

**restaurant-system**

Generated by Doxygen 1.9.8



---

<b>1 README</b>	<b>1</b>
1.0.1 Configuración de la BBDD (MySQL / MariaDB) . . . . .	3
<b>2 Namespace Index</b>	<b>5</b>
2.1 Namespace List . . . . .	5
<b>3 Hierarchical Index</b>	<b>7</b>
3.1 Class Hierarchy . . . . .	7
<b>4 Class Index</b>	<b>9</b>
4.1 Class List . . . . .	9
<b>5 File Index</b>	<b>11</b>
5.1 File List . . . . .	11
<b>6 Namespace Documentation</b>	<b>13</b>
6.1 Package es.ull.esit.app . . . . .	13
6.2 Package es.ull.esit.app.middleware . . . . .	13
6.3 Package es.ull.esit.app.middleware.model . . . . .	14
6.4 Package es.ull.esit.app.middleware.service . . . . .	14
6.5 Package es.ull.esit.server . . . . .	14
6.6 Package es.ull.esit.server.config . . . . .	15
6.7 Package es.ull.esit.server.controller . . . . .	15
6.8 Package es.ull.esit.server.middleware.model . . . . .	15
6.9 Package es.ull.esit.server.repo . . . . .	16
<b>7 Class Documentation</b>	<b>17</b>
7.1 es.ull.esit.app.AboutUs Class Reference . . . . .	17
7.1.1 Detailed Description . . . . .	19
7.1.2 Constructor & Destructor Documentation . . . . .	19
7.1.2.1 AboutUs() . . . . .	19
7.1.3 Member Function Documentation . . . . .	20
7.1.3.1 initComponents() . . . . .	20
7.1.3.2 jButton3ActionPerformed() . . . . .	21
7.1.3.3 main() . . . . .	22
7.1.4 Member Data Documentation . . . . .	23
7.1.4.1 jButton3 . . . . .	23
7.1.4.2 jLabel1 . . . . .	23
7.1.4.3 jPanel1 . . . . .	23
7.1.4.4 jScrollPane1 . . . . .	23
7.1.4.5 ourStory . . . . .	23
7.1.4.6 textBlock . . . . .	24
7.2 es.ull.esit.app.AdminLogin Class Reference . . . . .	24
7.2.1 Detailed Description . . . . .	27

---

7.2.2 Constructor & Destructor Documentation . . . . .	28
7.2.2.1 AdminLogin() . . . . .	28
7.2.3 Member Function Documentation . . . . .	28
7.2.3.1 initComponents() . . . . .	28
7.2.3.2 jButton1ActionPerformed() . . . . .	30
7.2.3.3 jButton2ActionPerformed() . . . . .	31
7.2.3.4 jButton3ActionPerformed() . . . . .	31
7.2.3.5 main() . . . . .	32
7.2.4 Member Data Documentation . . . . .	32
7.2.4.1 jButton1 . . . . .	32
7.2.4.2 jButton2 . . . . .	33
7.2.4.3 jButton3 . . . . .	33
7.2.4.4 jLabel1 . . . . .	33
7.2.4.5 jLabel3 . . . . .	33
7.2.4.6 jPanel1 . . . . .	33
7.2.4.7 LOGGER . . . . .	34
7.2.4.8 PRIMARY_FONT . . . . .	34
7.3 es.ull.esit.app.AdminProducts Class Reference . . . . .	34
7.3.1 Detailed Description . . . . .	38
7.3.2 Constructor & Destructor Documentation . . . . .	38
7.3.2.1 AdminProducts() . . . . .	38
7.3.3 Member Function Documentation . . . . .	39
7.3.3.1 initComponents() . . . . .	39
7.3.3.2 jButton1ActionPerformed() . . . . .	45
7.3.3.3 jButton2ActionPerformed() . . . . .	46
7.3.3.4 jButton3ActionPerformed() . . . . .	46
7.3.3.5 jButton4ActionPerformed() . . . . .	47
7.3.3.6 jButton5ActionPerformed() . . . . .	47
7.3.3.7 jButton6ActionPerformed() . . . . .	48
7.3.3.8 jButton7ActionPerformed() . . . . .	48
7.3.3.9 jTable1KeyPressed() . . . . .	49
7.3.3.10 jTable1MouseClicked() . . . . .	49
7.3.3.11 jTable2MouseClicked() . . . . .	50
7.3.3.12 jTable3MouseClicked() . . . . .	51
7.3.3.13 main() . . . . .	51
7.3.3.14 refreshAllTables() . . . . .	52
7.3.4 Member Data Documentation . . . . .	52
7.3.4.1 FIRST_COLUMN_HEADER . . . . .	52
7.3.4.2 LOGGER . . . . .	53
7.3.4.3 PRIMARY_FONT . . . . .	53
7.3.4.4 productService . . . . .	53
7.3.4.5 SECOND_COLUMN_HEADER . . . . .	53

---

7.3.4.6 SECONDARY_FONT . . . . .	53
7.3.4.7 THIRD_COLUMN_HEADER . . . . .	54
7.3.4.8 UPDATE_STRING . . . . .	54
7.4 es.ull.esit.app.middleware.ApiClient Class Reference . . . . .	54
7.4.1 Detailed Description . . . . .	57
7.4.2 Constructor & Destructor Documentation . . . . .	57
7.4.2.1 ApiClient() . . . . .	57
7.4.3 Member Function Documentation . . . . .	58
7.4.3.1 createAppetizer() . . . . .	58
7.4.3.2 createDrink() . . . . .	58
7.4.3.3 createMainCourse() . . . . .	59
7.4.3.4 deleteAppetizer() . . . . .	60
7.4.3.5 deleteDrink() . . . . .	60
7.4.3.6 deleteMainCourse() . . . . .	61
7.4.3.7 getAllAppetizers() . . . . .	61
7.4.3.8 getAllCashiers() . . . . .	62
7.4.3.9 getAllDrinks() . . . . .	62
7.4.3.10 getAllMainCourses() . . . . .	63
7.4.3.11 getAppetizerById() . . . . .	64
7.4.3.12 getCashierById() . . . . .	64
7.4.3.13 getCashierByName() . . . . .	65
7.4.3.14 getDrinkById() . . . . .	65
7.4.3.15 getMainCourseById() . . . . .	66
7.4.3.16 login() [1/2] . . . . .	67
7.4.3.17 login() [2/2] . . . . .	67
7.4.3.18 updateAppetizer() . . . . .	68
7.4.3.19 updateCashier() . . . . .	69
7.4.3.20 updateDrink() . . . . .	69
7.4.3.21 updateMainCourse() . . . . .	70
7.4.4 Member Data Documentation . . . . .	71
7.4.4.1 API_APPETIZERS . . . . .	71
7.4.4.2 API_DRINKS . . . . .	71
7.4.4.3 API_LOGIN . . . . .	71
7.4.4.4 API_MAINCOURSES . . . . .	72
7.4.4.5 APPLICATION_JSON . . . . .	72
7.4.4.6 baseUrl . . . . .	72
7.4.4.7 CONTENT_TYPE . . . . .	72
7.4.4.8 http . . . . .	72
7.4.4.9 LOGGER . . . . .	73
7.4.4.10 mapper . . . . .	73
7.5 es.ull.esit.app.middleware.ApiClientException Class Reference . . . . .	73
7.5.1 Detailed Description . . . . .	74

7.5.2 Constructor & Destructor Documentation . . . . .	74
7.5.2.1 ApiClientException() [1/2] . . . . .	74
7.5.2.2 ApiClientException() [2/2] . . . . .	75
7.6 es.ull.esit.app.middleware.model.Appetizer Class Reference . . . . .	75
7.6.1 Detailed Description . . . . .	77
7.6.2 Constructor & Destructor Documentation . . . . .	77
7.6.2.1 Appetizer() [1/2] . . . . .	77
7.6.2.2 Appetizer() [2/2] . . . . .	77
7.6.3 Member Function Documentation . . . . .	78
7.6.3.1 getAppetizersId() . . . . .	78
7.6.3.2 getAppetizersPrice() . . . . .	78
7.6.3.3 getItemAppetizers() . . . . .	78
7.6.3.4 getReceiptId() . . . . .	79
7.6.3.5 setAppetizersId() . . . . .	79
7.6.3.6 setAppetizersPrice() . . . . .	79
7.6.3.7 setItemAppetizers() . . . . .	80
7.6.3.8 setReceiptId() . . . . .	80
7.6.4 Member Data Documentation . . . . .	80
7.6.4.1 appetizersId . . . . .	80
7.6.4.2 appetizersPrice . . . . .	81
7.6.4.3 itemAppetizers . . . . .	81
7.6.4.4 receiptId . . . . .	81
7.7 es.ull.esit.server.middleware.model.Appetizer Class Reference . . . . .	82
7.7.1 Detailed Description . . . . .	83
7.7.2 Constructor & Destructor Documentation . . . . .	83
7.7.2.1 Appetizer() [1/2] . . . . .	83
7.7.2.2 Appetizer() [2/2] . . . . .	83
7.7.3 Member Function Documentation . . . . .	84
7.7.3.1 getAppetizersId() . . . . .	84
7.7.3.2 getAppetizersPrice() . . . . .	84
7.7.3.3 getItemAppetizers() . . . . .	84
7.7.3.4 setAppetizersId() . . . . .	84
7.7.3.5 setAppetizersPrice() . . . . .	85
7.7.3.6 setItemAppetizers() . . . . .	85
7.7.4 Member Data Documentation . . . . .	85
7.7.4.1 appetizersId . . . . .	85
7.7.4.2 appetizersPrice . . . . .	86
7.7.4.3 itemAppetizers . . . . .	86
7.8 es.ull.esit.server.controller.AppetizerController Class Reference . . . . .	86
7.8.1 Detailed Description . . . . .	87
7.8.2 Member Function Documentation . . . . .	87
7.8.2.1 createAppetizer() . . . . .	87

---

7.8.2.2 deleteAppetizer()	87
7.8.2.3 getAllAppetizers()	88
7.8.2.4 getAppetizerById()	88
7.8.2.5 updateAppetizer()	89
7.8.3 Member Data Documentation	89
7.8.3.1 appetizerRepository	89
7.9 es.ull.esit.server.repo.AppetizerRepository Interface Reference	90
7.9.1 Detailed Description	90
7.10 es.ull.esit.app.ApplicationLauncher Class Reference	91
7.10.1 Detailed Description	91
7.10.2 Member Function Documentation	92
7.10.2.1 main()	92
7.10.3 Member Data Documentation	92
7.10.3.1 loginFactory	92
7.11 es.ull.esit.server.controller.AuthController Class Reference	93
7.11.1 Detailed Description	93
7.11.2 Member Function Documentation	94
7.11.2.1 login()	94
7.11.3 Member Data Documentation	94
7.11.3.1 LOG	94
7.11.3.2 passwordEncoder	95
7.11.3.3 userRepository	95
7.12 es.ull.esit.app.middleware.service.AuthService Class Reference	95
7.12.1 Detailed Description	97
7.12.2 Constructor & Destructor Documentation	97
7.12.2.1 AuthService()	97
7.12.3 Member Function Documentation	97
7.12.3.1 authenticate()	97
7.12.4 Member Data Documentation	98
7.12.4.1 client	98
7.13 es.ull.esit.app.middleware.model.BillResult Class Reference	99
7.13.1 Detailed Description	99
7.13.2 Constructor & Destructor Documentation	100
7.13.2.1 BillResult()	100
7.13.3 Member Function Documentation	100
7.13.3.1 getSubTotal()	100
7.13.3.2 getTotal()	100
7.13.3.3 getVat()	101
7.13.4 Member Data Documentation	101
7.13.4.1 subTotal	101
7.13.4.2 total	101
7.13.4.3 vat	101

7.14 es.ull.esit.app.middleware.model.Cashier Class Reference . . . . .	102
7.14.1 Detailed Description . . . . .	103
7.14.2 Constructor & Destructor Documentation . . . . .	103
7.14.2.1 Cashier() [1/2] . . . . .	103
7.14.2.2 Cashier() [2/2] . . . . .	103
7.14.3 Member Function Documentation . . . . .	104
7.14.3.1 getId() . . . . .	104
7.14.3.2 getName() . . . . .	104
7.14.3.3 getSalary() . . . . .	104
7.14.3.4 setId() . . . . .	104
7.14.3.5 setName() . . . . .	105
7.14.3.6 setSalary() . . . . .	105
7.14.4 Member Data Documentation . . . . .	105
7.14.4.1 id . . . . .	105
7.14.4.2 name . . . . .	106
7.14.4.3 salary . . . . .	106
7.15 es.ull.esit.server.middleware.model.Cashier Class Reference . . . . .	106
7.15.1 Detailed Description . . . . .	107
7.15.2 Constructor & Destructor Documentation . . . . .	107
7.15.2.1 Cashier() [1/2] . . . . .	107
7.15.2.2 Cashier() [2/2] . . . . .	107
7.15.3 Member Function Documentation . . . . .	108
7.15.3.1 getId() . . . . .	108
7.15.3.2 getName() . . . . .	108
7.15.3.3 getSalary() . . . . .	108
7.15.3.4 setId() . . . . .	109
7.15.3.5 setName() . . . . .	109
7.15.3.6 setSalary() . . . . .	109
7.15.4 Member Data Documentation . . . . .	110
7.15.4.1 id . . . . .	110
7.15.4.2 name . . . . .	110
7.15.4.3 salary . . . . .	110
7.16 es.ull.esit.server.controller.CashierController Class Reference . . . . .	110
7.16.1 Detailed Description . . . . .	111
7.16.2 Member Function Documentation . . . . .	111
7.16.2.1 getAllCashiers() . . . . .	111
7.16.2.2 getCashierById() . . . . .	111
7.16.2.3 getCashierByName() . . . . .	112
7.16.2.4 updateCashier() . . . . .	112
7.16.3 Member Data Documentation . . . . .	113
7.16.3.1 cashierRepository . . . . .	113
7.17 es.ull.esit.app.CashierLogin Class Reference . . . . .	113

