph for this function:



7.32.3.8 getDrinkById()

```
Drink es.ull.esit.app.middleware.service.ProductService.getDrinkById (
            Long id ) throws Exception  [inline]
```

Retrieves a single drink by its identifier.

Parameters

id	[Long] Unique identifier of the drink.

Returns

[Drink] Drink object corresponding to the given id.

Exceptions

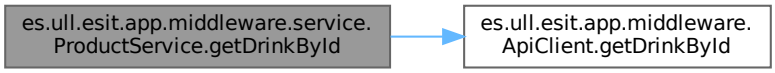
Exception	If an error occurs during the request.

Definition at line 49 of file ProductService.java.

```
00049                                                {
00050     return client.getDrinkById(id);
00051   }
```

References es.ull.esit.app.middleware.service.ProductService.client, and es.ull.esit.app.middleware.ApiClient.getDrinkById().

Here is the call graph for this function:



7.32.3.9 getMainCourseById()

```
MainCourse es.ull.esit.app.middleware.service.ProductService.getMainCourseById (
            Long id ) throws Exception  [inline]
```

Retrieves a single main course by its identifier.

Parameters

id	[Long] Unique identifier of the main course.

Returns

[MainCourse] MainCourse object corresponding to the given id.

Exceptions

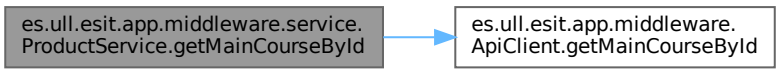
Exception	If an error occurs during the request.

Definition at line 171 of file ProductService.java.

```
00171                                                                    {
00172     return client.getMainCourseById(id);
00173   }
```

References es.ull.esit.app.middleware.service.ProductService.client, and es.ull.esit.app.middleware.ApiClient.getMainCourseById().

Here is the call graph for this function:



7.32.3.10 updateAppetizer()

```
void es.ull.esit.app.middleware.service.ProductService.updateAppetizer (
            Long id,
            String name,
            String priceStr ) throws Exception  [inline]
```

Updates an existing appetizer in the backend menu.

Parameters

id	[Long] Identifier of the appetizer to update.
name	[String] New name for the appetizer.
priceStr	[String] New price entered as a string.

Exceptions

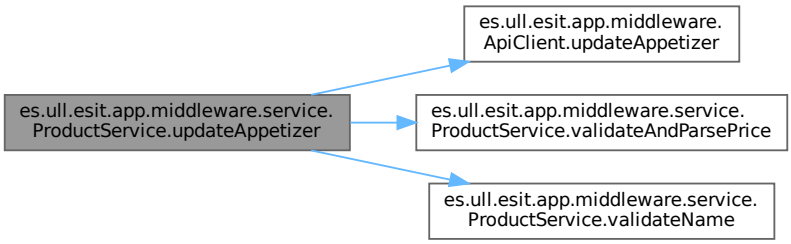
Exception	If no appetizer is selected, validation fails, or the backend returns an error.

Definition at line 140 of file ProductService.java.

```
00140                                                                                {
00141     if (id == null) {
00142       throw new IllegalArgumentException("No appetizer selected!");
00143     }
00144
00145     int price = validateAndParsePrice(priceStr);
00146     validateName(name);
00147
00148     Appetizer updatedAppetizer = new Appetizer(id, name, price, null);
00149     client.updateAppetizer(id, updatedAppetizer);
00150   }
```

References es.ull.esit.app.middleware.service.ProductService.client, es.ull.esit.app.middleware.ApiClient.updateAppetizer(), es.ull.esit.app.middleware.service.ProductService.validateAndParsePrice(), and es.ull.esit.app.middleware.service.ProductService.va

Here is the call graph for this function:



7.32.3.11 updateDrink()

```
void es.ull.esit.app.middleware.service.ProductService.updateDrink (
            Long id,
            String name,
            String priceStr ) throws Exception  [inline]
```

Updates an existing drink in the backend menu.

Parameters

id	[Long] Identifier of the drink to update.
name	[String] New name for the drink.
priceStr	[String] New price entered as a string.

Exceptions

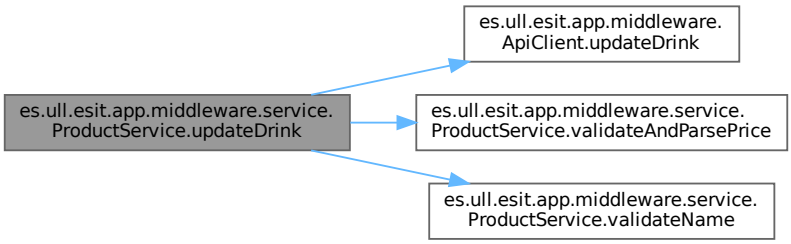
Exception	If no drink is selected, validation fails, or the backend returns an error.

Definition at line 79 of file ProductService.java.

```
00079                                                                    {
00080     if (id == null) {
00081       throw new IllegalArgumentException("No drink selected!");
00082     }
00083
00084     int price = validateAndParsePrice(priceStr);
00085     validateName(name);
00086
00087     Drink updatedDrink = new Drink(id, name, price, null);
00088     client.updateDrink(id, updatedDrink);
00089   }
```

References es.ull.esit.app.middleware.service.ProductService.client, es.ull.esit.app.middleware.ApiClient.updateDrink(), es.ull.esit.app.middleware.service.ProductService.validateAndParsePrice(), and es.ull.esit.app.middleware.service.ProductService.va

Here is the call graph for this function:



7.32.3.12 updateMainCourse()

```
void es.ull.esit.app.middleware.service.ProductService.updateMainCourse (
            Long id,
            String name,
            String priceStr ) throws Exception  [inline]
```

Updates an existing main course in the backend menu.

Parameters

id	[Long] Identifier of the main course to update.
name	[String] New name for the main course.
priceStr	[String] New price entered as a string.

Exceptions

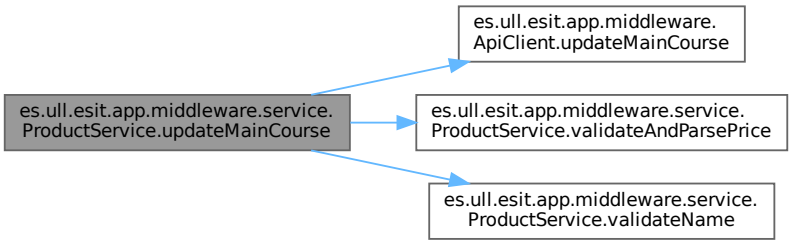
Exception	If no main course is selected, validation fails, or the backend returns an error.

Definition at line 201 of file ProductService.java.

```
00201                                                                                {
00202     if (id == null) {
00203       throw new IllegalArgumentException("No main course selected!");
00204     }
00205
00206     int price = validateAndParsePrice(priceStr);
00207     validateName(name);
00208
00209     MainCourse updatedMainCourse = new MainCourse(id, name, price, null);
00210     client.updateMainCourse(id, updatedMainCourse);
00211   }
```

References es.ull.esit.app.middleware.service.ProductService.client, es.ull.esit.app.middleware.ApiClient.updateMainCourse(), es.ull.esit.app.middleware.service.ProductService.validateAndParsePrice(), and es.ull.esit.app.middleware.service.ProductService.va

Here is the call graph for this function:



7.32.3.13 validateAndParsePrice()

```
int es.ull.esit.app.middleware.service.ProductService.validateAndParsePrice (
            String priceStr )  [inline], [private]
```

Validates and parses the price from string to integer.

```
Ensures the price is a non-negative integer value.
```

Parameters

priceStr	[String] Price entered as a string.

Returns

[int] Parsed price value.

Exceptions

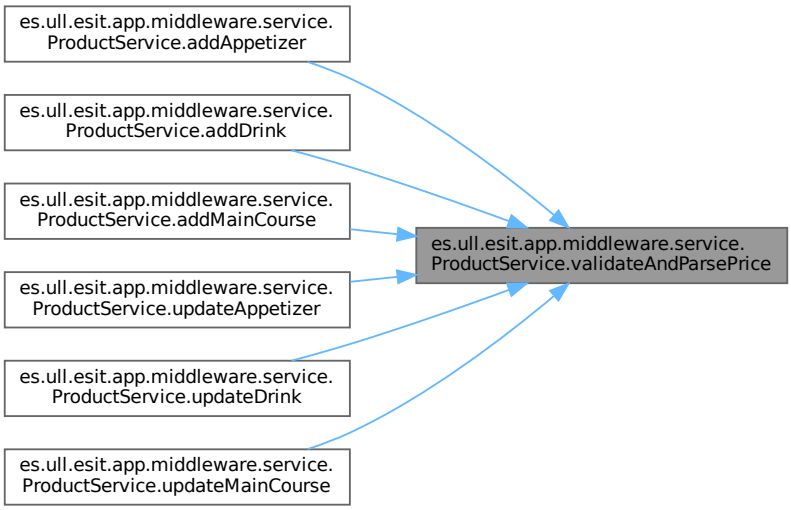
IllegalArgumentException	If the price is empty, negative or not a valid number.

Definition at line 236 of file ProductService.java.

```
00236                                                   {
00237     if (priceStr == null || priceStr.trim().isEmpty()) {
00238       throw new IllegalArgumentException("Price cannot be empty.");
00239     }
00240     try {
00241       int price = Integer.parseInt(priceStr.trim());
00242       if (price < 0) {
00243         throw new IllegalArgumentException("Price cannot be negative.");
00244       }
00245       return price;
00246     } catch (NumberFormatException e) {
00247       throw new IllegalArgumentException("Price must be a valid whole number.");
00248     }
00249   }
```

Referenced by es.ull.esit.app.middleware.service.ProductService.addAppetizer(), es.ull.esit.app.middleware.service.ProductService.a
es.ull.esit.app.middleware.service.ProductService.addMainCourse(), es.ull.esit.app.middleware.service.ProductService.updateAppeti
es.ull.esit.app.middleware.service.ProductService.updateDrink(), and es.ull.esit.app.middleware.service.ProductService.updateMainC

Here is the caller graph for this function:



7.32.3.14 validateName()

```
void es.ull.esit.app.middleware.service.ProductService.validateName (
            String name )  [inline], [private]
```

Validates that the item name is not null or empty.

Parameters

name	[String] Name of the item to validate.

Exceptions

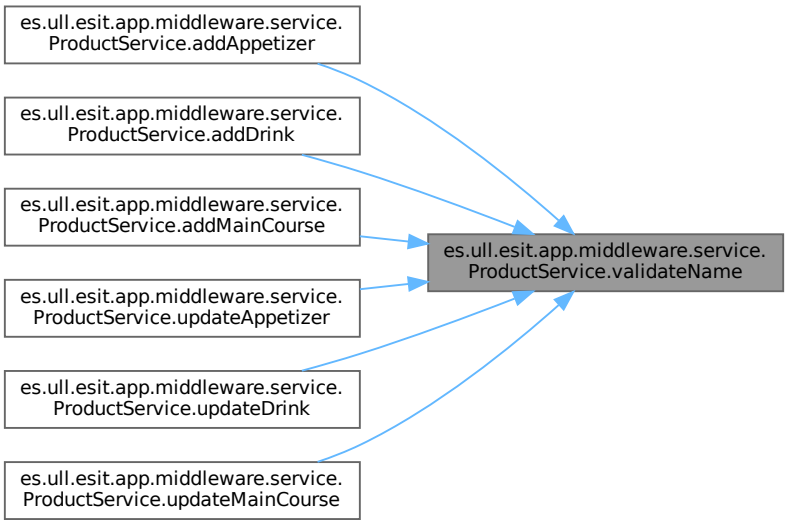
IllegalArgumentException	If the name is null or empty.

Definition at line 221 of file ProductService.java.

```
00221                                       {
00222     if (name == null || name.trim().isEmpty()) {
00223       throw new IllegalArgumentException("Item name cannot be empty.");
00224     }
00225   }
```

Referenced by es.ull.esit.app.middleware.service.ProductService.addAppetizer(), es.ull.esit.app.middleware.service.ProductService.a
es.ull.esit.app.middleware.service.ProductService.addMainCourse(), es.ull.esit.app.middleware.service.ProductService.updateAppeti
es.ull.esit.app.middleware.service.ProductService.updateDrink(), and es.ull.esit.app.middleware.service.ProductService.updateMainC

Here is the caller graph for this function:



7.32.4 Member Data Documentation

7.32.4.1 client

```
final ApiClient es.ull.esit.app.middleware.service.ProductService.client  [private]
```

REST API client used to communicate with the backend menu endpoints.

Definition at line 19 of file ProductService.java.

Referenced by es.ull.esit.app.middleware.service.ProductService.addAppetizer(), es.ull.esit.app.middleware.service.ProductService.a
es.ull.esit.app.middleware.service.ProductService.addMainCourse(), es.ull.esit.app.middleware.service.ProductService.getAllAppetiz
es.ull.esit.app.middleware.service.ProductService.getAllDrinks(), es.ull.esit.app.middleware.service.ProductService.getAllMainCourse
es.ull.esit.app.middleware.service.ProductService.getAppetizerById(), es.ull.esit.app.middleware.service.ProductService.getDrinkByI
es.ull.esit.app.middleware.service.ProductService.getMainCourseById(), es.ull.esit.app.middleware.service.ProductService.ProductS
es.ull.esit.app.middleware.service.ProductService.updateAppetizer(), es.ull.esit.app.middleware.service.ProductService.updateDrink(
and es.ull.esit.app.middleware.service.ProductService.updateMainCourse().

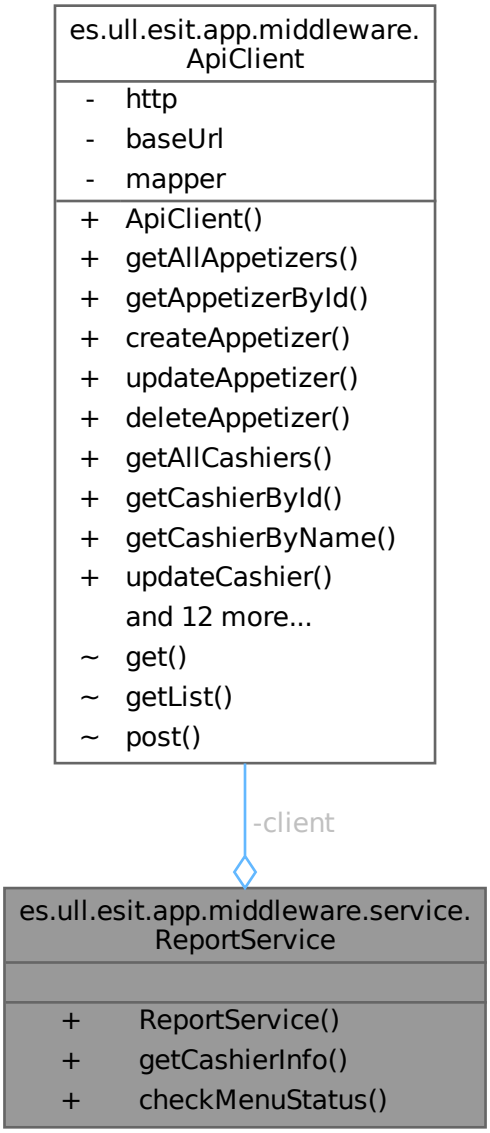
The documentation for this class was generated from the following file:

- ProductService.java

7.33 es.ull.esit.app.middleware.service.ReportService Class Reference

Service providing reporting and system status operations.

Collaboration diagram for es.ull.esit.app.middleware.service.ReportService:



Public Member Functions

- ReportService (ApiClient client)

  Constructs a ReportService with the given API client.
- List< Cashier > getCashierInfo () throws Exception

  Loads information about all registered cashiers.
- String checkMenuStatus () throws Exception

  Checks backend connectivity and counts menu items.

Private Attributes

- final ApiClient client

  REST API client used to communicate with the backend.

7.33.1 Detailed Description

Service providing reporting and system status operations.

```
It offers methods to retrieve cashier information and to verify that
the menu endpoints of the backend are accessible.
```

Definition at line 16 of file ReportService.java.

7.33.2 Constructor & Destructor Documentation

7.33.2.1 ReportService()

```
es.ull.esit.app.middleware.service.ReportService.ReportService (
            ApiClient client )  [inline]
```

Constructs a ReportService with the given API client.

Parameters

client	[ApiClient] Client used to perform HTTP requests.

Definition at line 26 of file ReportService.java.

```
00026                                       {
00027     this.client = client;
00028   }
```

References es.ull.esit.app.middleware.service.ReportService.client.

7.33.3 Member Function Documentation

7.33.3.1 checkMenuStatus()

```
String es.ull.esit.app.middleware.service.ReportService.checkMenuStatus ( ) throws Exception
[inline]
```

Checks backend connectivity and counts menu items.

```
Performs GET requests to appetizers, drinks and main courses endpoints
and returns a textual summary of the available items.
```

Returns

[String] Human-readable status message about the menu system.

Exceptions

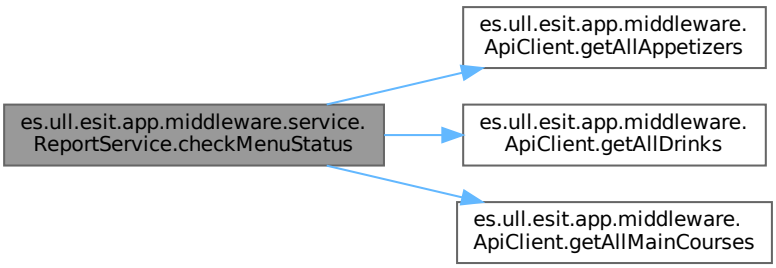
Exception	If any of the API calls fail.

Definition at line 51 of file ReportService.java.

```
00051                                       {
00052     // Fetch lists to verify connectivity.
00053     List<Appetizer> appetizers = client.getAllAppetizers();
00054     List<Drink> drinks = client.getAllDrinks();
00055     List<MainCourse> mainCourses = client.getAllMainCourses();
00056
00057     return String.format(
00058         "Menu System Online: %d appetizers, %d drinks, %d main courses available.",
00059         appetizers.size(), drinks.size(), mainCourses.size());
00060   }
```

References es.ull.esit.app.middleware.service.ReportService.client, es.ull.esit.app.middleware.ApiClient.getAllAppetizers(), es.ull.esit.app.middleware.ApiClient.getAllDrinks(), and es.ull.esit.app.middleware.ApiClient.getAllMainCourses().

Here is the call graph for this function:



7.33.3.2 getCashierInfo()

```
List< Cashier > es.ull.esit.app.middleware.service.ReportService.getCashierInfo ( ) throws
Exception  [inline]
```

Loads information about all registered cashiers.

```
Uses the "/api/cashiers" endpoint to obtain the list.
```

Returns

[List<Cashier>] List of cashiers returned by the backend.

Exceptions

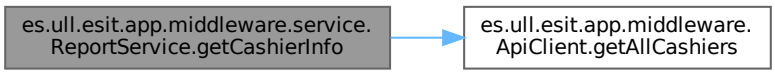
Exception	If an error occurs while contacting the backend.

Definition at line 38 of file ReportService.java.

```
00038                                                   {
00039     return client.getAllCashiers();
00040   }
```

References es.ull.esit.app.middleware.service.ReportService.client, and es.ull.esit.app.middleware.ApiClient.getAllCashiers().

Here is the call graph for this function:



7.33.4 Member Data Documentation

7.33.4.1 client

```
final ApiClient es.ull.esit.app.middleware.service.ReportService.client  [private]
```

REST API client used to communicate with the backend.

Definition at line 19 of file ReportService.java.

Referenced by es.ull.esit.app.middleware.service.ReportService.checkMenuStatus(), es.ull.esit.app.middleware.service.ReportServic
and es.ull.esit.app.middleware.service.ReportService.ReportService().

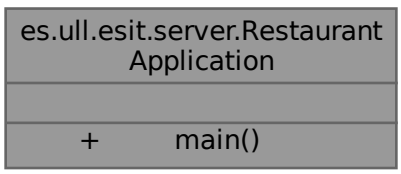
The documentation for this class was generated from the following file:

- ReportService.java

7.34 es.ull.esit.server.RestaurantApplication Class Reference

Main Spring Boot application class for the restaurant backend.

Collaboration diagram for es.ull.esit.server.RestaurantApplication:



Static Public Member Functions

- static void main (String[ ] args)

  Main entry point for the Spring Boot backend.

7.34.1 Detailed Description

Main Spring Boot application class for the restaurant backend.

```
Bootstraps the Spring Boot application.
Enables component scanning, auto-configuration and starts the embedded
web server that
exposes the REST controllers defined in the project.

The Swing frontend communicates with this backend via the HTTP
endpoints
provided by the controllers (Drinks, Appetizers, MainCourse, Menu,
etc.).
```

Definition at line 20 of file RestaurantApplication.java.

7.34.2 Member Function Documentation

7.34.2.1 main()

```
static void es.ull.esit.server.RestaurantApplication.main (
            String[] args )  [inline], [static]
```

Main entry point for the Spring Boot backend.

```
Delegates to SpringApplication.run(), which starts the application
context, configures the environment and launches the embedded server.
```

Parameters

args	[String[]] Command-line arguments (not used).

Definition at line 30 of file RestaurantApplication.java.

```
00030                                           {
00031     SpringApplication.run(RestaurantApplication.class, args);
00032   }
```

The documentation for this class was generated from the following file:

- RestaurantApplication.java

7.35 es.ull.esit.server.config.SecurityConfig Class Reference

Spring Security configuration for the backend.

Collaboration diagram for es.ull.esit.server.config.SecurityConfig:

Public Member Functions

- PasswordEncoder passwordEncoder ()
  Creates a password encoder using BCrypt hashing algorithm.
- SecurityFilterChain filterChain (HttpSecurity http) throws Exception
  Configures the HTTP security settings:

7.35.1 Detailed Description

Spring Security configuration for the backend.

Defines:

- Password encoder used for hashing passwords.
- HTTP endpoint security policies: all endpoints are currently open (no authentication required).

The login logic is handled manually in the AuthController.

Definition at line 20 of file SecurityConfig.java.

7.35.2 Member Function Documentation

7.35.2.1 filterChain()

```
SecurityFilterChain es.ull.esit.server.config.SecurityConfig.filterChain (
            HttpSecurity http ) throws Exception  [inline]
```

Configures the HTTP security settings:

- Disables CSRF protection.
- Permits unauthenticated access to specified endpoints (currently all).
- Enables HTTP Basic auth (username and password in headers).

Parameters

http	[HttpSecurity] The HttpSecurity builder to define security policies.

Returns

[SecurityFilterChain] The configured security filter chain.

Exceptions

Exception	If an error occurs during configuration.

Definition at line 45 of file SecurityConfig.java.

```
00045                                                                   {
00046     http
00047         .csrf().disable();
00048
00049     http
00050         .authorizeRequests()
00051         .anyRequest().permitAll();
00052
00053     http
00054         .httpBasic().disable()
00055         .formLogin().disable();
00056
00057     return http.build();
00058   }
```

7.35.2.2 passwordEncoder()

```
PasswordEncoder es.ull.esit.server.config.SecurityConfig.passwordEncoder ( )  [inline]
```

Creates a password encoder using BCrypt hashing algorithm.

Returns

[PasswordEncoder] The BCrypt-based password encoder.

Definition at line 28 of file SecurityConfig.java.

```
00028                                       {
00029     return new BCryptPasswordEncoder();
00030   }
```

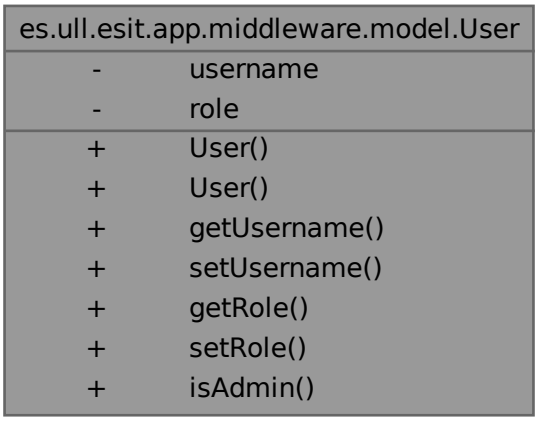
The documentation for this class was generated from the following file:

- SecurityConfig.java

7.36 es.ull.esit.app.middleware.model.User Class Reference

Client-side model representing an authenticated user.

Collaboration diagram for es.ull.esit.app.middleware.model.User:



Public Member Functions

- User ()
  Default constructor required for JSON deserialization.
- User (String username, String role)
  Constructs a user with the given username and role.
- String getUsername ()
  Gets the username.
- void setUsername (String username)
  Sets the username.
- String getRole ()
  Gets the user role.
- void setRole (String role)
  Sets the user role.
- boolean isAdmin ()
  Checks if the user has the ADMIN role.

Private Attributes

- String username
  Username of the authenticated user (JSON property "username").
- String role
  Role of the user (JSON property "role"), e.g., ADMIN or CASHIER.

7.36.1 Detailed Description

Client-side model representing an authenticated user.

```
It maps the JSON returned by the "/api/login" endpoint and contains
only the fields needed by the Swing application: username and role.
```

Definition at line 13 of file src/main/java/es/ull/esit/app/middleware/model/User.java.

7.36.2 Constructor & Destructor Documentation

7.36.2.1 User() [1/2]

```
es.ull.esit.app.middleware.model.User.User ( )  [inline]
```

Default constructor required for JSON deserialization.

Definition at line 26 of file src/main/java/es/ull/esit/app/middleware/model/User.java.

```
00026                {
00027    }
```

7.36.2.2 User() [2/2]

```
es.ull.esit.app.middleware.model.User.User (
              String username,
              String role )  [inline]
```

Constructs a user with the given username and role.

Parameters

username	[String] Username of the user.
role	[String] Role of the user.

Definition at line 35 of file src/main/java/es/ull/esit/app/middleware/model/User.java.

```
00035                                              {
00036     this.username = username;
00037     this.role = role;
00038    }
```

7.36.3 Member Function Documentation

7.36.3.1 getRole()

```
String es.ull.esit.app.middleware.model.User.getRole ( )  [inline]
```

Gets the user role.

Returns

[String] Role of the user.

Definition at line 63 of file src/main/java/es/ull/esit/app/middleware/model/User.java.

```
00063                          {
00064     return role;
00065    }
```

7.36.3.2 getUsername()

```
String es.ull.esit.app.middleware.model.User.getUsername ( )  [inline]
```

Gets the username.

Returns

[String] Username of the user.

Definition at line 45 of file src/main/java/es/ull/esit/app/middleware/model/User.java.

```
00045                                {
00046     return username;
00047   }
```

7.36.3.3 isAdmin()

```
boolean es.ull.esit.app.middleware.model.User.isAdmin ( )  [inline]
```

Checks if the user has the ADMIN role.

Returns

[boolean] True if the user role is ADMIN (case-insensitive), false otherwise.

Definition at line 81 of file src/main/java/es/ull/esit/app/middleware/model/User.java.

```
00081                          {
00082     return "ADMIN".equalsIgnoreCase(this.role);
00083   }
```

7.36.3.4 setRole()

```
void es.ull.esit.app.middleware.model.User.setRole (
            String role )  [inline]
```

Sets the user role.

Parameters

role	[String] Role of the user.

Definition at line 72 of file src/main/java/es/ull/esit/app/middleware/model/User.java.

```
00072                                  {
00073     this.role = role;
00074   }
```

7.36.3.5 setUsername()

```
void es.ull.esit.app.middleware.model.User.setUsername (
            String username )  [inline]
```

Sets the username.

Parameters

username	[String] Username of the user.

Definition at line 54 of file src/main/java/es/ull/esit/app/middleware/model/User.java.

```
00054                                           {
00055     this.username = username;
00056   }
```

7.36.4 Member Data Documentation

7.36.4.1 role

```
String es.ull.esit.app.middleware.model.User.role  [private]
```

Role of the user (JSON property "role"), e.g., ADMIN or CASHIER.

Definition at line 21 of file src/main/java/es/ull/esit/app/middleware/model/User.java.

7.36.4.2 username

```
String es.ull.esit.app.middleware.model.User.username  [private]
```

Username of the authenticated user (JSON property "username").

Definition at line 17 of file src/main/java/es/ull/esit/app/middleware/model/User.java.

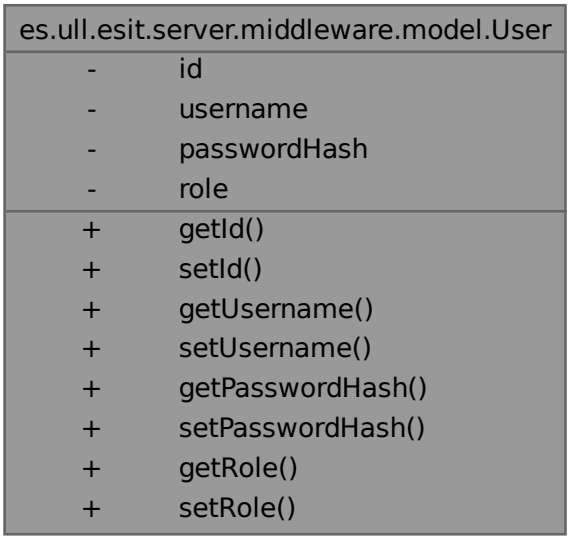
The documentation for this class was generated from the following file:

- src/main/java/es/ull/esit/app/middleware/model/User.java

7.37 es.ull.esit.server.middleware.model.User Class Reference

JPA entity that represents a user in the system.

Collaboration diagram for es.ull.esit.server.middleware.model.User:



Public Member Functions

- Long getId ()

  Gets the database identifier of the user.
- void setId (Long id)

  Sets the database identifier of the user.
- String getUsername ()

  Gets the user's username.
- void setUsername (String username)

  Sets the user's username.
- String getPasswordHash ()

  Gets the BCrypt hashed password of the user.
- void setPasswordHash (String passwordHash)

  Sets the BCrypt hashed password of the user.
- String getRole ()

  Gets the role of the user.
- void setRole (String role)

  Sets the role of the user.

Private Attributes

- Long id

  Unique identifier for the user (primary key).
- String username

  Unique username for the user.
- String passwordHash

  BCrypt hashed password for the user.
- String role

  Role of the user: ADMIN or CASHIER.

7.37.1 Detailed Description

JPA entity that represents a user in the system.

```
It is mapped to the "users" table in the database.
Stores user information such as username, password hash, and role.
```

Definition at line 22 of file server/src/main/java/es/ull/esit/server/middleware/model/User.java.

7.37.2 Member Function Documentation

7.37.2.1 getId()

```
Long es.ull.esit.server.middleware.model.User.getId ( )  [inline]
```

Gets the database identifier of the user.

Returns

[Long] The user's ID.