---

7.17.1 Detailed Description . . . . .	117
7.17.2 Constructor & Destructor Documentation . . . . .	117
7.17.2.1 CashierLogin() [1/2] . . . . .	117
7.17.2.2 CashierLogin() [2/2] . . . . .	117
7.17.3 Member Function Documentation . . . . .	118
7.17.3.1 initComponents() . . . . .	118
7.17.3.2 jButton1ActionPerformed() . . . . .	120
7.17.3.3 jButton2ActionPerformed() . . . . .	120
7.17.3.4 jButton3ActionPerformed() . . . . .	121
7.17.3.5 main() . . . . .	122
7.17.3.6 updateWelcomeMessage() . . . . .	122
7.17.4 Member Data Documentation . . . . .	123
7.17.4.1 cashierSupplier . . . . .	123
7.17.4.2 ERROR_CHECK_MENU_STATUS . . . . .	123
7.17.4.3 jButton1 . . . . .	123
7.17.4.4 jButton2 . . . . .	124
7.17.4.5 jButton3 . . . . .	124
7.17.4.6 jLabel1 . . . . .	124
7.17.4.7 jPanel1 . . . . .	124
7.17.4.8 LOGGER . . . . .	124
7.17.4.9 PRIMARY_FONT . . . . .	125
7.17.4.10 reportService . . . . .	125
7.17.4.11 WARN_FETCH_CASHIER_STATS . . . . .	125
7.17.4.12 welcomeTxt . . . . .	125
7.18 es.ull.esit.server.repo.CashierRepository Interface Reference . . . . .	126
7.18.1 Detailed Description . . . . .	127
7.18.2 Member Function Documentation . . . . .	127
7.18.2.1 findByName() . . . . .	127
7.19 es.ull.esit.app.middleware.model.Drink Class Reference . . . . .	127
7.19.1 Detailed Description . . . . .	129
7.19.2 Constructor & Destructor Documentation . . . . .	129
7.19.2.1 Drink() [1/2] . . . . .	129
7.19.2.2 Drink() [2/2] . . . . .	129
7.19.3 Member Function Documentation . . . . .	130
7.19.3.1 getDrinksId() . . . . .	130
7.19.3.2 getDrinksPrice() . . . . .	130
7.19.3.3 getItemDrinks() . . . . .	130
7.19.3.4 getReceiptId() . . . . .	131
7.19.3.5 setDrinksId() . . . . .	131
7.19.3.6 setDrinksPrice() . . . . .	131
7.19.3.7 setItemDrinks() . . . . .	132
7.19.3.8 setReceiptId() . . . . .	132

7.19.4 Member Data Documentation . . . . .	132
7.19.4.1 drinkId . . . . .	132
7.19.4.2 drinkPrice . . . . .	133
7.19.4.3 itemDrinks . . . . .	133
7.19.4.4 receiptId . . . . .	133
7.20 es.ull.esit.server.middleware.model.Drink Class Reference . . . . .	134
7.20.1 Detailed Description . . . . .	135
7.20.2 Constructor & Destructor Documentation . . . . .	135
7.20.2.1 Drink() [1/2] . . . . .	135
7.20.2.2 Drink() [2/2] . . . . .	135
7.20.3 Member Function Documentation . . . . .	136
7.20.3.1 getDrinkId() . . . . .	136
7.20.3.2 getDrinkPrice() . . . . .	136
7.20.3.3 getItemDrinks() . . . . .	136
7.20.3.4 setDrinkId() . . . . .	136
7.20.3.5 setDrinkPrice() . . . . .	137
7.20.3.6 setItemDrinks() . . . . .	137
7.20.4 Member Data Documentation . . . . .	137
7.20.4.1 drinkId . . . . .	137
7.20.4.2 drinkPrice . . . . .	138
7.20.4.3 itemDrinks . . . . .	138
7.21 es.ull.esit.server.controller.DrinkController Class Reference . . . . .	138
7.21.1 Detailed Description . . . . .	139
7.21.2 Member Function Documentation . . . . .	139
7.21.2.1 createDrink() . . . . .	139
7.21.2.2 deleteDrink() . . . . .	139
7.21.2.3 getAllDrinks() . . . . .	140
7.21.2.4 getDrinkById() . . . . .	140
7.21.2.5 updateDrink() . . . . .	141
7.21.3 Member Data Documentation . . . . .	141
7.21.3.1 drinkRepository . . . . .	141
7.22 es.ull.esit.server.repo.DrinkRepository Interface Reference . . . . .	142
7.22.1 Detailed Description . . . . .	142
7.23 es.ull.esit.server.controller.HealthController Class Reference . . . . .	143
7.23.1 Detailed Description . . . . .	143
7.23.2 Member Function Documentation . . . . .	144
7.23.2.1 checkDatabase() . . . . .	144
7.23.2.2 health() . . . . .	144
7.23.3 Member Data Documentation . . . . .	145
7.23.3.1 dataSource . . . . .	145
7.24 es.ull.esit.app.middleware.ApiClient.IOCallable< T > Interface Template Reference . . . . .	145
7.24.1 Detailed Description . . . . .	146

---

7.24.2 Member Function Documentation . . . . .	146
7.24.2.1 call() . . . . .	146
7.25 es.ull.esit.app.Login Class Reference . . . . .	146
7.25.1 Detailed Description . . . . .	150
7.25.2 Constructor & Destructor Documentation . . . . .	150
7.25.2.1 Login() . . . . .	150
7.25.3 Member Function Documentation . . . . .	151
7.25.3.1 initComponents() . . . . .	151
7.25.3.2 jButton1ActionPerformed() . . . . .	152
7.25.3.3 main() . . . . .	153
7.25.3.4 usertypecmbActionPerformed() . . . . .	154
7.25.4 Member Data Documentation . . . . .	154
7.25.4.1 authService . . . . .	154
7.25.4.2 jButton1 . . . . .	154
7.25.4.3 jLabel1 . . . . .	155
7.25.4.4 jLabel3 . . . . .	155
7.25.4.5 jPanel1 . . . . .	155
7.25.4.6 jPasswordField1 . . . . .	155
7.25.4.7 LOGIN_STRING . . . . .	155
7.25.4.8 loginFactory . . . . .	156
7.25.4.9 PRIMARY_FONT . . . . .	156
7.25.4.10 usernametxt . . . . .	156
7.25.4.11 usertypecmb . . . . .	156
7.26 es.ull.esit.server.controller.AuthController.LoginRequest Class Reference . . . . .	157
7.26.1 Detailed Description . . . . .	157
7.26.2 Member Data Documentation . . . . .	157
7.26.2.1 password . . . . .	157
7.26.2.2 username . . . . .	157
7.27 es.ull.esit.app.middleware.model.MainCourse Class Reference . . . . .	158
7.27.1 Detailed Description . . . . .	159
7.27.2 Constructor & Destructor Documentation . . . . .	159
7.27.2.1 MainCourse() [1/2] . . . . .	159
7.27.2.2 MainCourse() [2/2] . . . . .	159
7.27.3 Member Function Documentation . . . . .	160
7.27.3.1 getFoodId() . . . . .	160
7.27.3.2 getFoodPrice() . . . . .	160
7.27.3.3 getItemFood() . . . . .	160
7.27.3.4 getReceiptId() . . . . .	161
7.27.3.5 setFoodId() . . . . .	161
7.27.3.6 setFoodPrice() . . . . .	161
7.27.3.7 setItemFood() . . . . .	162
7.27.3.8 setReceiptId() . . . . .	162

---

7.27.4 Member Data Documentation . . . . .	162
7.27.4.1 foodId . . . . .	162
7.27.4.2 foodPrice . . . . .	163
7.27.4.3 itemFood . . . . .	163
7.27.4.4 receiptId . . . . .	163
7.28 es.ull.esit.server.middleware.model.MainCourse Class Reference . . . . .	164
7.28.1 Detailed Description . . . . .	165
7.28.2 Constructor & Destructor Documentation . . . . .	165
7.28.2.1 MainCourse() [1/2] . . . . .	165
7.28.2.2 MainCourse() [2/2] . . . . .	165
7.28.3 Member Function Documentation . . . . .	166
7.28.3.1 getFoodId() . . . . .	166
7.28.3.2 getFoodPrice() . . . . .	166
7.28.3.3 getItemFood() . . . . .	166
7.28.3.4 setFoodId() . . . . .	166
7.28.3.5 setFoodPrice() . . . . .	167
7.28.3.6 setItemFood() . . . . .	167
7.28.4 Member Data Documentation . . . . .	167
7.28.4.1 foodId . . . . .	167
7.28.4.2 foodPrice . . . . .	168
7.28.4.3 itemFood . . . . .	168
7.29 es.ull.esit.server.controller.MainCourseController Class Reference . . . . .	168
7.29.1 Detailed Description . . . . .	169
7.29.2 Member Function Documentation . . . . .	169
7.29.2.1 createMainCourse() . . . . .	169
7.29.2.2 deleteMainCourse() . . . . .	169
7.29.2.3 getAllMainCourses() . . . . .	170
7.29.2.4 getMainCourseById() . . . . .	170
7.29.2.5 updateMainCourse() . . . . .	171
7.29.3 Member Data Documentation . . . . .	171
7.29.3.1 mainCourseRepository . . . . .	171
7.30 es.ull.esit.server.repo.MainCourseRepository Interface Reference . . . . .	172
7.30.1 Detailed Description . . . . .	172
7.31 es.ull.esit.server.controller.MenuController Class Reference . . . . .	173
7.31.1 Detailed Description . . . . .	174
7.31.2 Member Function Documentation . . . . .	174
7.31.2.1 getFullMenu() . . . . .	174
7.31.3 Member Data Documentation . . . . .	174
7.31.3.1 appetizerRepository . . . . .	174
7.31.3.2 drinkRepository . . . . .	175
7.31.3.3 mainCourseRepository . . . . .	175
7.32 es.ull.esit.app.Order Class Reference . . . . .	175

---

7.32.1 Detailed Description . . . . .	180
7.32.2 Constructor & Destructor Documentation . . . . .	180
7.32.2.1 Order() . . . . .	180
7.32.3 Member Function Documentation . . . . .	181
7.32.3.1 fillAppetizers() . . . . .	181
7.32.3.2 fillDrinks() . . . . .	181
7.32.3.3 fillMains() . . . . .	182
7.32.3.4 goBackMenueBtnActionPerformed() . . . . .	183
7.32.3.5 initComponents() . . . . .	183
7.32.3.6 loadMenuFromDatabase() . . . . .	187
7.32.3.7 main() . . . . .	188
7.32.3.8 newReceiptBtnActionPerformed() . . . . .	189
7.32.3.9 payBtnActionPerformed() . . . . .	190
7.32.3.10 resetQtyColumn() . . . . .	191
7.32.3.11 saveReceiptBtnActionPerformed() . . . . .	192
7.32.3.12 setupTables() . . . . .	192
7.32.3.13 sumFromModel() . . . . .	194
7.32.4 Member Data Documentation . . . . .	195
7.32.4.1 appetizerPnl . . . . .	195
7.32.4.2 appetizersModel . . . . .	195
7.32.4.3 appetizersScroll . . . . .	195
7.32.4.4 appetizersTable . . . . .	195
7.32.4.5 drinksModel . . . . .	195
7.32.4.6 drinksPnl . . . . .	196
7.32.4.7 drinksScroll . . . . .	196
7.32.4.8 drinksTable . . . . .	196
7.32.4.9 goBackMenueBtn . . . . .	196
7.32.4.10 ID_COLUMN_HEADER . . . . .	196
7.32.4.11 ITEM_COLUMN_HEADER . . . . .	197
7.32.4.12 jLabel1 . . . . .	197
7.32.4.13 jLabel2 . . . . .	197
7.32.4.14 jPanel1 . . . . .	197
7.32.4.15 jPanel2 . . . . .	197
7.32.4.16 lastBill . . . . .	198
7.32.4.17 LOGGER . . . . .	198
7.32.4.18 mainCoursePnl . . . . .	198
7.32.4.19 mainsModel . . . . .	198
7.32.4.20 mainsScroll . . . . .	198
7.32.4.21 mainsTable . . . . .	199
7.32.4.22 newReceiptBtn . . . . .	199
7.32.4.23 orderService . . . . .	199
7.32.4.24 payBtn . . . . .	199

---

7.32.4.25 PRICE_COLUMN_HEADER . . . . .	199
7.32.4.26 PRIMARY_FONT . . . . .	200
7.32.4.27 PRIMARY_FONT_LIGHT . . . . .	200
7.32.4.28 productService . . . . .	200
7.32.4.29 QUANTITY_COLUMN_HEADER . . . . .	200
7.32.4.30 RECEIPT_NO_PREFIX . . . . .	200
7.32.4.31 receiptNoLbl . . . . .	201
7.32.4.32 saveReceiptBtn . . . . .	201
7.32.4.33 subTotalLbl . . . . .	201
7.32.4.34 totalLbl . . . . .	201
7.32.4.35 vatLbl . . . . .	201
7.33 es.ull.esit.app.middleware.service.OrderService Class Reference . . . . .	202
7.33.1 Detailed Description . . . . .	202
7.33.2 Member Function Documentation . . . . .	202
7.33.2.1 calculateBill() . . . . .	202
7.33.2.2 generateReceiptFile() . . . . .	203
7.33.3 Member Data Documentation . . . . .	203
7.33.3.1 VAT_RATE . . . . .	203
7.34 es.ull.esit.app.middleware.service.ProductService Class Reference . . . . .	204
7.34.1 Detailed Description . . . . .	205
7.34.2 Constructor & Destructor Documentation . . . . .	205
7.34.2.1 ProductService() . . . . .	205
7.34.3 Member Function Documentation . . . . .	206
7.34.3.1 addAppetizer() . . . . .	206
7.34.3.2 addDrink() . . . . .	207
7.34.3.3 addMainCourse() . . . . .	207
7.34.3.4 getAllAppetizers() . . . . .	208
7.34.3.5 getAllDrinks() . . . . .	209
7.34.3.6 getAllMainCourses() . . . . .	209
7.34.3.7 getAppetizerById() . . . . .	209
7.34.3.8 getDrinkById() . . . . .	210
7.34.3.9 getMainCourseById() . . . . .	211
7.34.3.10 updateAppetizer() . . . . .	211
7.34.3.11 updateDrink() . . . . .	212
7.34.3.12 updateMainCourse() . . . . .	213
7.34.3.13 validateAndParsePrice() . . . . .	214
7.34.3.14 validateName() . . . . .	215
7.34.4 Member Data Documentation . . . . .	216
7.34.4.1 client . . . . .	216
7.35 es.ull.esit.app.middleware.service.ReportService Class Reference . . . . .	217
7.35.1 Detailed Description . . . . .	218
7.35.2 Constructor & Destructor Documentation . . . . .	218

---

7.35.2.1 ReportService() . . . . .	218
7.35.3 Member Function Documentation . . . . .	218
7.35.3.1 checkMenuStatus() . . . . .	218
7.35.3.2 getCashierInfo() . . . . .	219
7.35.4 Member Data Documentation . . . . .	220
7.35.4.1 client . . . . .	220
7.36 es.ull.esit.app.middleware.ApiClient.ResponseHandler< R > Interface Template Reference . . . . .	220
7.36.1 Detailed Description . . . . .	221
7.36.2 Member Function Documentation . . . . .	221
7.36.2.1 handle() . . . . .	221
7.37 es.ull.esit.server.RestaurantApplication Class Reference . . . . .	221
7.37.1 Detailed Description . . . . .	222
7.37.2 Member Function Documentation . . . . .	222
7.37.2.1 main() . . . . .	222
7.38 es.ull.esit.server.config.SecurityConfig Class Reference . . . . .	222
7.38.1 Detailed Description . . . . .	223
7.38.2 Member Function Documentation . . . . .	223
7.38.2.1 filterChain() . . . . .	223
7.38.2.2 passwordEncoder() . . . . .	224
7.39 es.ull.esit.app.middleware.model.User Class Reference . . . . .	224
7.39.1 Detailed Description . . . . .	225
7.39.2 Constructor & Destructor Documentation . . . . .	226
7.39.2.1 User() [1/2] . . . . .	226
7.39.2.2 User() [2/2] . . . . .	226
7.39.3 Member Function Documentation . . . . .	226
7.39.3.1 getRole() . . . . .	226
7.39.3.2 getUsername() . . . . .	227
7.39.3.3 isAdmin() . . . . .	227
7.39.3.4 setRole() . . . . .	227
7.39.3.5 setUsername() . . . . .	227
7.39.4 Member Data Documentation . . . . .	228
7.39.4.1 role . . . . .	228
7.39.4.2 username . . . . .	228
7.40 es.ull.esit.server.middleware.model.User Class Reference . . . . .	228
7.40.1 Detailed Description . . . . .	230
7.40.2 Member Function Documentation . . . . .	230
7.40.2.1 getId() . . . . .	230
7.40.2.2 getPasswordHash() . . . . .	230
7.40.2.3 getRole() . . . . .	230
7.40.2.4 getUsername() . . . . .	231
7.40.2.5 setId() . . . . .	231
7.40.2.6 setPasswordHash() . . . . .	231

7.40.2.7 setRole() . . . . .	231
7.40.2.8 setUsername() . . . . .	232
7.40.3 Member Data Documentation . . . . .	232
7.40.3.1 id . . . . .	232
7.40.3.2 passwordHash . . . . .	232
7.40.3.3 role . . . . .	232
7.40.3.4 username . . . . .	233
7.41 es.ull.esit.server.repo.UserRepository Interface Reference . . . . .	233
7.41.1 Detailed Description . . . . .	234
7.41.2 Member Function Documentation . . . . .	234
7.41.2.1 findByUsername() . . . . .	234
<b>8 File Documentation</b> . . . . .	<b>237</b>
8.1 00-create-db.sql File Reference . . . . .	237
8.2 00-create-db.sql . . . . .	237
8.3 01-tables.sql File Reference . . . . .	237
8.4 01-tables.sql . . . . .	237
8.5 02-procedures.sql File Reference . . . . .	240
8.6 02-procedures.sql . . . . .	240
8.7 03-triggers.sql File Reference . . . . .	240
8.8 03-triggers.sql . . . . .	240
8.9 04-privileges.sql File Reference . . . . .	241
8.10 04-privileges.sql . . . . .	241
8.11 05-data.sql File Reference . . . . .	241
8.12 05-data.sql . . . . .	241
8.13 init.sql File Reference . . . . .	243
8.14 init.sql . . . . .	243
8.15 README.md File Reference . . . . .	247
8.16 SecurityConfig.java File Reference . . . . .	247
8.17 SecurityConfig.java . . . . .	247
8.18 AppetizerController.java File Reference . . . . .	248
8.19 AppetizerController.java . . . . .	248
8.20 AuthController.java File Reference . . . . .	249
8.21 AuthController.java . . . . .	250
8.22 CashierController.java File Reference . . . . .	251
8.23 CashierController.java . . . . .	251
8.24 DrinkController.java File Reference . . . . .	252
8.25 DrinkController.java . . . . .	253
8.26 HealthController.java File Reference . . . . .	254
8.27 HealthController.java . . . . .	254
8.28 MainCourseController.java File Reference . . . . .	255
8.29 MainCourseController.java . . . . .	256

---

8.30 MenuController.java File Reference . . . . .	257
8.31 MenuController.java . . . . .	257
8.32 AppetizerRepository.java File Reference . . . . .	258
8.33 AppetizerRepository.java . . . . .	258
8.34 CashierRepository.java File Reference . . . . .	259
8.35 CashierRepository.java . . . . .	259
8.36 DrinkRepository.java File Reference . . . . .	260
8.37 DrinkRepository.java . . . . .	260
8.38 MainCourseRepository.java File Reference . . . . .	261
8.39 MainCourseRepository.java . . . . .	261
8.40 UserRepository.java File Reference . . . . .	262
8.41 UserRepository.java . . . . .	262
8.42 RestaurantApplication.java File Reference . . . . .	263
8.43 RestaurantApplication.java . . . . .	263
8.44 AboutUs.java File Reference . . . . .	263
8.45 AboutUs.java . . . . .	264
8.46 AdminLogin.java File Reference . . . . .	266
8.47 AdminLogin.java . . . . .	266
8.48 AdminProducts.java File Reference . . . . .	269
8.49 AdminProducts.java . . . . .	269
8.50 ApplicationLauncher.java File Reference . . . . .	281
8.51 ApplicationLauncher.java . . . . .	281
8.52 CashierLogin.java File Reference . . . . .	282
8.53 CashierLogin.java . . . . .	282
8.54 Login.java File Reference . . . . .	286
8.55 Login.java . . . . .	287
8.56 ApiClient.java File Reference . . . . .	290
8.57 ApiClient.java . . . . .	291
8.58 ApiClientException.java File Reference . . . . .	295
8.59 ApiClientException.java . . . . .	295
8.60 server/src/main/java/es/ull/esit/server/middleware/model/Appetizer.java File Reference . . . . .	295
8.61 server/src/main/java/es/ull/esit/server/middleware/model/Appetizer.java . . . . .	296
8.62 src/main/java/es/ull/esit/app/middleware/model/Appetizer.java File Reference . . . . .	297
8.63 src/main/java/es/ull/esit/app/middleware/model/Appetizer.java . . . . .	297
8.64 BillResult.java File Reference . . . . .	298
8.65 BillResult.java . . . . .	298
8.66 server/src/main/java/es/ull/esit/server/middleware/model/Cashier.java File Reference . . . . .	299
8.67 server/src/main/java/es/ull/esit/server/middleware/model/Cashier.java . . . . .	299
8.68 src/main/java/es/ull/esit/app/middleware/model/Cashier.java File Reference . . . . .	300
8.69 src/main/java/es/ull/esit/app/middleware/model/Cashier.java . . . . .	301
8.70 server/src/main/java/es/ull/esit/server/middleware/model/Drink.java File Reference . . . . .	302
8.71 server/src/main/java/es/ull/esit/server/middleware/model/Drink.java . . . . .	302

8.72 src/main/java/es/ull/esit/app/middleware/model/Drink.java File Reference . . . . .	303
8.73 src/main/java/es/ull/esit/app/middleware/model/Drink.java . . . . .	304
8.74 server/src/main/java/es/ull/esit/server/middleware/model/MainCourse.java File Reference . . . . .	305
8.75 server/src/main/java/es/ull/esit/server/middleware/model/MainCourse.java . . . . .	305
8.76 src/main/java/es/ull/esit/app/middleware/model/MainCourse.java File Reference . . . . .	306
8.77 src/main/java/es/ull/esit/app/middleware/model/MainCourse.java . . . . .	307
8.78 server/src/main/java/es/ull/esit/server/middleware/model/User.java File Reference . . . . .	308
8.79 server/src/main/java/es/ull/esit/server/middleware/model/User.java . . . . .	308
8.80 src/main/java/es/ull/esit/app/middleware/model/User.java File Reference . . . . .	309
8.81 src/main/java/es/ull/esit/app/middleware/model/User.java . . . . .	310
8.82 AuthService.java File Reference . . . . .	310
8.83 AuthService.java . . . . .	311
8.84 OrderService.java File Reference . . . . .	311
8.85 OrderService.java . . . . .	312
8.86 ProductService.java File Reference . . . . .	313
8.87 ProductService.java . . . . .	313
8.88 ReportService.java File Reference . . . . .	315
8.89 ReportService.java . . . . .	315
8.90 Order.java File Reference . . . . .	316
8.91 Order.java . . . . .	316
<b>Index</b>	<b>325</b>

# **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/a1/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:

<a href="#">es.ull.esit.app</a> . . . . .	13
<a href="#">es.ull.esit.app.middleware</a> . . . . .	13
<a href="#">es.ull.esit.app.middleware.model</a> . . . . .	14
<a href="#">es.ull.esit.app.middleware.service</a> . . . . .	14
<a href="#">es.ull.esit.server</a> . . . . .	14
<a href="#">es.ull.esit.server.config</a> . . . . .	15
<a href="#">es.ull.esit.server.controller</a> . . . . .	15
<a href="#">es.ull.esit.server.middleware.model</a> . . . . .	15
<a href="#">es.ull.esit.server.repo</a> . . . . .	16



# Chapter 3

## Hierarchical Index

### 3.1 Class Hierarchy

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

es.ull.esit.app.middleware.ApiClient . . . . .	54
es.ull.esit.app.middleware.model.Appetizer . . . . .	75
es.ull.esit.server.middleware.model.Appetizer . . . . .	82
es.ull.esit.server.controller.AppetizerController . . . . .	86
es.ull.esit.app.ApplicationLauncher . . . . .	91
es.ull.esit.server.controller.AuthController . . . . .	93
es.ull.esit.app.middleware.service.AuthService . . . . .	95
es.ull.esit.app.middleware.model.BillResult . . . . .	99
es.ull.esit.app.middleware.model.Cashier . . . . .	102
es.ull.esit.server.middleware.model.Cashier . . . . .	106
es.ull.esit.server.controller.CashierController . . . . .	110
es.ull.esit.app.middleware.model.Drink . . . . .	127
es.ull.esit.server.middleware.model.Drink . . . . .	134
es.ull.esit.server.controller.DrinkController . . . . .	138
es.ull.esit.server.controller.HealthController . . . . .	143
es.ull.esit.app.middleware.ApiClient.IOCallable< T > . . . . .	145
javax.swing.JFrame	
es.ull.esit.app.AboutUs . . . . .	17
es.ull.esit.app.AdminLogin . . . . .	24
es.ull.esit.app.AdminProducts . . . . .	34
es.ull.esit.app.CashierLogin . . . . .	113
es.ull.esit.app.Login . . . . .	146
es.ull.esit.app.Order . . . . .	175
JpaRepository	
es.ull.esit.server.repo.AppetizerRepository . . . . .	90
es.ull.esit.server.repo.CashierRepository . . . . .	126
es.ull.esit.server.repo.DrinkRepository . . . . .	142
es.ull.esit.server.repo.MainCourseRepository . . . . .	172
es.ull.esit.server.repo.UserRepository . . . . .	233
es.ull.esit.server.controller.AuthController.LoginRequest . . . . .	157
es.ull.esit.app.middleware.model.MainCourse . . . . .	158
es.ull.esit.server.middleware.model.MainCourse . . . . .	164
es.ull.esit.server.controller.MainCourseController . . . . .	168
es.ull.esit.server.controller.MenuController . . . . .	173
es.ull.esit.app.middleware.service.OrderService . . . . .	202

es.ull.esit.app.middleware.service.ProductService . . . . .	204
es.ull.esit.app.middleware.service.ReportService . . . . .	217
es.ull.esit.app.middleware.ApiClient.ResponseHandler< R > . . . . .	220
es.ull.esit.server.RestaurantApplication . . . . .	221
RuntimeException	
es.ull.esit.app.middleware.ApiClientException . . . . .	73
es.ull.esit.server.config.SecurityConfig . . . . .	222
es.ull.esit.app.middleware.model.User . . . . .	224
es.ull.esit.server.middleware.model.User . . . . .	228

# Chapter 4

## Class Index

### 4.1 Class List

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

<a href="#">es.ull.esit.app.AboutUs</a>	"About us" window displaying the restaurant's history and info . . . . .	17
<a href="#">es.ull.esit.app.AdminLogin</a>	Login window for authenticating administrators . . . . .	24
<a href="#">es.ull.esit.app.AdminProducts</a>	Administrative window for managing products and prices . . . . .	34
<a href="#">es.ull.esit.app.middleware.ApiClient</a>	API client for interacting with the backend REST API . . . . .	54
<a href="#">es.ull.esit.app.middleware.ApiClientException</a>	Custom exception class for API client errors . . . . .	73
<a href="#">es.ull.esit.app.middleware.model.Appetizer</a>	Client-side model representing an appetizer received from the backend API . . . . .	75
<a href="#">es.ull.esit.server.middleware.model.Appetizer</a>	JPA entity that represents an appetizer in the menu . . . . .	82
<a href="#">es.ull.esit.server.controller.AppetizerController</a>	REST controller for managing appetizers . . . . .	86
<a href="#">es.ull.esit.server.repo.AppetizerRepository</a>	Repository interface for managing appetizers in the database . . . . .	90
<a href="#">es.ull.esit.app.ApplicationLauncher</a>	Auxiliary entry point used to start the application's UI . . . . .	91
<a href="#">es.ull.esit.server.controller.AuthController</a>	Rest controller for handling authentication-related requests . . . . .	93
<a href="#">es.ull.esit.app.middleware.service.AuthService</a>	Service responsible for handling authentication logic on the client side . . . . .	95
<a href="#">es.ull.esit.app.middleware.model.BillResult</a>	Data Transfer Object representing the result of a bill calculation . . . . .	99
<a href="#">es.ull.esit.app.middleware.model.Cashier</a>	Client-side model representing a cashier returned by the backend . . . . .	102
<a href="#">es.ull.esit.server.middleware.model.Cashier</a>	JPA entity that represents a cashier in the system . . . . .	106
<a href="#">es.ull.esit.server.controller.CashierController</a>	REST controller for managing cashiers . . . . .	110
<a href="#">es.ull.esit.app.CashierLogin</a>	Login window for authenticating cashiers . . . . .	113
<a href="#">es.ull.esit.server.repo.CashierRepository</a>	Repository interface for managing cashiers in the database . . . . .	126