Definition at line 46 of file server/src/main/java/es/ull/esit/server/middleware/model/User.java.

```
00046                     {
00047         return id;
00048     }
```

7.37.2.2 getPasswordHash()

```
String es.ull.esit.server.middleware.model.User.getPasswordHash ( )  [inline]
```

Gets the BCrypt hashed password of the user.

Returns

[String] The user's BCrypt hashed password.

Definition at line 82 of file server/src/main/java/es/ull/esit/server/middleware/model/User.java.

```
00082                                 {
00083         return passwordHash;
00084     }
```

7.37.2.3 getRole()

```
String es.ull.esit.server.middleware.model.User.getRole ( )  [inline]
```

Gets the role of the user.

Returns

[String] The user's role.

Definition at line 100 of file server/src/main/java/es/ull/esit/server/middleware/model/User.java.

```
00100                           {
00101         return role;
00102     }
```

7.37.2.4 getUsername()

```
String es.ull.esit.server.middleware.model.User.getUsername ( )  [inline]
```

Gets the user's username.

Returns

[String] The user's username.

Definition at line 64 of file server/src/main/java/es/ull/esit/server/middleware/model/User.java.

```
00064                                {
00065     return username;
00066   }
```

7.37.2.5 setId()

```
void es.ull.esit.server.middleware.model.User.setId (
            Long id )  [inline]
```

Sets the database identifier of the user.

Parameters

id	[Long] The user's ID.

Definition at line 55 of file server/src/main/java/es/ull/esit/server/middleware/model/User.java.

```
00055                               {
00056     this.id = id;
00057   }
```

7.37.2.6 setPasswordHash()

```
void es.ull.esit.server.middleware.model.User.setPasswordHash (
            String passwordHash )  [inline]
```

Sets the BCrypt hashed password of the user.

Parameters

passwordHash	[String] The user's BCrypt hashed password.

Definition at line 91 of file server/src/main/java/es/ull/esit/server/middleware/model/User.java.

```
00091                                                {
00092     this.passwordHash = passwordHash;
00093   }
```

7.37.2.7 setRole()

```
void es.ull.esit.server.middleware.model.User.setRole (
            String role )  [inline]
```

Sets the role of the user.

Parameters

role	[String] The user's role.

Definition at line 109 of file server/src/main/java/es/ull/esit/server/middleware/model/User.java.

```
00109                                   {
00110     this.role = role;
00111   }
```

7.37.2.8 setUsername()

```
void es.ull.esit.server.middleware.model.User.setUsername (
             String username )  [inline]
```

Sets the user's username.

Parameters

username	[String] The user's username.

Definition at line 73 of file server/src/main/java/es/ull/esit/server/middleware/model/User.java.

```
00073                                       {
00074     this.username = username;
00075   }
```

7.37.3 Member Data Documentation

7.37.3.1 id

```
Long es.ull.esit.server.middleware.model.User.id  [private]
```

Unique identifier for the user (primary key).

Definition at line 27 of file server/src/main/java/es/ull/esit/server/middleware/model/User.java.

7.37.3.2 passwordHash

```
String es.ull.esit.server.middleware.model.User.passwordHash  [private]
```

BCrypt hashed password for the user.

Definition at line 36 of file server/src/main/java/es/ull/esit/server/middleware/model/User.java.

7.37.3.3 role

```
String es.ull.esit.server.middleware.model.User.role  [private]
```

Role of the user: ADMIN or CASHIER.

Definition at line 39 of file server/src/main/java/es/ull/esit/server/middleware/model/User.java.

7.37.3.4 username

```
String es.ull.esit.server.middleware.model.User.username  [private]
```

Unique username for the user.

Definition at line 31 of file server/src/main/java/es/ull/esit/server/middleware/model/User.java.

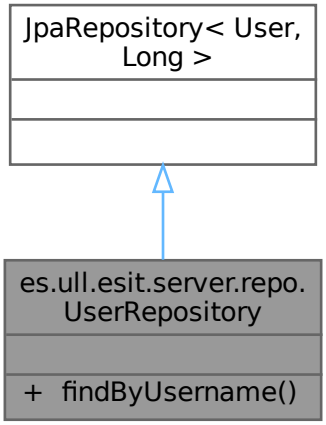
The documentation for this class was generated from the following file:

- server/src/main/java/es/ull/esit/server/middleware/model/User.java

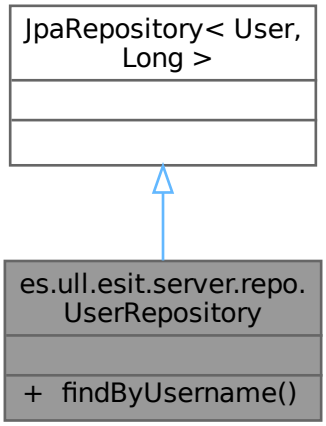
7.38 es.ull.esit.server.repo.UserRepository Interface Reference

Repository interface for accessing users in the database.

Inheritance diagram for es.ull.esit.server.repo.UserRepository:



Collaboration diagram for es.ull.esit.server.repo.UserRepository:



Public Member Functions

- Optional< User > findByUsername (String username)

  Finds a user by their unique username.

7.38.1 Detailed Description

Repository interface for accessing users in the database.

Extends JpaRepository to provide standard CRUD operations on the "users" table and defines an extra method to find a user by username.

Definition at line 13 of file UserRepository.java.

7.38.2 Member Function Documentation

7.38.2.1 findByUsername()

```
Optional< User > es.ull.esit.server.repo.UserRepository.findByUsername (
            String username )
```

Finds a user by their unique username.

Parameters

username	[String] The unique username of the user to find.

Returns

[Optional<User>] An Optional containing the User if found, or empty if not found.

The documentation for this interface was generated from the following file:

- UserRepository.java

Chapter 8

File Documentation

8.1 00-create-db.sql File Reference

8.2 00-create-db.sql

Go to the documentation of this file.

```
00001 -- 00-create-db.sql
00002 -- Crea la base de datos principal del proyecto.
00003
00004 CREATE DATABASE IF NOT EXISTS project3
00005   CHARACTER SET utf8mb4
00006   COLLATE utf8mb4_unicode_ci;
00007
00008 USE project3;
```

8.3 01-tables.sql File Reference

8.4 01-tables.sql

Go to the documentation of this file.

```
00001 -- 01-tables.sql
00002 -- Define todas las tablas de la base de datos.
00003
00004 USE project3;
00005
00006 -- ======================================
00007 -- USUARIOS DEL SISTEMA (para la API)
00008 -- ======================================
00009 CREATE TABLE IF NOT EXISTS users (
00010     id            BIGINT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
00011     username      VARCHAR(255) NOT NULL UNIQUE,
00012     password_hash VARCHAR(255) NOT NULL,
00013     role          VARCHAR(50)  NOT NULL
00014 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
00015
00016 -- ======================================
00017 -- MANAGER
00018 -- ======================================
00019 CREATE TABLE IF NOT EXISTS manager (
00020     manager_id     BIGINT UNSIGNED PRIMARY KEY,
00021     manager_fname  VARCHAR(30),
00022     manager_mname  VARCHAR(30),
00023     manager_lname  VARCHAR(30),
00024     manager_number BIGINT,
00025     manager_salary INT
00026 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
00027
```

```
00028 -- ====================================
00029 -- CASHIER (vinculado a users.id)
00030 -- ====================================
00031 CREATE TABLE IF NOT EXISTS cashier (
00032     cashier_id     BIGINT UNSIGNED PRIMARY KEY,
00033     cashier_name   VARCHAR(30),
00034     cashier_salary INT,
00035     CONSTRAINT fk_cashier_user
00036       FOREIGN KEY (cashier_id) REFERENCES users(id)
00037 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
00038
00039 -- ====================================
00040 -- CHEF
00041 -- ====================================
00042 CREATE TABLE IF NOT EXISTS chef (
00043     chef_id      BIGINT UNSIGNED PRIMARY KEY,
00044     chef_name    VARCHAR(30),
00045     chef_manager VARCHAR(30),
00046     chef_salary  INT,
00047     manager_id   BIGINT UNSIGNED,
00048     CONSTRAINT fk_chef_manager
00049       FOREIGN KEY (manager_id) REFERENCES manager (manager_id)
00050 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
00051
00052 -- ====================================
00053 -- RECEIPT
00054 -- ====================================
00055 CREATE TABLE IF NOT EXISTS receipt (
00056     receipt_id    BIGINT UNSIGNED PRIMARY KEY,
00057     receipt_time  TIME,
00058     receipt_date  DATE,
00059     receipt_total INT
00060 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
00061
00062 -- ====================================
00063 -- RESTAURANT MANAGEMENT
00064 -- ====================================
00065 CREATE TABLE IF NOT EXISTS restaurant_management (
00066     restaurant_id      BIGINT UNSIGNED PRIMARY KEY,
00067     restaurant_name    VARCHAR(50),
00068     restaurant_address VARCHAR(100),
00069     manager_id         BIGINT UNSIGNED,
00070     CONSTRAINT fk_restaurant_manager
00071       FOREIGN KEY (manager_id) REFERENCES manager (manager_id)
00072 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
00073
00074 -- ====================================
00075 -- DEPENDENTS (MANAGER)
00076 -- ====================================
00077 CREATE TABLE IF NOT EXISTS dependent_manager (
00078     dependent_name     VARCHAR(30) PRIMARY KEY,
00079     dependent_relation VARCHAR(30),
00080     dependent_sex      ENUM('M','F'),
00081     manager_id         BIGINT UNSIGNED,
00082     CONSTRAINT fk_dep_manager
00083       FOREIGN KEY (manager_id) REFERENCES manager (manager_id)
00084 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
00085
00086 -- ====================================
00087 -- DEPENDENTS (CHEF)
00088 -- ====================================
00089 CREATE TABLE IF NOT EXISTS dependent_chef (
00090     dependent_name     VARCHAR(30) PRIMARY KEY,
00091     dependent_relation VARCHAR(30),
00092     dependent_sex      ENUM('M','F'),
00093     chef_id            BIGINT UNSIGNED,
00094     CONSTRAINT fk_dep_chef
00095       FOREIGN KEY (chef_id) REFERENCES chef (chef_id)
00096 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
00097
00098 -- ====================================
00099 -- DEPENDENTS (CASHIER)
00100 -- ====================================
00101 CREATE TABLE IF NOT EXISTS dependent_cashier (
00102     dependent_name     VARCHAR(30) PRIMARY KEY,
00103     dependent_relation VARCHAR(30),
00104     dependent_sex      ENUM('M','F'),
00105     cashier_id         BIGINT UNSIGNED,
00106     CONSTRAINT fk_dep_cashier
00107       FOREIGN KEY (cashier_id) REFERENCES cashier (cashier_id)
00108 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
00109
00110 -- ====================================
00111 -- ITEM (productos combinados en un ticket)
00112 -- ====================================
00113 CREATE TABLE IF NOT EXISTS item (
00114     item_food         VARCHAR(30),
```

```
00115     food_price        INT,
00116     food_id           INT,
00117
00118     item_appetizers   VARCHAR(30),
00119     appetizers_price  INT,
00120     appetizers_id     INT,
00121
00122     item_drinks       VARCHAR(30),
00123     drinks_price      INT,
00124     drinks_id         INT,
00125
00126     receipt_id        BIGINT UNSIGNED,
00127
00128     PRIMARY KEY (food_id, appetizers_id, drinks_id),
00129     CONSTRAINT fk_item_receipt
00130       FOREIGN KEY (receipt_id) REFERENCES receipt (receipt_id)
00131 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
00132
00133 CREATE INDEX item_index ON item (item_food);
00134
00135 -- =====================================
00136 -- CHEF_PREPARE_ITEM
00137 -- =====================================
00138 CREATE TABLE IF NOT EXISTS chef_prepare_item (
00139     chef_id   BIGINT UNSIGNED,
00140     food_id   INT,
00141     sweet_id  INT,
00142     drinks_id INT,
00143     CONSTRAINT fk_cpi_chef
00144       FOREIGN KEY (chef_id) REFERENCES chef (chef_id)
00145 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
00146
00147 -- =====================================
00148 -- RECEIPT_TAKEN_BY_CASHIER
00149 -- =====================================
00150 CREATE TABLE IF NOT EXISTS receipt_taken_by_cashier (
00151     receipt_id BIGINT UNSIGNED,
00152     cashier_id BIGINT UNSIGNED,
00153     CONSTRAINT fk_rtc_receipt
00154       FOREIGN KEY (receipt_id) REFERENCES receipt (receipt_id),
00155     CONSTRAINT fk_rtc_cashier
00156       FOREIGN KEY (cashier_id) REFERENCES cashier (cashier_id)
00157 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
00158
00159 -- =====================================
00160 -- RESTAURANT_ADDRESS
00161 -- =====================================
00162 CREATE TABLE IF NOT EXISTS restaurant_address (
00163     restaurant_address VARCHAR(100) PRIMARY KEY,
00164     restaurant_id      BIGINT UNSIGNED,
00165     CONSTRAINT fk_restaurant_addr
00166       FOREIGN KEY (restaurant_id) REFERENCES restaurant_management (restaurant_id)
00167 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
00168
00169 -- =====================================
00170 -- TABLAS DE PRODUCTOS PARA LA API
00171 -- =====================================
00172 CREATE TABLE IF NOT EXISTS appetizers (
00173   id    INT NOT NULL AUTO_INCREMENT,
00174   name  VARCHAR(250) NOT NULL,
00175   price INT NOT NULL,
00176   PRIMARY KEY (id)
00177 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
00178
00179 CREATE TABLE IF NOT EXISTS drinks (
00180   id    INT NOT NULL AUTO_INCREMENT,
00181   name  VARCHAR(250) NOT NULL,
00182   price INT NOT NULL,
00183   PRIMARY KEY (id)
00184 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
00185
00186 CREATE TABLE IF NOT EXISTS maincourse (
00187   id    INT NOT NULL AUTO_INCREMENT,
00188   name  VARCHAR(250) NOT NULL,
00189   price INT NOT NULL,
00190   PRIMARY KEY (id)
00191 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
00192
00193 -- =====================================
00194 -- TABLA DE PRUEBAS: receipt_copy
00195 -- =====================================
00196 CREATE TABLE IF NOT EXISTS receipt_copy (
00197     receipt_id    BIGINT UNSIGNED PRIMARY KEY,
00198     receipt_time  TIME,
00199     receipt_date  DATE,
00200     receipt_total INT
00201 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

8.5 02-procedures.sql File Reference

8.6 02-procedures.sql

Go to the documentation of this file.

```
00001 -- 02-procedures.sql
00002 -- Define procedimientos almacenados de la base de datos.
00003
00004 USE project3;
00005
00006 DROP PROCEDURE IF EXISTS chefname;
00007
00008 DELIMITER $$
00009
00010 CREATE PROCEDURE chefname (IN chef_name_param VARCHAR(10))
00011 BEGIN
00012     SELECT *
00013     FROM chef
00014     WHERE chef_name = chef_name_param;
00015 END $$
00016
00017 DELIMITER ;
00018
```

8.7 03-triggers.sql File Reference

8.8 03-triggers.sql

Go to the documentation of this file.

```
00001 -- 03-triggers.sql
00002 -- Añade triggers a la base de datos.
00003
00004 USE project3;
00005
00006 -- ====================================
00007 -- Trigger: nombres de manager en mayúsculas
00008 -- ====================================
00009 DROP TRIGGER IF EXISTS trg_uppercase_manager_fname;
00010
00011 DELIMITER $$
00012
00013 CREATE TRIGGER trg_uppercase_manager_fname
00014 BEFORE INSERT ON manager
00015 FOR EACH ROW
00016 BEGIN
00017   SET NEW.manager_fname = UPPER(NEW.manager_fname);
00018 END $$
00019
00020 DELIMITER ;
00021
00022 -- ====================================
00023 -- Trigger: crear cashier automáticamente al insertar un usuario CASHIER
00024 -- Asigna salario base = 1500 €
00025 -- ====================================
00026 DROP TRIGGER IF EXISTS trg_create_cashier_from_user;
00027
00028 DELIMITER $$
00029
00030 CREATE TRIGGER trg_create_cashier_from_user
00031 AFTER INSERT ON users
00032 FOR EACH ROW
00033 BEGIN
00034   IF NEW.role = 'CASHIER' THEN
00035     INSERT INTO cashier (cashier_id, cashier_name, cashier_salary)
00036     VALUES (NEW.id, NEW.username, 1500);
00037   END IF;
00038 END $$
00039
00040 DELIMITER ;
00041
```

8.9 04-privileges.sql File Reference

8.10 04-privileges.sql

Go to the documentation of this file.

```
00001 -- 04-privileges.sql
00002 -- Asigna permisos al usuario usado por Spring Boot.
00003
00004 USE project3;
00005
00006 GRANT ALL PRIVILEGES ON project3.* TO 'restaurant'@'%';
00007 FLUSH PRIVILEGES;
```

8.11 05-data.sql File Reference

8.12 05-data.sql

Go to the documentation of this file.

```
00001 -- 05-data.sql
00002 -- Inserción de datos realistas para el restaurante "Black Plate"
00003
00004 USE project3;
00005
00006 -- =====================================================================
00007 -- MANAGER
00008 -- =====================================================================
00009 INSERT INTO manager (manager_id, manager_fname, manager_mname, manager_lname, manager_number,
      manager_salary)
00010 VALUES
00011 (1001, 'marta', 'elena', 'rodriguez', 659123456, 3200);
00012
00013 -- =====================================================================
00014 -- RESTAURANT
00015 -- =====================================================================
00016 INSERT INTO restaurant_management (restaurant_id, restaurant_name, restaurant_address, manager_id)
00017 VALUES
00018 (2001, 'black plate', 'Calle Luna 45, Madrid', 1001);
00019
00020 -- =====================================================================
00021 -- CHEFS DE BLACK PLATE
00022 -- =====================================================================
00023 INSERT INTO chef (chef_id, chef_name, chef_manager, chef_salary, manager_id)
00024 VALUES
00025 (3001, 'Luis Gómez', 'Marta', 1800, 1001),
00026 (3002, 'Carmen Ruiz', 'Marta', 1750, 1001),
00027 (3003, 'Javier Soto', 'Marta', 1700, 1001);
00028
00029 -- =====================================================================
00030 -- RECEIPTS (Tickets reales)
00031 -- =====================================================================
00032 INSERT INTO receipt (receipt_id, receipt_time, receipt_date, receipt_total)
00033 VALUES
00034 (1, '13:15:20', '2024-10-01', 42),
00035 (2, '14:22:10', '2024-10-01', 27),
00036 (3, '20:45:50', '2024-10-02', 58),
00037 (4, '21:10:35', '2024-10-03', 73),
00038 (5, '12:30:10', '2024-10-04', 19);
00039
00040 -- =====================================================================
00041 -- DEPENDENTS DEL MANAGER
00042 -- =====================================================================
00043 INSERT INTO dependent_manager (dependent_name, dependent_relation, dependent_sex, manager_id)
00044 VALUES
00045 ('lucia', 'daughter', 'F', 1001),
00046 ('pablo', 'son',      'M', 1001);
00047
00048 -- =====================================================================
00049 -- DEPENDENTS DE CHEF
00050 -- =====================================================================
00051 INSERT INTO dependent_chef (dependent_name, dependent_relation, dependent_sex, chef_id)
00052 VALUES
00053 ('marcos', 'son',  'M', 3001),
00054 ('irene',  'wife', 'F', 3002);
```

```
00055
00056 -- ========================================================================
00057 -- USUARIOS DEL SISTEMA
00058 -- El trigger creará automáticamente los CASHIERS en la tabla cashier
00059 -- ========================================================================
00060
00061 -- ADMIN (NO genera cashier)
00062 INSERT INTO users (id, username, password_hash, role)
00063 VALUES (1, 'admin', '$2b$10$TAxhf.ziwuv7ynXlSQYjMeftWo47ft8M4JfdXjLoBPdERwuo7zD06', 'ADMIN');
00064
00065 -- CAJEROS (generan automáticamente filas en cashier gracias al trigger)
00066 -- password: cashier123 (hash reutilizado)
00067 INSERT INTO users (id, username, password_hash, role)
00068 VALUES
00069 (2, 'laura.mendez', '$2b$10$HLeNcr00JL6qpGJUd.klqud4PfSRci2qvuKr7fnklGjtMOO6OhwLC', 'CASHIER'),
00070 (3, 'diego.santos', '$2b$10$HLeNcr00JL6qpGJUd.klqud4PfSRci2qvuKr7fnklGjtMOO6OhwLC', 'CASHIER');
00071
00072 -- En este punto, gracias al trigger:
00073 --   cashier tendrá:
00074 --   (2, 'laura.mendez', 1500) y (3, 'diego.santos', 1500)
00075
00076 -- ========================================================================
00077 -- DEPENDENTS DE LOS CASHIERS
00078 -- ========================================================================
00079 INSERT INTO dependent_cashier (dependent_name, dependent_relation, dependent_sex, cashier_id)
00080 VALUES
00081 ('emma', 'daughter', 'F', 2),
00082 ('alex', 'son',      'M', 3);
00083
00084 -- ========================================================================
00085 -- ITEMS DEL TICKET (combinaciones reales del menú Black Plate)
00086 -- ========================================================================
00087 INSERT INTO item (item_food, food_price, food_id,
00088                   item_appetizers, appetizers_price, appetizers_id,
00089                   item_drinks, drinks_price, drinks_id,
00090                   receipt_id)
00091 VALUES
00092 ('Grilled Salmon',      18, 1, 'Truffle Fries',      6, 1, 'Coca Cola',       2, 1, 1),
00093 ('Black Angus Burger',  16, 2, 'Garlic Bread',       4, 2, 'Sparkling Water', 2, 2, 2),
00094 ('Chicken Teriyaki',    14, 3, 'Hummus Bowl',        5, 3, 'Iced Tea',        3, 3, 3),
00095 ('Vegan Buddha Bowl',   13, 4, 'Sweet Potato Chips', 4, 4, 'Lemonade',        3, 4, 4),
00096 ('Pasta Alfredo',       11, 5, 'Bruschetta',         4, 5, 'Coca Cola',       2, 1, 5);
00097
00098 -- ========================================================================
00099 -- CHEF PREPARA PLATOS
00100 -- ========================================================================
00101 INSERT INTO chef_prepare_item (chef_id, food_id, sweet_id, drinks_id)
00102 VALUES
00103 (3001, 1, 1, 1),
00104 (3002, 2, 2, 2),
00105 (3003, 3, 3, 3);
00106
00107 -- ========================================================================
00108 -- RECEIPTS TOMADOS POR CASHIERS (usa IDs 2 y 3)
00109 -- ========================================================================
00110 INSERT INTO receipt_taken_by_cashier (receipt_id, cashier_id)
00111 VALUES
00112 (1, 2),
00113 (2, 2),
00114 (3, 3),
00115 (4, 3),
00116 (5, 2);
00117
00118 -- ========================================================================
00119 -- Tabla receipt_copy (pruebas)
00120 -- ========================================================================
00121 INSERT INTO receipt_copy (receipt_id, receipt_time, receipt_date, receipt_total)
00122 VALUES
00123 (1, '12:45:51', '2024-10-01', 42);
00124
00125 DELETE FROM receipt_copy WHERE receipt_id = 1;
00126
00127 -- ========================================================================
00128 -- Productos para la API (Black Plate)
00129 -- ========================================================================
00130 INSERT INTO appetizers (name, price) VALUES
00131 ('Truffle Fries',      6),
00132 ('Garlic Bread',       4),
00133 ('Hummus Bowl',        5),
00134 ('Sweet Potato Chips', 4),
00135 ('Bruschetta',         4);
00136
00137 INSERT INTO drinks (name, price) VALUES
00138 ('Coca Cola',       2),
00139 ('Sparkling Water', 2),
00140 ('Iced Tea',        3),
00141 ('Lemonade',        3),
```

```
00142 ('Red Fruit Juice', 4);
00143
00144 INSERT INTO maincourse (name, price) VALUES
00145 ('Grilled Salmon',        18),
00146 ('Black Angus Burger',    16),
00147 ('Chicken Teriyaki',      14),
00148 ('Vegan Buddha Bowl',     13),
00149 ('Pasta Alfredo',         11);
```

8.13 init.sql File Reference

8.14 init.sql

Go to the documentation of this file.

```
00001 /**
00002  * init.sql
00003  * Script de inicialización de la base de datos project3 para el contenedor Docker MySQL.
00004  */
00005
00006 -- Desactivar restricciones de claves foráneas.
00007 SET FOREIGN_KEY_CHECKS = 0;
00008
00009 -- Crear BD si no existe.
00010 CREATE DATABASE IF NOT EXISTS project3;
00011 USE project3;
00012
00013 ----- CREACIÓN DE TABLAS -----
00014 -- Tabla 1. RestaurantManagement.
00015 CREATE TABLE IF NOT EXISTS RestaurantManagement (
00016     Restaurant_id      NUMERIC(10) PRIMARY KEY,
00017     Restaurant_name    VARCHAR(30),
00018     Restaurant_address VARCHAR(30),
00019     Manager_id         NUMERIC(10)
00020         REFERENCES Manager (Manager_id)
00021 );
00022
00023 -- Tabla 2. Manager.
00024 CREATE TABLE IF NOT EXISTS Manager (
00025     Manager_id     NUMERIC(10) PRIMARY KEY,
00026     Manager_Fname  VARCHAR(30),
00027     Manager_Mname  VARCHAR(30),
00028     Manager_Lname  VARCHAR(30),
00029     Manager_number NUMERIC(10),
00030     Manager_salary NUMERIC(5)
00031 );
00032
00033 -- Tabla 3. Item.
00034 CREATE TABLE IF NOT EXISTS Item (
00035     Item_food         VARCHAR(30),
00036     food_price        NUMERIC(4),
00037     food_id           NUMERIC(5),
00038
00039     Item_appetizers   VARCHAR(30),
00040     appetizers_price  NUMERIC(4),
00041     appetizers_id     NUMERIC(5),
00042
00043     Item_drinks       VARCHAR(30),
00044     drinks_price      NUMERIC(4),
00045     drinks_id         NUMERIC(5),
00046
00047     Receipt_id NUMERIC(10) REFERENCES Receipt (Receipt_id),
00048
00049     PRIMARY KEY (food_id, appetizers_id, drinks_id)
00050 );
00051
00052 -- Tabla 4. Chef.
00053 CREATE TABLE IF NOT EXISTS Chef (
00054     Chef_id      NUMERIC(10) PRIMARY KEY,
00055     Chef_name    VARCHAR(30),
00056     Chef_manager VARCHAR(30),
00057     Chef_salary  NUMERIC(5),
00058     Manager_id   NUMERIC(10) REFERENCES Manager (Manager_id)
00059 );
00060
00061 -- Tabla 5. Cashier.
00062 CREATE TABLE IF NOT EXISTS Cashier (
00063     Cashier_id     NUMERIC(10) PRIMARY KEY,
00064     Cashier_name   VARCHAR(30),
```

```
00065     Cashier_salary NUMERIC(5)
00066 );
00067
00068 -- Tabla 6. Receipt.
00069 CREATE TABLE IF NOT EXISTS Receipt (
00070     Receipt_id    NUMERIC(10) PRIMARY KEY,
00071     Receipt_time  TIME,
00072     Receipt_date  DATE,
00073     Receipt_total NUMERIC(10)
00074 );
00075
00076 -- Tabla 7. Dependent_Manager.
00077 CREATE TABLE IF NOT EXISTS Dependent_Manager (
00078     Dependent_name     VARCHAR(30) PRIMARY KEY,
00079     Dependent_relation VARCHAR(30),
00080     Dependent_sex      ENUM('M','F'),
00081     Manager_id         NUMERIC(10) REFERENCES Manager (Manager_id)
00082 );
00083
00084 -- Tabla 8. Dependent_chef.
00085 CREATE TABLE IF NOT EXISTS Dependent_chef (
00086     Dependent_name     VARCHAR(30) PRIMARY KEY,
00087     Dependent_relation VARCHAR(30),
00088     Dependent_sex      ENUM('M','F'),
00089     Chef_id            NUMERIC(10) REFERENCES Chef (Chef_id)
00090 );
00091
00092 -- Tabla 9. Dependent_Cashier.
00093 CREATE TABLE IF NOT EXISTS Dependent_Cashier (
00094     Dependent_name     VARCHAR(30) PRIMARY KEY,
00095     Dependent_relation VARCHAR(30),
00096     Dependent_sex      ENUM('M','F'),
00097     Cashier_id         NUMERIC(10) REFERENCES Cashier (Cashier_id)
00098 );
00099
00100 -- Tabla 10. Restaurant_address.
00101 CREATE TABLE IF NOT EXISTS Restaurant_address (
00102     Restaurant_address VARCHAR(30) PRIMARY KEY,
00103     Restaurant_id      NUMERIC(10) REFERENCES RestaurantManagement (Restaurant_id)
00104 );
00105
00106 -- Tabla 11. Chef_Prepare_item.
00107 CREATE TABLE IF NOT EXISTS Chef_Prepare_item (
00108     Chef_id   NUMERIC(10) REFERENCES Chef (Chef_id),
00109     food_id   NUMERIC(5) REFERENCES Item (food_id),
00110     sweet_id  NUMERIC(5),
00111     drinks_id NUMERIC(5)
00112 );
00113
00114 -- Tabla 12. Receipt_takenBy_Cashier.
00115 CREATE TABLE IF NOT EXISTS Receipt_takenBy_Cashier (
00116     Receipt_id NUMERIC(10) REFERENCES Receipt (Receipt_id),
00117     Cashier_id NUMERIC(10) REFERENCES Cashier (Cashier_id)
00118 );
00119
00120 ----- INSERCIÓN DE DATOS -----
00121 -- Restaurant --
00122 INSERT INTO RestaurantManagement VALUES (56471,'project_resturant','khobar',3214);
00123
00124 -- Manager --
00125 INSERT INTO Manager VALUES
00126 (3214,'ahmed','khalid','alfahad',053276488,7000);
00127
00128 -- Chefs --
00129 INSERT INTO Chef VALUES
00130 (2561,'abduallah','ahmed',6000,3214),
00131 (2562,'saleh','ahmed',6000,3214),
00132 (2563,'mohammad','ahmed',6000,3214),
00133 (2564,'khalid','ahmed',6000,3214);
00134
00135 -- Cashiers --
00136 INSERT INTO Cashier VALUES
00137 (4231,'abdualmajeed',7000),
00138 (4232,'abdualrahman',7000);
00139
00140 -- Receipts --
00141 INSERT INTO Receipt VALUES
00142 (1,'12:45:56','2022-01-23',300),
00143 (2,'12:44:32','2022-01-23',200),
00144 (3,'01:30:50','2022-01-24',300),
00145 (4,'03:30:55','2022-01-24',360),
00146 (5,'03:00:50','2022-01-25',400),
00147 (6,'02:00:00','2022-01-25',200),
00148 (7,'03:00:00','2022-01-26',150);
00149
00150 -- Dependents (Manager) --
00151 INSERT INTO Dependent_Manager VALUES
```

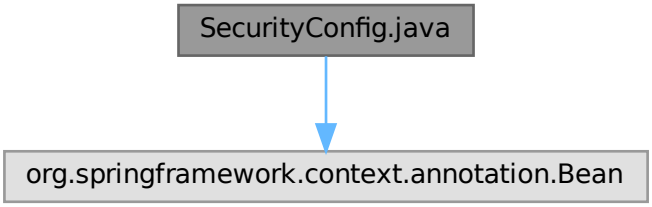
```
00152 ('maram','wife','F',3214),
00153 ('marwa','child','F',3214);
00154
00155 -- Dependents (Chef) --
00156 INSERT INTO Dependent_chef VALUES
00157 ('saad','son','M',2562),
00158 ('sara','wife','F',2563),
00159 ('norah','wife','F',2561);
00160
00161 -- Dependents (Cashier) --
00162 INSERT INTO Dependent_Cashier VALUES
00163 ('sara','daughter','F',4231),
00164 ('lama','daughter','F',4232);
00165
00166 -- Items --
00167 INSERT INTO Item VALUES
00168 ('buratta pizza',56,1,'Dynamite shrimp',39,1,'cola',5,1,1),
00169 ('pink pasta',37,2,'mac&cheese balls',45,2,'7up',5,2,2),
00170 ('spaghetti',40,3,'tiramisu',42,3,'orang juice',15,3,3),
00171 ('rosemary salmon',87,4,'molten chocolate',19,4,'mojito',25,4,4);
00172
00173 -- Chef Prepare Item --
00174 INSERT INTO Chef_Prepare_item VALUES
00175 (2561,1,1,1),
00176 (2562,2,2,2),
00177 (2563,3,3,3),
00178 (2564,4,4,4);
00179
00180 -- Receipt taken by cashier --
00181 INSERT INTO Receipt_takenBy_Cashier VALUES
00182 (1,1),(2,1),(3,1),(4,1),(5,2),(6,2),(7,2);
00183
00184 -- CONSULTAS, PRUEBAS Y PROCEDIMIENTOS --
00185 -- SELECTs de prueba --
00186 SELECT DISTINCT Receipt_total FROM Receipt;
00187
00188 SELECT AVG(Receipt_total) AS avg_total FROM Receipt;
00189 SELECT MAX(Receipt_total) AS max_total FROM Receipt;
00190 SELECT MIN(Receipt_total) AS min_total FROM Receipt;
00191
00192 SELECT * FROM Chef WHERE NOT Chef_name='khalid';
00193 SELECT * FROM Chef WHERE Chef_name LIKE 'k%';
00194
00195 SELECT * FROM Item ORDER BY Item_food DESC;
00196 SELECT * FROM Item ORDER BY Item_food ASC;
00197
00198 SELECT COUNT(Chef_id), Chef_name FROM Chef GROUP BY Chef_name HAVING COUNT(Chef_id) > 2561;
00199
00200 SELECT * FROM Receipt WHERE Receipt_date IN ('2022-01-24','2022-01-25');
00201 SELECT * FROM Receipt WHERE Receipt_total BETWEEN 300 AND 400;
00202
00203 -- CASE example --
00204 INSERT INTO Receipt VALUES (10,'13:46:51','2022-02-23',650);
00205
00206 SELECT
00207     Receipt_id,
00208     Receipt_time,
00209     Receipt_date,
00210     Receipt_total,
00211     CASE
00212         WHEN Receipt_total BETWEEN 600 AND 700 THEN 'salary increasing'
00213         ELSE 'normal'
00214     END AS salary
00215 FROM Receipt;
00216
00217 -- Receipt copy table --
00218 CREATE TABLE IF NOT EXISTS Receipt_copy (
00219     Receipt_id    NUMERIC(10) PRIMARY KEY,
00220     Receipt_time  TIME,
00221     Receipt_date  DATE,
00222     Receipt_total NUMERIC(10)
00223 );
00224
00225 INSERT INTO Receipt_copy VALUES (1,'12:45:51','2022-01-23',300);
00226 DELETE FROM Receipt_copy WHERE Receipt_id = 1;
00227
00228 -- Index --
00229 CREATE INDEX Itemindex ON Item (Item_food);
00230
00231 -- PROCEDURE: Chefname --
00232 DELIMITER $$
00233 CREATE PROCEDURE Chefname (IN Cheffname VARCHAR(10))
00234 BEGIN
00235     SELECT * FROM Chef WHERE Chef_name = Cheffname;
00236 END $$
00237 DELIMITER ;
00238
```

```
00239 CALL Chefname('mohammad');
00240 CALL Chefname('abduallah');
00241
00242 -- TRIGGER: uppercase names in Manager --
00243 CREATE TRIGGER uppercase BEFORE INSERT ON Manager
00244 FOR EACH ROW
00245 SET NEW.Manager_Fname = UPPER(NEW.Manager_Fname);
00246
00247 INSERT INTO Manager VALUES (3217,'Yazeed','fahad','alahmad',053276488,7000);
00248
00249 SELECT * FROM Manager;
00250
00251 -- CREACIÓN E INSERCIÓN DE DATOS EN TABLAS ADICIONALES --
00252 -- Tabla Appetizers --
00253 CREATE TABLE IF NOT EXISTS appetizers (
00254   id    INT NOT NULL AUTO_INCREMENT,
00255   name  VARCHAR(250) NOT NULL,
00256   price INT NOT NULL,
00257   PRIMARY KEY (id)
00258 );
00259
00260 INSERT INTO appetizers (id, name, price) VALUES
00261 (1,'Truffel Fries',23),
00262 (2,'Molten Chocolate',12),
00263 (3,'Mac&Cheese Balls',12),
00264 (4,'Dynamite Shrimp',32),
00265 (5,'Kheera',10);
00266
00267 -- Tabla Drinks --
00268 CREATE TABLE IF NOT EXISTS drinks (
00269   id    INT NOT NULL AUTO_INCREMENT,
00270   name  VARCHAR(250) NOT NULL,
00271   price INT NOT NULL,
00272   PRIMARY KEY (id)
00273 );
00274
00275 INSERT INTO drinks (id, name, price) VALUES
00276 (1,'cola',6),
00277 (2,'7up',6),
00278 (3,'orange juice',10),
00279 (4,'mojito',14),
00280 (5,'Red Bull',8);
00281
00282 -- Tabla MainCourse --
00283 CREATE TABLE IF NOT EXISTS maincourse (
00284   id    INT NOT NULL AUTO_INCREMENT,
00285   name  VARCHAR(250) NOT NULL,
00286   price INT NOT NULL,
00287   PRIMARY KEY (id)
00288 );
00289
00290 INSERT INTO maincourse (id, name, price) VALUES
00291 (1,'Buratta Pizza',54),
00292 (2,'Pink Pasta',12),
00293 (3,'Rosemary Salmon',30),
00294 (4,'Spaghetti',8),
00295 (5,'Crown Pizza',50);
00296
00297 COMMIT;
00298
00299 -- CONCESIÓN DE PRIVILEGIOS AL USUARIO RESTAURANT --
00300 GRANT ALL PRIVILEGES ON project3.* TO 'restaurant'@'%';
00301 FLUSH PRIVILEGES;
00302
00303 -- Reactivar restricciones de claves foráneas. --
00304 SET FOREIGN_KEY_CHECKS = 1;
```

8.15 README.md File Reference

8.16 SecurityConfig.java File Reference

`import org.springframework.context.annotation.Bean;`
Include dependency graph for SecurityConfig.java:



Classes

- class es.ull.esit.server.config.SecurityConfig

  Spring Security configuration for the backend.

Packages

- package es.ull.esit.server.config

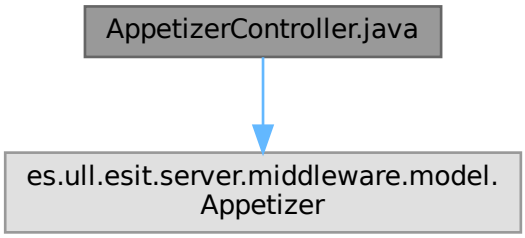
8.17 SecurityConfig.java

Go to the documentation of this file.

```
00001 package es.ull.esit.server.config;
00002
00003 import org.springframework.context.annotation.Bean;
00004 import org.springframework.context.annotation.Configuration;
00005 import org.springframework.security.config.annotation.web.builders.HttpSecurity;
00006 import org.springframework.security.web.SecurityFilterChain;
00007 import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
00008 import org.springframework.security.crypto.password.PasswordEncoder;
00009
00019 @Configuration
00020 public class SecurityConfig {
00021
00027   @Bean
00028   public PasswordEncoder passwordEncoder() {
00029     return new BCryptPasswordEncoder();
00030   }
00031
00044   @Bean
00045   public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {
00046     http
00047         .csrf().disable();
00048
00049     http
00050         .authorizeRequests()
00051         .anyRequest().permitAll();
00052
00053     http
00054         .httpBasic().disable()
00055         .formLogin().disable();
00056
00057     return http.build();
00058   }
00059 }
```

8.18 AppetizerController.java File Reference

`import es.ull.esit.server.middleware.model.Appetizer;`
Include dependency graph for AppetizerController.java:



Classes

- class es.ull.esit.server.controller.AppetizerController

  REST controller for managing appetizers.

Packages

- package es.ull.esit.server.controller

8.19 AppetizerController.java

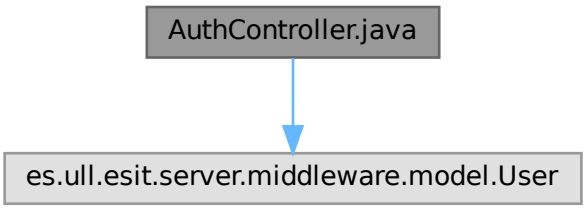
Go to the documentation of this file.

```
00001 package es.ull.esit.server.controller;
00002
00003 import es.ull.esit.server.middleware.model.Appetizer;
00004 import es.ull.esit.server.repo.AppetizerRepository;
00005 import org.springframework.beans.factory.annotation.Autowired;
00006 import org.springframework.http.*;
00007 import org.springframework.web.bind.annotation.*;
00008
00009 import java.util.List;
00010 import java.util.Optional;
00011
00019 @RestController
00020 @RequestMapping("/api/appetizers")
00021 public class AppetizerController {
00022
00024   @Autowired
00025   private AppetizerRepository appetizerRepository;
00026
00034   @GetMapping
00035   public ResponseEntity<List<Appetizer>> getAllAppetizers() {
00036     List<Appetizer> appetizers = appetizerRepository.findAll();
00037     return ResponseEntity.ok(appetizers);
00038   }
00039
00048   @GetMapping("/{id}")
00049   public ResponseEntity<Appetizer> getAppetizerById(@PathVariable Long id) {
00050     Optional<Appetizer> appetizer = appetizerRepository.findById(id);
00051     return appetizer
00052         .map(ResponseEntity::ok)
```

```
00053         .orElseGet(() -> ResponseEntity.notFound().build());
00054   }
00055
00064   @PostMapping
00065   public ResponseEntity<Appetizer> createAppetizer(@RequestBody Appetizer appetizer) {
00066     Appetizer saved = appetizerRepository.save(appetizer);
00067     return ResponseEntity.status(HttpStatus.CREATED).body(saved);
00068   }
00069
00079   @PutMapping("/{id}")
00080   public ResponseEntity<Appetizer> updateAppetizer(@PathVariable Long id,
00081                                                    @RequestBody Appetizer appetizer) {
00082     if (!appetizerRepository.existsById(id)) {
00083       return ResponseEntity.notFound().build();
00084     }
00085     appetizer.setAppetizersId(id);
00086     Appetizer updated = appetizerRepository.save(appetizer);
00087     return ResponseEntity.ok(updated);
00088   }
00089
00098   @DeleteMapping("/{id}")
00099   public ResponseEntity<Void> deleteAppetizer(@PathVariable Long id) {
00100     if (!appetizerRepository.existsById(id)) {
00101       return ResponseEntity.notFound().build();
00102     }
00103     appetizerRepository.deleteById(id);
00104     return ResponseEntity.noContent().build();
00105   }
00106 }
```

8.20 AuthController.java File Reference

`import es.ull.esit.server.middleware.model.User;`
Include dependency graph for AuthController.java:



Classes

- class es.ull.esit.server.controller.AuthController

  Rest controller for handling authentication-related requests.
- class es.ull.esit.server.controller.AuthController.LoginRequest

  Simple DTO (Data Transfer Object) for login requests payload.

Packages

- package es.ull.esit.server.controller

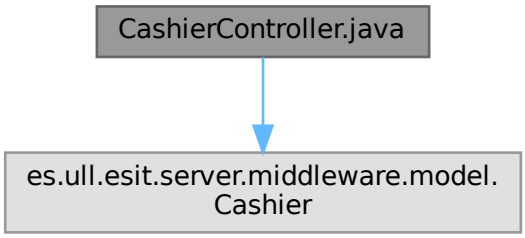
8.21 AuthController.java

Go to the documentation of this file.

```
00001 package es.ull.esit.server.controller;
00002
00003 import es.ull.esit.server.middleware.model.User;
00004 import es.ull.esit.server.repo.UserRepository;
00005
00006 import org.springframework.beans.factory.annotation.Autowired;
00007 import org.springframework.http.ResponseEntity;
00008 import org.springframework.security.crypto.password.PasswordEncoder;
00009 import org.springframework.web.bind.annotation.*;
00010 import java.util.Optional;
00011 import org.slf4j.Logger;
00012 import org.slf4j.LoggerFactory;
00013
00021 @RestController
00022 @RequestMapping("/api")
00023 public class AuthController {
00024
00026   private static final Logger LOG = LoggerFactory.getLogger(AuthController.class);
00027
00029   @Autowired
00030   private UserRepository userRepository;
00031
00033   @Autowired
00034   private PasswordEncoder passwordEncoder;
00035
00042   public static class LoginRequest {
00043     public String username;
00044     public String password;
00045   }
00046
00060   @PostMapping("/login")
00061   public ResponseEntity<?> login(@RequestBody LoginRequest request) {
00062
00063     LOG.info("Login attempt for user: {}", request.username);
00064
00065     Optional<User> userOpt = userRepository.findByUsername(request.username);
00066
00067     if (!userOpt.isPresent()) {
00068       LOG.warn("User not found in DB: {}", request.username);
00069       return ResponseEntity.status(401).body("Invalid credentials");
00070     }
00071
00072     User user = userOpt.get();
00073     LOG.debug("User found. Stored hash = {}", user.getPasswordHash()); // Mejor debug, no info
00074
00075     if (!passwordEncoder.matches(request.password, user.getPasswordHash())) {
00076       LOG.warn("Password mismatch for user: {}", request.username);
00077       return ResponseEntity.status(401).body("Invalid credentials");
00078     }
00079
00080     LOG.info("Login OK for user: {}", request.username);
00081
00082     return ResponseEntity.ok(user);
00083
00084   }
00085
00086 }
```

8.22 CashierController.java File Reference

`import es.ull.esit.server.middleware.model.Cashier;`
Include dependency graph for CashierController.java:



Classes

- class es.ull.esit.server.controller.CashierController

  REST controller for managing cashiers.

Packages

- package es.ull.esit.server.controller

8.23 CashierController.java

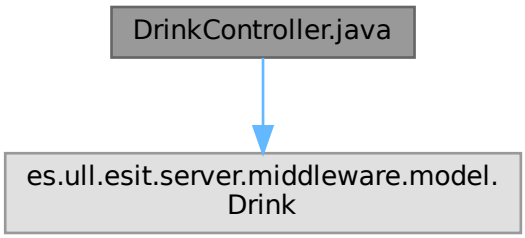
Go to the documentation of this file.

```
00001 package es.ull.esit.server.controller;
00002
00003 import es.ull.esit.server.middleware.model.Cashier;
00004 import es.ull.esit.server.repo.CashierRepository;
00005 import org.springframework.beans.factory.annotation.Autowired;
00006 import org.springframework.http.HttpStatus;
00007 import org.springframework.http.ResponseEntity;
00008 import org.springframework.web.bind.annotation.*;
00009 import java.util.List;
00010 import java.util.Optional;
00011
00023 @RestController
00024 @RequestMapping("/api/cashiers")
00025 public class CashierController {
00026
00027   /* Repository used to perform database operations for Cashier entities. */
00028   @Autowired
00029   private CashierRepository cashierRepository;
00030
00039   @GetMapping
00040   public ResponseEntity<List<Cashier» getAllCashiers() {
00041     List<Cashier> cashiers = cashierRepository.findAll();
00042     return ResponseEntity.ok(cashiers);
00043   }
00044
00054   @GetMapping("/{id}")
00055   public ResponseEntity<Cashier> getCashierById(@PathVariable Long id) {
00056     Optional<Cashier> cashier = cashierRepository.findById(id);
00057     return cashier
```

```
00058         .map(ResponseEntity::ok)
00059         .orElseGet(() -> ResponseEntity.notFound().build());
00060   }
00061
00071   @GetMapping("/name/{name}")
00072   public ResponseEntity<Cashier> getCashierByName(@PathVariable String name) {
00073     Optional<Cashier> cashier = cashierRepository.findByName(name);
00074     return cashier
00075         .map(ResponseEntity::ok)
00076         .orElseGet(() -> ResponseEntity.notFound().build());
00077   }
00078
00089   @PutMapping("/{id}")
00090   public ResponseEntity<Cashier> updateCashier(@PathVariable Long id,
00091       @RequestBody Cashier cashier) {
00092     Optional<Cashier> existingOpt = cashierRepository.findById(id);
00093     if (!existingOpt.isPresent()) {
00094       return ResponseEntity.notFound().build();
00095     }
00096
00097     Cashier existing = existingOpt.get();
00098     existing.setName(cashier.getName());
00099     existing.setSalary(cashier.getSalary());
00100
00101     Cashier updated = cashierRepository.save(existing);
00102     return ResponseEntity.ok(updated);
00103   }
00104
00105 }
```

8.24 DrinkController.java File Reference

`import es.ull.esit.server.middleware.model.Drink;`
Include dependency graph for DrinkController.java:



Classes

- class es.ull.esit.server.controller.DrinkController

  REST controller for managing drinks.

Packages

- package es.ull.esit.server.controller

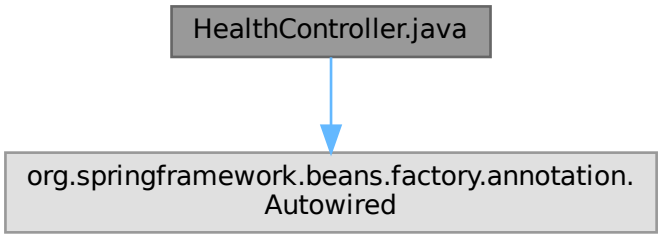
8.25 DrinkController.java

Go to the documentation of this file.

```
00001 package es.ull.esit.server.controller;
00002
00003 import es.ull.esit.server.middleware.model.Drink;
00004 import es.ull.esit.server.repo.DrinkRepository;
00005 import org.springframework.beans.factory.annotation.Autowired;
00006 import org.springframework.http.HttpStatus;
00007 import org.springframework.http.ResponseEntity;
00008 import org.springframework.web.bind.annotation.*;
00009
00010 import java.util.List;
00011 import java.util.Optional;
00012
00020 @RestController
00021 @RequestMapping("/api/drinks")
00022 public class DrinkController {
00023
00025   @Autowired
00026   private DrinkRepository drinkRepository;
00027
00035   @GetMapping
00036   public ResponseEntity<List<Drink» getAllDrinks() {
00037     List<Drink> drinks = drinkRepository.findAll();
00038     return ResponseEntity.ok(drinks);
00039   }
00040
00049   @GetMapping("/{id}")
00050   public ResponseEntity<Drink> getDrinkById(@PathVariable Long id) {
00051     Optional<Drink> drink = drinkRepository.findById(id);
00052     return drink.map(ResponseEntity::ok)
00053         .orElseGet(() -> ResponseEntity.notFound().build());
00054   }
00055
00064   @PostMapping
00065   public ResponseEntity<Drink> createDrink(@RequestBody Drink drink) {
00066     Drink saved = drinkRepository.save(drink);
00067     return ResponseEntity.status(HttpStatus.CREATED).body(saved);
00068   }
00069
00079   @PutMapping("/{id}")
00080   public ResponseEntity<Drink> updateDrink(@PathVariable Long id,
00081       @RequestBody Drink drink) {
00082     if (!drinkRepository.existsById(id)) {
00083       return ResponseEntity.notFound().build();
00084     }
00085     drink.setDrinksId(id);
00086     Drink updated = drinkRepository.save(drink);
00087     return ResponseEntity.ok(updated);
00088   }
00089
00098   @DeleteMapping("/{id}")
00099   public ResponseEntity<Void> deleteDrink(@PathVariable Long id) {
00100     if (!drinkRepository.existsById(id)) {
00101       return ResponseEntity.notFound().build();
00102     }
00103     drinkRepository.deleteById(id);
00104     return ResponseEntity.noContent().build();
00105   }
00106 }
```

8.26 HealthController.java File Reference

`import org.springframework.beans.factory.annotation.Autowired;`
Include dependency graph for HealthController.java:



Classes

- class es.ull.esit.server.controller.HealthController

  REST controller for health and database connectivity checks.

Packages

- package es.ull.esit.server.controller

8.27 HealthController.java

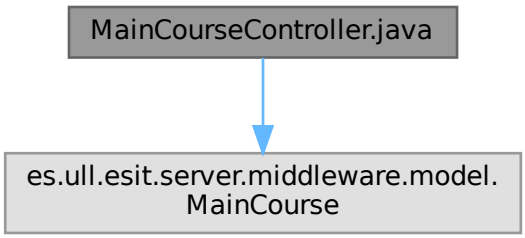
Go to the documentation of this file.

```
00001 package es.ull.esit.server.controller;
00002
00003 import org.springframework.beans.factory.annotation.Autowired;
00004 import org.springframework.http.HttpStatus;
00005 import org.springframework.http.ResponseEntity;
00006 import org.springframework.web.bind.annotation.GetMapping;
00007 import org.springframework.web.bind.annotation.RequestMapping;
00008 import org.springframework.web.bind.annotation.RestController;
00009
00010 import javax.sql.DataSource;
00011 import java.sql.Connection;
00012 import java.util.HashMap;
00013 import java.util.Map;
00014
00025 @RestController
00026 @RequestMapping("/api")
00027 public class HealthController {
00028
00030   @Autowired
00031   private DataSource dataSource;
00032
00043   @GetMapping("/health")
00044   public ResponseEntity<Map<String, Object>> health() {
00045     Map<String, Object> response = new HashMap<>();
00046     response.put("status", "UP");
00047     response.put("timestamp", System.currentTimeMillis());
00048     response.put("service", "Restaurant Server");
00049     return ResponseEntity.ok(response);
00050   }
```

```
00051
00072   @GetMapping("/db-check")
00073   public ResponseEntity<Map<String, Object» checkDatabase() {
00074     Map<String, Object> response = new HashMap<>();
00075
00076     try (Connection conn = dataSource.getConnection()) {
00077       boolean isValid = conn.isValid(2);
00078
00079       if (isValid) {
00080         response.put("status", "UP");
00081         response.put("database", "Connected");
00082         response.put("catalog", conn.getCatalog());
00083         response.put("url", conn.getMetaData().getURL());
00084         response.put("timestamp", System.currentTimeMillis());
00085         return ResponseEntity.ok(response);
00086       } else {
00087         response.put("status", "DOWN");
00088         response.put("database", "Connection not valid");
00089         return ResponseEntity.status(HttpStatus.SERVICE_UNAVAILABLE).body(response);
00090       }
00091     } catch (Exception e) {
00092       response.put("status", "DOWN");
00093       response.put("database", "Connection failed");
00094       response.put("error", e.getMessage());
00095       response.put("timestamp", System.currentTimeMillis());
00096       return ResponseEntity.status(HttpStatus.SERVICE_UNAVAILABLE).body(response);
00097     }
00098   }
00099 }
```

8.28 MainCourseController.java File Reference

`import es.ull.esit.server.middleware.model.MainCourse;`
Include dependency graph for MainCourseController.java:



Classes

- class es.ull.esit.server.controller.MainCourseController

  REST controller for managing main courses.

Packages

- package es.ull.esit.server.controller

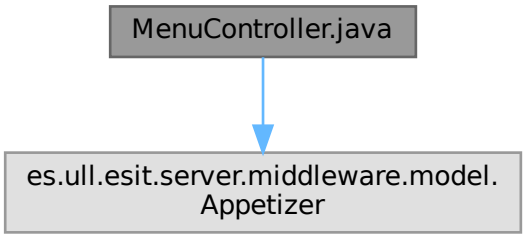
8.29 MainCourseController.java

Go to the documentation of this file.

```
00001 package es.ull.esit.server.controller;
00002
00003 import es.ull.esit.server.middleware.model.MainCourse;
00004 import es.ull.esit.server.repo.MainCourseRepository;
00005 import org.springframework.beans.factory.annotation.Autowired;
00006 import org.springframework.http.*;
00007 import org.springframework.web.bind.annotation.*;
00008
00009 import java.util.List;
00010 import java.util.Optional;
00011
00020 @RestController
00021 @RequestMapping("/api/maincourses")
00022 public class MainCourseController {
00023
00025   @Autowired
00026   private MainCourseRepository mainCourseRepository;
00027
00036   @GetMapping
00037   public ResponseEntity<List<MainCourse» getAllMainCourses() {
00038     List<MainCourse> courses = mainCourseRepository.findAll();
00039     return ResponseEntity.ok(courses);
00040   }
00041
00051   @GetMapping("/{id}")
00052   public ResponseEntity<MainCourse> getMainCourseById(@PathVariable Long id) {
00053     Optional<MainCourse> course = mainCourseRepository.findById(id);
00054     return course
00055         .map(ResponseEntity::ok)
00056         .orElseGet(() -> ResponseEntity.notFound().build());
00057   }
00058
00067   @PostMapping
00068   public ResponseEntity<MainCourse> createMainCourse(@RequestBody MainCourse mainCourse) {
00069     MainCourse saved = mainCourseRepository.save(mainCourse);
00070     return ResponseEntity.status(HttpStatus.CREATED).body(saved);
00071   }
00072
00084   @PutMapping("/{id}")
00085   public ResponseEntity<MainCourse> updateMainCourse(@PathVariable Long id, @RequestBody MainCourse
      mainCourse) {
00086
00087     if (!mainCourseRepository.existsById(id)) {
00088       return ResponseEntity.notFound().build();
00089     }
00090
00091     mainCourse.setFoodId(id);
00092     MainCourse updated = mainCourseRepository.save(mainCourse);
00093     return ResponseEntity.ok(updated);
00094   }
00095
00104   @DeleteMapping("/{id}")
00105   public ResponseEntity<Void> deleteMainCourse(@PathVariable Long id) {
00106     if (!mainCourseRepository.existsById(id)) {
00107       return ResponseEntity.notFound().build();
00108     }
00109
00110     mainCourseRepository.deleteById(id);
00111     return ResponseEntity.noContent().build();
00112   }
00113 }
```

8.30 MenuController.java File Reference

`import es.ull.esit.server.middleware.model.Appetizer;`
Include dependency graph for MenuController.java:



Classes

- class es.ull.esit.server.controller.MenuController

  REST controller that exposes a consolidated restaurant menu.

Packages

- package es.ull.esit.server.controller

8.31 MenuController.java

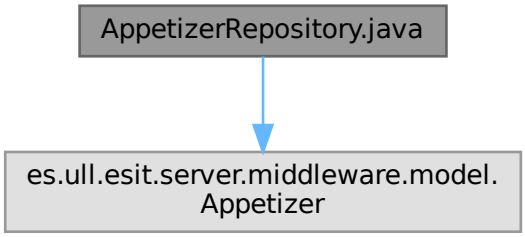
Go to the documentation of this file.

```
00001 package es.ull.esit.server.controller;
00002
00003 import es.ull.esit.server.middleware.model.Appetizer;
00004 import es.ull.esit.server.middleware.model.Drink;
00005 import es.ull.esit.server.middleware.model.MainCourse;
00006 import es.ull.esit.server.repo.AppetizerRepository;
00007 import es.ull.esit.server.repo.DrinkRepository;
00008 import es.ull.esit.server.repo.MainCourseRepository;
00009 import org.springframework.beans.factory.annotation.Autowired;
00010 import org.springframework.http.ResponseEntity;
00011 import org.springframework.web.bind.annotation.GetMapping;
00012 import org.springframework.web.bind.annotation.RequestMapping;
00013 import org.springframework.web.bind.annotation.RestController;
00014
00015 import java.util.HashMap;
00016 import java.util.List;
00017 import java.util.Map;
00018
00027 @RestController
00028 @RequestMapping("/api/menu")
00029 public class MenuController {
00030
00032   @Autowired
00033   private MainCourseRepository mainCourseRepository;
00034
00036   @Autowired
00037   private AppetizerRepository appetizerRepository;
00038
00040   @Autowired
```

```
00041   private DrinkRepository drinkRepository;
00042
00056   @GetMapping
00057   public ResponseEntity<Map<String, Object» getFullMenu() {
00058     Map<String, Object> menu = new HashMap<>();
00059
00060     List<MainCourse> mainCourses = mainCourseRepository.findAll();
00061     List<Appetizer> appetizers = appetizerRepository.findAll();
00062     List<Drink> drinks = drinkRepository.findAll();
00063
00064     menu.put("mainCourses", mainCourses);
00065     menu.put("appetizers", appetizers);
00066     menu.put("drinks", drinks);
00067     menu.put("totalItems", mainCourses.size() + appetizers.size() + drinks.size());
00068
00069     return ResponseEntity.ok(menu);
00070   }
00071 }
```

8.32 AppetizerRepository.java File Reference

`import es.ull.esit.server.middleware.model.Appetizer;`
Include dependency graph for AppetizerRepository.java:



Classes

- interface es.ull.esit.server.repo.AppetizerRepository

  Repository interface for managing appetizers in the database.

Packages

- package es.ull.esit.server.repo

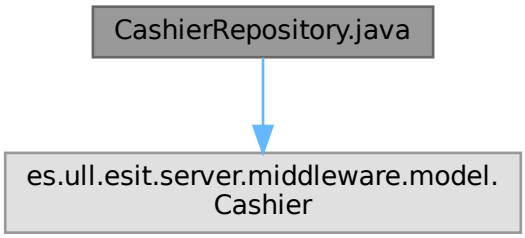
8.33 AppetizerRepository.java

Go to the documentation of this file.