<a href="#">es.ull.esit.app.middleware.model.Drink</a>	Client-side model representing a drink returned by the backend API . . . . .	127
<a href="#">es.ull.esit.server.middleware.model.Drink</a>	JPA entity that represents a drink in the menu . . . . .	134
<a href="#">es.ull.esit.server.controller.DrinkController</a>	REST controller for managing drinks . . . . .	138
<a href="#">es.ull.esit.server.repo.DrinkRepository</a>	Repository interface for managing drinks in the database . . . . .	142
<a href="#">es.ull.esit.server.controller.HealthController</a>	REST controller for health and database connectivity checks . . . . .	143
<a href="#">es.ull.esit.app.middleware.ApiClient.IOCallable&lt; T &gt;</a>	Functional interface for lambdas that may throw InterruptedException or IOException . . . . .	145
<a href="#">es.ull.esit.app.Login</a>	Login window for the Restaurant System . . . . .	146
<a href="#">es.ull.esit.server.controller.AuthController.LoginRequest</a>	Simple DTO (Data Transfer Object) for login requests payload . . . . .	157
<a href="#">es.ull.esit.app.middleware.model.MainCourse</a>	Client-side model representing a main course returned by the backend . . . . .	158
<a href="#">es.ull.esit.server.middleware.model.MainCourse</a>	JPA entity that represents a main course in the menu . . . . .	164
<a href="#">es.ull.esit.server.controller.MainCourseController</a>	REST controller for managing main courses . . . . .	168
<a href="#">es.ull.esit.server.repo.MainCourseRepository</a>	Repository interface for managing main courses in the database . . . . .	172
<a href="#">es.ull.esit.server.controller.MenuController</a>	REST controller that exposes a consolidated restaurant menu . . . . .	173
<a href="#">es.ull.esit.app.Order</a>	Main menu window for taking customer orders . . . . .	175
<a href="#">es.ull.esit.app.middleware.service.OrderService</a>	Service that handles order-related calculations and receipt generation . . . . .	202
<a href="#">es.ull.esit.app.middleware.service.ProductService</a>	Service handling business logic for menu products . . . . .	204
<a href="#">es.ull.esit.app.middleware.service.ReportService</a>	Service providing reporting and system status operations . . . . .	217
<a href="#">es.ull.esit.app.middleware.ApiClient.ResponseHandler&lt; R &gt;</a>	Functional interface for processing HTTP responses and possibly throwing IOException from JSON parsing . . . . .	220
<a href="#">es.ull.esit.server.RestaurantApplication</a>	Main Spring Boot application class for the restaurant backend . . . . .	221
<a href="#">es.ull.esit.server.config.SecurityConfig</a>	Spring Security configuration for the backend . . . . .	222
<a href="#">es.ull.esit.app.middleware.model.User</a>	Client-side model representing an authenticated user . . . . .	224
<a href="#">es.ull.esit.server.middleware.model.User</a>	JPA entity that represents a user in the system . . . . .	228
<a href="#">es.ull.esit.server.repo.UserRepository</a>	Repository interface for accessing users in the database . . . . .	233

# Chapter 5

## File Index

### 5.1 File List

Here is a list of all files with brief descriptions:

00-create-db.sql . . . . .	237
01-tables.sql . . . . .	237
02-procedures.sql . . . . .	240
03-triggers.sql . . . . .	240
04-privileges.sql . . . . .	241
05-data.sql . . . . .	241
init.sql . . . . .	243
SecurityConfig.java . . . . .	247
AppetizerController.java . . . . .	248
AuthController.java . . . . .	249
CashierController.java . . . . .	251
DrinkController.java . . . . .	252
HealthController.java . . . . .	254
MainCourseController.java . . . . .	255
MenuController.java . . . . .	257
AppetizerRepository.java . . . . .	258
CashierRepository.java . . . . .	259
DrinkRepository.java . . . . .	260
MainCourseRepository.java . . . . .	261
UserRepository.java . . . . .	262
RestaurantApplication.java . . . . .	263
AboutUs.java . . . . .	263
AdminLogin.java . . . . .	266
AdminProducts.java . . . . .	269
ApplicationLauncher.java . . . . .	281
CashierLogin.java . . . . .	282
Login.java . . . . .	286
ApiClient.java . . . . .	290
ApiClientException.java . . . . .	295
server/src/main/java/es/ull/esit/server/middleware/model/Appetizer.java . . . . .	295
src/main/java/es/ull/esit/app/middleware/model/Appetizer.java . . . . .	297
BillResult.java . . . . .	298
server/src/main/java/es/ull/esit/server/middleware/model/Cashier.java . . . . .	299
src/main/java/es/ull/esit/app/middleware/model/Cashier.java . . . . .	300
server/src/main/java/es/ull/esit/server/middleware/model/Drink.java . . . . .	302

src/main/java/es/ull/esit/app/middleware/model/Drink.java . . . . .	303
server/src/main/java/es/ull/esit/server/middleware/model/MainCourse.java . . . . .	305
src/main/java/es/ull/esit/app/middleware/model/MainCourse.java . . . . .	306
server/src/main/java/es/ull/esit/server/middleware/model/User.java . . . . .	308
src/main/java/es/ull/esit/app/middleware/model/User.java . . . . .	309
AuthService.java . . . . .	310
OrderService.java . . . . .	311
ProductService.java . . . . .	313
ReportService.java . . . . .	315
Order.java . . . . .	316

# 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.*
- class [ApiClientException](#)  
*Custom exception class for API client errors.*

## 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.*

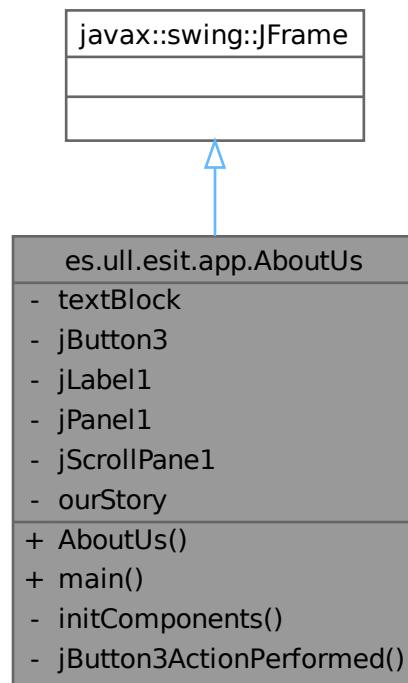
# Chapter 7

## Class Documentation

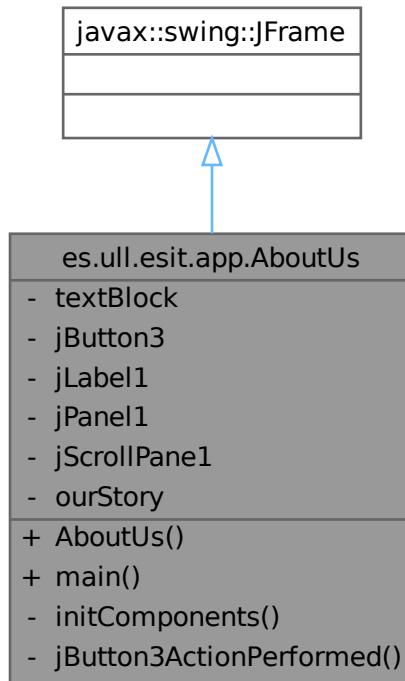
### 7.1 es.ull.esit.app.AboutUs Class Reference

"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

- String [textBlock](#)  
*Static text block with the restaurant story.*
- 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 30 of file [AboutUs.java](#).

```
00030      {
00031      initComponents();
00032  }
```

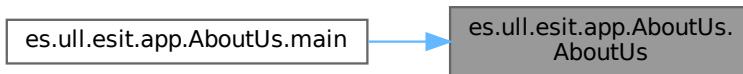
References [es.ull.esit.app.AboutUs.initComponents\(\)](#).

Referenced by [es.ull.esit.app.AboutUs.main\(\)](#).

Here is the call graph for this function:



Here is the caller graph for this function:



### 7.1.3 Member Function Documentation

#### 7.1.3.1 initComponents()

```
void es.ul.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 48 of file [AboutUs.java](#).

```

00048
00049
00050     jPanell = new javax.swing.JPanel();
00051     jButton3 = new javax.swing.JButton();
00052     jScrollPane = new javax.swing.JScrollPane();
00053     ourStory = new javax.swing.JTextArea();
00054     jLabel1 = new javax.swing.JLabel();
00055
00056     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00057     setTitle("Info");
00058     setResizable(false);
00059
00060     jPanell.setBackground(new java.awt.Color(248, 244, 230));
00061
00062     jButton3.setBackground(new java.awt.Color(255, 255, 255));
00063     jButton3.setFont(new java.awt.Font("Yu Gothic UI", 1, 18)); // NOI18N
00064     jButton3.setText("Go Back");
00065     jButton3.addActionListener(this::jButton3ActionPerformed);
00066
00067     ourStory.setEditable(false);
00068     ourStory.setBackground(new java.awt.Color(248, 244, 230));
00069     ourStory.setColumns(20);

```

```

00070     ourStory.setFont(new java.awt.Font("Yu Gothic UI Light", 0, 14)); // NOI18N
00071     ourStory.setRows(5);
00072     ourStory.setText(textBlock);
00073     ourStory.setBorder(javax.swing.BorderFactory.createLineBorder(new java.awt.Color(0, 0, 0)));
00074     jScrollPane1.setViewportView(ourStory);
00075
00076     // Updated path to match other UI resources
00077     jLabel1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/ui/screenshot.png")));
00078
00079     javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
00080     jPanel1.setLayout(jPanel1Layout);
00081     jPanel1Layout.setHorizontalGroup(
00082         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00083             .addGroup(jPanel1Layout.createSequentialGroup()
00084                 .addGroup(jPanel1Layout.createSequentialGroup()
00085                     .addGroup(jPanel1Layout.createSequentialGroup()
00086                         .addGap(55, 55, 55)
00087
00088                         .addGroup(jPanel1Layout.createSequentialGroup()
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                         .addGap(159, Short.MAX_VALUE)))
00097
00098             .addGroup(jPanel1Layout.createSequentialGroup()
00099                 .addGroup(jPanel1Layout.createSequentialGroup()
00100                     .addComponent(jLabel1)
00101                     .addGap(67, 67, 67)
00102                     .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 203,
00103                         javax.swing.GroupLayout.PREFERRED_SIZE)
00104                     .addGap(26, 26, 26)
00105                     .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00106                         javax.swing.GroupLayout.PREFERRED_SIZE)
00107                     .addGap(30, 30, 30)));
00108
00109     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00110     getContentPane().setLayout(layout);
00111     layout.setHorizontalGroup(
00112         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00113             .addGroup(layout.createSequentialGroup()
00114                 .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
00115                     javax.swing.GroupLayout.DEFAULT_SIZE,
00116                     Short.MAX_VALUE));
00117
00118     layout.setVerticalGroup(
00119         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00120             .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
00121                     javax.swing.GroupLayout.DEFAULT_SIZE,
00122                     Short.MAX_VALUE));
00123
00124     pack();
00125     setLocationRelativeTo(null);
00126 } // </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](#), [es.ull.esit.app.AboutUs.ourStory](#), and [es.ull.esit.app.AboutUs.textBlock](#).

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

<i>evt</i>	Action event triggered by button click.
------------	---

Definition at line 132 of file [AboutUs.java](#).

```
00132     GEN-FIRST:event_jButton3ActionPerformed
00133         new CashierLogin().setVisible(true);
00134         this.dispose(); // Close current window
00135     } // 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

<i>args</i>	Command line arguments (not used).
-------------	------------------------------------

Definition at line 147 of file [AboutUs.java](#).

```
00147
00148     try {
00149         for (javax.swing.UIManager.LookAndFeelInfo info :
00150             javax.swing.UIManager.getInstalledLookAndFeels()) {
00151             if ("Nimbus".equals(info.getName())) {
00152                 javax.swing.UIManager.setLookAndFeel(info.getClassName());
00153                 break;
00154             }
00155         } catch (ClassNotFoundException | InstantiationException | IllegalAccessException
00156             | javax.swing.UnsupportedLookAndFeelException ex) {
00157             java.util.logging.Logger.getLogger(AboutUs.class.getName())
00158                 .log(java.util.logging.Level.SEVERE, null, ex);
00159         }
00160     // </editor-fold>
00161
00162     // Create and display the form
00163     java.awt.EventQueue.invokeLater(() -> new AboutUs().setVisible(true));
00164 }
```

References [es.ull.esit.app.AboutUs.AboutUs\(\)](#).

Here is the call graph for this function:



## 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 171 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 173 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 175 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 177 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 179 of file [AboutUs.java](#).

Referenced by [es.ull.esit.app.AboutUs.initComponents\(\)](#).

#### 7.1.4.6 textBlock

```
String es.ull.esit.app.AboutUs.textBlock [private]
```

**Initial value:**

```
= """  
    Our Story  
    Collecting flavors from all over the world to give everyone a taste of what they love.  
    Black Plate came in with this vision in mind, to be more of a home rather than an establishment.  
    2021 marked the beginning of the journey of Black Plate, starting from Al Khobar.  
    Because your visit means a lot to us, we would love to hear your suggestions and concerns so we  
    can improve!  
"""
```

Static text block with the restaurant story.

Definition at line 17 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](#)

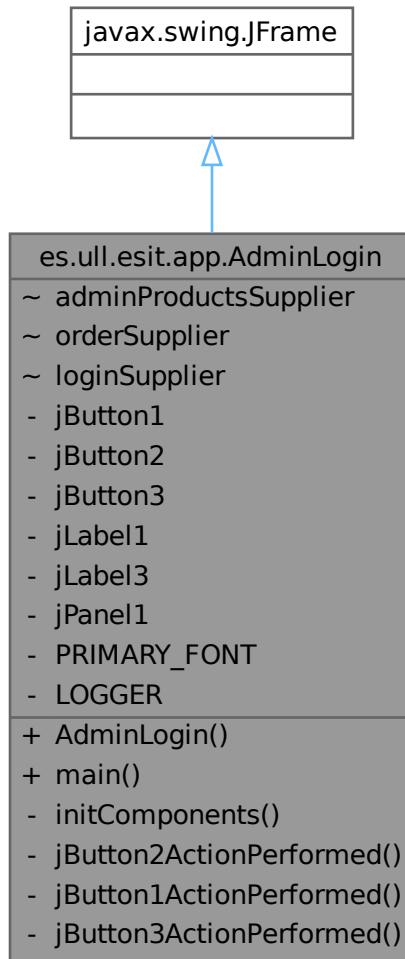
## 7.2 es.ull.esit.app.AdminLogin Class Reference

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.*

### Static Private Attributes

- static final String [PRIMARY\\_FONT](#) = "Yu Gothic UI"  
*Primary font used in the UI components.*
- static final Logger [LOGGER](#) = LoggerFactory.getLogger(AdminLogin.class)  
*Logger for logging events and errors.*

## 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 16 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.

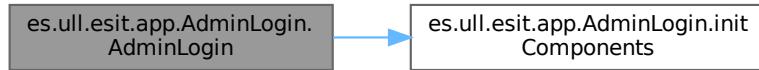
Definition at line 52 of file [AdminLogin.java](#).

```
00052      {
00053      initComponents();
00054 }
```

References [es.ull.esit.app.AdminLogin.initComponents\(\)](#).

Referenced by [es.ull.esit.app.AdminLogin.main\(\)](#).

Here is the call graph for this function:



Here is the caller 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 66 of file [AdminLogin.java](#).

```

00066
00067
00068     jPanell = new javax.swing.JPanel();
00069     jLabel3 = new javax.swing.JLabel();
00070     jLabel11 = new javax.swing.JLabel();
00071     jButton2 = new javax.swing.JButton();
00072     jButton1 = new javax.swing.JButton();
00073     jButton3 = new javax.swing.JButton();
00074
00075     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00076     setTitle("Admin ");
00077     setResizable(false);
00078
00079     jPanell.setBackground(new java.awt.Color(248, 244, 230));
00080
00081     jLabel3.setFont(new java.awt.Font(PRIMARY_FONT, 1, 24)); // NOI18N
00082     jLabel3.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00083     jLabel3.setText("Welcome Admin");
00084
00085     jLabel11.setIcon(new javax.swing.ImageIcon(getClass().getResource("/ui/screenshot.png"))); // NOI18N
00086
00087     jButton2.setBackground(new java.awt.Color(153, 153, 153));
00088     jButton2.setFont(new java.awt.Font(PRIMARY_FONT, 1, 18)); // NOI18N
00089     jButton2.setText("Update Prices");
00090     jButton2.addActionListener(this::jButton2ActionPerformed);
00091
00092     jButton1.setBackground(new java.awt.Color(153, 153, 153));
00093     jButton1.setFont(new java.awt.Font(PRIMARY_FONT, 1, 18)); // NOI18N
00094     jButton1.setText("Menu");
00095     jButton1.addActionListener(this::jButton1ActionPerformed);
00096
00097     jButton3.setFont(new java.awt.Font(PRIMARY_FONT, 1, 18)); // NOI18N
00098     jButton3.setText("LogOut");
00099     jButton3.addActionListener(this::jButton3ActionPerformed);
00100
00101     javax.swing.GroupLayout jPanellLayout = new javax.swing.GroupLayout(jPanell);
00102     jPanell.setLayout(jPanellLayout);
00103     jPanellLayout.setHorizontalGroup(
00104         jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00105             .addGroup(jPanellLayout.createSequentialGroup()
00106                 .addGroup(jPanellLayout.createSequentialGroup()
00107                     .addGroup(jPanellLayout.createSequentialGroup()
00108                         .addGap(140, 140, 140)
00109                         .addComponent(jLabel11))
00110                     .addGroup(jPanellLayout.createSequentialGroup()
00111                         .addGap(66, 66, 66)
00112
00113                         .addGroup(jPanellLayout.createSequentialGroup()
00114                             .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 320,
00115                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00116                             .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 320,
00117                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00118                             .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 306,
00119                                 javax.swing.GroupLayout.PREFERRED_SIZE)))
00120                             .addGap(152, 152, 152)
00121                             .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 141,
00122                                 javax.swing.GroupLayout.PREFERRED_SIZE))
00123                             .addGap(69, Short.MAX_VALUE)));
00124             jPanellLayout.setVerticalGroup(
00125                 jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00126                     .addGroup(jPanellLayout.createSequentialGroup()
00127                         .addGap(31, 31, 31)
00128                         .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
00129                             javax.swing.GroupLayout.PREFERRED_SIZE)
00130                         .addGap(18, 18, 18)
00131                         .addComponent(jLabel11)
00132                         .addGap(29, 29, 29)
00133                         .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00134                             javax.swing.GroupLayout.PREFERRED_SIZE)
00135                         .addGap(18, 18, 18)
00136                         .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00137                             javax.swing.GroupLayout.PREFERRED_SIZE)
00138                         .addGap(29, 29, 29)
00139                         .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00140                             javax.swing.GroupLayout.PREFERRED_SIZE))
00141                         .addGap(115, Short.MAX_VALUE)));
00142
00143     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00144     getContentPane().setLayout(layout);
00145     layout.setHorizontalGroup(
00146         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00147             .addGroup(layout.createSequentialGroup()
00148                 .addComponent(jPanell, javax.swing.GroupLayout.DEFAULT_SIZE,
00149                     javax.swing.GroupLayout.PREFERRED_SIZE))
00149             .addGap(0, 0, 0)
00150         );
00151     
```

```

00149     layout.setVerticalGroup(
00150         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00151             .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
00152                 javax.swing.GroupLayout.DEFAULT_SIZE,
00153                 Short.MAX_VALUE));
00154     pack();
00155     setLocationRelativeTo(null);
00156 } // </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](#), [es.ull.esit.app.AdminLogin.JPanel1](#), and [es.ull.esit.app.AdminLogin.PRIMARY\\_FONT](#).

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

<code>evt</code>	<code>[java.awt.event.ActionEvent]</code> Action event triggered by button click.
------------------	---

Definition at line 189 of file [AdminLogin.java](#).

```

00189
00190     GEN-FIRST:event_jButton1ActionPerformed
00191     try {
00192         orderSupplier.get().setVisible(true);
00193         this.dispose();
00194     } catch (Exception ex) {
00195         LOGGER.error("Error opening menu window", ex);
00196         javax.swing.JOptionPane.showMessageDialog(
00197             this,
00198             "Error opening menu window:\n" + ex.getMessage(),
00199             "Error",
00200             javax.swing.JOptionPane.ERROR_MESSAGE);
00201 } // GEN-LAST:event_jButton1ActionPerformed

```

References [es.ull.esit.app.AdminLogin.LOGGER](#).

### 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

<code>evt</code>	[ <code>java.awt.event.ActionEvent</code> ] Action event triggered by button click.
------------------	---

Definition at line 167 of file [AdminLogin.java](#).

```
00167 // GEN-FIRST:event_jButton2ActionPerformed
00168     try {
00169         adminProductsSupplier.get().setVisible(true);
00170         this.dispose();
00171     } catch (Exception ex) {
00172         LOGGER.error("Error opening product admin window", ex);
00173         javax.swing.JOptionPane.showMessageDialog(
00174             this,
00175             "Error opening product admin window:\n" + ex.getMessage(),
00176             "Error",
00177             javax.swing.JOptionPane.ERROR_MESSAGE);
00178     }
00179 // GEN-LAST:event_jButton2ActionPerformed
```

References [es.ull.esit.app.AdminLogin.LOGGER](#).

### 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

<code>evt</code>	[ <code>java.awt.event.ActionEvent</code> ] Action event triggered by button click.
------------------	---

Definition at line 211 of file [AdminLogin.java](#).

```
00211 // GEN-FIRST:event_jButton3ActionPerformed
00212     loginSupplier.get().setVisible(true);
00213     this.dispose();
00214 // 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

<i>args</i>	[String[]] Command line arguments (not used).
-------------	---

Definition at line 225 of file [AdminLogin.java](#).

```
00225             {
00226     try {
00227         for (javax.swing.UIManager.LookAndFeelInfo info :
00228             javax.swing.UIManager.getInstalledLookAndFeels()) {
00229             if ("Nimbus".equals(info.getName())) {
00230                 javax.swing.UIManager.setLookAndFeel(info.getClassName());
00231                 break;
00232             }
00233         } catch (ClassNotFoundException | InstantiationException | IllegalAccessException
00234             | javax.swing.UnsupportedLookAndFeel ex) {
00235             java.util.logging.Logger.getLogger(AdminLogin.class.getName()).log(java.util.logging.Level.SEVERE,
00236             null, ex);
00237         // </editor-fold>
00238     }
00239     java.awt.EventQueue.invokeLater(() -> new AdminLogin().setVisible(true));
00240 }
```

References [es.ull.esit.app.AdminLogin.AdminLogin\(\)](#).

Here is the call graph for this function:



## 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 247 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 249 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 251 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 253 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 255 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 257 of file [AdminLogin.java](#).

Referenced by [es.ull.esit.app.AdminLogin.initComponents\(\)](#).

#### 7.2.4.7 LOGGER

```
final Logger es.ull.esit.app.AdminLogin.LOGGER = LoggerFactory.getLogger(AdminLogin.class)  
[static], [private]
```

Logger for logging events and errors.

Definition at line 22 of file [AdminLogin.java](#).

Referenced by [es.ull.esit.app.AdminLogin.jButton1ActionPerformed\(\)](#), and [es.ull.esit.app.AdminLogin.jButton2ActionPerformed\(\)](#).

#### 7.2.4.8 PRIMARY\_FONT

```
final String es.ull.esit.app.AdminLogin.PRIMARY_FONT = "Yu Gothic UI" [static], [private]
```

Primary font used in the UI components.

Definition at line 19 of file [AdminLogin.java](#).

Referenced by [es.ull.esit.app.AdminLogin.initComponents\(\)](#).

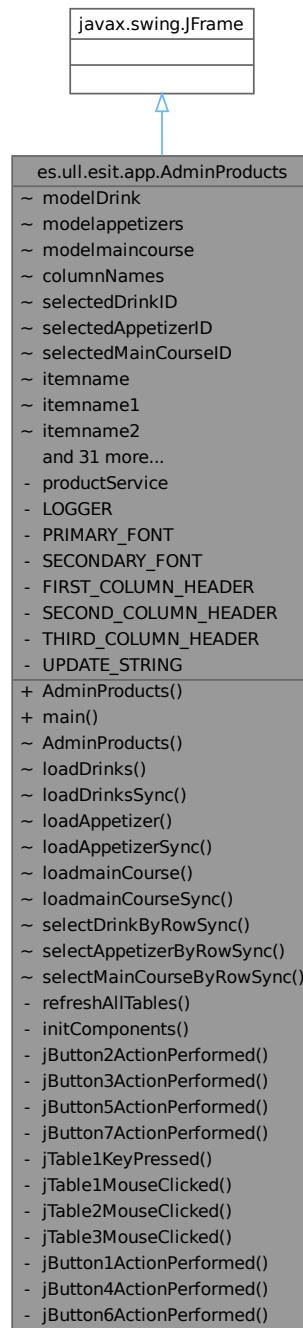
The documentation for this class was generated from the following file:

- [AdminLogin.java](#)

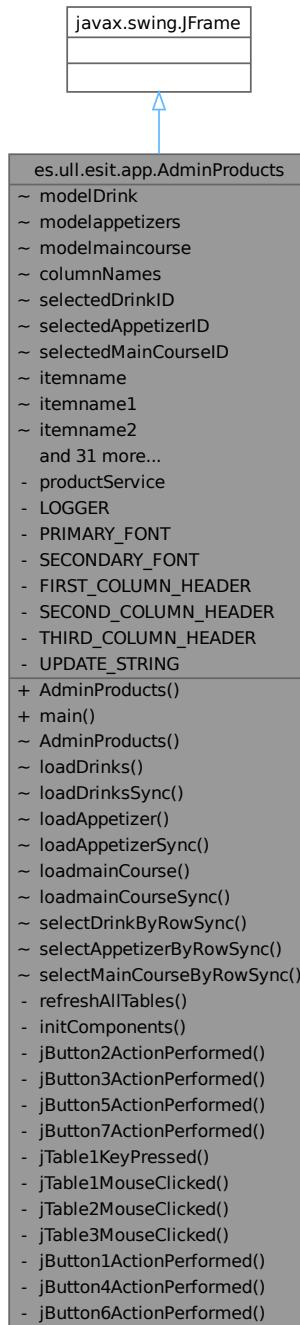
### 7.3 es.ull.esit.app.AdminProducts Class Reference

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 transient ProductService `productService`  
*Service used to call the REST API and apply product-related logic.*

### Static Private Attributes

- static final Logger `LOGGER` = LoggerFactory.getLogger(AdminProducts.class)  
*Logger for logging events and errors.*
- static final String `PRIMARY_FONT` = "Yu Gothic UI"  
*Primary font used throughout the GUI.*
- static final String `SECONDARY_FONT` = "Thonburi"  
*Secondary font used for buttons and smaller text.*
- static final String `FIRST_COLUMN_HEADER` = "ID"  
*Column header for the ID column in product tables.*
- static final String `SECOND_COLUMN_HEADER` = "Item Name"  
*Column header for the name column in product tables.*
- static final String `THIRD_COLUMN_HEADER` = "Item Price"  
*Column header for the price column in product tables.*
- static final String `UPDATE_STRING` = "Update"  
*String constant for the "Update" button label.*

### 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 26 of file [AdminProducts.java](#).

### 7.3.2 Constructor & Destructor Documentation

#### 7.3.2.1 AdminProducts()

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

Constructor.

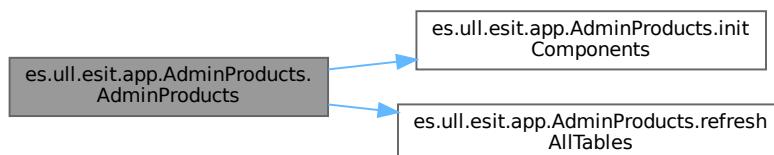
Creates the admin products window, initializes GUI components, configures the table models and loads the initial data from the backend service.

Definition at line 73 of file [AdminProducts.java](#).