```
00001 package es.ull.esit.server.repo;
00002
00003 import es.ull.esit.server.middleware.model.Appetizer;
00004 import org.springframework.data.jpa.repository.JpaRepository;
00005
00012 public interface AppetizerRepository extends JpaRepository<Appetizer, Long> {
00013   // No extra methods are added for now, but custom queries can be defined here if
00014   // needed in the future.
00015 }
```

8.34 CashierRepository.java File Reference

`import es.ull.esit.server.middleware.model.Cashier;`
Include dependency graph for CashierRepository.java:



Classes

- interface es.ull.esit.server.repo.CashierRepository

  Repository interface for managing cashiers in the database.

Packages

- package es.ull.esit.server.repo

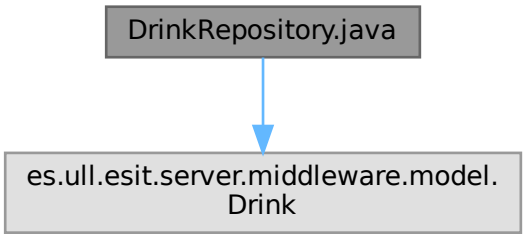
8.35 CashierRepository.java

Go to the documentation of this file.

```
00001 package es.ull.esit.server.repo;
00002
00003 import es.ull.esit.server.middleware.model.Cashier;
00004 import org.springframework.data.jpa.repository.JpaRepository;
00005
00006 import java.util.Optional;
00007
00014 public interface CashierRepository extends JpaRepository<Cashier, Long> {
00015
00022   Optional<Cashier> findByName(String name);
00023 }
```

8.36 DrinkRepository.java File Reference

`import es.ull.esit.server.middleware.model.Drink;`
Include dependency graph for DrinkRepository.java:



Classes

- interface es.ull.esit.server.repo.DrinkRepository

  Repository interface for managing drinks in the database.

Packages

- package es.ull.esit.server.repo

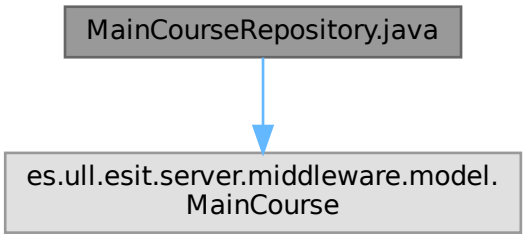
8.37 DrinkRepository.java

Go to the documentation of this file.

```
00001 package es.ull.esit.server.repo;
00002
00003 import es.ull.esit.server.middleware.model.Drink;
00004 import org.springframework.data.jpa.repository.JpaRepository;
00005
00012 public interface DrinkRepository extends JpaRepository<Drink, Long> {
00013   // No extra methods are added for now, but custom queries can be defined here if
00014   // needed in the future.
00015 }
```

8.38 MainCourseRepository.java File Reference

`import es.ull.esit.server.middleware.model.MainCourse;`
Include dependency graph for MainCourseRepository.java:



Classes

- interface es.ull.esit.server.repo.MainCourseRepository

  Repository interface for managing main courses in the database.

Packages

- package es.ull.esit.server.repo

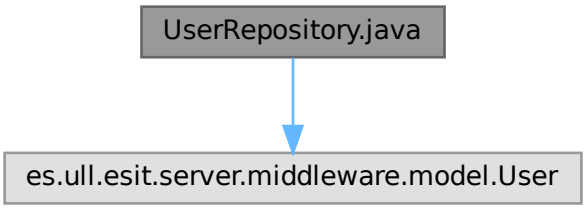
8.39 MainCourseRepository.java

Go to the documentation of this file.

```
00001 package es.ull.esit.server.repo;
00002
00003 import es.ull.esit.server.middleware.model.MainCourse;
00004 import org.springframework.data.jpa.repository.JpaRepository;
00005
00013 public interface MainCourseRepository extends JpaRepository<MainCourse, Long> {
00014   // No extra methods are added for now, but custom queries can be defined here if
00015   // needed in the future.
00016 }
```

8.40 UserRepository.java File Reference

`import es.ull.esit.server.middleware.model.User;`
Include dependency graph for UserRepository.java:



Classes

- interface es.ull.esit.server.repo.UserRepository

  Repository interface for accessing users in the database.

Packages

- package es.ull.esit.server.repo

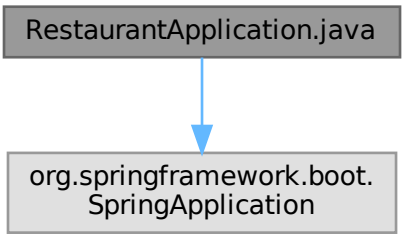
8.41 UserRepository.java

Go to the documentation of this file.

```
00001 package es.ull.esit.server.repo;
00002
00003 import es.ull.esit.server.middleware.model.User;
00004 import org.springframework.data.jpa.repository.JpaRepository;
00005 import java.util.Optional;
00006
00013 public interface UserRepository extends JpaRepository<User, Long> {
00020   Optional<User> findByUsername(String username);
00021 }
```

8.42 RestaurantApplication.java File Reference

`import org.springframework.boot.SpringApplication;`
Include dependency graph for RestaurantApplication.java:



Classes

- class es.ull.esit.server.RestaurantApplication

  Main Spring Boot application class for the restaurant backend.

Packages

- package es.ull.esit.server

8.43 RestaurantApplication.java

Go to the documentation of this file.

```
00001 package es.ull.esit.server;
00002
00003 import org.springframework.boot.SpringApplication;
00004 import org.springframework.boot.autoconfigure.SpringBootApplication;
00005
00019 @SpringBootApplication
00020 public class RestaurantApplication {
00021
00030   public static void main(String[] args) {
00031     SpringApplication.run(RestaurantApplication.class, args);
00032   }
00033 }
```

8.44 AboutUs.java File Reference

Classes

- class es.ull.esit.app.AboutUs

  "About us" window displaying the restaurant's history and info.

Packages

- package es.ull.esit.app

8.45 AboutUs.java

Go to the documentation of this file.

```
00001 package es.ull.esit.app;
00002
00014 public class AboutUs extends javax.swing.JFrame {
00015
00021   public AboutUs() {
00022     initComponents();
00023   }
00024
00036   @SuppressWarnings("unchecked")
00037   // <editor-fold defaultstate="collapsed" desc="Generated
00038   // Code">//GEN-BEGIN:initComponents
00039   private void initComponents() {
00040
00041     jPanel1 = new javax.swing.JPanel();
00042     jButton3 = new javax.swing.JButton();
00043     jScrollPane1 = new javax.swing.JScrollPane();
00044     ourStory = new javax.swing.JTextArea();
00045     jLabel1 = new javax.swing.JLabel();
00046
00047     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00048     setTitle("Info");
00049     setResizable(false);
00050
00051     jPanel1.setBackground(new java.awt.Color(248, 244, 230));
00052
00053     jButton3.setBackground(new java.awt.Color(255, 255, 255));
00054     jButton3.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00055     jButton3.setText("Go Back");
00056     jButton3.addActionListener(new java.awt.event.ActionListener() {
00057       public void actionPerformed(java.awt.event.ActionEvent evt) {
00058         jButton3ActionPerformed(evt);
00059       }
00060     });
00061
00062     ourStory.setEditable(false);
00063     ourStory.setBackground(new java.awt.Color(248, 244, 230));
00064     ourStory.setColumns(20);
00065     ourStory.setFont(new java.awt.Font("Yu Gothic UI Light", 0, 14)); // NOI18N
00066     ourStory.setRows(5);
00067     ourStory.setText(
00068         "  Our Story |\n" +
00069             "  Collecting flavors from all over the world to give everyone a taste of what they love.
      \n\n" +
00070             "  Black Plate came in with this vision in mind, to be more of a home rather than an
      establishment.\n\n" +
00071             "  2021 marked the beginning of the journey of Black Plate, starting from Al Khobar.\n\n"
      +
00072             "  Because your visit means a lot to us, we would love to hear your suggestions and
      concerns \n\n" +
00073             "  so we can improve!\n");
00074     ourStory.setBorder(javax.swing.BorderFactory.createLineBorder(new java.awt.Color(0, 0, 0)));
00075     jScrollPane1.setViewportView(ourStory);
00076
00077     // Updated path to match other UI resources
00078     jLabel1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/ui/screenshot.png")));
00079
00080     javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
00081     jPanel1.setLayout(jPanel1Layout);
00082     jPanel1Layout.setHorizontalGroup(
00083         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00084             .addGroup(jPanel1Layout.createSequentialGroup()
00085                 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00086                     .addGroup(jPanel1Layout.createSequentialGroup()
00087                         .addGap(55, 55, 55)
00088
      .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00089                             .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 141,
00090                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00091                             .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 1004,
00092                                 javax.swing.GroupLayout.PREFERRED_SIZE)))
00093                     .addGroup(jPanel1Layout.createSequentialGroup()
00094                         .addGap(489, 489, 489)
00095                         .addComponent(jLabel1)))
```

```
00096                 .addContainerGap(159, Short.MAX_VALUE)));
00097     jPanel1Layout.setVerticalGroup(
00098         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00099             .addGroup(jPanel1Layout.createSequentialGroup()
00100                 .addContainerGap(71, Short.MAX_VALUE)
00101                 .addComponent(jLabel1)
00102                 .addGap(67, 67, 67)
00103                 .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 203,
00104                     javax.swing.GroupLayout.PREFERRED_SIZE)
00105                 .addGap(26, 26, 26)
00106                 .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00107                     javax.swing.GroupLayout.PREFERRED_SIZE)
00108                 .addGap(30, 30, 30)));
00109 
00110     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00111     getContentPane().setLayout(layout);
00112     layout.setHorizontalGroup(
00113         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00114             .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00115                 Short.MAX_VALUE));
00116     layout.setVerticalGroup(
00117         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00118             .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00119                 Short.MAX_VALUE));
00120 
00121     pack();
00122     setLocationRelativeTo(null);
00123   }// </editor-fold>//GEN-END:initComponents
00124 
00133   private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {//
      GEN-FIRST:event_jButton3ActionPerformed
00134     new CashierLogin().setVisible(true);
00135     this.dispose(); // Close current window
00136   }// GEN-LAST:event_jButton3ActionPerformed
00137 
00148   public static void main(String[] args) {
00149     try {
00150       for (javax.swing.UIManager.LookAndFeelInfo info :
      javax.swing.UIManager.getInstalledLookAndFeels()) {
00151         if ("Nimbus".equals(info.getName())) {
00152           javax.swing.UIManager.setLookAndFeel(info.getClassName());
00153           break;
00154         }
00155       }
00156     } catch (ClassNotFoundException ex) {
00157       java.util.logging.Logger.getLogger(AboutUs.class.getName())
00158           .log(java.util.logging.Level.SEVERE, null, ex);
00159     } catch (InstantiationException ex) {
00160       java.util.logging.Logger.getLogger(AboutUs.class.getName())
00161           .log(java.util.logging.Level.SEVERE, null, ex);
00162     } catch (IllegalAccessException ex) {
00163       java.util.logging.Logger.getLogger(AboutUs.class.getName())
00164           .log(java.util.logging.Level.SEVERE, null, ex);
00165     } catch (javax.swing.UnsupportedLookAndFeelException ex) {
00166       java.util.logging.Logger.getLogger(AboutUs.class.getName())
00167           .log(java.util.logging.Level.SEVERE, null, ex);
00168     }
00169     // </editor-fold>
00170 
00171     /* Create and display the form */
00172     java.awt.EventQueue.invokeLater(() -> new AboutUs().setVisible(true));
00173   }
00174 
00175   // Variables declaration - do not modify//GEN-BEGIN:variables
00177   private javax.swing.JButton jButton3;
00179   private javax.swing.JLabel jLabel1;
00181   private javax.swing.JPanel jPanel1;
00183   private javax.swing.JScrollPane jScrollPane1;
00185   private javax.swing.JTextArea ourStory;
00186   // End of variables declaration//GEN-END:variables
00187 }
```

8.46 AdminLogin.java File Reference

Classes

- class es.ull.esit.app.AdminLogin

  Login window for authenticating administrators.

Packages

- package es.ull.esit.app

8.47 AdminLogin.java

Go to the documentation of this file.

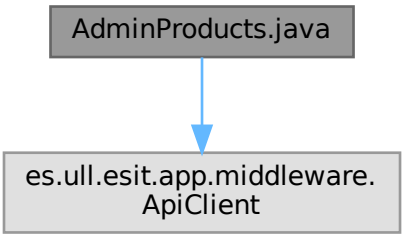
```
00001 package es.ull.esit.app;
00002
00012 public class AdminLogin extends javax.swing.JFrame {
00013
00019   public AdminLogin() {
00020     initComponents();
00021   }
00022
00030   @SuppressWarnings("unchecked")
00031   // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
00032   private void initComponents() {
00033
00034     jPanel1 = new javax.swing.JPanel();
00035     jLabel3 = new javax.swing.JLabel();
00036     jLabel1 = new javax.swing.JLabel();
00037     jButton2 = new javax.swing.JButton();
00038     jButton1 = new javax.swing.JButton();
00039     jButton3 = new javax.swing.JButton();
00040
00041     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00042     setTitle("Admin ");
00043     setResizable(false);
00044
00045     jPanel1.setBackground(new java.awt.Color(248, 244, 230));
00046
00047     jLabel3.setFont(new java.awt.Font("Yu Gothic UI", 1, 24)); // NOI18N
00048     jLabel3.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00049     jLabel3.setText("Welcome Admin");
00050
00051     jLabel1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/ui/screenshot.png"))); //
      NOI18N
00052
00053     jButton2.setBackground(new java.awt.Color(153, 153, 153));
00054     jButton2.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00055     jButton2.setText("Update Prices");
00056     jButton2.addActionListener(new java.awt.event.ActionListener() {
00057
00058       public void actionPerformed(java.awt.event.ActionEvent evt) {
00059         jButton2ActionPerformed(evt);
00060       }
00061     });
00062
00063     jButton1.setBackground(new java.awt.Color(153, 153, 153));
00064     jButton1.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00065     jButton1.setText("Menu");
00066     jButton1.addActionListener(new java.awt.event.ActionListener() {
00067
00068       public void actionPerformed(java.awt.event.ActionEvent evt) {
00069         jButton1ActionPerformed(evt);
00070       }
00071     });
00072
00073     jButton3.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00074     jButton3.setText("LogOut");
00075     jButton3.addActionListener(new java.awt.event.ActionListener() {
00076
00077       public void actionPerformed(java.awt.event.ActionEvent evt) {
00078         jButton3ActionPerformed(evt);
00079       }
00080     });
00081
00082     javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
00083     jPanel1.setLayout(jPanel1Layout);
00084     jPanel1Layout.setHorizontalGroup(
00085         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00086             .addGroup(jPanel1Layout.createSequentialGroup()
00087                 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00088                     .addGroup(jPanel1Layout.createSequentialGroup()
00089                         .addGap(140, 140, 140)
00090                         .addComponent(jLabel1))
00091                     .addGroup(jPanel1Layout.createSequentialGroup()
00092                         .addGap(66, 66, 66)
00093
      .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
```

```
00094                               .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 320,
00095                                   javax.swing.GroupLayout.PREFERRED_SIZE)
00096                               .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 320,
00097                                   javax.swing.GroupLayout.PREFERRED_SIZE)
00098                               .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 306,
00099                                   javax.swing.GroupLayout.PREFERRED_SIZE)))
00100                       .addGroup(jPanel1Layout.createSequentialGroup()
00101                           .addGap(152, 152, 152)
00102                           .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 141,
00103                               javax.swing.GroupLayout.PREFERRED_SIZE)))
00104                   .addContainerGap(69, Short.MAX_VALUE)));
00105       jPanel1Layout.setVerticalGroup(
00106           jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00107               .addGroup(jPanel1Layout.createSequentialGroup()
00108                   .addGap(31, 31, 31)
00109                   .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
00110                       javax.swing.GroupLayout.PREFERRED_SIZE)
00111                   .addGap(18, 18, 18)
00112                   .addComponent(jLabel1)
00113                   .addGap(29, 29, 29)
00114                   .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00115                       javax.swing.GroupLayout.PREFERRED_SIZE)
00116                   .addGap(18, 18, 18)
00117                   .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00118                       javax.swing.GroupLayout.PREFERRED_SIZE)
00119                   .addGap(29, 29, 29)
00120                   .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00121                       javax.swing.GroupLayout.PREFERRED_SIZE)
00122                   .addContainerGap(115, Short.MAX_VALUE)));
00123
00124       javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00125       getContentPane().setLayout(layout);
00126       layout.setHorizontalGroup(
00127           layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00128               .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00129                   javax.swing.GroupLayout.PREFERRED_SIZE));
00130       layout.setVerticalGroup(
00131           layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00132               .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00133                   Short.MAX_VALUE));
00134
00135       pack();
00136       setLocationRelativeTo(null);
00137   }// </editor-fold>//GEN-END:initComponents
00138
00149   private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {//
      GEN-FIRST:event_jButton2ActionPerformed
00150       try {
00151           new AdminProducts().setVisible(true);
00152           this.dispose();
00153       } catch (Exception ex) {
00154           ex.printStackTrace();
00155           javax.swing.JOptionPane.showMessageDialog(
00156               this,
00157               "Error opening product admin window:\n" + ex.getMessage(),
00158               "Error",
00159               javax.swing.JOptionPane.ERROR_MESSAGE
00160           );
00161       }
00162   }// GEN-LAST:event_jButton2ActionPerformed
00163
00172   private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//
      GEN-FIRST:event_jButton1ActionPerformed
00173       try {
00174           new Order().setVisible(true);
00175           this.dispose();
00176       } catch (Exception ex) {
00177           ex.printStackTrace();
00178           javax.swing.JOptionPane.showMessageDialog(
00179               this,
00180               "Error opening menu window:\n" + ex.getMessage(),
00181               "Error",
00182               javax.swing.JOptionPane.ERROR_MESSAGE
00183           );
00184       }
00185   }// GEN-LAST:event_jButton1ActionPerformed
00186
00195   private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {//
      GEN-FIRST:event_jButton3ActionPerformed
00196       new Login().setVisible(true);
00197       this.dispose();
00198   }// GEN-LAST:event_jButton3ActionPerformed
00199
00209   public static void main(String[] args) {
00210       try {
```

```
00211       for (javax.swing.UIManager.LookAndFeelInfo info :
      javax.swing.UIManager.getInstalledLookAndFeels()) {
00212         if ("Nimbus".equals(info.getName())) {
00213           javax.swing.UIManager.setLookAndFeel(info.getClassName());
00214           break;
00215         }
00216       }
00217     } catch (ClassNotFoundException | InstantiationException | IllegalAccessException
00218         | javax.swing.UnsupportedLookAndFeelException ex) {
00219
      java.util.logging.Logger.getLogger(AdminLogin.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
00220     }
00221     // </editor-fold>
00222
00223     java.awt.EventQueue.invokeLater(new Runnable() {
00224       public void run() {
00225         new AdminLogin().setVisible(true);
00226       }
00227     });
00228   }
00229
00230   // Variables declaration - do not modify//GEN-BEGIN:variables
00232   private javax.swing.JButton jButton1;
00234   private javax.swing.JButton jButton2;
00236   private javax.swing.JButton jButton3;
00238   private javax.swing.JLabel jLabel1;
00240   private javax.swing.JLabel jLabel3;
00242   private javax.swing.JPanel jPanel1;
00243   // End of variables declaration//GEN-END:variables
00244 }
```

8.48 AdminProducts.java File Reference

`import es.ull.esit.app.middleware.ApiClient;`
Include dependency graph for AdminProducts.java:



Classes

- class es.ull.esit.app.AdminProducts

  Administrative window for managing products and prices.

Packages

- package es.ull.esit.app

8.49 AdminProducts.java

Go to the documentation of this file.

```
00001 package es.ull.esit.app;
00002
00003 import es.ull.esit.app.middleware.ApiClient;
00004 import es.ull.esit.app.middleware.model.Appetizer;
00005 import es.ull.esit.app.middleware.model.Drink;
00006 import es.ull.esit.app.middleware.model.MainCourse;
00007 import es.ull.esit.app.middleware.service.ProductService;
00008 import java.util.List;
00009 import javax.swing.JOptionPane;
00010 import javax.swing.SwingUtilities;
00011 import javax.swing.table.DefaultTableModel;
00012
00024 public class AdminProducts extends javax.swing.JFrame {
00025
00027   private final ProductService productService;
00028
00030   DefaultTableModel modelDrink;
00032   DefaultTableModel modelappetizers;
00034   DefaultTableModel modelmaincourse;
00035
00037   String[] columnNames = { "ID", "Item Name", "Item Price" };
00038
00040   Long selectedDrinkID;
00042   Long selectedAppetizerID;
00044   Long selectedMainCourseID;
00045
00053   public AdminProducts() {
00054     initComponents();
00055
00056     // Initialize the Service Layer.
00057     ApiClient client = new ApiClient("http://localhost:8080");
00058     this.productService = new ProductService(client);
00059
00060     // Initialize table models and set shared headers.
00061     modelDrink = new DefaultTableModel();
00062     modelDrink.setColumnIdentifiers(columnNames);
00063     modelappetizers = new DefaultTableModel();
00064     modelappetizers.setColumnIdentifiers(columnNames);
00065     modelmaincourse = new DefaultTableModel();
00066     modelmaincourse.setColumnIdentifiers(columnNames);
00067
00068     // Link models to tables.
00069     jTable1.setModel(modelDrink);
00070     jTable2.setModel(modelappetizers);
00071     jTable3.setModel(modelmaincourse);
00072
00073     // Load initial data from backend.
00074     refreshAllTables();
00075   }
00076
00083   private void refreshAllTables() {
00084     loadDrinks();
00085     loadAppetizer();
00086     loadmainCourse();
00087   }
00088
00096   void loadDrinks() {
00097     new Thread(() -> {
00098       try {
00099         List<Drink> drinks = productService.getAllDrinks();
00100         SwingUtilities.invokeLater(() -> {
00101           modelDrink.setRowCount(0);
00102           for (Drink drink : drinks) {
00103             modelDrink.addRow(new Object[] {
00104                 drink.getDrinksId(),
00105                 drink.getItemDrinks(),
00106                 drink.getDrinksPrice()
00107             });
00108           }
00109         });
00110       } catch (Exception ex) {
00111         SwingUtilities
00112             .invokeLater(() -> JOptionPane.showMessageDialog(null, "Error loading drinks: " +
      ex.getMessage()));
00113       }
00114     }).start();
00115   }
00116
00123   void loadAppetizer() {
00124     new Thread(() -> {
00125       try {
00126         List<Appetizer> appetizers = productService.getAllAppetizers();
```

```
00127         SwingUtilities.invokeLater(() -> {
00128           modelappetizers.setRowCount(0);
00129           for (Appetizer appetizer : appetizers) {
00130             modelappetizers.addRow(new Object[] {
00131                 appetizer.getAppetizersId(),
00132                 appetizer.getItemAppetizers(),
00133                 appetizer.getAppetizersPrice()
00134             });
00135           }
00136         });
00137       } catch (Exception ex) {
00138         SwingUtilities
00139             .invokeLater(() -> JOptionPane.showMessageDialog(null, "Error loading appetizers: " +
      ex.getMessage()));
00140       }
00141     }).start();
00142   }
00143
00150   void loadmainCourse() {
00151     new Thread(() -> {
00152       try {
00153         List<MainCourse> mainCourses = productService.getAllMainCourses();
00154         SwingUtilities.invokeLater(() -> {
00155           modelmaincourse.setRowCount(0);
00156           for (MainCourse mainCourse : mainCourses) {
00157             modelmaincourse.addRow(new Object[] {
00158                 mainCourse.getFoodId(),
00159                 mainCourse.getItemFood(),
00160                 mainCourse.getFoodPrice()
00161             });
00162           }
00163         });
00164       } catch (Exception ex) {
00165         SwingUtilities
00166             .invokeLater(() -> JOptionPane.showMessageDialog(null, "Error loading main courses: " +
      ex.getMessage()));
00167       }
00168     }).start();
00169   }
00170
00179   @SuppressWarnings("unchecked")
00180   // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
00181   private void initComponents() {
00182
00183     jPanel1 = new javax.swing.JPanel();
00184     jLabel1 = new javax.swing.JLabel();
00185     jTabbedPane1 = new javax.swing.JTabbedPane();
00186     jPanel2 = new javax.swing.JPanel();
00187     itemname = new javax.swing.JTextField();
00188     itemprice = new javax.swing.JTextField();
00189     jScrollPane1 = new javax.swing.JScrollPane();
00190     jTable1 = new javax.swing.JTable();
00191     jButton1 = new javax.swing.JButton();
00192     jButton2 = new javax.swing.JButton();
00193     jLabel2 = new javax.swing.JLabel();
00194     jLabel3 = new javax.swing.JLabel();
00195     jLabel4 = new javax.swing.JLabel();
00196     jPanel3 = new javax.swing.JPanel();
00197     jLabel5 = new javax.swing.JLabel();
00198     jLabel6 = new javax.swing.JLabel();
00199     jLabel7 = new javax.swing.JLabel();
00200     jScrollPane2 = new javax.swing.JScrollPane();
00201     jTable2 = new javax.swing.JTable();
00202     itemname1 = new javax.swing.JTextField();
00203     itemprice1 = new javax.swing.JTextField();
00204     jButton4 = new javax.swing.JButton();
00205     jButton5 = new javax.swing.JButton();
00206     jPanel4 = new javax.swing.JPanel();
00207     jLabel8 = new javax.swing.JLabel();
00208     jLabel9 = new javax.swing.JLabel();
00209     jLabel10 = new javax.swing.JLabel();
00210     jScrollPane3 = new javax.swing.JScrollPane();
00211     jTable3 = new javax.swing.JTable();
00212     itemname2 = new javax.swing.JTextField();
00213     itemprice2 = new javax.swing.JTextField();
00214     jButton6 = new javax.swing.JButton();
00215     jButton7 = new javax.swing.JButton();
00216     jButton3 = new javax.swing.JButton();
00217
00218     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00219     setTitle("Admin");
00220     setResizable(false);
00221
00222     jPanel1.setBackground(new java.awt.Color(248, 244, 230));
00223
00224     jLabel1.setBackground(new java.awt.Color(255, 153, 0));
00225     jLabel1.setFont(new java.awt.Font("Yu Gothic UI", 1, 36)); // NOI18N
```

```
00226     jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00227     jLabel1.setText("Items Prices Update Portal");
00228
00229     jTabbedPane1.setBackground(new java.awt.Color(248, 244, 230));
00230     jTabbedPane1.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));
00231     jTabbedPane1.setCursor(new java.awt.Cursor(java.awt.Cursor.DEFAULT_CURSOR));
00232     jTabbedPane1.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00233
00234     jPanel2.setBackground(new java.awt.Color(248, 244, 230));
00235
00236     itemname.setFont(new java.awt.Font("Yu Gothic UI", 0, 18)); // NOI18N
00237     itemname.setHorizontalAlignment(javax.swing.JTextField.CENTER);
00238     itemname.setBorder(javax.swing.BorderFactory.createTitledBorder(
00239         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Item Name",
00240         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00241         new java.awt.Font("Yu Gothic UI", 0, 18))); // NOI18N
00242
00243     itemprice.setFont(new java.awt.Font("Yu Gothic UI", 0, 18)); // NOI18N
00244     itemprice.setHorizontalAlignment(javax.swing.JTextField.CENTER);
00245     itemprice.setBorder(javax.swing.BorderFactory.createTitledBorder(
00246         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Item Price",
00247         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00248         new java.awt.Font("Yu Gothic UI", 0, 18))); // NOI18N
00249
00250     jTable1.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00251     jTable1.setModel(new javax.swing.table.DefaultTableModel(
00252         new Object[][] {
00253             {},
00254             {},
00255             {},
00256             {}
00257         },
00258         new String[] {
00259
00260         }));
00261     jTable1.setInheritsPopupMenu(true);
00262     jTable1.getTableHeader().setResizingAllowed(false);
00263     jTable1.getTableHeader().setReorderingAllowed(false);
00264     jTable1.addMouseListener(new java.awt.event.MouseAdapter() {
00265       public void mouseClicked(java.awt.event.MouseEvent evt) {
00266         jTable1MouseClicked(evt);
00267       }
00268     });
00269     jTable1.addKeyListener(new java.awt.event.KeyAdapter() {
00270       public void keyPressed(java.awt.event.KeyEvent evt) {
00271         jTable1KeyPressed(evt);
00272       }
00273     });
00274     jScrollPane1.setViewportView(jTable1);
00275
00276     jButton1.setBackground(new java.awt.Color(255, 255, 255));
00277     jButton1.setFont(new java.awt.Font("Thonburi", 0, 18)); // NOI18N
00278     jButton1.setText("Update");
00279     jButton1.addActionListener(new java.awt.event.ActionListener() {
00280       public void actionPerformed(java.awt.event.ActionEvent evt) {
00281         jButton1ActionPerformed(evt);
00282       }
00283     });
00284
00285     jButton2.setBackground(new java.awt.Color(255, 255, 255));
00286     jButton2.setFont(new java.awt.Font("Thonburi", 0, 18)); // NOI18N
00287     jButton2.setText("Add");
00288     jButton2.addActionListener(new java.awt.event.ActionListener() {
00289       public void actionPerformed(java.awt.event.ActionEvent evt) {
00290         jButton2ActionPerformed(evt);
00291       }
00292     });
00293
00294     jLabel2.setBackground(new java.awt.Color(255, 153, 0));
00295     jLabel2.setFont(new java.awt.Font("Yu Gothic UI", 1, 25)); // NOI18N
00296     jLabel2.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00297     jLabel2.setText("Available Drinks");
00298
00299     jLabel3.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00300     jLabel3.setText("Enter new drink information carefully to add.");
00301
00302     jLabel4.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00303     jLabel4.setText("Please select the drink to update the price.");
00304
00305     javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);
00306     jPanel2.setLayout(jPanel2Layout);
00307     jPanel2Layout.setHorizontalGroup(
00308         jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00309             .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
      jPanel2Layout.createSequentialGroup()
00310                 .addGap(0, 27, Short.MAX_VALUE)
00311                 .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
```