```
00073         {
00074     initComponents();
00075
00076     // Initialize the Service Layer.
00077     ApiClient client = new ApiClient("http://localhost:8080");
00078     this.productService = new ProductService(client);
00079
00080     // Initialize table models and set shared headers.
00081     modelDrink = new DefaultTableModel();
00082     modelDrink.setColumnIdentifiers(columnNames);
00083     modelAppetizers = new DefaultTableModel();
00084     modelAppetizers.setColumnIdentifiers(columnNames);
00085     modelMainCourse = new DefaultTableModel();
00086     modelMainCourse.setColumnIdentifiers(columnNames);
00087
00088     // Link models to tables.
00089     jTable1.setModel(modelDrink);
00090     jTable2.setModel(modelAppetizers);
00091     jTable3.setModel(modelMainCourse);
00092
00093     // Load initial data from backend.
00094     refreshAllTables();
00095 }
```

References [es.ull.esit.app.AdminProducts.initComponents\(\)](#), 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 361 of file [AdminProducts.java](#).

```
00361     {
00362
00363     jPanel1 = new javax.swing.JPanel();
00364     jLabel1 = new javax.swing.JLabel();
00365     jTabbedPane1 = new javax.swing.JTabbedPane();
00366     jPanel12 = new javax.swing.JPanel();
00367     itemname = new javax.swing.JTextField();
00368     itemprice = new javax.swing.JTextField();
00369     jScrollPane1 = new javax.swing.JScrollPane();
00370     jTable1 = new javax.swing.JTable();
00371     jButton1 = new javax.swing.JButton();
00372     jButton2 = new javax.swing.JButton();
00373     jLabel2 = new javax.swing.JLabel();
00374     jLabel3 = new javax.swing.JLabel();
00375     jLabel4 = new javax.swing.JLabel();
00376     jPanel13 = new javax.swing.JPanel();
00377     jLabel5 = new javax.swing.JLabel();
00378     jLabel6 = new javax.swing.JLabel();
00379     jLabel7 = new javax.swing.JLabel();
00380     jScrollPane2 = new javax.swing.JScrollPane();
00381     jTable2 = new javax.swing.JTable();
00382     itemname1 = new javax.swing.JTextField();
00383     itemprice1 = new javax.swing.JTextField();
00384     jButton4 = new javax.swing.JButton();
00385     jButton5 = new javax.swing.JButton();
00386     jPanel14 = new javax.swing.JPanel();
00387     jLabel8 = new javax.swing.JLabel();
00388     jLabel9 = new javax.swing.JLabel();
00389     jLabel10 = new javax.swing.JLabel();
00390     jScrollPane3 = new javax.swing.JScrollPane();
00391     jTable3 = new javax.swing.JTable();
00392     itemname2 = new javax.swing.JTextField();
00393     itemprice2 = new javax.swing.JTextField();
00394     jButton6 = new javax.swing.JButton();
00395     jButton7 = new javax.swing.JButton();
00396     jButton3 = new javax.swing.JButton();
00397
00398     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00399     setTitle("Admin");
00400     setResizable(false);
00401
00402     jPanel1.setBackground(new java.awt.Color(248, 244, 230));
00403
00404     jLabel1.setBackground(new java.awt.Color(255, 153, 0));
00405     jLabel1.setFont(new java.awt.Font(PRIMARY_FONT, 1, 36)); // NOI18N
00406     jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00407     jLabel1.setText("Items Prices Update Portal");
00408
00409     jTabbedPane1.setBackground(new java.awt.Color(248, 244, 230));
00410     jTabbedPane1.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));
00411     jTabbedPane1.setCursor(new java.awt.Cursor(java.awt.Cursor.DEFAULT_CURSOR));
00412     jTabbedPane1.setFont(new java.awt.Font(PRIMARY_FONT, 1, 18)); // NOI18N
00413
00414     jPanel12.setBackground(new java.awt.Color(248, 244, 230));
00415
00416     itemname.setFont(new java.awt.Font(PRIMARY_FONT, 0, 18)); // NOI18N
00417     itemname.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00418     itemname.setBorder(javax.swing.BorderFactory.createTitledBorder(
00419         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), SECOND_COLUMN_HEADER,
00420         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00421         new java.awt.Font(PRIMARY_FONT, 0, 18))); // NOI18N
00422
00423     itemprice.setFont(new java.awt.Font(PRIMARY_FONT, 0, 18)); // NOI18N
00424     itemprice.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
```



```

00507                     javax.swing.GroupLayout.PREFERRED_SIZE))
00508             .addGap(115, 115, 115)
00509             .addGroup(jPanel2Layout.createSequentialGroup()
00510                 .addComponent(jLabel12, javax.swing.GroupLayout.PREFERRED_SIZE, 1111,
00511                     javax.swing.GroupLayout.PREFERRED_SIZE)
00512                     .addGap(18, 18, 18))))))
00513     jPanel2Layout.setVerticalGroup(
00514         jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00515             .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
00516                 jPanel2Layout.createSequentialGroup()
00517                     .addContainerGap()
00518                     .addComponent(jLabel12)
00519                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 21,
00520                         Short.MAX_VALUE)
00521             .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00522                 .addComponent(jLabel13)
00523                     .addComponent(jLabel14)
00524                     .addGap(18, 18, 18)
00525                     .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00526                         .addGroup(jPanel2Layout.createSequentialGroup()
00527                             .addComponent(itemname, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00528                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00529                             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
00530                             .addComponent(itemprice, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00531                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00532                             .addGap(37, 37, 37)
00533             .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00534                 .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00535                     javax.swing.GroupLayout.PREFERRED_SIZE)
00536                 .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 203,
00537                     javax.swing.GroupLayout.PREFERRED_SIZE))
00538                 .addGap(48, 48, 48))))))
00539
00540     jTabbedPane.addTab("Drinks", jPanel12);
00541
00542     jPanel13.setBackground(new java.awt.Color(248, 244, 230));
00543
00544     jLabel15.setFont(new java.awt.Font(PRIMARY_FONT, 0, 14)); // NOI18N
00545     jLabel15.setText("Enter new appetizer information carefully to add.");
00546
00547     jLabel16.setBackground(new java.awt.Color(255, 153, 0));
00548     jLabel16.setFont(new java.awt.Font(PRIMARY_FONT, 1, 25)); // NOI18N
00549     jLabel16.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00550     jLabel16.setText("Available Appetizer");
00551
00552     jLabel17.setFont(new java.awt.Font(PRIMARY_FONT, 0, 14)); // NOI18N
00553     jLabel17.setText("Please select the appetizer to update the price.");
00554
00555     jTable2.setFont(new java.awt.Font(PRIMARY_FONT, 0, 14)); // NOI18N
00556     jTable2.setModel(new javax.swing.table.DefaultTableModel(
00557         new Object[][] {
00558             {},
00559             {},
00560             {},
00561             {}
00562         },
00563         new String[] {
00564             {}
00565         }));
00566     jTable2.addMouseListener(new java.awt.event.MouseAdapter() {
00567         @Override
00568         public void mouseClicked(java.awt.event.MouseEvent evt) {
00569             jTable2MouseClicked(evt);
00570         }
00571     });
00572     jScrollPane2.setViewportView(jTable2);
00573
00574     itemname1.setFont(new java.awt.Font(PRIMARY_FONT, 0, 18)); // NOI18N
00575     itemname1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00576     itemname1.setBorder(javax.swing.BorderFactory.createTitledBorder(
00577         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), SECOND_COLUMN_HEADER,
00578         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00579         new java.awt.Font(PRIMARY_FONT, 0, 18))); // NOI18N
00580
00581     itemprice1.setFont(new java.awt.Font(PRIMARY_FONT, 0, 18)); // NOI18N
00582     itemprice1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00583     itemprice1.setBorder(javax.swing.BorderFactory.createTitledBorder(
00584         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), THIRD_COLUMN_HEADER,
00585         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00586         new java.awt.Font(PRIMARY_FONT, 0, 18))); // NOI18N
00587
00588     jButton4.setBackground(new java.awt.Color(255, 255, 255));
00589     jButton4.setFont(new java.awt.Font(SECONDARY_FONT, 0, 18)); // NOI18N

```

```

00590     jButton4.setText(UPDATE_STRING);
00591     jButton4.addActionListener(this::jButton4ActionPerformed);
00592
00593     jButton5.setBackground(new java.awt.Color(255, 255, 255));
00594     jButton5.setFont(new java.awt.Font(SECONDARY_FONT, 0, 18)); // NOI18N
00595     jButton5.setText("Add");
00596     jButton5.addActionListener(this::jButton5ActionPerformed);
00597
00598     javax.swing.GroupLayout jPanel3Layout = new javax.swing.GroupLayout(jPanel3);
00599     jPanel3.setLayout(jPanel3Layout);
00600     jPanel3Layout.setHorizontalGroup(
00601         jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00602             .addGroup(jPanel3Layout.createSequentialGroup()
00603                 .addGap(27, Short.MAX_VALUE)
00604                 .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00605                     .addGroup(jPanel3Layout.createSequentialGroup()
00606                         .addGap(24, 24, 24)
00607                         .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00608                             .addGroup(jPanel3Layout.createSequentialGroup()
00609                                 .addGap(0, 27, Short.MAX_VALUE)
00610                                 .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00611                                     .addComponent(jButton5, javax.swing.GroupLayout.PREFERRED_SIZE, 91,
00612                                         javax.swing.GroupLayout.PREFERRED_SIZE)
00613                                     .addGapPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
00614                                         javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
00615                                     .addComponent(jButton4, javax.swing.GroupLayout.PREFERRED_SIZE, 103,
00616                                         javax.swing.GroupLayout.PREFERRED_SIZE)
00617                                     .addGapGap(8, 8, 8))
00618                                     .addComponent(itemprice1)
00619                                     .addComponent(itemname1)
00620                                     .addComponent(jLabel5, javax.swing.GroupLayout.DEFAULT_SIZE,
00621                                         javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
00622                                     .addGapPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED,
00623                                         javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
00624                                     .addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED_SIZE, 586,
00625                                         javax.swing.GroupLayout.PREFERRED_SIZE)
00626                                     .addComponent(jLabel7, javax.swing.GroupLayout.PREFERRED_SIZE, 408,
00627                                         javax.swing.GroupLayout.PREFERRED_SIZE)
00628                                     .addGapGap(115, 115, 115)
00629                                     .addGroup(jPanel3Layout.createParallelGroup()
00630                                         .addComponent(jLabel6, javax.swing.GroupLayout.PREFERRED_SIZE, 1111,
00631                                         javax.swing.GroupLayout.PREFERRED_SIZE)
00632                                         .addGapGap(18, 18, 18)))
00633                                     .addGapVerticalGroup()
00634                                     .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00635                                         .addComponent(jLabel16)
00636                                         .addGapPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
00637                                         Short.MAX_VALUE)
00638                                         .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00639                                             .addComponent(jLabel15)
00640                                             .addComponent(jLabel17))
00641                                         .addGapGap(18, 18, 18)
00642                                         .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00643                                             .addGroup(jPanel3Layout.createSequentialGroup()
00644                                                 .addComponent(itemname1, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00645                                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00646                                                 .addGapPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
00647                                                 .addComponent(itemprice1, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00648                                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00649                                                 .addGapGap(37, 37, 37)
00650                                         .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00651                                             .addComponent(jButton4, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00652                                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00653                                             .addComponent(jButton5, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00654                                                 javax.swing.GroupLayout.PREFERRED_SIZE))
00655                                             .addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED_SIZE, 203,
00656                                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00657                                             .addGapGap(48, 48, 48)));
00658
00659     jTabbedPane.addTab("Appetizers", jPanel3);
00660
00661     jPanel4.setBackground(new java.awt.Color(248, 244, 230));
00662     jPanel4.setFont(new java.awt.Font(PRIMARY_FONT, 0, 18)); // NOI18N
00663
00664     jLabel8.setBackground(new java.awt.Color(248, 244, 230));
00665     jLabel8.setFont(new java.awt.Font(PRIMARY_FONT, 0, 14)); // NOI18N
00666     jLabel8.setText("Enter new item information carefully to add.");
00667
00668     jLabel9.setBackground(new java.awt.Color(255, 153, 0));
00669     jLabel9.setFont(new java.awt.Font(PRIMARY_FONT, 1, 25)); // NOI18N

```



```
00752         .addGap(18, 18, 18))));  
00753     jPanel4Layout.setVerticalGroup()  
00754         jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
00755             .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,  
00756                 jPanel4Layout.createSequentialGroup()  
00757                     .addContainerGap()  
00758                     .addComponent(jLabel9)  
00759                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 21,  
Short.MAX_VALUE)  
00760         .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  
00761             .addComponent(jLabel8)  
00762             .addComponent(jLabel10))  
00763             .addGap(18, 18, 18)  
00764             .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
00765                 .addGroup(jPanel4Layout.createSequentialGroup()  
00766                     .addComponent(itemname2, javax.swing.GroupLayout.PREFERRED_SIZE, 59,  
javax.swing.GroupLayout.PREFERRED_SIZE)  
00767                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  
00768                     .addComponent(itemprice2, javax.swing.GroupLayout.PREFERRED_SIZE, 59,  
javax.swing.GroupLayout.PREFERRED_SIZE)  
00769                     .addGap(37, 37, 37)  
00770             .addGap(37, 37, 37)  
00771         .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  
00772             .addComponent(jButton6, javax.swing.GroupLayout.PREFERRED_SIZE, 42,  
javax.swing.GroupLayout.PREFERRED_SIZE)  
00773             .addComponent(jButton7, javax.swing.GroupLayout.PREFERRED_SIZE, 42,  
javax.swing.GroupLayout.PREFERRED_SIZE))  
00774             .addComponent(jScrollPane3, javax.swing.GroupLayout.PREFERRED_SIZE, 203,  
javax.swing.GroupLayout.PREFERRED_SIZE))  
00775             .addGap(48, 48, 48));  
00776  
00777     jTabbedPane.addTab("MainCourse", jPanel1);  
00778  
00779     jButton3.setBackground(new java.awt.Color(255, 255, 255));  
00780     jButton3.setFont(new java.awt.Font(PRIMARY_FONT, 1, 18)); // NOI18N  
00781     jButton3.setText("Go Back");  
00782     jButton3.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));  
00783     jButton3.addActionListener(this::jButton3ActionPerformed);  
00784  
00785     javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);  
00786     jPanel1.setLayout(jPanel1Layout);  
00787     jPanel1Layout.setHorizontalGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
00788         .addGroup(jPanel1Layout.createSequentialGroup()  
00789             .addComponent(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
00790                 .setHorizontalGroup()  
00791                     jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
00792                         .addGroup(jPanel1Layout.createSequentialGroup()  
00793                             .addGap(25, 25, 25)  
00794                             .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
00795                                 .addGroup(jPanel1Layout.createSequentialGroup()  
00796                                     .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 122,  
javax.swing.GroupLayout.PREFERRED_SIZE)  
00797                                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,  
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)  
00798                                     .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 689,  
javax.swing.GroupLayout.PREFERRED_SIZE)  
00799                                     .addGap(218, 218, 218))  
00800                                 .addGroup(jPanel1Layout.createSequentialGroup()  
00801                                     .addComponent(jTabbedPane, javax.swing.GroupLayout.PREFERRED_SIZE,  
javax.swing.GroupLayout.DEFAULT_SIZE,  
00802                                         javax.swing.GroupLayout.PREFERRED_SIZE)  
00803                                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)  
00804                                         .addComponent(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
00805                                             .addGroup(jPanel1Layout.createSequentialGroup()  
00806                                                 .addGap(29, Short.MAX_VALUE))));  
00807                                         .setVerticalGroup()  
00808                                         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
00809                                         .addGroup(jPanel1Layout.createSequentialGroup()  
00810                                             .addComponent(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)  
00811                                                 .addGroup(jPanel1Layout.createSequentialGroup()  
00812                                                 .addComponent(jLabel11, javax.swing.GroupLayout.PREFERRED_SIZE, 60,  
javax.swing.GroupLayout.PREFERRED_SIZE)  
00813                                                 .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 37,  
javax.swing.GroupLayout.PREFERRED_SIZE)  
00814                                                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED,  
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)  
00815                                                 .addComponent(jTabbedPane, javax.swing.GroupLayout.PREFERRED_SIZE, 392,  
javax.swing.GroupLayout.PREFERRED_SIZE)  
00816                                                 .addGap(29, 29, 29));  
00817                                         javax.swing.GroupLayout.PREFERRED_SIZE)  
00818                                         .addComponent(layout, javax.swing.GroupLayout.PREFERRED_SIZE, 392,  
00819                                         javax.swing.GroupLayout.PREFERRED_SIZE)  
00820                                         .addGap(29, 29, 29));  
00821                                         javax.swing.GroupLayout.PREFERRED_SIZE)  
00822                                         .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,  
00823                                         javax.swing.GroupLayout.PREFERRED_SIZE)  
00824                                         .setHorizontalGroup()  
00825                                         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
00826                                         .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,  
00827                                         javax.swing.GroupLayout.PREFERRED_SIZE,  
Short.MAX_VALUE));  
00828                                         layout.setVerticalGroup()  
00829                                         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
00830                                         .addGroup(layout.createSequentialGroup()  
00831                                             .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
```

```

00832         javax.swing.GroupLayout.DEFAULT_SIZE,
00833             javax.swing.GroupLayout.PREFERRED_SIZE)
00834         .addGap(0, 0, Short.MAX_VALUE)));
00835     pack();
00836     setLocationRelativeTo(null);
00837 } // </editor-fold> //GEN-END:initComponents

```

References [es.ull.esit.app.AdminProducts.PRIMARY\\_FONT](#), [es.ull.esit.app.AdminProducts.SECOND\\_COLUMN\\_HEADER](#), and [es.ull.esit.app.AdminProducts.THIRD\\_COLUMN\\_HEADER](#).

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

<code>evt</code>	[ <code>java.awt.event.ActionEvent</code> ] Event triggered by the button click.
------------------	--

Definition at line 1052 of file [AdminProducts.java](#).

```

01052 //GEN-FIRST:event_jButton1ActionPerformed
01053     String name = itemname.getText();
01054     String price = itemprice.getText();
01055
01056     new Thread(() -> {
01057         try {
01058             productService.updateDrink(selectedDrinkID, name, price);
01059             SwingUtilities.invokeLater(() -> {
01060                 JOptionPane.showMessageDialog(this, "Drink updated successfully.");
01061                 itemname.setText("");
01062                 itemprice.setText("");
01063                 selectedDrinkID = null;
01064                 loadDrinks();
01065             });
01066         } catch (Exception ex) {
01067             SwingUtilities
01068                 .invokeLater(() -> JOptionPane.showMessageDialog(this, "Error updating drink: " +
ex.getMessage()));
01069         }
01070     }).start();
01071 //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

<code>evt</code>	[java.awt.event.ActionEvent] Event triggered by the button click.
------------------	---

Definition at line 849 of file AdminProducts.java.

```
00849 //////////////////////////////////////////////////////////////////
00850     GEN-FIRST:event_jButton2ActionPerformed
00851     String name = itemname.getText();
00852     String price = itemprice.getText();
00853     new Thread(() -> {
00854         try {
00855             productService.addDrink(name, price);
00856             SwingUtilities.invokeLater(() -> {
00857                 JOptionPane.showMessageDialog(this, "Drink Added Successfully.");
00858                 itemname.setText("");
00859                 itemprice.setText("");
00860                 loadDrinks();
00861             });
00862         } catch (Exception ex) {
00863             SwingUtilities.invokeLater(() -> JOptionPane.showMessageDialog(this, "Error adding drink: " +
00864                         ex.getMessage()));
00865         }).start();
00866 } // 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

<code>evt</code>	[java.awt.event.ActionEvent] Event triggered by the button click.
------------------	---

Definition at line 876 of file AdminProducts.java.

```
00876 //////////////////////////////////////////////////////////////////
00877     GEN-FIRST:event_jButton3ActionPerformed
00878     new AdminLogin().setVisible(true);
00879     this.dispose();
00880 } // 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

<code>evt</code>	[ <code>java.awt.event.ActionEvent</code> ] Event triggered by the button click.
------------------	--

Definition at line 1081 of file [AdminProducts.java](#).

```
01081 // GEN-FIRST:event_jButton4ActionPerformed
01082     String name = itemnamel.getText();
01083     String price = itempricel.getText();
01084
01085     new Thread(() -> {
01086         try {
01087             productService.updateAppetizer(selectedAppetizerID, name, price);
01088             SwingUtilities.invokeLater(() -> {
01089                 JOptionPane.showMessageDialog(this, "Appetizer updated successfully.");
01090                 itemnamel.setText("");
01091                 itempricel.setText("");
01092                 selectedAppetizerID = null;
01093                 loadAppetizer();
01094             });
01095         } catch (Exception ex) {
01096             SwingUtilities
01097                 .invokeLater(() -> JOptionPane.showMessageDialog(this, "Error updating appetizer: " +
01098                         ex.getMessage()));
01099         });
01100     } // 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

<code>evt</code>	[ <code>java.awt.event.ActionEvent</code> ] Event triggered by the button click.
------------------	--

Definition at line 889 of file [AdminProducts.java](#).

```
00889 // GEN-FIRST:event_jButton5ActionPerformed
00890     String name = itemnamel.getText();
00891     String price = itempricel.getText();
00892
00893     new Thread(() -> {
00894         try {
```

```

00895     productService.addAppetizer(name, price);
00896     SwingUtilities.invokeLater(() -> {
00897         JOptionPane.showMessageDialog(this, "Appetizer Added Successfully.");
00898         itemname1.setText("");
00899         itemprice1.setText("");
00900         loadAppetizer();
00901     });
00902 } catch (Exception ex) {
00903     SwingUtilities
00904         .invokeLater(() -> JOptionPane.showMessageDialog(this, "Error adding appetizer: " +
00905             ex.getMessage()));
00906 }
00907 } // GEN-LAST:event_jButton5ActionPerformed

```

### 7.3.3.7 jButton6ActionPerformed()

```

void es.ul.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

<code>evt</code>	[ <code>java.awt.event.ActionEvent</code> ] Event triggered by the button click.
------------------	--

Definition at line 1110 of file [AdminProducts.java](#).

```

01110                                     { // 
01111     GEN-FIRST:event_jButton6ActionPerformed
01112     String name = itemname2.getText();
01113     String price = itemprice2.getText();
01114     new Thread(() -> {
01115         try {
01116             productService.updateMainCourse(selectedMainCourseID, name, price);
01117             SwingUtilities.invokeLater(() -> {
01118                 JOptionPane.showMessageDialog(this, "Main course updated successfully.");
01119                 itemname2.setText("");
01120                 itemprice2.setText("");
01121                 selectedMainCourseID = null;
01122                 loadmainCourse();
01123             });
01124         } catch (Exception ex) {
01125             SwingUtilities
01126                 .invokeLater(() -> JOptionPane.showMessageDialog(this, "Error updating main course: " +
01127                     ex.getMessage()));
01128         }
01129     } // GEN-LAST:event_jButton6ActionPerformed

```

### 7.3.3.8 jButton7ActionPerformed()

```

void es.ul.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**

<code>evt</code>	[ <code>java.awt.event.ActionEvent</code> ] Event triggered by the button click.
------------------	--

Definition at line 917 of file [AdminProducts.java](#).

```
00917     GEN-FIRST:event_jButton7ActionPerformed
00918     String name = itemname2.getText();
00919     String price = itemprice2.getText();
00920
00921     new Thread(() -> {
00922         try {
00923             productService.addMainCourse(name, price);
00924             SwingUtilities.invokeLater(() -> {
00925                 JOptionPane.showMessageDialog(this, "Main Course Added Successfully.");
00926                 itemname2.setText("");
00927                 itemprice2.setText("");
00928                 loadmainCourse();
00929             });
00930         } catch (Exception ex) {
00931             SwingUtilities
00932                 .invokeLater(() -> JOptionPane.showMessageDialog(this, "Error adding main course: " +
ex.getMessage()));
00933         }
00934     }).start();
00935 } // 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**

<code>evt</code>	[ <code>java.awt.event.KeyEvent</code> ] Key event generated by the table.
------------------	--

Definition at line 945 of file [AdminProducts.java](#).

```
00945
00946     // No action needed for key press in this version.
00947 } // GEN-LAST:event_jTable1KeyPressed
```

{// GEN-FIRST: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**

<code>evt</code>	[ <code>java.awt.event.MouseEvent</code> ] Mouse event generated by the table.
------------------	--

Definition at line 959 of file `AdminProducts.java`.

```

00959             { ///
00960     GEN-FIRST:event_jTable1MouseClicked
00961     int row = jTable1.getSelectedRow();
00962     if (row == -1)
00963         return;
00964     String idStr = jTable1.getValueAt(row, 0).toString();
00965     selectedDrinkID = Long.valueOf(idStr);
00966     LOGGER.info("Selected Drink ID: {}", selectedDrinkID);
00967
00968     new Thread(() -> {
00969         try {
00970             Drink drink = productService.getDrinkById(selectedDrinkID);
00971             SwingUtilities.invokeLater(() -> {
00972                 itemname.setText(drink.getItemDrinks());
00973                 itemprice.setText(String.valueOf(drink.getDrinksPrice()));
00974             });
00975         } catch (Exception ex) {
00976             SwingUtilities
00977                 .invokeLater(() -> JOptionPane.showMessageDialog(null, "Error loading drink details: " +
00978                     ex.getMessage()));
00979         });
00980     }).start();
00981 } // 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**

<code>evt</code>	[ <code>java.awt.event.MouseEvent</code> ] Mouse event generated by the table.
------------------	--

Definition at line 990 of file `AdminProducts.java`.

```

00990             { ///
00991     GEN-FIRST:event_jTable2MouseClicked
00992     int row = jTable2.getSelectedRow();
00993     if (row == -1)
00994         return;
00995     String idStr = jTable2.getValueAt(row, 0).toString();
00996     selectedAppetizerID = Long.valueOf(idStr);
00997     LOGGER.info("Selected Appetizer ID: {}", selectedAppetizerID);
00998
00999     new Thread(() -> {
01000         try {
01001             Appetizer appetizer = productService.getAppetizerById(selectedAppetizerID);
01002             SwingUtilities.invokeLater(() -> {
01003                 itemnamel.setText(appetizer.getItemAppetizers());
01004                 itempricel.setText(String.valueOf(appetizer.getAppetizersPrice()));
01005             });
01006         } catch (Exception ex) {
01007             SwingUtilities.invokeLater(
01008                 () -> JOptionPane.showMessageDialog(null, "Error loading appetizer details: " +
01009                     ex.getMessage()));
01010         });
01011     }).start();
01012 } // 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

<code>evt</code>	[ <code>java.awt.event.MouseEvent</code> ] Mouse event generated by the table.
------------------	--

Definition at line 1021 of file `AdminProducts.java`.

```
01021                                     { //
01022     GEN-FIRST:event_jTable3MouseClicked
01023     int row = jTable3.getSelectedRow();
01024     if (row == -1)
01025         return;
01026
01027     String idStr = jTable3.getValueAt(row, 0).toString();
01028     selectedMainCourseID = Long.valueOf(idStr);
01029     LOGGER.info("Selected Main Course ID: {}", selectedMainCourseID);
01030
01031     new Thread(() -> {
01032         try {
01033             MainCourse mainCourse = productService.getMainCourseById(selectedMainCourseID);
01034             SwingUtilities.invokeLater(() -> {
01035                 itemname2.setText(mainCourse.getItemFood());
01036                 itemprice2.setText(String.valueOf(mainCourse.getFoodPrice()));
01037             });
01038         } catch (Exception ex) {
01039             SwingUtilities.invokeLater(
01040                 () -> JOptionPane.showMessageDialog(null, "Error loading main course details: " +
01041                     ex.getMessage()));
01042         }
01043     }).start();
01044 } // 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

<code>args</code>	[ <code>String[]</code> ] Command line arguments (not used).
-------------------	--

Definition at line 1140 of file `AdminProducts.java`.