```
00312                     .addGroup(jPanel2Layout.createSequentialGroup()
00313                         .addGap(24, 24, 24)
00314
    .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
00315                             .addGroup(jPanel2Layout.createSequentialGroup()
00316                                 .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 91,
00317                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00318                                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
    109,
00319                                     Short.MAX_VALUE)
00320                                 .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 103,
00321                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00322                                 .addGap(8, 8, 8))
00323                             .addComponent(itemprice)
00324                             .addComponent(itemname, javax.swing.GroupLayout.DEFAULT_SIZE, 311,
    Short.MAX_VALUE)
00325                             .addComponent(jLabel3, javax.swing.GroupLayout.DEFAULT_SIZE,
00326                                 javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
00327                         .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED,
00328                             javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
00329
    .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00330                             .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 586,
00331                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00332                             .addComponent(jLabel4, javax.swing.GroupLayout.PREFERRED_SIZE, 408,
00333                                 javax.swing.GroupLayout.PREFERRED_SIZE))
00334                         .addGap(115, 115, 115))
00335                     .addGroup(jPanel2Layout.createSequentialGroup()
00336                         .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE, 1111,
00337                             javax.swing.GroupLayout.PREFERRED_SIZE)
00338                         .addGap(18, 18, 18)))));
00339     jPanel2Layout.setVerticalGroup(
00340         jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00341             .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
    jPanel2Layout.createSequentialGroup()
00342                 .addContainerGap()
00343                 .addComponent(jLabel2)
00344                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 21,
    Short.MAX_VALUE)
00345
    .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00346                     .addComponent(jLabel3)
00347                     .addComponent(jLabel4))
00348                 .addGap(18, 18, 18)
00349                 .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00350                     .addGroup(jPanel2Layout.createSequentialGroup()
00351                         .addComponent(itemname, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00352                             javax.swing.GroupLayout.PREFERRED_SIZE)
00353                         .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
00354                         .addComponent(itemprice, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00355                             javax.swing.GroupLayout.PREFERRED_SIZE)
00356                         .addGap(37, 37, 37)
00357
    .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00358                             .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00359                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00360                             .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00361                                 javax.swing.GroupLayout.PREFERRED_SIZE)))
00362                     .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 203,
00363                         javax.swing.GroupLayout.PREFERRED_SIZE))
00364                 .addGap(48, 48, 48)));
00365
00366     jTabbedPane1.addTab("Drinks", jPanel2);
00367
00368     jPanel3.setBackground(new java.awt.Color(248, 244, 230));
00369
00370     jLabel5.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00371     jLabel5.setText("Enter new appetizer information carefully to add.");
00372
00373     jLabel6.setBackground(new java.awt.Color(255, 153, 0));
00374     jLabel6.setFont(new java.awt.Font("Yu Gothic UI", 1, 25)); // NOI18N
00375     jLabel6.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00376     jLabel6.setText("Available Appetizer");
00377
00378     jLabel7.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00379     jLabel7.setText("Please select the appetizer to update the price.");
00380
00381     jTable2.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00382     jTable2.setModel(new javax.swing.table.DefaultTableModel(
00383         new Object[][] {
00384             {},
00385             {},
00386             {},
00387             {}
00388         },
00389         new String[] {
00390
```

```
00391         }));
00392     jTable2.addMouseListener(new java.awt.event.MouseAdapter() {
00393       public void mouseClicked(java.awt.event.MouseEvent evt) {
00394         jTable2MouseClicked(evt);
00395       }
00396     });
00397     jScrollPane2.setViewportView(jTable2);
00398 
00399     itemname1.setFont(new java.awt.Font("Yu Gothic UI", 0, 18)); // NOI18N
00400     itemname1.setHorizontalAlignment(javax.swing.JTextField.CENTER);
00401     itemname1.setBorder(javax.swing.BorderFactory.createTitledBorder(
00402         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Item Name",
00403         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00404         new java.awt.Font("Yu Gothic UI", 0, 18))); // NOI18N
00405 
00406     itemprice1.setFont(new java.awt.Font("Yu Gothic UI", 0, 18)); // NOI18N
00407     itemprice1.setHorizontalAlignment(javax.swing.JTextField.CENTER);
00408     itemprice1.setBorder(javax.swing.BorderFactory.createTitledBorder(
00409         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Item Price",
00410         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00411         new java.awt.Font("Yu Gothic UI", 0, 18))); // NOI18N
00412 
00413     jButton4.setBackground(new java.awt.Color(255, 255, 255));
00414     jButton4.setFont(new java.awt.Font("Thonburi", 0, 18)); // NOI18N
00415     jButton4.setText("Update");
00416     jButton4.addActionListener(new java.awt.event.ActionListener() {
00417       public void actionPerformed(java.awt.event.ActionEvent evt) {
00418         jButton4ActionPerformed(evt);
00419       }
00420     });
00421 
00422     jButton5.setBackground(new java.awt.Color(255, 255, 255));
00423     jButton5.setFont(new java.awt.Font("Thonburi", 0, 18)); // NOI18N
00424     jButton5.setText("Add");
00425     jButton5.addActionListener(new java.awt.event.ActionListener() {
00426       public void actionPerformed(java.awt.event.ActionEvent evt) {
00427         jButton5ActionPerformed(evt);
00428       }
00429     });
00430 
00431     javax.swing.GroupLayout jPanel3Layout = new javax.swing.GroupLayout(jPanel3);
00432     jPanel3.setLayout(jPanel3Layout);
00433     jPanel3Layout.setHorizontalGroup(
00434         jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00435             .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
    jPanel3Layout.createSequentialGroup()
00436                 .addGap(0, 27, Short.MAX_VALUE)
00437                 .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00438                     .addGroup(jPanel3Layout.createSequentialGroup()
00439                         .addGap(24, 24, 24)
00440 
    .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
00441                             .addGroup(jPanel3Layout.createSequentialGroup()
00442                                 .addComponent(jButton5, javax.swing.GroupLayout.PREFERRED_SIZE, 91,
00443                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00444                                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
00445                                     javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
00446                                 .addComponent(jButton4, javax.swing.GroupLayout.PREFERRED_SIZE, 103,
00447                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00448                                 .addGap(8, 8, 8))
00449                             .addComponent(itemprice1)
00450                             .addComponent(itemname1)
00451                             .addComponent(jLabel5, javax.swing.GroupLayout.DEFAULT_SIZE,
00452                                 javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
00453                         .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED,
00454                             javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
00455 
    .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00456                             .addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED_SIZE, 586,
00457                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00458                             .addComponent(jLabel7, javax.swing.GroupLayout.PREFERRED_SIZE, 408,
00459                                 javax.swing.GroupLayout.PREFERRED_SIZE))
00460                         .addGap(115, 115, 115))
00461                     .addGroup(jPanel3Layout.createSequentialGroup()
00462                         .addComponent(jLabel6, javax.swing.GroupLayout.PREFERRED_SIZE, 1111,
00463                             javax.swing.GroupLayout.PREFERRED_SIZE)
00464                         .addGap(18, 18, 18)))));
00465     jPanel3Layout.setVerticalGroup(
00466         jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00467             .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
    jPanel3Layout.createSequentialGroup()
00468                 .addContainerGap()
00469                 .addComponent(jLabel6)
00470                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 21,
    Short.MAX_VALUE)
00471 
    .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
```

```
00472                     .addComponent(jLabel5)
00473                     .addComponent(jLabel7))
00474                 .addGap(18, 18, 18)
00475                 .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00476                     .addGroup(jPanel3Layout.createSequentialGroup()
00477                         .addComponent(itemname1, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00478                             javax.swing.GroupLayout.PREFERRED_SIZE)
00479                         .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
00480                         .addComponent(itemprice1, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00481                             javax.swing.GroupLayout.PREFERRED_SIZE)
00482                         .addGap(37, 37, 37)
00483
    .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00484                         .addComponent(jButton4, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00485                             javax.swing.GroupLayout.PREFERRED_SIZE)
00486                         .addComponent(jButton5, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00487                             javax.swing.GroupLayout.PREFERRED_SIZE)))
00488                     .addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED_SIZE, 203,
00489                         javax.swing.GroupLayout.PREFERRED_SIZE))
00490                 .addGap(48, 48, 48)));
00491
00492     jTabbedPane1.addTab("Appetizers", jPanel3);
00493
00494     jPanel4.setBackground(new java.awt.Color(248, 244, 230));
00495     jPanel4.setFont(new java.awt.Font("Yu Gothic UI", 0, 18)); // NOI18N
00496
00497     jLabel8.setBackground(new java.awt.Color(248, 244, 230));
00498     jLabel8.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00499     jLabel8.setText("Enter new item information carefully to add.");
00500
00501     jLabel9.setBackground(new java.awt.Color(255, 153, 0));
00502     jLabel9.setFont(new java.awt.Font("Yu Gothic UI", 1, 25)); // NOI18N
00503     jLabel9.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00504     jLabel9.setText("Avaliable MainCourse");
00505
00506     jLabel10.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00507     jLabel10.setText("Please select the item to update the price.");
00508
00509     jTable3.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00510     jTable3.setModel(new javax.swing.table.DefaultTableModel(
00511         new Object[][] {
00512             {},
00513             {},
00514             {},
00515             {}
00516         },
00517         new String[] {
00518
00519         }));
00520     jTable3.addMouseListener(new java.awt.event.MouseAdapter() {
00521       public void mouseClicked(java.awt.event.MouseEvent evt) {
00522         jTable3MouseClicked(evt);
00523       }
00524     });
00525     jScrollPane3.setViewportView(jTable3);
00526
00527     itemname2.setFont(new java.awt.Font("Yu Gothic UI", 0, 18)); // NOI18N
00528     itemname2.setHorizontalAlignment(javax.swing.JTextField.CENTER);
00529     itemname2.setBorder(javax.swing.BorderFactory.createTitledBorder(
00530         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Item Name",
00531         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00532         new java.awt.Font("Yu Gothic UI", 0, 18))); // NOI18N
00533
00534     itemprice2.setFont(new java.awt.Font("Yu Gothic UI", 0, 18)); // NOI18N
00535     itemprice2.setHorizontalAlignment(javax.swing.JTextField.CENTER);
00536     itemprice2.setBorder(javax.swing.BorderFactory.createTitledBorder(
00537         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Item Price",
00538         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00539         new java.awt.Font("Yu Gothic UI", 0, 18))); // NOI18N
00540
00541     jButton6.setBackground(new java.awt.Color(255, 255, 255));
00542     jButton6.setFont(new java.awt.Font("Thonburi", 0, 18)); // NOI18N
00543     jButton6.setText("Update");
00544     jButton6.addActionListener(new java.awt.event.ActionListener() {
00545       public void actionPerformed(java.awt.event.ActionEvent evt) {
00546         jButton6ActionPerformed(evt);
00547       }
00548     });
00549
00550     jButton7.setBackground(new java.awt.Color(255, 255, 255));
00551     jButton7.setFont(new java.awt.Font("Thonburi", 0, 18)); // NOI18N
00552     jButton7.setText("Add");
00553     jButton7.addActionListener(new java.awt.event.ActionListener() {
00554       public void actionPerformed(java.awt.event.ActionEvent evt) {
00555         jButton7ActionPerformed(evt);
00556       }
00557     });
```

```
00558
00559     javax.swing.GroupLayout jPanel4Layout = new javax.swing.GroupLayout(jPanel4);
00560     jPanel4.setLayout(jPanel4Layout);
00561     jPanel4Layout.setHorizontalGroup(
00562         jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00563             .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
      jPanel4Layout.createSequentialGroup()
00564                 .addGap(0, 27, Short.MAX_VALUE)
00565                 .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00566                     .addGroup(jPanel4Layout.createSequentialGroup()
00567                         .addGap(24, 24, 24)
00568
      .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
00569                             .addGroup(jPanel4Layout.createSequentialGroup()
00570                                 .addComponent(jButton7, javax.swing.GroupLayout.PREFERRED_SIZE, 91,
00571                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00572                                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
      109,
00573                                     Short.MAX_VALUE)
00574                                 .addComponent(jButton6, javax.swing.GroupLayout.PREFERRED_SIZE, 103,
00575                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00576                                 .addGap(8, 8, 8))
00577                             .addComponent(itemprice2)
00578                             .addComponent(itemname2, javax.swing.GroupLayout.DEFAULT_SIZE, 311,
      Short.MAX_VALUE)
00579                             .addComponent(jLabel8, javax.swing.GroupLayout.DEFAULT_SIZE,
00580                                 javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
00581                         .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED,
00582                             javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
00583
      .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00584                             .addComponent(jScrollPane3, javax.swing.GroupLayout.PREFERRED_SIZE, 586,
00585                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00586                             .addComponent(jLabel10, javax.swing.GroupLayout.PREFERRED_SIZE, 408,
00587                                 javax.swing.GroupLayout.PREFERRED_SIZE))
00588                         .addGap(115, 115, 115))
00589                     .addGroup(jPanel4Layout.createSequentialGroup()
00590                         .addComponent(jLabel9, javax.swing.GroupLayout.PREFERRED_SIZE, 1111,
00591                             javax.swing.GroupLayout.PREFERRED_SIZE)
00592                         .addGap(18, 18, 18)))));
00593     jPanel4Layout.setVerticalGroup(
00594         jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00595             .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
      jPanel4Layout.createSequentialGroup()
00596                 .addContainerGap()
00597                 .addComponent(jLabel9)
00598                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 21,
      Short.MAX_VALUE)
00599
      .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00600                     .addComponent(jLabel8)
00601                     .addComponent(jLabel10))
00602                 .addGap(18, 18, 18)
00603                 .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00604                     .addGroup(jPanel4Layout.createSequentialGroup()
00605                         .addComponent(itemname2, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00606                             javax.swing.GroupLayout.PREFERRED_SIZE)
00607                         .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
00608                         .addComponent(itemprice2, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00609                             javax.swing.GroupLayout.PREFERRED_SIZE)
00610                         .addGap(37, 37, 37)
00611
      .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00612                             .addComponent(jButton6, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00613                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00614                             .addComponent(jButton7, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00615                                 javax.swing.GroupLayout.PREFERRED_SIZE)))
00616                     .addComponent(jScrollPane3, javax.swing.GroupLayout.PREFERRED_SIZE, 203,
00617                         javax.swing.GroupLayout.PREFERRED_SIZE))
00618                 .addGap(48, 48, 48)));
00619
00620     jTabbedPane1.addTab("MainCourse", jPanel4);
00621
00622     jButton3.setBackground(new java.awt.Color(255, 255, 255));
00623     jButton3.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00624     jButton3.setText("Go Back");
00625     jButton3.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));
00626     jButton3.addActionListener(new java.awt.event.ActionListener() {
00627       public void actionPerformed(java.awt.event.ActionEvent evt) {
00628         jButton3ActionPerformed(evt);
00629       }
00630     });
00631
00632     javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
00633     jPanel1.setLayout(jPanel1Layout);
00634     jPanel1Layout.setHorizontalGroup(
00635         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
```

```
00636             .addGroup(jPanel1Layout.createSequentialGroup()
00637                 .addGap(25, 25, 25)
00638                 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00639                     .addGroup(jPanel1Layout.createSequentialGroup()
00640                         .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 122,
00641                             javax.swing.GroupLayout.PREFERRED_SIZE)
00642                         .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
00643                             javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
00644                         .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 689,
00645                             javax.swing.GroupLayout.PREFERRED_SIZE)
00646                         .addGap(218, 218, 218))
00647                     .addGroup(jPanel1Layout.createSequentialGroup()
00648                         .addComponent(jTabbedPane1, javax.swing.GroupLayout.PREFERRED_SIZE,
00649                             javax.swing.GroupLayout.DEFAULT_SIZE,
      javax.swing.GroupLayout.PREFERRED_SIZE)
00650                         .addContainerGap(29, Short.MAX_VALUE)))));
00651     jPanel1Layout.setVerticalGroup(
00652         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00653             .addGroup(jPanel1Layout.createSequentialGroup()
00654                 .addContainerGap()
00655
      .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
00656                     .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 60,
00657                         javax.swing.GroupLayout.PREFERRED_SIZE)
00658                     .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 37,
00659                         javax.swing.GroupLayout.PREFERRED_SIZE))
00660                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED,
00661                     javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
00662                 .addComponent(jTabbedPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 392,
00663                     javax.swing.GroupLayout.PREFERRED_SIZE)
00664                 .addGap(29, 29, 29)));
00665
00666     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00667     getContentPane().setLayout(layout);
00668     layout.setHorizontalGroup(
00669         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00670             .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00671                 Short.MAX_VALUE));
00672     layout.setVerticalGroup(
00673         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00674             .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
00675                 .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00676                     javax.swing.GroupLayout.PREFERRED_SIZE)
00677                 .addGap(0, 0, Short.MAX_VALUE)));
00678
00679     pack();
00680     setLocationRelativeTo(null);
00681   }// </editor-fold>//GEN-END:initComponents
00682
00693   private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {//
      GEN-FIRST:event_jButton2ActionPerformed
00694     String name = itemname.getText();
00695     String price = itemprice.getText();
00696
00697     new Thread(() -> {
00698       try {
00699         productService.addDrink(name, price);
00700         SwingUtilities.invokeLater(() -> {
00701           JOptionPane.showMessageDialog(this, "Drink Added Successfully.");
00702           itemname.setText("");
00703           itemprice.setText("");
00704           loadDrinks();
00705         });
00706       } catch (Exception ex) {
00707         SwingUtilities.invokeLater(() -> JOptionPane.showMessageDialog(this, "Error adding drink: " +
      ex.getMessage()));
00708       }
00709     }).start();
00710   }// GEN-LAST:event_jButton2ActionPerformed
00711
00720   private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {//
      GEN-FIRST:event_jButton3ActionPerformed
00721     new AdminLogin().setVisible(true);
00722     this.dispose();
00723   }// GEN-LAST:event_jButton3ActionPerformed
00724
00733   private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {//
      GEN-FIRST:event_jButton5ActionPerformed
00734     String name = itemname1.getText();
00735     String price = itemprice1.getText();
00736
00737     new Thread(() -> {
00738       try {
00739         productService.addAppetizer(name, price);
00740         SwingUtilities.invokeLater(() -> {
```

```
00741           JOptionPane.showMessageDialog(this, "Appetizer Added Successfully.");
00742           itemname1.setText("");
00743           itemprice1.setText("");
00744           loadAppetizer();
00745         });
00746       } catch (Exception ex) {
00747         SwingUtilities
00748             .invokeLater(() -> JOptionPane.showMessageDialog(this, "Error adding appetizer: " +
      ex.getMessage()));
00749       }
00750     }).start();
00751   }// GEN-LAST:event_jButton5ActionPerformed
00752
00761   private void jButton7ActionPerformed(java.awt.event.ActionEvent evt) {//
      GEN-FIRST:event_jButton7ActionPerformed
00762     String name = itemname2.getText();
00763     String price = itemprice2.getText();
00764
00765     new Thread(() -> {
00766       try {
00767         productService.addMainCourse(name, price);
00768         SwingUtilities.invokeLater(() -> {
00769           JOptionPane.showMessageDialog(this, "Main Course Added Successfully.");
00770           itemname2.setText("");
00771           itemprice2.setText("");
00772           loadmainCourse();
00773         });
00774       } catch (Exception ex) {
00775         SwingUtilities
00776             .invokeLater(() -> JOptionPane.showMessageDialog(this, "Error adding main course: " +
      ex.getMessage()));
00777       }
00778     }).start();
00779   }// GEN-LAST:event_jButton7ActionPerformed
00780
00789   private void jTable1KeyPressed(java.awt.event.KeyEvent evt) {// GEN-FIRST:event_jTable1KeyPressed
00790     // No action needed for key press in this version.
00791   }// GEN-LAST:event_jTable1KeyPressed
00792
00803   private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {//
      GEN-FIRST:event_jTable1MouseClicked
00804     int row = jTable1.getSelectedRow();
00805     if (row == -1)
00806       return;
00807
00808     String idStr = jTable1.getValueAt(row, 0).toString();
00809     selectedDrinkID = Long.valueOf(idStr);
00810     System.out.println("Selected Drink ID: " + selectedDrinkID);
00811
00812     new Thread(() -> {
00813       try {
00814         Drink drink = productService.getDrinkById(selectedDrinkID);
00815         SwingUtilities.invokeLater(() -> {
00816           itemname.setText(drink.getItemDrinks());
00817           itemprice.setText(String.valueOf(drink.getDrinksPrice()));
00818         });
00819       } catch (Exception ex) {
00820         SwingUtilities
00821             .invokeLater(() -> JOptionPane.showMessageDialog(null, "Error loading drink details: " +
      ex.getMessage()));
00822       }
00823     }).start();
00824   }// GEN-LAST:event_jTable1MouseClicked
00825
00834   private void jTable2MouseClicked(java.awt.event.MouseEvent evt) {//
      GEN-FIRST:event_jTable2MouseClicked
00835     int row = jTable2.getSelectedRow();
00836     if (row == -1)
00837       return;
00838
00839     String idStr = jTable2.getValueAt(row, 0).toString();
00840     selectedAppetizerID = Long.valueOf(idStr);
00841     System.out.println("Selected Appetizer ID: " + selectedAppetizerID);
00842
00843     new Thread(() -> {
00844       try {
00845         Appetizer appetizer = productService.getAppetizerById(selectedAppetizerID);
00846         SwingUtilities.invokeLater(() -> {
00847           itemname1.setText(appetizer.getItemAppetizers());
00848           itemprice1.setText(String.valueOf(appetizer.getAppetizersPrice()));
00849         });
00850       } catch (Exception ex) {
00851         SwingUtilities.invokeLater(
00852             () -> JOptionPane.showMessageDialog(null, "Error loading appetizer details: " +
      ex.getMessage()));
00853       }
00854     }).start();
```

```
00855   }// GEN-LAST:event_jTable2MouseClicked
00856
00865   private void jTable3MouseClicked(java.awt.event.MouseEvent evt) {//
      GEN-FIRST:event_jTable3MouseClicked
00866     int row = jTable3.getSelectedRow();
00867     if (row == -1)
00868       return;
00869
00870     String idStr = jTable3.getValueAt(row, 0).toString();
00871     selectedMainCourseID = Long.valueOf(idStr);
00872     System.out.println("Selected Main Course ID: " + selectedMainCourseID);
00873
00874     new Thread(() -> {
00875       try {
00876         MainCourse mainCourse = productService.getMainCourseById(selectedMainCourseID);
00877         SwingUtilities.invokeLater(() -> {
00878           itemname2.setText(mainCourse.getItemFood());
00879           itemprice2.setText(String.valueOf(mainCourse.getFoodPrice()));
00880         });
00881       } catch (Exception ex) {
00882         SwingUtilities.invokeLater(
00883             () -> JOptionPane.showMessageDialog(null, "Error loading main course details: " +
      ex.getMessage()));
00884       }
00885     }).start();
00886   }// GEN-LAST:event_jTable3MouseClicked
00887
00896   private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//
      GEN-FIRST:event_jButton1ActionPerformed
00897     String name = itemname.getText();
00898     String price = itemprice.getText();
00899
00900     new Thread(() -> {
00901       try {
00902         productService.updateDrink(selectedDrinkID, name, price);
00903         SwingUtilities.invokeLater(() -> {
00904           JOptionPane.showMessageDialog(this, "Drink updated successfully.");
00905           itemname.setText("");
00906           itemprice.setText("");
00907           selectedDrinkID = null;
00908           loadDrinks();
00909         });
00910       } catch (Exception ex) {
00911         SwingUtilities
00912             .invokeLater(() -> JOptionPane.showMessageDialog(this, "Error updating drink: " +
      ex.getMessage()));
00913       }
00914     }).start();
00915   }// GEN-LAST:event_jButton1ActionPerformed
00916
00925   private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {//
      GEN-FIRST:event_jButton4ActionPerformed
00926     String name = itemname1.getText();
00927     String price = itemprice1.getText();
00928
00929     new Thread(() -> {
00930       try {
00931         productService.updateAppetizer(selectedAppetizerID, name, price);
00932         SwingUtilities.invokeLater(() -> {
00933           JOptionPane.showMessageDialog(this, "Appetizer updated successfully.");
00934           itemname1.setText("");
00935           itemprice1.setText("");
00936           selectedAppetizerID = null;
00937           loadAppetizer();
00938         });
00939       } catch (Exception ex) {
00940         SwingUtilities
00941             .invokeLater(() -> JOptionPane.showMessageDialog(this, "Error updating appetizer: " +
      ex.getMessage()));
00942       }
00943     }).start();
00944   }// GEN-LAST:event_jButton4ActionPerformed
00945
00954   private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {//
      GEN-FIRST:event_jButton6ActionPerformed
00955     String name = itemname2.getText();
00956     String price = itemprice2.getText();
00957
00958     new Thread(() -> {
00959       try {
00960         productService.updateMainCourse(selectedMainCourseID, name, price);
00961         SwingUtilities.invokeLater(() -> {
00962           JOptionPane.showMessageDialog(this, "Main course updated successfully.");
00963           itemname2.setText("");
00964           itemprice2.setText("");
00965           selectedMainCourseID = null;
00966           loadmainCourse();
```

```
00967         });
00968       } catch (Exception ex) {
00969         SwingUtilities
00970             .invokeLater(() -> JOptionPane.showMessageDialog(this, "Error updating main course: " +
      ex.getMessage()));
00971       }
00972     }).start();
00973   }// GEN-LAST:event_jButton6ActionPerformed
00974
00984   public static void main(String args[]) {
00985     try {
00986       for (javax.swing.UIManager.LookAndFeelInfo info :
      javax.swing.UIManager.getInstalledLookAndFeels()) {
00987         if ("Nimbus".equals(info.getName())) {
00988           javax.swing.UIManager.setLookAndFeel(info.getClassName());
00989           break;
00990         }
00991       }
00992     } catch (ClassNotFoundException ex) {
00993
      java.util.logging.Logger.getLogger(AdminProducts.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
00994     } catch (InstantiationException ex) {
00995
      java.util.logging.Logger.getLogger(AdminProducts.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
00996     } catch (IllegalAccessException ex) {
00997
      java.util.logging.Logger.getLogger(AdminProducts.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
00998     } catch (javax.swing.UnsupportedLookAndFeelException ex) {
00999
      java.util.logging.Logger.getLogger(AdminProducts.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
01000     }
01001     // </editor-fold>
01002
01003     /* Create and display the form */
01004     java.awt.EventQueue.invokeLater(new Runnable() {
01005       public void run() {
01006         new AdminProducts().setVisible(true);
01007       }
01008     });
01009   }
01010
01011   // Variables declaration - do not modify//GEN-BEGIN:variables
01013   private javax.swing.JTextField itemname;
01015   private javax.swing.JTextField itemname1;
01017   private javax.swing.JTextField itemname2;
01019   private javax.swing.JTextField itemprice;
01021   private javax.swing.JTextField itemprice1;
01023   private javax.swing.JTextField itemprice2;
01025   private javax.swing.JButton jButton1;
01027   private javax.swing.JButton jButton2;
01029   private javax.swing.JButton jButton3;
01031   private javax.swing.JButton jButton4;
01033   private javax.swing.JButton jButton5;
01035   private javax.swing.JButton jButton6;
01037   private javax.swing.JButton jButton7;
01039   private javax.swing.JLabel jLabel1;
01041   private javax.swing.JLabel jLabel10;
01043   private javax.swing.JLabel jLabel2;
01045   private javax.swing.JLabel jLabel3;
01047   private javax.swing.JLabel jLabel4;
01049   private javax.swing.JLabel jLabel5;
01051   private javax.swing.JLabel jLabel6;
01053   private javax.swing.JLabel jLabel7;
01055   private javax.swing.JLabel jLabel8;
01057   private javax.swing.JLabel jLabel9;
01059   private javax.swing.JPanel jPanel1;
01061   private javax.swing.JPanel jPanel2;
01063   private javax.swing.JPanel jPanel3;
01065   private javax.swing.JPanel jPanel4;
01067   private javax.swing.JScrollPane jScrollPane1;
01069   private javax.swing.JScrollPane jScrollPane2;
01071   private javax.swing.JScrollPane jScrollPane3;
01073   private javax.swing.JTabbedPane jTabbedPane1;
01075   private javax.swing.JTable jTable1;
01077   private javax.swing.JTable jTable2;
01079   private javax.swing.JTable jTable3;
01080   // End of variables declaration//GEN-END:variables
01081 }
```

8.50 ApplicationLauncher.java File Reference

Classes

- class es.ull.esit.app.ApplicationLauncher

  Auxiliary entry point used to start the application's UI.

Packages

- package es.ull.esit.app

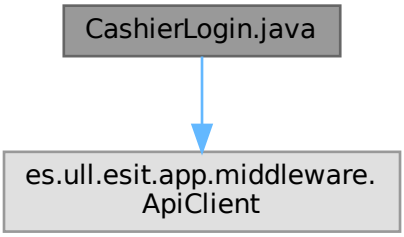
8.51 ApplicationLauncher.java

Go to the documentation of this file.

```
00001 package es.ull.esit.app;
00002
00010 public class ApplicationLauncher {
00011
00019   public static void main(String[] args) {
00020     new Login().setVisible(true);
00021   }
00022 }
```

8.52 CashierLogin.java File Reference

`import es.ull.esit.app.middleware.ApiClient;`
Include dependency graph for CashierLogin.java:



Classes

- class es.ull.esit.app.CashierLogin

  Login window for authenticating cashiers.

Packages

- package es.ull.esit.app

8.53 CashierLogin.java

Go to the documentation of this file.

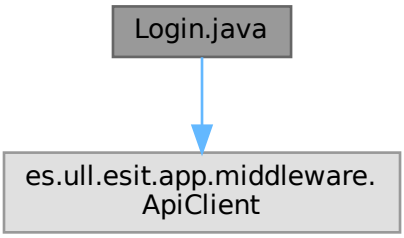
```
00001 package es.ull.esit.app;
00002
00003 import es.ull.esit.app.middleware.ApiClient;
00004 import es.ull.esit.app.middleware.service.ReportService;
00005 import java.util.List;
00006 import javax.swing.SwingUtilities;
00007
00023 public class CashierLogin extends javax.swing.JFrame {
00024
00026   private final ReportService reportService;
00027
00035   public CashierLogin() {
00036     this((String) null);
00037   }
00038
00049   public CashierLogin(String name) {
00050     initComponents();
00051
00052     ApiClient client = new ApiClient("http://localhost:8080");
00053     this.reportService = new ReportService(client);
00054
00055     updateWelcomeMessage(name);
00056   }
00057
00066   private void updateWelcomeMessage(String name) {
00067     if (name == null || name.trim().isEmpty()) {
00068       welcomeTxt.setText("Welcome Cashier");
00069     } else {
00070       welcomeTxt.setText("Welcome " + name);
00071     }
00072   }
00073
00090   private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
00091     new Thread(() -> {
00092       try {
00093         List<es.ull.esit.app.middleware.model.Cashier> cashiers = reportService.getCashierInfo();
00094         System.out.println("Found " + cashiers.size() + " cashiers in DB.");
00095
00096       } catch (Exception ex) {
00097         System.err.println("Warning: Could not fetch cashier stats: " + ex.getMessage());
00098       }
00099       SwingUtilities.invokeLater(() -> {
00100         new AboutUs().setVisible(true);
00101         dispose();
00102       });
00103     }).start();
00104   }
00105
00123   private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
00124     new Thread(() -> {
00125       try {
00126         String status = reportService.checkMenuStatus();
00127         System.out.println(status);
00128
00129       } catch (Exception ex) {
00130         System.err.println("Error checking menu status: " + ex.getMessage());
00131       }
00132
00133       SwingUtilities.invokeLater(() -> {
00134         new Order().setVisible(true);
00135         dispose();
00136       });
00137     }).start();
00138   }
00139
00153   private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
00154     new Login().setVisible(true);
00155     dispose();
00156   }
00157
00168   public static void main(String args[]) {
00169     try {
00170       for (javax.swing.UIManager.LookAndFeelInfo info :
      javax.swing.UIManager.getInstalledLookAndFeels()) {
00171         if ("Nimbus".equals(info.getName())) {
00172           javax.swing.UIManager.setLookAndFeel(info.getClassName());
00173           break;
00174         }
00175       }
00176     } catch (ClassNotFoundException | InstantiationException | IllegalAccessException
00177         | javax.swing.UnsupportedLookAndFeelException ex) {
```

```
00178
      java.util.logging.Logger.getLogger(CashierLogin.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
00179     }
00180
00181     java.awt.EventQueue.invokeLater(() -> new CashierLogin().setVisible(true));
00182   }
00183
00192   @SuppressWarnings("unchecked")
00193   // <editor-fold defaultstate="collapsed" desc="Generated
00194   // Code">//GEN-BEGIN:initComponents
00195   private void initComponents() {
00196
00197     jPanel1 = new javax.swing.JPanel();
00198     welcomeTxt = new javax.swing.JLabel();
00199     jButton1 = new javax.swing.JButton();
00200     jButton2 = new javax.swing.JButton();
00201     jLabel1 = new javax.swing.JLabel();
00202     jButton3 = new javax.swing.JButton();
00203
00204     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00205     setTitle("Cashier");
00206     setResizable(false);
00207
00208     jPanel1.setBackground(new java.awt.Color(248, 244, 230));
00209
00210     welcomeTxt.setFont(new java.awt.Font("Yu Gothic UI", 1, 24)); // NOI18N
00211     welcomeTxt.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00212     welcomeTxt.setText("Welcome Cashier");
00213
00214     jButton1.setBackground(new java.awt.Color(153, 153, 153));
00215     jButton1.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00216     jButton1.setText("About us");
00217     jButton1.addActionListener(new java.awt.event.ActionListener() {
00218       public void actionPerformed(java.awt.event.ActionEvent evt) {
00219         jButton1ActionPerformed(evt);
00220       }
00221     });
00222
00223     jButton2.setBackground(new java.awt.Color(153, 153, 153));
00224     jButton2.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00225     jButton2.setText("Menu");
00226     jButton2.addActionListener(new java.awt.event.ActionListener() {
00227       public void actionPerformed(java.awt.event.ActionEvent evt) {
00228         jButton2ActionPerformed(evt);
00229       }
00230     });
00231
00232     jLabel1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/ui/screenshot.png"))); //
      NOI18N
00233
00234     jButton3.setBackground(new java.awt.Color(255, 255, 255));
00235     jButton3.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00236     jButton3.setText("LogOut");
00237     jButton3.addActionListener(new java.awt.event.ActionListener() {
00238       public void actionPerformed(java.awt.event.ActionEvent evt) {
00239         jButton3ActionPerformed(evt);
00240       }
00241     });
00242
00243     javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
00244     jPanel1.setLayout(jPanel1Layout);
00245     jPanel1Layout.setHorizontalGroup(
00246         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00247             .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
      jPanel1Layout.createSequentialGroup()
00248                 .addContainerGap(109, Short.MAX_VALUE)
00249                 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00250                     .addComponent(welcomeTxt, javax.swing.GroupLayout.PREFERRED_SIZE, 299,
00251                         javax.swing.GroupLayout.PREFERRED_SIZE)
00252
      .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00253                         .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
      jPanel1Layout.createSequentialGroup()
00254                             .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 320,
00255                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00256                             .addGap(98, 98, 98))
00257                         .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
      jPanel1Layout.createSequentialGroup()
00258                             .addComponent(jLabel1)
00259                             .addGap(183, 183, 183))
00260                         .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
      jPanel1Layout.createSequentialGroup()
00261                             .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 141,
00262                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00263                             .addGap(190, 190, 190)))))
00264             .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
```

```
00265                 .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
      jPanel1Layout.createSequentialGroup()
00266                     .addContainerGap(108, Short.MAX_VALUE)
00267                     .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 320,
00268                         javax.swing.GroupLayout.PREFERRED_SIZE)
00269                     .addGap(99, 99, 99))));
00270     jPanel1Layout.setVerticalGroup(
00271         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00272             .addGroup(jPanel1Layout.createSequentialGroup()
00273                 .addGap(45, 45, 45)
00274                 .addComponent(welcomeTxt, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
00275                     javax.swing.GroupLayout.PREFERRED_SIZE)
00276                 .addGap(18, 18, 18)
00277                 .addComponent(jLabel1)
00278                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 99,
      Short.MAX_VALUE)
00279                 .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00280                     javax.swing.GroupLayout.PREFERRED_SIZE)
00281                 .addGap(57, 57, 57)
00282                 .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00283                     javax.swing.GroupLayout.PREFERRED_SIZE)
00284                 .addGap(68, 68, 68))
00285             .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00286                 .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
      jPanel1Layout.createSequentialGroup()
00287                     .addContainerGap(290, Short.MAX_VALUE)
00288                     .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00289                         javax.swing.GroupLayout.PREFERRED_SIZE)
00290                     .addGap(242, 242, 242))));
00291 
00292     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00293     getContentPane().setLayout(layout);
00294     layout.setHorizontalGroup(
00295         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00296             .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00297                 Short.MAX_VALUE));
00298     layout.setVerticalGroup(
00299         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00300             .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00301                 Short.MAX_VALUE));
00302 
00303     pack();
00304     setLocationRelativeTo(null);
00305   }// </editor-fold>//GEN-END:initComponents
00306 
00307   // Variables declaration - do not modify//GEN-BEGIN:variables
00309   private javax.swing.JButton jButton1;
00311   private javax.swing.JButton jButton2;
00313   private javax.swing.JButton jButton3;
00315   private javax.swing.JLabel jLabel1;
00317   private javax.swing.JPanel jPanel1;
00319   private javax.swing.JLabel welcomeTxt;
00320   // End of variables declaration//GEN-END:variables
00321 }
```

8.54 Login.java File Reference

`import es.ull.esit.app.middleware.ApiClient;`
Include dependency graph for Login.java:



Classes

- class es.ull.esit.app.Login

  Login window for the Restaurant System.

Packages

- package es.ull.esit.app

8.55 Login.java

Go to the documentation of this file.

```
00001 package es.ull.esit.app;
00002
00003 import es.ull.esit.app.middleware.ApiClient;
00004 import es.ull.esit.app.middleware.model.User;
00005 import es.ull.esit.app.middleware.service.AuthService;
00006 import javax.swing.JOptionPane;
00007 import javax.swing.SwingUtilities;
00008
00023 public class Login extends javax.swing.JFrame {
00024
00026   private final AuthService authService;
00027
00037   public Login() {
00038     initComponents();
00039     ApiClient client = new ApiClient("http://localhost:8080");
00040     this.authService = new AuthService(client);
00041   }
00042
00049   @SuppressWarnings("unchecked")
00050   // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
00051   private void initComponents() {
00052
00053     jPanel1 = new javax.swing.JPanel();
00054     jLabel1 = new javax.swing.JLabel();
00055     jLabel3 = new javax.swing.JLabel();
00056     usernametxt = new javax.swing.JTextField();
00057     usertypecmbo = new javax.swing.JComboBox<>();
00058     jButton1 = new javax.swing.JButton();
00059     jPasswordField1 = new javax.swing.JPasswordField();
```

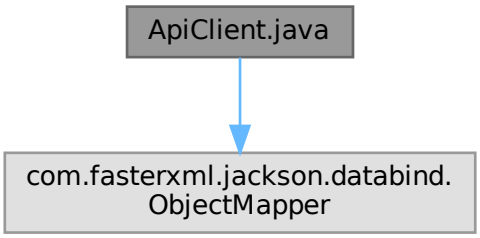
```
00060
00061     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00062     setTitle("Login");
00063     setResizable(false);
00064
00065     jPanel1.setBackground(new java.awt.Color(248, 244, 230));
00066
00067     jLabel1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/ui/screenshot.png"))); //
      NOI18N
00068
00069     jLabel3.setFont(new java.awt.Font("Yu Gothic UI", 1, 24)); // NOI18N
00070     jLabel3.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00071     jLabel3.setText("User Login ");
00072
00073     usernametxt.setFont(new java.awt.Font("Yu Gothic UI", 0, 18)); // NOI18N
00074     usernametxt.setHorizontalAlignment(javax.swing.JTextField.CENTER);
00075     usernametxt.setBorder(javax.swing.BorderFactory.createTitledBorder(
00076         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Username",
00077         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00078         new java.awt.Font("Yu Gothic UI", 0, 18))); // NOI18N
00079
00080     usertypecmbo.setEditable(true);
00081     usertypecmbo.setFont(new java.awt.Font("Yu Gothic UI", 0, 18)); // NOI18N
00082     usertypecmbo.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] { "admin", "cashier"
      }));
00083     usertypecmbo.addActionListener(new java.awt.event.ActionListener() {
00084       public void actionPerformed(java.awt.event.ActionEvent evt) {
00085         usertypecmboActionPerformed(evt);
00086       }
00087     });
00088
00089     jButton1.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00090     jButton1.setText("Log In");
00091     jButton1.addActionListener(new java.awt.event.ActionListener() {
00092       public void actionPerformed(java.awt.event.ActionEvent evt) {
00093         jButton1ActionPerformed(evt);
00094       }
00095     });
00096
00097     jPasswordField1.setFont(new java.awt.Font("Thonburi", 0, 18)); // NOI18N
00098     jPasswordField1.setHorizontalAlignment(javax.swing.JTextField.CENTER);
00099     jPasswordField1.setBorder(javax.swing.BorderFactory.createTitledBorder(
00100         javax.swing.BorderFactory.createLineBorder(new java.awt.Color(0, 0, 0)), "Password",
00101         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00102         new java.awt.Font("Lucida Grande", 0, 18))); // NOI18N
00103
00104     javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
00105     jPanel1.setLayout(jPanel1Layout);
00106     jPanel1Layout.setHorizontalGroup(
00107         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00108             .addGroup(jPanel1Layout.createSequentialGroup()
00109                 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00110                     .addGroup(jPanel1Layout.createSequentialGroup()
00111                         .addGap(132, 132, 132)
00112                         .addComponent(jLabel1))
00113                     .addGroup(jPanel1Layout.createSequentialGroup()
00114                         .addGap(61, 61, 61)
00115                         .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 309,
00116                             javax.swing.GroupLayout.PREFERRED_SIZE))
00117                     .addGroup(jPanel1Layout.createSequentialGroup()
00118                         .addGap(74, 74, 74)
00119
      .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00120                             .addComponent(jPasswordField1, javax.swing.GroupLayout.PREFERRED_SIZE,
      311,
00121                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00122                             .addComponent(usernametxt, javax.swing.GroupLayout.PREFERRED_SIZE, 311,
00123                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00124                             .addComponent(usertypecmbo, javax.swing.GroupLayout.PREFERRED_SIZE, 311,
00125                                 javax.swing.GroupLayout.PREFERRED_SIZE)))
00126                     .addGroup(jPanel1Layout.createSequentialGroup()
00127                         .addGap(146, 146, 146)
00128                         .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 141,
00129                             javax.swing.GroupLayout.PREFERRED_SIZE)))
00130                 .addContainerGap(86, Short.MAX_VALUE)));
00131     jPanel1Layout.setVerticalGroup(
00132         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00133             .addGroup(jPanel1Layout.createSequentialGroup()
00134                 .addGap(39, 39, 39)
00135                 .addComponent(jLabel1)
00136                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
00137                 .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
00138                     javax.swing.GroupLayout.PREFERRED_SIZE)
00139                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
00140                 .addComponent(usernametxt, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00141                     javax.swing.GroupLayout.PREFERRED_SIZE)
00142                 .addGap(18, 18, 18)
```

```
00143                 .addComponent(jPasswordField1, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00144                     javax.swing.GroupLayout.PREFERRED_SIZE)
00145                 .addGap(18, 18, 18)
00146                 .addComponent(usertypecmbo, javax.swing.GroupLayout.PREFERRED_SIZE, 46,
00147                     javax.swing.GroupLayout.PREFERRED_SIZE)
00148                 .addGap(18, 18, 18)
00149                 .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00150                     javax.swing.GroupLayout.PREFERRED_SIZE)
00151                 .addContainerGap(39, Short.MAX_VALUE)));
00152
00153     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00154     getContentPane().setLayout(layout);
00155     layout.setHorizontalGroup(
00156         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00157             .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00158                 javax.swing.GroupLayout.PREFERRED_SIZE));
00159     layout.setVerticalGroup(
00160         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00161             .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00162                 Short.MAX_VALUE));
00163
00164     pack();
00165     setLocationRelativeTo(null);
00166   }// </editor-fold>//GEN-END:initComponents
00167
00181   private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//
      GEN-FIRST:event_jButton1ActionPerformed
00182     // Get username and password from input fields.
00183     String usernameInput = usernametxt.getText();
00184     String passwordInput = new String(jPasswordField1.getPassword());
00185
00186     jButton1.setEnabled(false);
00187     jButton1.setText("Loading...");
00188
00189     new Thread(() -> {
00190       try {
00191         // Authenticate user through AuthService.
00192         User loggedInUser = authService.authenticate(usernameInput, passwordInput);
00193
00194         SwingUtilities.invokeLater(() -> {
00195
00196           // Open the corresponding window based on user role.
00197           if ("ADMIN".equalsIgnoreCase(loggedInUser.getRole())) {
00198             // Open the admin window if the role user is ADMIN.
00199             new AdminLogin().setVisible(true);
00200           } else if ("CASHIER".equalsIgnoreCase(loggedInUser.getRole())) {
00201             // Open the cashier window if the role user is CASHIER.
00202             new CashierLogin(loggedInUser.getUsername()).setVisible(true);
00203           } else {
00204             JOptionPane.showMessageDialog(this, "Unknown Role: " + loggedInUser.getRole());
00205             jButton1.setEnabled(true);
00206             jButton1.setText("Log In");
00207             return;
00208           }
00209
00210           this.dispose();
00211         });
00212
00213       } catch (Exception e) {
00214         SwingUtilities.invokeLater(() -> {
00215           JOptionPane.showMessageDialog(this,
00216               "Login failed: " + e.getMessage(),
00217               "Login Error",
00218               JOptionPane.ERROR_MESSAGE);
00219
00220           jButton1.setEnabled(true);
00221           jButton1.setText("Log In");
00222         });
00223       }
00224     }).start();
00225   }// GEN-LAST:event_jButton1ActionPerformed
00226
00235   private void usertypecmboActionPerformed(java.awt.event.ActionEvent evt) {//
      GEN-FIRST:event_usertypecmboActionPerformed
00236     // No action needed here.
00237   }// GEN-LAST:event_usertypecmboActionPerformed
00238
00244   public static void main(String[] args) {
00245     try {
00246       for (javax.swing.UIManager.LookAndFeelInfo info :
      javax.swing.UIManager.getInstalledLookAndFeels()) {
00247         if ("Nimbus".equals(info.getName())) {
00248           javax.swing.UIManager.setLookAndFeel(info.getClassName());
00249           break;
00250         }
```

```
00251       }
00252     } catch (ClassNotFoundException | InstantiationException | IllegalAccessException
00253         | javax.swing.UnsupportedLookAndFeelException ex) {
00254       java.util.logging.Logger.getLogger(Login.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
00255     }
00256
00257     java.awt.EventQueue.invokeLater(() -> new Login().setVisible(true));
00258   }
00259
00260   // Variables declaration - do not modify//GEN-BEGIN:variables
00262   private javax.swing.JButton jButton1;
00264   private javax.swing.JLabel jLabel1;
00266   private javax.swing.JLabel jLabel3;
00268   private javax.swing.JPanel jPanel1;
00270   private javax.swing.JPasswordField jPasswordField1;
00272   private javax.swing.JTextField usernametxt;
00277   private javax.swing.JComboBox<String> usertypecmbo;
00278   // End of variables declaration//GEN-END:variables
00279 }
```

8.56 ApiClient.java File Reference

`import com.fasterxml.jackson.databind.ObjectMapper;`
Include dependency graph for ApiClient.java:



Classes

- class es.ull.esit.app.middleware.ApiClient

  API client for interacting with the backend REST API.

Packages

- package es.ull.esit.app.middleware

8.57 ApiClient.java

Go to the documentation of this file.

```
00001 package es.ull.esit.app.middleware;
00002
00003 import com.fasterxml.jackson.databind.ObjectMapper;
00004 import com.fasterxml.jackson.core.type.TypeReference;
00005 import java.net.http.*;
```

```
00006 import java.net.URI;
00007 import java.time.Duration;
00008 import java.util.List;
00009 import java.util.Map;
00010
00011 import es.ull.esit.app.middleware.model.Appetizer;
00012 import es.ull.esit.app.middleware.model.Cashier;
00013 import es.ull.esit.app.middleware.model.Drink;
00014 import es.ull.esit.app.middleware.model.MainCourse;
00015 import es.ull.esit.app.middleware.model.User;
00016
00025 public class ApiClient {
00027   private final HttpClient http;
00028
00030   private final String baseUrl;
00031
00036   private final ObjectMapper mapper;
00037
00046   public ApiClient(String baseUrl) {
00047     this.baseUrl = baseUrl.endsWith("/") ? baseUrl.substring(0, baseUrl.length() - 1) : baseUrl;
00048     this.http = HttpClient.newBuilder()
00049         .connectTimeout(Duration.ofSeconds(5))
00050         .build();
00051     this.mapper = new ObjectMapper();
00052   }
00053
00068   private <T> T get(String path, Class<T> responseType) throws Exception {
00069     HttpRequest req = HttpRequest.newBuilder()
00070         .uri(URI.create(baseUrl + path))
00071         .GET()
00072         .header("Accept", "application/json")
00073         .build();
00074
00075     HttpResponse<String> res = http.send(req, HttpResponse.BodyHandlers.ofString());
00076     int status = res.statusCode();
00077     String body = res.body();
00078
00079     System.out.println("GET " + path + " -> HTTP " + status);
00080     System.out.println("Response body: '" + body + "'");
00081
00082     if (status == 204 || body == null || body.trim().isEmpty()) {
00083       return null;
00084     }
00085
00086     if (status < 200 || status >= 300) {
00087       throw new RuntimeException("GET " + path + " failed with HTTP " + status + " body: " + body);
00088     }
00089
00090     return mapper.readValue(body, responseType);
00091   }
00092
00107   private <T> List<T> getList(String path, TypeReference<List<T» typeRef) throws Exception {
00108     HttpRequest req = HttpRequest.newBuilder()
00109         .uri(URI.create(baseUrl + path))
00110         .GET()
00111         .header("Accept", "application/json")
00112         .build();
00113
00114     HttpResponse<String> res = http.send(req, HttpResponse.BodyHandlers.ofString());
00115     int status = res.statusCode();
00116     String body = res.body();
00117
00118     System.out.println("GET " + path + " -> HTTP " + status);
00119     System.out.println("Response body: '" + body + "'");
00120
00121     if (status == 204 || body == null || body.trim().isEmpty()) {
00122       return java.util.Collections.emptyList();
00123     }
00124
00125     if (status < 200 || status >= 300) {
00126       throw new RuntimeException("GET " + path + " failed with HTTP " + status + " body: " + body);
00127     }
00128
00129     return mapper.readValue(body, typeRef);
00130   }
00131
00148   private <T, R> R post(String path, T body, Class<R> responseType) throws Exception {
00149     String json = mapper.writeValueAsString(body);
00150     HttpRequest req = HttpRequest.newBuilder()
00151         .uri(URI.create(baseUrl + path))
00152         .POST(HttpRequest.BodyPublishers.ofString(json))
00153         .header("Content-Type", "application/json")
00154         .build();
00155     HttpResponse<String> res = http.send(req, HttpResponse.BodyHandlers.ofString());
00156
00157     int status = res.statusCode();
00158     String responseBody = res.body();
```

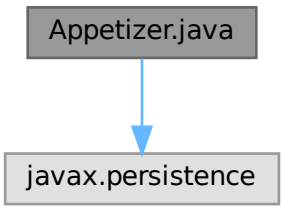
```
00159
00160     System.out.println("POST " + path + " -> HTTP " + status);
00161     System.out.println("Response body: '" + responseBody + "'");
00162
00163     if (status < 200 || status >= 300) {
00164       throw new RuntimeException("POST " + path + " failed with HTTP " + status + " body: " +
      responseBody);
00165     }
00166
00167     return mapper.readValue(responseBody, responseType);
00168   }
00169
00170   // ------------- CRUD methods for Appetizers ----------------
00179   public List<Appetizer> getAllAppetizers() throws Exception {
00180     return getList("/api/appetizers", new TypeReference<List<Appetizer>>() {
00181     });
00182   }
00183
00193   public Appetizer getAppetizerById(Long id) throws Exception {
00194     return get("/api/appetizers/" + id, Appetizer.class);
00195   }
00196
00205   public Appetizer createAppetizer(Appetizer appetizer) throws Exception {
00206     return post("/api/appetizers", appetizer, Appetizer.class);
00207   }
00208
00218   public Appetizer updateAppetizer(Long id, Appetizer appetizer) throws Exception {
00219     String json = mapper.writeValueAsString(appetizer);
00220     HttpRequest req = HttpRequest.newBuilder()
00221         .uri(URI.create(baseUrl + "/api/appetizers/" + id))
00222         .PUT(HttpRequest.BodyPublishers.ofString(json))
00223         .header("Content-Type", "application/json")
00224         .build();
00225     HttpResponse<String> res = http.send(req, HttpResponse.BodyHandlers.ofString());
00226
00227     int status = res.statusCode();
00228     String body = res.body();
00229
00230     if (status < 200 || status >= 300) {
00231       throw new RuntimeException("PUT /api/appetizers/" + id + " failed with HTTP " + status + " body:
      " + body);
00232     }
00233
00234     return mapper.readValue(body, Appetizer.class);
00235   }
00236
00244   public void deleteAppetizer(Long id) throws Exception {
00245     HttpRequest req = HttpRequest.newBuilder()
00246         .uri(URI.create(baseUrl + "/api/appetizers/" + id))
00247         .DELETE()
00248         .build();
00249     http.send(req, HttpResponse.BodyHandlers.ofString());
00250   }
00251
00252   // ------------- READ / UPDATE methods for Cashiers ----------------
00253
00262   public List<Cashier> getAllCashiers() throws Exception {
00263     return getList("/api/cashiers", new TypeReference<List<Cashier>>() {
00264     });
00265   }
00266
00275   public Cashier getCashierById(Long id) throws Exception {
00276     return get("/api/cashiers/" + id, Cashier.class);
00277   }
00278
00287   public Cashier getCashierByName(String name) throws Exception {
00288     return get("/api/cashiers/name/" + name, Cashier.class);
00289   }
00290
00301   public Cashier updateCashier(Long id, Cashier cashier) throws Exception {
00302     String json = mapper.writeValueAsString(cashier);
00303     HttpRequest req = HttpRequest.newBuilder()
00304         .uri(URI.create(baseUrl + "/api/cashiers/" + id))
00305         .PUT(HttpRequest.BodyPublishers.ofString(json))
00306         .header("Content-Type", "application/json")
00307         .build();
00308     HttpResponse<String> res = http.send(req, HttpResponse.BodyHandlers.ofString());
00309
00310     int status = res.statusCode();
00311     String body = res.body();
00312
00313     if (status < 200 || status >= 300) {
00314       throw new RuntimeException("PUT /api/cashiers/" + id + " failed with HTTP " + status + " body: "
      + body);
00315     }
00316
00317     return mapper.readValue(body, Cashier.class);
```

```
00318   }
00319
00320   // ------------- CRUD methods for Drinks ----------------
00329   public List<Drink> getAllDrinks() throws Exception {
00330     return getList("/api/drinks", new TypeReference<List<Drink>>() {
00331     });
00332   }
00333
00343   public Drink getDrinkById(Long id) throws Exception {
00344     return get("/api/drinks/" + id, Drink.class);
00345   }
00346
00356   public Drink createDrink(Drink drink) throws Exception {
00357     return post("/api/drinks", drink, Drink.class);
00358   }
00359
00370   public Drink updateDrink(Long id, Drink drink) throws Exception {
00371     String json = mapper.writeValueAsString(drink);
00372     HttpRequest req = HttpRequest.newBuilder()
00373         .uri(URI.create(baseUrl + "/api/drinks/" + id))
00374         .PUT(HttpRequest.BodyPublishers.ofString(json))
00375         .header("Content-Type", "application/json")
00376         .build();
00377     HttpResponse<String> res = http.send(req, HttpResponse.BodyHandlers.ofString());
00378
00379     int status = res.statusCode();
00380     String body = res.body();
00381
00382     if (status < 200 || status >= 300) {
00383       throw new RuntimeException("PUT /api/drinks/" + id + " failed with HTTP " + status + " body: " +
      body);
00384     }
00385
00386     return mapper.readValue(body, Drink.class);
00387   }
00388
00397   public void deleteDrink(Long id) throws Exception {
00398     HttpRequest req = HttpRequest.newBuilder()
00399         .uri(URI.create(baseUrl + "/api/drinks/" + id))
00400         .DELETE()
00401         .build();
00402     http.send(req, HttpResponse.BodyHandlers.ofString());
00403   }
00404
00405   // ------------- CRUD methods for MainCourses ----------------
00414   public List<MainCourse> getAllMainCourses() throws Exception {
00415     return getList("/api/maincourses", new TypeReference<List<MainCourse>>() {
00416     });
00417   }
00418
00428   public MainCourse getMainCourseById(Long id) throws Exception {
00429     return get("/api/maincourses/" + id, MainCourse.class);
00430   }
00431
00441   public MainCourse createMainCourse(MainCourse mainCourse) throws Exception {
00442     return post("/api/maincourses", mainCourse, MainCourse.class);
00443   }
00444
00455   public MainCourse updateMainCourse(Long id, MainCourse mainCourse) throws Exception {
00456     String json = mapper.writeValueAsString(mainCourse);
00457     HttpRequest req = HttpRequest.newBuilder()
00458         .uri(URI.create(baseUrl + "/api/maincourses/" + id))
00459         .PUT(HttpRequest.BodyPublishers.ofString(json))
00460         .header("Content-Type", "application/json")
00461         .build();
00462     HttpResponse<String> res = http.send(req, HttpResponse.BodyHandlers.ofString());
00463
00464     int status = res.statusCode();
00465     String body = res.body();
00466
00467     if (status < 200 || status >= 300) {
00468       throw new RuntimeException("PUT /api/maincourses/" + id + " failed with HTTP " + status + "
      body: " + body);
00469     }
00470
00471     return mapper.readValue(body, MainCourse.class);
00472   }
00473
00482   public void deleteMainCourse(Long id) throws Exception {
00483     HttpRequest req = HttpRequest.newBuilder()
00484         .uri(URI.create(baseUrl + "/api/maincourses/" + id))
00485         .DELETE()
00486         .build();
00487     http.send(req, HttpResponse.BodyHandlers.ofString());
00488   }
00489
00490   // ------------- Authentication methods ----------------
```

```
00491
00500   public void login(String ignored) throws Exception {
00501     // No-op: kept for compatibility.
00502   }
00503
00516   public User login(String username, String password) throws Exception {
00517
00518     String jsonBody = mapper.writeValueAsString(Map.of(
00519         "username", username,
00520         "password", password));
00521
00522     HttpRequest request = HttpRequest.newBuilder()
00523
00524         .uri(URI.create(baseUrl + "/api/login"))
00525         .header("Content-Type", "application/json")
00526         .POST(HttpRequest.BodyPublishers.ofString(jsonBody))
00527         .build();
00528
00529     HttpResponse<String> response = http.send(request, HttpResponse.BodyHandlers.ofString());
00530
00531     System.out.println("POST /api/login -> HTTP " + response.statusCode());
00532     System.out.println("Response body: '" + response.body() + "'");
00533
00534     if (response.statusCode() == 200) {
00535       return mapper.readValue(response.body(), User.class);
00536     } else {
00537       throw new RuntimeException("Login failed with status: " + response.statusCode());
00538     }
00539   }
00540
00541 }
```

8.58 server/src/main/java/es/ull/esit/server/middleware/model/↩ Appetizer.java File Reference

`import javax.persistence;`
Include dependency graph for server/src/main/java/es/ull/esit/server/middleware/model/Appetizer.java:



Classes

- class es.ull.esit.server.middleware.model.Appetizer

  JPA entity that represents an appetizer in the menu.

Packages

- package es.ull.esit.server.middleware.model

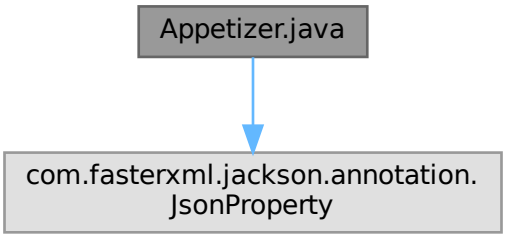
8.59 server/src/main/java/es/ull/esit/server/middleware/model/↩ Appetizer.java

Go to the documentation of this file.

```
00001 package es.ull.esit.server.middleware.model;
00002
00003 import javax.persistence.*;
00004 import com.fasterxml.jackson.annotation.JsonIgnoreProperties;
00005
00012 @Entity
00013 @Table(name = "appetizers")
00014 @JsonIgnoreProperties({ "hibernateLazyInitializer", "handler" })
00015 public class Appetizer {
00016
00018   @Id
00019   @GeneratedValue(strategy = GenerationType.IDENTITY)
00020   @Column(name = "id")
00021   private Long appetizersId;
00022
00024   @Column(name = "name", nullable = false)
00025   private String itemAppetizers;
00026
00028   @Column(name = "price", nullable = false)
00029   private Integer appetizersPrice;
00030
00034   public Appetizer() {
00035   }
00036
00044   public Appetizer(Long appetizersId, String itemAppetizers, Integer appetizersPrice) {
00045     this.appetizersId = appetizersId;
00046     this.itemAppetizers = itemAppetizers;
00047     this.appetizersPrice = appetizersPrice;
00048   }
00049
00055   public Long getAppetizersId() {
00056     return appetizersId;
00057   }
00058
00065   public void setAppetizersId(Long appetizersId) {
00066     this.appetizersId = appetizersId;
00067   }
00068
00074   public String getItemAppetizers() {
00075     return itemAppetizers;
00076   }
00077
00083   public void setItemAppetizers(String itemAppetizers) {
00084     this.itemAppetizers = itemAppetizers;
00085   }
00086
00092   public Integer getAppetizersPrice() {
00093     return appetizersPrice;
00094   }
00095
00101   public void setAppetizersPrice(Integer appetizersPrice) {
00102     this.appetizersPrice = appetizersPrice;
00103   }
00104 }
```

8.60 src/main/java/es/ull/esit/app/middleware/model/Appetizer.java File Reference

`import com.fasterxml.jackson.annotation.JsonProperty;`
Include dependency graph for src/main/java/es/ull/esit/app/middleware/model/Appetizer.java:



Classes

- class es.ull.esit.app.middleware.model.Appetizer

  Client-side model representing an appetizer received from the backend API.

Packages

- package es.ull.esit.app.middleware.model

8.61 src/main/java/es/ull/esit/app/middleware/model/Appetizer.java

Go to the documentation of this file.

```
00001 package es.ull.esit.app.middleware.model;
00002
00003 import com.fasterxml.jackson.annotation.JsonProperty;
00004
00011 public class Appetizer {
00012
00014   @JsonProperty("appetizersId")
00015   private Long appetizersId;
00016
00018   @JsonProperty("itemAppetizers")
00019   private String itemAppetizers;
00020
00022   @JsonProperty("appetizersPrice")
00023   private Integer appetizersPrice;
00024
00029   @JsonProperty("receiptId")
00030   private Long receiptId;
00031
00035   public Appetizer() {
00036   }
00037
00046   public Appetizer(Long appetizersId, String itemAppetizers, Integer appetizersPrice, Long receiptId)
      {
00047     this.appetizersId = appetizersId;
00048     this.itemAppetizers = itemAppetizers;
00049     this.appetizersPrice = appetizersPrice;
00050     this.receiptId = receiptId;
```

```
00051   }
00052
00058   public Long getAppetizersId() {
00059     return appetizersId;
00060   }
00061
00067   public void setAppetizersId(Long appetizersId) {
00068     this.appetizersId = appetizersId;
00069   }
00070
00076   public String getItemAppetizers() {
00077     return itemAppetizers;
00078   }
00079
00085   public void setItemAppetizers(String itemAppetizers) {
00086     this.itemAppetizers = itemAppetizers;
00087   }
00088
00094   public Integer getAppetizersPrice() {
00095     return appetizersPrice;
00096   }
00097
00103   public void setAppetizersPrice(Integer appetizersPrice) {
00104     this.appetizersPrice = appetizersPrice;
00105   }
00106
00112   public Long getReceiptId() {
00113     return receiptId;
00114   }
00115
00121   public void setReceiptId(Long receiptId) {
00122     this.receiptId = receiptId;
00123   }
00124 }
```

8.62 BillResult.java File Reference

Classes

- class es.ull.esit.app.middleware.model.BillResult

  Data Transfer Object representing the result of a bill calculation.

Packages

- package es.ull.esit.app.middleware.model

8.63 BillResult.java

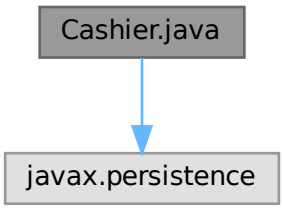
Go to the documentation of this file.

```
00001 package es.ull.esit.app.middleware.model;
00002
00009 public class BillResult {
00010
00012   private final double subTotal;
00013
00015   private final double vat;
00016
00018   private final double total;
00019
00027   public BillResult(double subTotal, double vat, double total) {
00028     this.subTotal = subTotal;
00029     this.vat = vat;
00030     this.total = total;
00031   }
00032
00038   public double getSubTotal() {
00039     return subTotal;
00040   }
00041
00047   public double getVat() {
```

```
00048     return vat;
00049   }
00050
00056   public double getTotal() {
00057     return total;
00058   }
00059 }
```

8.64 server/src/main/java/es/ull/esit/server/middleware/model/↩ Cashier.java File Reference

`import javax.persistence;`
Include dependency graph for server/src/main/java/es/ull/esit/server/middleware/model/Cashier.java:



Classes

- class es.ull.esit.server.middleware.model.Cashier

  JPA entity that represents a cashier in the system.

Packages

- package es.ull.esit.server.middleware.model

8.65 server/src/main/java/es/ull/esit/server/middleware/model/↩ Cashier.java

Go to the documentation of this file.

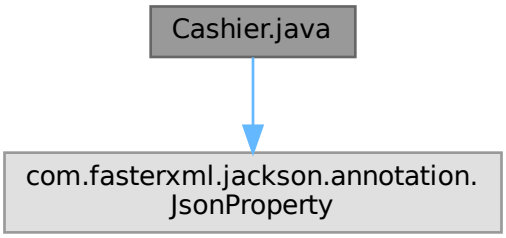
```
00001 package es.ull.esit.server.middleware.model;
00002
00003 import javax.persistence.*;
00004 import com.fasterxml.jackson.annotation.JsonIgnoreProperties;
00005
00012 @Entity
00013 @Table(name = "cashier")
00014 @JsonIgnoreProperties({"hibernateLazyInitializer", "handler"})
00015 public class Cashier {
00016
00018   @Id
00019   @Column(name = "cashier_id")
00020   private Long id;
00021
```

```
00023   @Column(name = "cashier_name")
00024   private String name;
00025
00027   @Column(name = "cashier_salary")
00028   private Integer salary;
00029
00033   public Cashier() {
00034   }
00035
00043   public Cashier(Long id, String name, Integer salary) {
00044     this.id = id;
00045     this.name = name;
00046     this.salary = salary;
00047   }
00048
00054   public Long getId() {
00055     return id;
00056   }
00057
00064   public void setId(Long id) {
00065     this.id = id;
00066   }
00067
00073   public String getName() {
00074     return name;
00075   }
00076
00082   public void setName(String name) {
00083     this.name = name;
00084   }
00085
00091   public Integer getSalary() {
00092     return salary;
00093   }
00094
00100   public void setSalary(Integer salary) {
00101     this.salary = salary;
00102   }
00103 }
```

8.66 src/main/java/es/ull/esit/app/middleware/model/Cashier.java File Reference

```
import com.fasterxml.jackson.annotation.JsonProperty;
```

Include dependency graph for src/main/java/es/ull/esit/app/middleware/model/Cashier.java:



Classes

- class es.ull.esit.app.middleware.model.Cashier

  Client-side model representing a cashier returned by the backend.

Packages

- package es.ull.esit.app.middleware.model

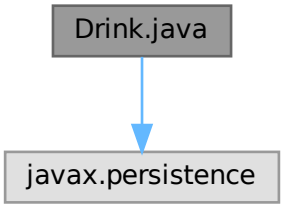
8.67 src/main/java/es/ull/esit/app/middleware/model/Cashier.java

Go to the documentation of this file.

```
00001 package es.ull.esit.app.middleware.model;
00002
00003 import com.fasterxml.jackson.annotation.JsonProperty;
00004
00011 public class Cashier {
00012
00014   @JsonProperty("id")
00015   private Long id;
00016
00018   @JsonProperty("name")
00019   private String name;
00020
00022   @JsonProperty("salary")
00023   private Integer salary;
00024
00028   public Cashier() {
00029   }
00030
00038   public Cashier(Long id, String name, Integer salary) {
00039     this.id = id;
00040     this.name = name;
00041     this.salary = salary;
00042   }
00043
00049   public Long getId() {
00050     return id;
00051   }
00052
00058   public void setId(Long id) {
00059     this.id = id;
00060   }
00061
00067   public String getName() {
00068     return name;
00069   }
00070
00076   public void setName(String name) {
00077     this.name = name;
00078   }
00079
00085   public Integer getSalary() {
00086     return salary;
00087   }
00088
00094   public void setSalary(Integer salary) {
00095     this.salary = salary;
00096   }
00097 }
```

8.68 server/src/main/java/es/ull/esit/server/middleware/model/Drink.java File Reference

`import javax.persistence;`
Include dependency graph for server/src/main/java/es/ull/esit/server/middleware/model/Drink.java:



Classes

- class es.ull.esit.server.middleware.model.Drink

  JPA entity that represents a drink in the menu.

Packages

- package es.ull.esit.server.middleware.model

8.69 server/src/main/java/es/ull/esit/server/middleware/model/Drink.java

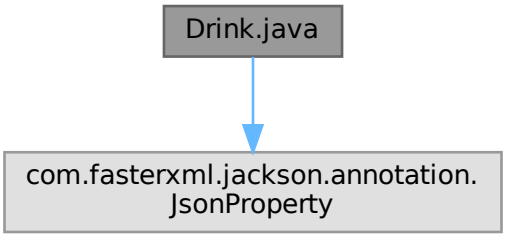
Go to the documentation of this file.

```
00001 package es.ull.esit.server.middleware.model;
00002
00003 import javax.persistence.*;
00004 import com.fasterxml.jackson.annotation.JsonIgnoreProperties;
00005
00012 @Entity
00013 @Table(name = "drinks")
00014 @JsonIgnoreProperties({ "hibernateLazyInitializer", "handler" })
00015 public class Drink {
00016
00018   @Id
00019   @GeneratedValue(strategy = GenerationType.IDENTITY)
00020   @Column(name = "id")
00021   private Long drinksId;
00022
00024   @Column(name = "name", nullable = false)
00025   private String itemDrinks;
00026
00028   @Column(name = "price", nullable = false)
00029   private Integer drinksPrice;
00030
00034   public Drink() {
00035   }
00036
00044   public Drink(Long drinksId, String itemDrinks, Integer drinksPrice) {
00045     this.drinksId = drinksId;
00046     this.itemDrinks = itemDrinks;
00047     this.drinksPrice = drinksPrice;
```

```
00048   }
00049
00055   public Long getDrinksId() {
00056     return drinksId;
00057   }
00058
00066   public void setDrinksId(Long drinksId) {
00067     this.drinksId = drinksId;
00068   }
00069
00075   public String getItemDrinks() {
00076     return itemDrinks;
00077   }
00078
00084   public void setItemDrinks(String itemDrinks) {
00085     this.itemDrinks = itemDrinks;
00086   }
00087
00093   public Integer getDrinksPrice() {
00094     return drinksPrice;
00095   }
00096
00102   public void setDrinksPrice(Integer drinksPrice) {
00103     this.drinksPrice = drinksPrice;
00104   }
00105 }
```

8.70 src/main/java/es/ull/esit/app/middleware/model/Drink.java File Reference

`import com.fasterxml.jackson.annotation.JsonProperty;`
Include dependency graph for src/main/java/es/ull/esit/app/middleware/model/Drink.java:



Classes

- class es.ull.esit.app.middleware.model.Drink

  Client-side model representing a drink returned by the backend API.

Packages

- package es.ull.esit.app.middleware.model

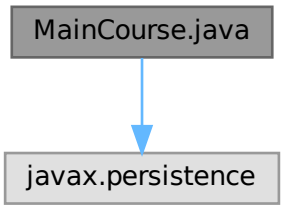
8.71 src/main/java/es/ull/esit/app/middleware/model/Drink.java

Go to the documentation of this file.

```
00001 package es.ull.esit.app.middleware.model;
00002
00003 import com.fasterxml.jackson.annotation.JsonProperty;
00004
00011 public class Drink {
00012
00014   @JsonProperty("drinksId")
00015   private Long drinksId;
00016
00018   @JsonProperty("itemDrinks")
00019   private String itemDrinks;
00020
00022   @JsonProperty("drinksPrice")
00023   private Integer drinksPrice;
00024
00029   @JsonProperty("receiptId")
00030   private Long receiptId;
00031
00035   public Drink() {
00036   }
00037
00046   public Drink(Long drinksId, String itemDrinks, Integer drinksPrice, Long receiptId) {
00047     this.drinksId = drinksId;
00048     this.itemDrinks = itemDrinks;
00049     this.drinksPrice = drinksPrice;
00050     this.receiptId = receiptId;
00051   }
00052
00058   public Long getDrinksId() {
00059     return drinksId;
00060   }
00061
00067   public void setDrinksId(Long drinksId) {
00068     this.drinksId = drinksId;
00069   }
00070
00076   public String getItemDrinks() {
00077     return itemDrinks;
00078   }
00079
00085   public void setItemDrinks(String itemDrinks) {
00086     this.itemDrinks = itemDrinks;
00087   }
00088
00094   public Integer getDrinksPrice() {
00095     return drinksPrice;
00096   }
00097
00103   public void setDrinksPrice(Integer drinksPrice) {
00104     this.drinksPrice = drinksPrice;
00105   }
00106
00112   public Long getReceiptId() {
00113     return receiptId;
00114   }
00115
00121   public void setReceiptId(Long receiptId) {
00122     this.receiptId = receiptId;
00123   }
00124 }
```

8.72 server/src/main/java/es/ull/esit/server/middleware/model/MainCourse.java File Reference

`import javax.persistence;`
Include dependency graph for server/src/main/java/es/ull/esit/server/middleware/model/MainCourse.java:



Classes

- class es.ull.esit.server.middleware.model.MainCourse

  JPA entity that represents a main course in the menu.

Packages

- package es.ull.esit.server.middleware.model

8.73 server/src/main/java/es/ull/esit/server/middleware/model/MainCourse.java

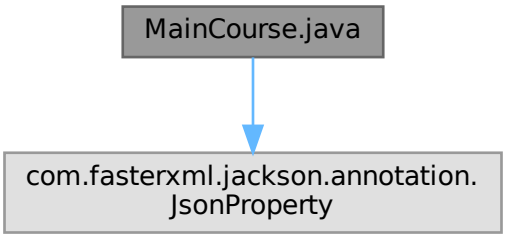
Go to the documentation of this file.

```
00001 package es.ull.esit.server.middleware.model;
00002
00003 import javax.persistence.*;
00004 import com.fasterxml.jackson.annotation.JsonIgnoreProperties;
00005
00012 @Entity
00013 @Table(name = "maincourse")
00014 @JsonIgnoreProperties({ "hibernateLazyInitializer", "handler" })
00015 public class MainCourse {
00016
00018   @Id
00019   @GeneratedValue(strategy = GenerationType.IDENTITY)
00020   @Column(name = "id")
00021   private Long foodId;
00022
00024   @Column(name = "name", nullable = false)
00025   private String itemFood;
00026
00028   @Column(name = "price", nullable = false)
00029   private Integer foodPrice;
00030
00034   public MainCourse() {
00035   }
00036
00044   public MainCourse(Long foodId, String itemFood, Integer foodPrice) {
00045     this.foodId = foodId;
```

```
00046     this.itemFood = itemFood;
00047     this.foodPrice = foodPrice;
00048   }
00049
00055   public Long getFoodId() {
00056     return foodId;
00057   }
00058
00066   public void setFoodId(Long foodId) {
00067     this.foodId = foodId;
00068   }
00069
00075   public String getItemFood() {
00076     return itemFood;
00077   }
00078
00084   public void setItemFood(String itemFood) {
00085     this.itemFood = itemFood;
00086   }
00087
00093   public Integer getFoodPrice() {
00094     return foodPrice;
00095   }
00096
00102   public void setFoodPrice(Integer foodPrice) {
00103     this.foodPrice = foodPrice;
00104   }
00105 }
```

8.74 src/main/java/es/ull/esit/app/middleware/model/MainCourse.java File Reference

`import com.fasterxml.jackson.annotation.JsonProperty;`
Include dependency graph for src/main/java/es/ull/esit/app/middleware/model/MainCourse.java:



Classes

- class es.ull.esit.app.middleware.model.MainCourse

  Client-side model representing a main course returned by the backend.

Packages

- package es.ull.esit.app.middleware.model

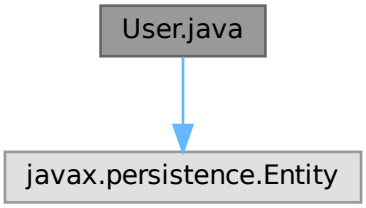
8.75 src/main/java/es/ull/esit/app/middleware/model/MainCourse.java

Go to the documentation of this file.

```
00001 package es.ull.esit.app.middleware.model;
00002
00003 import com.fasterxml.jackson.annotation.JsonProperty;
00004
00011 public class MainCourse {
00012
00014   @JsonProperty("foodId")
00015   private Long foodId;
00016
00018   @JsonProperty("itemFood")
00019   private String itemFood;
00020
00022   @JsonProperty("foodPrice")
00023   private Integer foodPrice;
00024
00029   @JsonProperty("receiptId")
00030   private Long receiptId;
00031
00035   public MainCourse() {
00036   }
00037
00046   public MainCourse(Long foodId, String itemFood, Integer foodPrice, Long receiptId) {
00047     this.foodId = foodId;
00048     this.itemFood = itemFood;
00049     this.foodPrice = foodPrice;
00050     this.receiptId = receiptId;
00051   }
00052
00058   public Long getFoodId() {
00059     return foodId;
00060   }
00061
00067   public void setFoodId(Long foodId) {
00068     this.foodId = foodId;
00069   }
00070
00076   public String getItemFood() {
00077     return itemFood;
00078   }
00079
00085   public void setItemFood(String itemFood) {
00086     this.itemFood = itemFood;
00087   }
00088
00094   public Integer getFoodPrice() {
00095     return foodPrice;
00096   }
00097
00103   public void setFoodPrice(Integer foodPrice) {
00104     this.foodPrice = foodPrice;
00105   }
00106
00112   public Long getReceiptId() {
00113     return receiptId;
00114   }
00115
00121   public void setReceiptId(Long receiptId) {
00122     this.receiptId = receiptId;
00123   }
00124 }
```

8.76 server/src/main/java/es/ull/esit/server/middleware/model/User.java File Reference

`import javax.persistence.Entity;`
Include dependency graph for server/src/main/java/es/ull/esit/server/middleware/model/User.java:



Classes

- class es.ull.esit.server.middleware.model.User

  JPA entity that represents a user in the system.

Packages

- package es.ull.esit.server.middleware.model

8.77 server/src/main/java/es/ull/esit/server/middleware/model/User.java

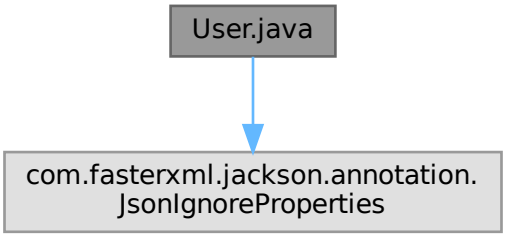
Go to the documentation of this file.

```
00001 package es.ull.esit.server.middleware.model;
00002
00003 import javax.persistence.Entity;
00004 import javax.persistence.Table;
00005 import javax.persistence.Id;
00006 import javax.persistence.GeneratedValue;
00007 import javax.persistence.GenerationType;
00008 import javax.persistence.Column;
00009
00010 import com.fasterxml.jackson.annotation.JsonIgnoreProperties;
00011 import com.fasterxml.jackson.annotation.JsonIgnore;
00012
00019 @Entity
00020 @Table(name = "users")
00021 @JsonIgnoreProperties({ "hibernateLazyInitializer", "handler" })
00022 public class User {
00023
00025   @Id
00026   @GeneratedValue(strategy = GenerationType.IDENTITY)
00027   private Long id;
00028
00030   @Column(unique = true, nullable = false)
00031   private String username;
00032
00034   @Column(name = "password_hash", nullable = false)
00035   @JsonIgnore
00036   private String passwordHash;
00037
```

```
00039   private String role;
00040
00046   public Long getId() {
00047     return id;
00048   }
00049
00055   public void setId(Long id) {
00056     this.id = id;
00057   }
00058
00064   public String getUsername() {
00065     return username;
00066   }
00067
00073   public void setUsername(String username) {
00074     this.username = username;
00075   }
00076
00082   public String getPasswordHash() {
00083     return passwordHash;
00084   }
00085
00091   public void setPasswordHash(String passwordHash) {
00092     this.passwordHash = passwordHash;
00093   }
00094
00100   public String getRole() {
00101     return role;
00102   }
00103
00109   public void setRole(String role) {
00110     this.role = role;
00111   }
00112 }
```

8.78 src/main/java/es/ull/esit/app/middleware/model/User.java File Reference

`import com.fasterxml.jackson.annotation.JsonIgnoreProperties;`
Include dependency graph for src/main/java/es/ull/esit/app/middleware/model/User.java:



Classes

- class es.ull.esit.app.middleware.model.User

  Client-side model representing an authenticated user.

Packages

- package es.ull.esit.app.middleware.model

8.79 src/main/java/es/ull/esit/app/middleware/model/User.java

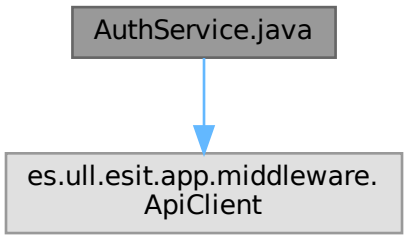
Go to the documentation of this file.

```
00001 package es.ull.esit.app.middleware.model;
00002
00003 import com.fasterxml.jackson.annotation.JsonIgnoreProperties;
00004 import com.fasterxml.jackson.annotation.JsonProperty;
00005
00012 @JsonIgnoreProperties(ignoreUnknown = true)
00013 public class User {
00014
00016   @JsonProperty("username")
00017   private String username;
00018
00020   @JsonProperty("role")
00021   private String role;
00022
00026   public User() {
00027   }
00028
00035   public User(String username, String role) {
00036     this.username = username;
00037     this.role = role;
00038   }
00039
00045   public String getUsername() {
00046     return username;
00047   }
00048
00054   public void setUsername(String username) {
00055     this.username = username;
00056   }
00057
00063   public String getRole() {
00064     return role;
00065   }
00066
00072   public void setRole(String role) {
00073     this.role = role;
00074   }
00075
00081   public boolean isAdmin() {
00082     return "ADMIN".equalsIgnoreCase(this.role);
00083   }
00084 }
```

8.80 AuthService.java File Reference

`import es.ull.esit.app.middleware.ApiClient;`

Include dependency graph for AuthService.java:



Classes

- class es.ull.esit.app.middleware.service.AuthService

  Service responsible for handling authentication logic on the client side.

Packages

- package es.ull.esit.app.middleware.service

8.81 AuthService.java

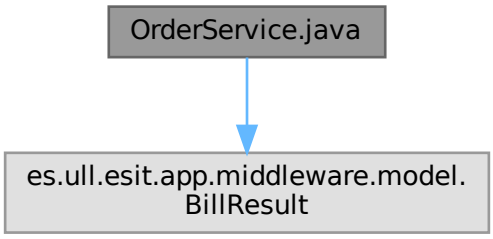
Go to the documentation of this file.

```
00001 package es.ull.esit.app.middleware.service;
00002
00003 import es.ull.esit.app.middleware.ApiClient;
00004 import es.ull.esit.app.middleware.model.User;
00005
00012 public class AuthService {
00013
00015   private final ApiClient client;
00016
00022   public AuthService(ApiClient client) {
00023     this.client = client;
00024   }
00025
00038   public User authenticate(String username, String password) throws Exception {
00039     // 1. Local Validation.
00040     if (username == null || username.trim().isEmpty()) {
00041       throw new IllegalArgumentException("Username cannot be empty.");
00042     }
00043     if (password == null || password.isEmpty()) {
00044       throw new IllegalArgumentException("Password cannot be empty.");
00045     }
00046
00047     // 2. Call API.
00048     return client.login(username, password);
00049   }
00050 }
```

8.82 OrderService.java File Reference

```
import es.ull.esit.app.middleware.model.BillResult;
```

Include dependency graph for OrderService.java:



Classes

- class es.ull.esit.app.middleware.service.OrderService

  Service that handles order-related calculations and receipt generation.

Packages

- package es.ull.esit.app.middleware.service

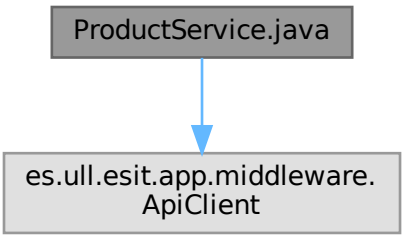
8.83 OrderService.java

Go to the documentation of this file.

```
00001 package es.ull.esit.app.middleware.service;
00002
00003 import es.ull.esit.app.middleware.model.BillResult;
00004
00011 public class OrderService {
00012
00014   private static final double VAT_RATE = 0.15;
00015
00024   public BillResult calculateBill(double itemsTotalSum) {
00025     double vat = itemsTotalSum * VAT_RATE;
00026     double total = itemsTotalSum + vat;
00027
00028     return new BillResult(
00029         Math.round(itemsTotalSum * 100.0) / 100.0,
00030         Math.round(vat * 100.0) / 100.0,
00031         Math.round(total * 100.0) / 100.0);
00032   }
00033
00044   public void generateReceiptFile(int receiptNo, BillResult bill) throws Exception {
00045     // Ensure the directory exists.
00046     new java.io.File("receipts").mkdirs();
00047
00048     try (java.io.PrintWriter output = new java.io.PrintWriter("receipts/billNo." + receiptNo +
    ".txt")) {
00049       output.println(" Bill number is: " + receiptNo);
00050       output.println("==============");
00051       output.println("--------------");
00052       output.println("Subtotal is: " + bill.getSubTotal() + " SR");
00053       output.println("vat: " + bill.getVat() + " SR");
00054       output.println("Total is: " + bill.getTotal() + " SR");
00055       output.println();
00056       output.println("THANK YOU FOR ORDERING");
00057     }
00058   }
00059 }
```

8.84 ProductService.java File Reference

`import es.ull.esit.app.middleware.ApiClient;`
Include dependency graph for ProductService.java:



Classes

- class es.ull.esit.app.middleware.service.ProductService

  Service handling business logic for menu products.

Packages

- package es.ull.esit.app.middleware.service

8.85 ProductService.java

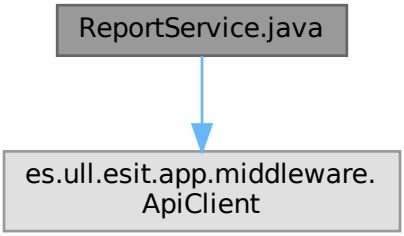
Go to the documentation of this file.

```
00001 package es.ull.esit.app.middleware.service;
00002
00003 import es.ull.esit.app.middleware.ApiClient;
00004 import es.ull.esit.app.middleware.model.Appetizer;
00005 import es.ull.esit.app.middleware.model.Drink;
00006 import es.ull.esit.app.middleware.model.MainCourse;
00007 import java.util.List;
00008
00016 public class ProductService {
00017
00019   private final ApiClient client;
00020
00026   public ProductService(ApiClient client) {
00027     this.client = client;
00028   }
00029
00030   // ==================== DRINKS LOGIC ====================
00031
00038   public List<Drink> getAllDrinks() throws Exception {
00039     return client.getAllDrinks();
00040   }
00041
00049   public Drink getDrinkById(Long id) throws Exception {
00050     return client.getDrinkById(id);
00051   }
00052
00062   public void addDrink(String name, String priceStr) throws Exception {
00063     int price = validateAndParsePrice(priceStr);
00064     validateName(name);
00065
00066     Drink newDrink = new Drink(null, name, price, null);
00067     client.createDrink(newDrink);
00068   }
00069
00079   public void updateDrink(Long id, String name, String priceStr) throws Exception {
00080     if (id == null) {
00081       throw new IllegalArgumentException("No drink selected!");
00082     }
00083
00084     int price = validateAndParsePrice(priceStr);
00085     validateName(name);
00086
00087     Drink updatedDrink = new Drink(id, name, price, null);
00088     client.updateDrink(id, updatedDrink);
00089   }
00090
00091   // ==================== APPETIZERS LOGIC ====================
00092
00099   public List<Appetizer> getAllAppetizers() throws Exception {
00100     return client.getAllAppetizers();
00101   }
00102
00110   public Appetizer getAppetizerById(Long id) throws Exception {
00111     return client.getAppetizerById(id);
00112   }
00113
00123   public void addAppetizer(String name, String priceStr) throws Exception {
00124     int price = validateAndParsePrice(priceStr);
00125     validateName(name);
00126
00127     Appetizer newAppetizer = new Appetizer(null, name, price, null);
00128     client.createAppetizer(newAppetizer);
00129   }
00130
```

```
00140   public void updateAppetizer(Long id, String name, String priceStr) throws Exception {
00141     if (id == null) {
00142       throw new IllegalArgumentException("No appetizer selected!");
00143     }
00144
00145     int price = validateAndParsePrice(priceStr);
00146     validateName(name);
00147
00148     Appetizer updatedAppetizer = new Appetizer(id, name, price, null);
00149     client.updateAppetizer(id, updatedAppetizer);
00150   }
00151
00152   // ==================== MAIN COURSE LOGIC ====================
00153
00160   public List<MainCourse> getAllMainCourses() throws Exception {
00161     return client.getAllMainCourses();
00162   }
00163
00171   public MainCourse getMainCourseById(Long id) throws Exception {
00172     return client.getMainCourseById(id);
00173   }
00174
00184   public void addMainCourse(String name, String priceStr) throws Exception {
00185     int price = validateAndParsePrice(priceStr);
00186     validateName(name);
00187
00188     MainCourse newMainCourse = new MainCourse(null, name, price, null);
00189     client.createMainCourse(newMainCourse);
00190   }
00191
00201   public void updateMainCourse(Long id, String name, String priceStr) throws Exception {
00202     if (id == null) {
00203       throw new IllegalArgumentException("No main course selected!");
00204     }
00205
00206     int price = validateAndParsePrice(priceStr);
00207     validateName(name);
00208
00209     MainCourse updatedMainCourse = new MainCourse(id, name, price, null);
00210     client.updateMainCourse(id, updatedMainCourse);
00211   }
00212
00213   // ==================== VALIDATION HELPERS ====================
00214
00221   private void validateName(String name) {
00222     if (name == null || name.trim().isEmpty()) {
00223       throw new IllegalArgumentException("Item name cannot be empty.");
00224     }
00225   }
00226
00236   private int validateAndParsePrice(String priceStr) {
00237     if (priceStr == null || priceStr.trim().isEmpty()) {
00238       throw new IllegalArgumentException("Price cannot be empty.");
00239     }
00240     try {
00241       int price = Integer.parseInt(priceStr.trim());
00242       if (price < 0) {
00243         throw new IllegalArgumentException("Price cannot be negative.");
00244       }
00245       return price;
00246     } catch (NumberFormatException e) {
00247       throw new IllegalArgumentException("Price must be a valid whole number.");
00248     }
00249   }
00250 }
```

8.86 ReportService.java File Reference

`import es.ull.esit.app.middleware.ApiClient;`
Include dependency graph for ReportService.java:



Classes

- class es.ull.esit.app.middleware.service.ReportService

  Service providing reporting and system status operations.

Packages

- package es.ull.esit.app.middleware.service

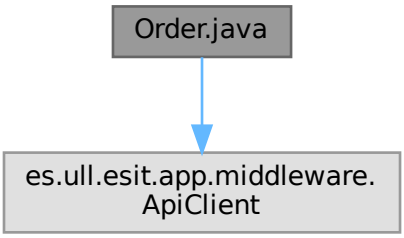
8.87 ReportService.java

Go to the documentation of this file.

```
00001 package es.ull.esit.app.middleware.service;
00002
00003 import es.ull.esit.app.middleware.ApiClient;
00004 import es.ull.esit.app.middleware.model.Appetizer;
00005 import es.ull.esit.app.middleware.model.Cashier;
00006 import es.ull.esit.app.middleware.model.Drink;
00007 import es.ull.esit.app.middleware.model.MainCourse;
00008 import java.util.List;
00009
00016 public class ReportService {
00017
00019   private final ApiClient client;
00020
00026   public ReportService(ApiClient client) {
00027     this.client = client;
00028   }
00029
00038   public List<Cashier> getCashierInfo() throws Exception {
00039     return client.getAllCashiers();
00040   }
00041
00051   public String checkMenuStatus() throws Exception {
00052     // Fetch lists to verify connectivity.
00053     List<Appetizer> appetizers = client.getAllAppetizers();
00054     List<Drink> drinks = client.getAllDrinks();
00055     List<MainCourse> mainCourses = client.getAllMainCourses();
00056
00057     return String.format(
00058         "Menu System Online: %d appetizers, %d drinks, %d main courses available.",
00059         appetizers.size(), drinks.size(), mainCourses.size());
00060   }
00061 }
```

8.88 Order.java File Reference

`import es.ull.esit.app.middleware.ApiClient;`
Include dependency graph for Order.java:



Classes

- class es.ull.esit.app.Order

  Main menu window for taking customer orders.

Packages

- package es.ull.esit.app

8.89 Order.java

Go to the documentation of this file.

```
00001 package es.ull.esit.app;
00002
00003 import es.ull.esit.app.middleware.ApiClient;
00004 import es.ull.esit.app.middleware.model.Appetizer;
00005 import es.ull.esit.app.middleware.model.BillResult;
00006 import es.ull.esit.app.middleware.model.Drink;
00007 import es.ull.esit.app.middleware.model.MainCourse;
00008 import es.ull.esit.app.middleware.service.OrderService;
00009 import es.ull.esit.app.middleware.service.ProductService;
00010
00011 import java.io.PrintWriter;
00012 import javax.swing.JOptionPane;
00013 import javax.swing.SwingUtilities;
00014 import javax.swing.table.DefaultTableModel;
00015
00028 public class Order extends javax.swing.JFrame {
00029
00031   private final ProductService productService;
00032
00034   private final OrderService orderService = new OrderService();
00035
00037   private DefaultTableModel drinksModel;
00038   private DefaultTableModel appetizersModel;
00039   private DefaultTableModel mainsModel;
00040
00042   private BillResult lastBill;
00043
00045   double subTotal;
00047   double vat;
```

```
00049   double total;
00050
00052   int receiptNo = 1;
00053
00058   PrintWriter output;
00059
00066   public Order() {
00067     initComponents();
00068
00069     // Initialize the Service Layer.
00070     ApiClient client = new ApiClient("http://localhost:8080");
00071     this.productService = new ProductService(client);
00072
00073     // Initialize receipt number label.
00074     receiptNoLbl.setText("Receipt No. : " + receiptNo);
00075
00076     // Configure tables and models.
00077     setupTables();
00078
00079     // Load menu items from database via REST API.
00080     loadMenuFromDatabase();
00081   }
00082
00092   private void setupTables() {
00093     // DRINKS
00094     drinksModel = new DefaultTableModel(
00095         new Object[] { "ID", "Item", "Price (SR)", "Qty" }, 0) {
00096       @Override
00097       public boolean isCellEditable(int row, int column) {
00098         // Only Qty column is editable
00099         return column == 3;
00100       }
00101
00102       @Override
00103       public Class<?> getColumnClass(int columnIndex) {
00104         return switch (columnIndex) {
00105           case 0 -> Long.class; // ID
00106           case 2, 3 -> Integer.class; // Price, Qty
00107           default -> String.class;
00108         };
00109       }
00110     };
00111     drinksTable.setModel(drinksModel);
00112
00113     // APPETIZERS
00114     appetizersModel = new DefaultTableModel(
00115         new Object[] { "ID", "Item", "Price (SR)", "Qty" }, 0) {
00116       @Override
00117       public boolean isCellEditable(int row, int column) {
00118         return column == 3;
00119       }
00120
00121       @Override
00122       public Class<?> getColumnClass(int columnIndex) {
00123         return switch (columnIndex) {
00124           case 0 -> Long.class;
00125           case 2, 3 -> Integer.class;
00126           default -> String.class;
00127         };
00128       }
00129     };
00130     appetizersTable.setModel(appetizersModel);
00131
00132     // MAIN COURSES
00133     mainsModel = new DefaultTableModel(
00134         new Object[] { "ID", "Item", "Price (SR)", "Qty" }, 0) {
00135       @Override
00136       public boolean isCellEditable(int row, int column) {
00137         return column == 3;
00138       }
00139
00140       @Override
00141       public Class<?> getColumnClass(int columnIndex) {
00142         return switch (columnIndex) {
00143           case 0 -> Long.class;
00144           case 2, 3 -> Integer.class;
00145           default -> String.class;
00146         };
00147       }
00148     };
00149     mainsTable.setModel(mainsModel);
00150   }
00151
00156   private void loadMenuFromDatabase() {
00157     new Thread(() -> {
00158       try {
00159         java.util.List<Drink> drinks = productService.getAllDrinks();
```

```
00160         java.util.List<Appetizer> appetizers = productService.getAllAppetizers();
00161         java.util.List<MainCourse> mains = productService.getAllMainCourses();
00162
00163         SwingUtilities.invokeLater(() -> {
00164           fillDrinks(drinks);
00165           fillAppetizers(appetizers);
00166           fillMains(mains);
00167         });
00168
00169       } catch (Exception ex) {
00170         ex.printStackTrace();
00171         SwingUtilities.invokeLater(() -> JOptionPane.showMessageDialog(
00172             this,
00173             "Error loading menu from backend:\n" + ex.getMessage(),
00174             "Menu loading error",
00175             JOptionPane.ERROR_MESSAGE));
00176       }
00177     }).start();
00178   }
00179
00185   private void fillDrinks(java.util.List<Drink> drinks) {
00186     drinksModel.setRowCount(0);
00187     for (Drink d : drinks) {
00188       drinksModel.addRow(new Object[] {
00189           d.getDrinksId(),
00190           d.getItemDrinks(),
00191           d.getDrinksPrice(),
00192           0 // initial quantity
00193       });
00194     }
00195   }
00196
00202   private void fillAppetizers(java.util.List<Appetizer> appetizers) {
00203     appetizersModel.setRowCount(0);
00204     for (Appetizer a : appetizers) {
00205       appetizersModel.addRow(new Object[] {
00206           a.getAppetizersId(),
00207           a.getItemAppetizers(),
00208           a.getAppetizersPrice(),
00209           0
00210       });
00211     }
00212   }
00213
00219   private void fillMains(java.util.List<MainCourse> mains) {
00220     mainsModel.setRowCount(0);
00221     for (MainCourse m : mains) {
00222       mainsModel.addRow(new Object[] {
00223           m.getFoodId(),
00224           m.getItemFood(),
00225           m.getFoodPrice(),
00226           0
00227       });
00228     }
00229   }
00230
00241   private double sumFromModel(DefaultTableModel model) {
00242     double sum = 0.0;
00243     for (int row = 0; row < model.getRowCount(); row++) {
00244       Object qtyObj = model.getValueAt(row, 3); // Qty
00245       Object priceObj = model.getValueAt(row, 2); // Price
00246
00247       if (qtyObj == null || priceObj == null)
00248         continue;
00249
00250       int qty;
00251       int price;
00252       try {
00253         qty = ((Number) qtyObj).intValue();
00254         price = ((Number) priceObj).intValue();
00255       } catch (ClassCastException e) {
00256         // Fallback in case something comes as String
00257         qty = Integer.parseInt(qtyObj.toString());
00258         price = Integer.parseInt(priceObj.toString());
00259       }
00260
00261       if (qty > 0) {
00262         sum += qty * price;
00263       }
00264     }
00265     return sum;
00266   }
00267
00271   private void resetQtyColumn(DefaultTableModel model) {
00272     for (int row = 0; row < model.getRowCount(); row++) {
00273       model.setValueAt(0, row, 3); // column 3 is Qty
00274     }
```

```
00275   }
00276
00283   void loadPrice() {
00284     // No-op
00285   }
00286
00295   @SuppressWarnings("unchecked")
00296   // <editor-fold defaultstate="collapsed" desc="Generated
00297   // Code">//GEN-BEGIN:initComponents
00298   private void initComponents() {
00299
00300     jPanel2 = new javax.swing.JPanel();
00301     DrinksPnl = new javax.swing.JPanel();
00302     drinksScroll = new javax.swing.JScrollPane();
00303     drinksTable = new javax.swing.JTable();
00304     AppetizerPnl = new javax.swing.JPanel();
00305     appetizersScroll = new javax.swing.JScrollPane();
00306     appetizersTable = new javax.swing.JTable();
00307     MainCoursePnl = new javax.swing.JPanel();
00308     mainsScroll = new javax.swing.JScrollPane();
00309     mainsTable = new javax.swing.JTable();
00310     jPanel1 = new javax.swing.JPanel();
00311     subTotalLbl = new javax.swing.JLabel();
00312     vatLbl = new javax.swing.JLabel();
00313     totalLbl = new javax.swing.JLabel();
00314     receiptNoLbl = new javax.swing.JLabel();
00315     payBtn = new javax.swing.JButton();
00316     newReceiptBtn = new javax.swing.JButton();
00317     saveReceiptBtn = new javax.swing.JButton();
00318     jLabel1 = new javax.swing.JLabel();
00319     goBackMenueBtn = new javax.swing.JButton();
00320     jLabel2 = new javax.swing.JLabel();
00321
00322     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00323     setTitle("Menu");
00324     setBackground(new java.awt.Color(204, 204, 204));
00325     setResizable(false);
00326
00327     jPanel2.setBackground(new java.awt.Color(248, 244, 230));
00328
00329     DrinksPnl.setBackground(new java.awt.Color(248, 244, 230));
00330     DrinksPnl.setBorder(javax.swing.BorderFactory.createTitledBorder(
00331         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Drinks",
00332         javax.swing.border.TitledBorder.CENTER, javax.swing.border.TitledBorder.TOP,
00333         new java.awt.Font("Yu Gothic UI", 0, 18))); // NOI18N
00334     DrinksPnl.setPreferredSize(new java.awt.Dimension(260, 278));
00335
00336     drinksTable.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00337     drinksTable.setModel(new javax.swing.table.DefaultTableModel(
00338         new Object[][] {
00339
00340         },
00341         new String[] {
00342
00343         }));
00344     drinksTable.getTableHeader().setReorderingAllowed(false);
00345     drinksScroll.setViewportView(drinksTable);
00346
00347     javax.swing.GroupLayout DrinksPnlLayout = new javax.swing.GroupLayout(DrinksPnl);
00348     DrinksPnl.setLayout(DrinksPnlLayout);
00349     DrinksPnlLayout.setHorizontalGroup(
00350         DrinksPnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00351             .addGroup(DrinksPnlLayout.createSequentialGroup()
00352                 .addContainerGap()
00353                 .addComponent(drinksScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 281,
      Short.MAX_VALUE)
00354                 .addContainerGap()));
00355     DrinksPnlLayout.setVerticalGroup(
00356         DrinksPnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00357             .addGroup(DrinksPnlLayout.createSequentialGroup()
00358                 .addContainerGap()
00359                 .addComponent(drinksScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 186,
      Short.MAX_VALUE)
00360                 .addContainerGap()));
00361
00362     AppetizerPnl.setBackground(new java.awt.Color(248, 244, 230));
00363     AppetizerPnl.setBorder(javax.swing.BorderFactory.createTitledBorder(
00364         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Appetizers",
00365         javax.swing.border.TitledBorder.CENTER, javax.swing.border.TitledBorder.TOP,
00366         new java.awt.Font("Yu Gothic UI", 0, 18))); // NOI18N
00367     AppetizerPnl.setPreferredSize(new java.awt.Dimension(260, 278));
00368
00369     appetizersTable.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00370     appetizersTable.setModel(new javax.swing.table.DefaultTableModel(
00371         new Object[][] {
00372
00373         },
```

```
00374         new String[] {
00375
00376         }));
00377     appetizersTable.getTableHeader().setReorderingAllowed(false);
00378     appetizersScroll.setViewportView(appetizersTable);
00379
00380     javax.swing.GroupLayout AppetizerPnlLayout = new javax.swing.GroupLayout(AppetizerPnl);
00381     AppetizerPnl.setLayout(AppetizerPnlLayout);
00382     AppetizerPnlLayout.setHorizontalGroup(
00383         AppetizerPnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00384             .addGroup(AppetizerPnlLayout.createSequentialGroup()
00385                 .addContainerGap()
00386                 .addComponent(appetizersScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 368,
      Short.MAX_VALUE)
00387                 .addContainerGap()));
00388     AppetizerPnlLayout.setVerticalGroup(
00389         AppetizerPnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00390             .addGroup(AppetizerPnlLayout.createSequentialGroup()
00391                 .addContainerGap()
00392                 .addComponent(appetizersScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 186,
      Short.MAX_VALUE)
00393                 .addContainerGap()));
00394
00395     MainCoursePnl.setBackground(new java.awt.Color(248, 244, 230));
00396     MainCoursePnl.setBorder(javax.swing.BorderFactory.createTitledBorder(
00397         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Main Course",
00398         javax.swing.border.TitledBorder.CENTER, javax.swing.border.TitledBorder.TOP,
00399         new java.awt.Font("Yu Gothic UI", 0, 18))); // NOI18N
00400     MainCoursePnl.setPreferredSize(new java.awt.Dimension(260, 278));
00401
00402     mainsTable.setFont(new java.awt.Font("Yu Gothic UI", 0, 14)); // NOI18N
00403     mainsTable.setModel(new javax.swing.table.DefaultTableModel(
00404         new Object[][] {
00405
00406         },
00407         new String[] {
00408
00409         }));
00410     mainsTable.getTableHeader().setReorderingAllowed(false);
00411     mainsScroll.setViewportView(mainsTable);
00412
00413     javax.swing.GroupLayout MainCoursePnlLayout = new javax.swing.GroupLayout(MainCoursePnl);
00414     MainCoursePnl.setLayout(MainCoursePnlLayout);
00415     MainCoursePnlLayout.setHorizontalGroup(
00416         MainCoursePnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00417             .addGroup(MainCoursePnlLayout.createSequentialGroup()
00418                 .addContainerGap()
00419                 .addComponent(mainsScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 427, Short.MAX_VALUE)
00420                 .addContainerGap()));
00421     MainCoursePnlLayout.setVerticalGroup(
00422         MainCoursePnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00423             .addGroup(MainCoursePnlLayout.createSequentialGroup()
00424                 .addContainerGap()
00425                 .addComponent(mainsScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 186, Short.MAX_VALUE)
00426                 .addContainerGap()));
00427
00428     jPanel1.setBackground(new java.awt.Color(248, 244, 230));
00429     jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(
00430         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Receipt",
00431         javax.swing.border.TitledBorder.CENTER, javax.swing.border.TitledBorder.TOP,
00432         new java.awt.Font("Yu Gothic UI", 0, 14))); // NOI18N
00433
00434     subTotalLbl.setFont(new java.awt.Font("Yu Gothic UI Light", 0, 18)); // NOI18N
00435     subTotalLbl.setText("SubTotal: 0.0 SR");
00436
00437     vatLbl.setFont(new java.awt.Font("Yu Gothic UI Light", 0, 18)); // NOI18N
00438     vatLbl.setText("VAT included: 0.0 SR");
00439
00440     totalLbl.setFont(new java.awt.Font("Yu Gothic UI Light", 0, 18)); // NOI18N
00441     totalLbl.setText("Total: 0.0 SR");
00442
00443     receiptNoLbl.setFont(new java.awt.Font("Yu Gothic UI Light", 0, 14)); // NOI18N
00444     receiptNoLbl.setText("Receipt No. : 0");
00445
00446     javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
00447     jPanel1.setLayout(jPanel1Layout);
00448     jPanel1Layout.setHorizontalGroup(
00449         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00450             .addGroup(jPanel1Layout.createSequentialGroup()
00451                 .addContainerGap()
00452                 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00453                     .addComponent(subTotalLbl)
00454                     .addComponent(vatLbl)
00455                     .addComponent(totalLbl)
00456                     .addComponent(receiptNoLbl))
00457                 .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)));
00458     jPanel1Layout.setVerticalGroup(
```

```
00459         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00460             .addGroup(jPanel1Layout.createSequentialGroup()
00461                 .addContainerGap()
00462                 .addComponent(subTotalLbl)
00463                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
00464                 .addComponent(vatLbl)
00465                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
00466                 .addComponent(totalLbl)
00467                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 10,
      Short.MAX_VALUE)
00468                 .addComponent(receiptNoLbl)));
00469 
00470     payBtn.setFont(new java.awt.Font("Lucida Grande", 1, 18)); // NOI18N
00471     payBtn.setText("Pay");
00472     payBtn.addActionListener(new java.awt.event.ActionListener() {
00473       public void actionPerformed(java.awt.event.ActionEvent evt) {
00474         payBtnActionPerformed(evt);
00475       }
00476     });
00477 
00478     newReceiptBtn.setFont(new java.awt.Font("Yu Gothic UI", 1, 14)); // NOI18N
00479     newReceiptBtn.setText("New Receipt");
00480     newReceiptBtn.addActionListener(new java.awt.event.ActionListener() {
00481       public void actionPerformed(java.awt.event.ActionEvent evt) {
00482         newReceiptBtnActionPerformed(evt);
00483       }
00484     });
00485 
00486     saveReceiptBtn.setFont(new java.awt.Font("Yu Gothic UI", 1, 14)); // NOI18N
00487     saveReceiptBtn.setText("Save Receipt");
00488     saveReceiptBtn.addActionListener(new java.awt.event.ActionListener() {
00489       public void actionPerformed(java.awt.event.ActionEvent evt) {
00490         saveReceiptBtnActionPerformed(evt);
00491       }
00492     });
00493 
00494     jLabel1.setBackground(new java.awt.Color(255, 153, 0));
00495     jLabel1.setFont(new java.awt.Font("Yu Gothic UI", 1, 36)); // NOI18N
00496     jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00497     jLabel1.setText("BLACK PLATE MENU");
00498 
00499     goBackMenueBtn.setFont(new java.awt.Font("Yu Gothic UI", 1, 14)); // NOI18N
00500     goBackMenueBtn.setText("Go Back");
00501     goBackMenueBtn.addActionListener(new java.awt.event.ActionListener() {
00502       public void actionPerformed(java.awt.event.ActionEvent evt) {
00503         goBackMenueBtnActionPerformed(evt);
00504       }
00505     });
00506 
00507     jLabel2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/ui/screenshot.png"))); //
      NOI18N
00508 
00509     javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);
00510     jPanel2.setLayout(jPanel2Layout);
00511     jPanel2Layout.setHorizontalGroup(
00512         jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00513             .addGroup(jPanel2Layout.createSequentialGroup()
00514                 .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00515                     .addGroup(jPanel2Layout.createSequentialGroup()
00516                         .addGap(40, 40, 40)
00517                         .addComponent(DrinksPnl, javax.swing.GroupLayout.PREFERRED_SIZE, 305,
00518                             javax.swing.GroupLayout.PREFERRED_SIZE)
00519                         .addGap(18, 18, 18)
00520                         .addComponent(AppetizerPnl, javax.swing.GroupLayout.PREFERRED_SIZE, 392,
00521                             javax.swing.GroupLayout.PREFERRED_SIZE)
00522                         .addGap(18, 18, 18)
00523                         .addComponent(MainCoursePnl, javax.swing.GroupLayout.PREFERRED_SIZE, 451,
00524                             javax.swing.GroupLayout.PREFERRED_SIZE))
00525                     .addGroup(jPanel2Layout.createSequentialGroup()
00526                         .addGap(251, 251, 251)
00527                         .addComponent(jLabel2)
00528                         .addGap(18, 18, 18)
00529                         .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 433,
00530                             javax.swing.GroupLayout.PREFERRED_SIZE))
00531                     .addGroup(jPanel2Layout.createSequentialGroup()
00532                         .addGap(16, 16, 16)
00533                         .addComponent(goBackMenueBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 130,
00534                             javax.swing.GroupLayout.PREFERRED_SIZE)
00535                         .addGap(147, 147, 147)
00536                         .addComponent(payBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 180,
00537                             javax.swing.GroupLayout.PREFERRED_SIZE)
00538                         .addGap(18, 18, 18)
00539 
      .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00540                         .addComponent(newReceiptBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 220,
00541                             javax.swing.GroupLayout.PREFERRED_SIZE)
00542                         .addComponent(saveReceiptBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 220,
```

```
00543                                 javax.swing.GroupLayout.PREFERRED_SIZE))
00544                         .addGap(28, 28, 28)
00545                         .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
00546                             javax.swing.GroupLayout.DEFAULT_SIZE,
      javax.swing.GroupLayout.PREFERRED_SIZE)))
00547                 .addContainerGap(30, Short.MAX_VALUE)));
00548     jPanel2Layout.setVerticalGroup(
00549         jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00550             .addGroup(jPanel2Layout.createSequentialGroup()
00551                 .addContainerGap(16, Short.MAX_VALUE)
00552                 .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00553                     .addComponent(jLabel2, javax.swing.GroupLayout.Alignment.TRAILING)
00554                     .addComponent(jLabel1, javax.swing.GroupLayout.Alignment.TRAILING,
00555                         javax.swing.GroupLayout.PREFERRED_SIZE, 60,
      javax.swing.GroupLayout.PREFERRED_SIZE))
00556                 .addGap(40, 40, 40)
00557                 .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING,
      false)
00558                     .addComponent(AppetizerPnl, javax.swing.GroupLayout.DEFAULT_SIZE, 230,
      Short.MAX_VALUE)
00559                     .addComponent(MainCoursePnl, javax.swing.GroupLayout.DEFAULT_SIZE, 230,
      Short.MAX_VALUE)
00560                     .addComponent(DrinksPnl, javax.swing.GroupLayout.DEFAULT_SIZE, 230,
      Short.MAX_VALUE))
00561                 .addGap(18, 18, 18)
00562                 .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00563                     .addComponent(goBackMenueBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 40,
00564                         javax.swing.GroupLayout.PREFERRED_SIZE)
00565                     .addGroup(jPanel2Layout.createSequentialGroup()
00566                         .addComponent(newReceiptBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 30,
00567                             javax.swing.GroupLayout.PREFERRED_SIZE)
00568                         .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
00569                         .addComponent(saveReceiptBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 29,
00570                             javax.swing.GroupLayout.PREFERRED_SIZE))
00571                     .addComponent(payBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 70,
00572                         javax.swing.GroupLayout.PREFERRED_SIZE)
00573                     .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00574                         javax.swing.GroupLayout.PREFERRED_SIZE))
00575                 .addGap(30, 30, 30)));
00576
00577     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00578     getContentPane().setLayout(layout);
00579     layout.setHorizontalGroup(
00580         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00581             .addComponent(jPanel2, javax.swing.GroupLayout.DEFAULT_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00582                 Short.MAX_VALUE));
00583     layout.setVerticalGroup(
00584         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00585             .addGroup(layout.createSequentialGroup()
00586                 .addComponent(jPanel2, javax.swing.GroupLayout.DEFAULT_SIZE,
      javax.swing.GroupLayout.DEFAULT_SIZE,
00587                     Short.MAX_VALUE)
00588                 .addContainerGap()));
00589
00590     pack();
00591     setLocationRelativeTo(null);
00592   }// </editor-fold>//GEN-END:initComponents
00593
00607   private void payBtnActionPerformed(java.awt.event.ActionEvent evt) {//
      GEN-FIRST:event_payBtnActionPerformed
00608     double itemsTotal = 0.0;
00609
00610     itemsTotal += sumFromModel(drinksModel);
00611     itemsTotal += sumFromModel(appetizersModel);
00612     itemsTotal += sumFromModel(mainsModel);
00613
00614     if (itemsTotal == 0.0) {
00615       JOptionPane.showMessageDialog(
00616           this,
00617           "Please select at least one item (Qty > 0).",
00618           "No items selected",
00619           JOptionPane.INFORMATION_MESSAGE);
00620       return;
00621     }
00622
00623     // Calculate bill using the dedicated service
00624     lastBill = orderService.calculateBill(itemsTotal);
00625
00626     subTotal = lastBill.getSubTotal();
00627     vat = lastBill.getVat();
00628     total = lastBill.getTotal();
00629
00630     subTotalLbl.setText("SubTotal: " + subTotal + " SR");
00631     vatLbl.setText("VAT included: " + vat + " SR");
00632     totalLbl.setText("Total: " + total + " SR");
```

```
00633
00634       JOptionPane.showMessageDialog(this, "Paid successfully");
00635     }// GEN-LAST:event_payBtnActionPerformed
00636
00650     private void saveReceiptBtnActionPerformed(java.awt.event.ActionEvent evt) {//
      GEN-FIRST:event_saveReceiptBtnActionPerformed
00651       try {
00652         if (lastBill == null || lastBill.getTotal() == 0.0) {
00653           JOptionPane.showMessageDialog(
00654               this,
00655               "There is no paid bill to save.\nPlease press Pay first.",
00656               "No bill",
00657               JOptionPane.INFORMATION_MESSAGE);
00658           return;
00659         }
00660
00661         orderService.generateReceiptFile(receiptNo, lastBill);
00662
00663         JOptionPane.showMessageDialog(
00664             this,
00665             "Receipt number: " + receiptNo + " has been saved successfully.",
00666             "Receipt saved",
00667             JOptionPane.INFORMATION_MESSAGE);
00668
00669       } catch (Exception ex) {
00670         ex.printStackTrace();
00671         JOptionPane.showMessageDialog(
00672             this,
00673             "Error saving receipt:\n" + ex.getMessage(),
00674             "Error",
00675             JOptionPane.ERROR_MESSAGE);
00676       }
00677     }// GEN-LAST:event_saveReceiptBtnActionPerformed
00678
00692     private void newReceiptBtnActionPerformed(java.awt.event.ActionEvent evt) {//
      GEN-FIRST:event_newReceiptBtnActionPerformed
00693       if (total == 0.0) {
00694         // Nothing to reset
00695         return;
00696       }
00697
00698       resetQtyColumn(drinksModel);
00699       resetQtyColumn(appetizersModel);
00700       resetQtyColumn(mainsModel);
00701
00702       subTotal = 0.0;
00703       vat = 0.0;
00704       total = 0.0;
00705       lastBill = null;
00706
00707       subTotalLbl.setText("SubTotal: 0.0 SR");
00708       vatLbl.setText("VAT included: 0.0 SR");
00709       totalLbl.setText("Total: 0.0 SR");
00710
00711       receiptNo++;
00712       receiptNoLbl.setText("Receipt No. : " + receiptNo);
00713     }// GEN-LAST:event_newReceiptBtnActionPerformed
00714
00724     private void goBackMenueBtnActionPerformed(java.awt.event.ActionEvent evt) {//
      GEN-FIRST:event_goBackMenueBtnActionPerformed
00725       this.dispose();
00726       new Login().setVisible(true);
00727     }// GEN-LAST:event_goBackMenueBtnActionPerformed
00728
00738     public static void main(String args[]) {
00739       /* Set the Nimbus look and feel */
00740       // <editor-fold defaultstate="collapsed" desc=" Look and feel setting code
00741       // (optional) ">
00742       /*
00743        * If Nimbus (introduced in Java SE 6) is not available, stay with the default
00744        * look and feel.
00745        * For details see
00746        * http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
00747        */
00748       try {
00749         for (javax.swing.UIManager.LookAndFeelInfo info :
      javax.swing.UIManager.getInstalledLookAndFeels()) {
00750           if ("Nimbus".equals(info.getName())) {
00751             javax.swing.UIManager.setLookAndFeel(info.getClassName());
00752             break;
00753           }
00754         }
00755       } catch (ClassNotFoundException ex) {
00756         java.util.logging.Logger.getLogger(Order.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
00757       } catch (InstantiationException ex) {
00758         java.util.logging.Logger.getLogger(Order.class.getName()).log(java.util.logging.Level.SEVERE,
```

```
      null, ex);
00759     } catch (IllegalAccessException ex) {
00760       java.util.logging.Logger.getLogger(Order.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
00761     } catch (javax.swing.UnsupportedLookAndFeelException ex) {
00762       java.util.logging.Logger.getLogger(Order.class.getName()).log(java.util.logging.Level.SEVERE,
      null, ex);
00763     }
00764     // </editor-fold>
00765
00766     /* Create and display the form */
00767     java.awt.EventQueue.invokeLater(new Runnable() {
00768       public void run() {
00769         new Order().setVisible(true);
00770       }
00771     });
00772   }
00773
00774   // Variables declaration - do not modify//GEN-BEGIN:variables
00776   private javax.swing.JPanel AppetizerPnl;
00778   private javax.swing.JPanel DrinksPnl;
00780   private javax.swing.JPanel MainCoursePnl;
00782   private javax.swing.JTable appetizersTable;
00784   private javax.swing.JScrollPane appetizersScroll;
00786   private javax.swing.JTable drinksTable;
00788   private javax.swing.JScrollPane drinksScroll;
00790   private javax.swing.JButton goBackMenueBtn;
00792   private javax.swing.JLabel jLabel1;
00794   private javax.swing.JLabel jLabel2;
00796   private javax.swing.JPanel jPanel1;
00798   private javax.swing.JPanel jPanel2;
00800   private javax.swing.JTable mainsTable;
00802   private javax.swing.JScrollPane mainsScroll;
00804   private javax.swing.JButton newReceiptBtn;
00806   private javax.swing.JButton payBtn;
00808   private javax.swing.JLabel receiptNoLbl;
00810   private javax.swing.JButton saveReceiptBtn;
00812   private javax.swing.JLabel subTotalLbl;
00814   private javax.swing.JLabel totalLbl;
00816   private javax.swing.JLabel vatLbl;
00817   // End of variables declaration//GEN-END:variables
00818 }
```

Index