```
01140
01141     try {
01142         for (javax.swing.UIManager.LookAndFeelInfo info :
01143             javax.swing.UIManager.getInstalledLookAndFeels()) {
```

```

01143     if ("Nimbus".equals(info.getName())) {
01144         javax.swing.UIManager.setLookAndFeel(info.getClassName());
01145         break;
01146     }
01147 }
01148 } catch (ClassNotFoundException | InstantiationException | IllegalAccessException
01149 | javax.swing.UnsupportedLookAndFeelException ex) {
01150     java.util.logging.Logger.getLogger(AdminProducts.class.getName()).log(java.util.logging.Level.SEVERE,
01151     null, ex);
01152 // </editor-fold>
01153
01154     /* Create and display the form */
01155     java.awt.EventQueue.invokeLater(() -> new AdminProducts().setVisible(true));
01156 }

```

### 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 140 of file [AdminProducts.java](#).

```

00140
00141     loadDrinks();
00142     loadAppetizer();
00143     loadmainCourse();
00144 }

```

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 FIRST\_COLUMN\_HEADER

final String es.ull.esit.app.AdminProducts.FIRST\_COLUMN\_HEADER = "ID" [static], [private]

Column header for the ID column in product tables.

Definition at line 40 of file [AdminProducts.java](#).

### 7.3.4.2 LOGGER

```
final Logger es.ull.esit.app.AdminProducts.LOGGER = LoggerFactory.getLogger(AdminProducts.class) [static], [private]
```

Logger for logging events and errors.

Definition at line 32 of file [AdminProducts.java](#).

### 7.3.4.3 PRIMARY\_FONT

```
final String es.ull.esit.app.AdminProducts.PRIMARY_FONT = "Yu Gothic UI" [static], [private]
```

Primary font used throughout the GUI.

Definition at line 35 of file [AdminProducts.java](#).

Referenced by [es.ull.esit.app.AdminProducts.initComponents\(\)](#).

### 7.3.4.4 productService

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

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

Definition at line 29 of file [AdminProducts.java](#).

### 7.3.4.5 SECOND\_COLUMN\_HEADER

```
final String es.ull.esit.app.AdminProducts.SECOND_COLUMN_HEADER = "Item Name" [static], [private]
```

Column header for the name column in product tables.

Definition at line 42 of file [AdminProducts.java](#).

Referenced by [es.ull.esit.app.AdminProducts.initComponents\(\)](#).

### 7.3.4.6 SECONDARY\_FONT

```
final String es.ull.esit.app.AdminProducts.SECONDARY_FONT = "Thonburi" [static], [private]
```

Secondary font used for buttons and smaller text.

Definition at line 37 of file [AdminProducts.java](#).

#### 7.3.4.7 THIRD\_COLUMN\_HEADER

```
final String es.ull.esit.app.AdminProducts.THIRD_COLUMN_HEADER = "Item Price" [static], [private]
```

Column header for the price column in product tables.

Definition at line 44 of file [AdminProducts.java](#).

Referenced by [es.ull.esit.app.AdminProducts.initComponents\(\)](#).

#### 7.3.4.8 UPDATE\_STRING

```
final String es.ull.esit.app.AdminProducts.UPDATE_STRING = "Update" [static], [private]
```

String constant for the "Update" button label.

Definition at line 47 of file [AdminProducts.java](#).

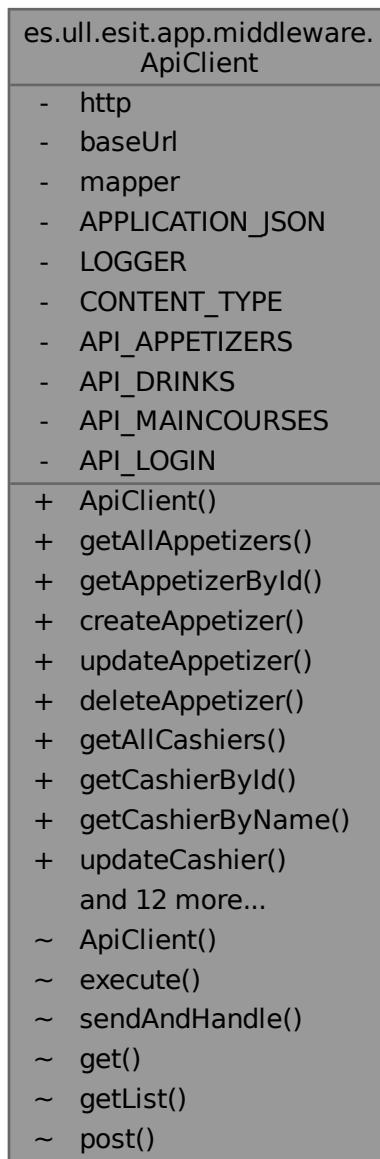
The documentation for this class was generated from the following file:

- [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.ul.esit.app.middleware.ApiClient:



## Classes

- interface [IOCallable](#)  
*Functional interface for lambdas that may throw InterruptedException or IOException.*
- interface [ResponseHandler](#)  
*Functional interface for processing HTTP responses and possibly throwing IOException from JSON parsing.*

## Public Member Functions

- [ApiClient \(String baseUrl\)](#)

- Constructs the ApiClient with the given base URL.*
- List< Appetizer > [getAllAppetizers \(\)](#)  
*GET all appetizers.*
  - Appetizer [getAppetizerById \(Long id\)](#)  
*GET appetizer by ID.*
  - Appetizer [createAppetizer \(Appetizer appetizer\)](#)  
*POST create new appetizer.*
  - Appetizer [updateAppetizer \(Long id, Appetizer appetizer\)](#)  
*PUT update appetizer by ID.*
  - void [deleteAppetizer \(Long id\)](#)  
*DELETE appetizer by ID.*
  - List< Cashier > [getAllCashiers \(\)](#)  
*GET all cashiers.*
  - Cashier [getCashierById \(Long id\)](#)  
*GET cashier by ID.*
  - Cashier [getCashierByName \(String name\)](#)  
*GET cashier by name.*
  - Cashier [updateCashier \(Long id, Cashier cashier\)](#)  
*PUT update cashier by ID.*
  - List< Drink > [getAllDrinks \(\)](#)  
*GET all drinks.*
  - Drink [getDrinkById \(Long id\)](#)  
*GET drink by ID.*
  - Drink [createDrink \(Drink drink\)](#)  
*POST create new drink.*
  - Drink [updateDrink \(Long id, Drink drink\)](#)  
*PUT update drink by ID.*
  - void [deleteDrink \(Long id\)](#)  
*DELETE drink by ID.*
  - List< MainCourse > [getAllMainCourses \(\)](#)  
*GET all maincourses.*
  - MainCourse [getMainCourseById \(Long id\)](#)  
*GET maincourse by ID.*
  - MainCourse [createMainCourse \(MainCourse mainCourse\)](#)  
*POST create new maincourse.*
  - MainCourse [updateMainCourse \(Long id, MainCourse mainCourse\)](#)  
*PUT update maincourse by ID.*
  - void [deleteMainCourse \(Long id\)](#)  
*DELETE maincourse by ID.*
  - void [login \(String ignored\)](#)  
*Legacy login method kept for compatibility.*
  - User [login \(String username, String password\)](#)  
*Authenticates a user against the backend.*

## Private Attributes

- final HttpClient [http](#)  
*Low-level HTTP client to send requests.*
- final String [baseUrl](#)  
*Base URL of the REST API.*
- final ObjectMapper [mapper](#)  
*JSON object mapper for serialization/deserialization: converts between JSON and Java objects.*

## Static Private Attributes

- static final String **APPLICATION\_JSON** = "application/json"  
*Constant for JSON content type.*
- static final Logger **LOGGER** = LoggerFactory.getLogger(ApiClient.class)  
*Logger for logging events and errors.*
- static final String **CONTENT\_TYPE** = "Content-Type"  
*HTTP header for content type.*
- static final String **API\_APPETIZERS** = "/api/appetizers/"  
*API endpoint for appetizers.*
- static final String **API\_DRINKS** = "/api/drinks/"  
*API endpoint for drinks.*
- static final String **API\_MAINCOURSES** = "/api/maincourses/"  
*API endpoint for main courses.*
- static final String **API\_LOGIN** = "/api/login"

### 7.4.1 Detailed Description

API client for interacting with the backend REST API.

Wraps HttpClient and provides methods to:  
 - perform CRUD operations on Appetizers, Cashiers, Drinks, MainCourses.  
 - send login requests to the backend.

Definition at line 31 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

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

Definition at line 74 of file [ApiClient.java](#).

```
00074
00075      this.baseUrl = baseUrl.endsWith("/") ? baseUrl.substring(0, baseUrl.length() - 1) : baseUrl;
00076      this.http = HttpClient.newBuilder()
00077          .connectTimeout(Duration.ofSeconds(5))
00078          .build();
00079      this.mapper = new ObjectMapper();
00080 }
```

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 ) [inline]
```

POST create new appetizer.

Creates a new appetizer in the backend.

##### Parameters

<code>appetizer</code>	[Appetizer] Appetizer object to create.
------------------------	---

##### Returns

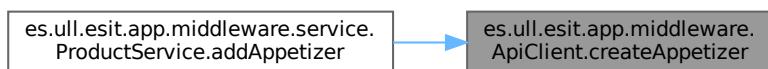
[Appetizer] Created Appetizer object returned from the backend.

Definition at line 274 of file [ApiClient.java](#).

```
00274     {
00275         return post("/api/appetizers", appetizer, Appetizer.class);
00276     }
```

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 ) [inline]
```

POST create new drink.

Creates a new drink in the backend.

**Parameters**

<code>drink</code>	[Drink] Drink object to create.
--------------------	---------------------------------

**Returns**

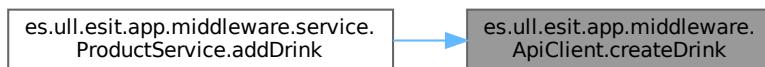
[Drink] Created Drink object returned from the backend.

Definition at line 416 of file [ApiClient.java](#).

```
00416             {
00417     return post("/api/drinks", drink, Drink.class);
00418 }
```

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 ) [inline]
```

POST create new maincourse.

Creates a new maincourse in the backend.

**Parameters**

<code>mainCourse</code>	[MainCourse] MainCourse object to create.
-------------------------	---

**Returns**

[MainCourse] Created MainCourse object returned from the backend.

Definition at line 498 of file [ApiClient.java](#).

```
00498             {
00499     return post("/api/maincourses", mainCourse, MainCourse.class);
00500 }
```

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 ) [inline]
```

DELETE appetizer by ID.

Deletes an existing appetizer in the backend.

##### Parameters

<i>id</i>	[Long] ID of the appetizer to delete.
-----------	---------------------------------------

Definition at line 308 of file [ApiClient.java](#).

```
00308
00309     String path = API_APPETIZERS + id;
00310     execute("deleteAppetizer id=" + id, () -> sendAndHandle("DELETE " + path,
00311             HttpRequest.newBuilder().uri(URI.create(baseUrl + path)).DELETE().build(), (status, body) -> {
00312                 if (status < 200 || status >= 300) {
00313                     throw new ApiClientException("DELETE " + path + " failed with HTTP " + status + " body: "
00314                         + body);
00315                 }
00316             }));
00317 }
```

#### 7.4.3.5 deleteDrink()

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

DELETE drink by ID.

Deletes an existing drink in the backend.

##### Parameters

<i>id</i>	[Long] ID of the drink to delete.
-----------	-----------------------------------

Definition at line 452 of file [ApiClient.java](#).

```
00452
00453     String path = API_DRINKS + id;
```

```

00454     execute("deleteDrink id=" + id, () -> sendAndHandle("DELETE " + path,
00455         HttpRequest.newBuilder().uri(URI.create(baseUrl + path)).DELETE().build(), (status, body) -> {
00456             if (status < 200 || status >= 300) {
00457                 throw new ApiClientException("DELETE " + path + " failed with HTTP " + status + " body: " +
00458                     + body);
00459             }
00460         });
00461     }

```

#### 7.4.3.6 deleteMainCourse()

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

**DELETE** maincourse by ID.

Deletes an existing maincourse in the backend by its ID.

##### Parameters

<i>id</i>	[Long] ID of the maincourse to delete.
-----------	--

**Definition at line 534 of file ApiClient.java.**

```

00534     String path = API_MAINCOURSES + id;
00535     execute("deleteMainCourse id=" + id, () -> sendAndHandle("DELETE " + path,
00536         HttpRequest.newBuilder().uri(URI.create(baseUrl + path)).DELETE().build(), (status, body) -> {
00537             if (status < 200 || status >= 300) {
00538                 throw new ApiClientException("DELETE " + path + " failed with HTTP " + status + " body: " +
00539                     + body);
00540             }
00541         });
00542     });
00543 }

```

#### 7.4.3.7 getAllAppetizers()

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

**GET** all appetizers.

Retrieves a list of all appetizers from the backend.

##### Returns

[List<Appetizer>] List of all appetizers.

**Definition at line 251 of file ApiClient.java.**

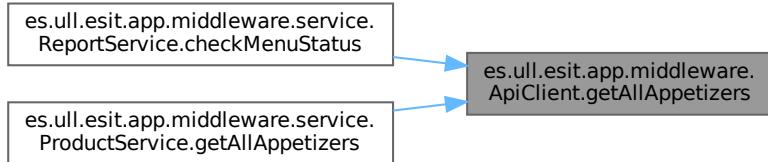
```

00251     {
00252         return getList("/api/appetizers", new TypeReference<List<Appetizer>>() { });
00253     }

```

Referenced by [es.ul.esit.app.middleware.service.ReportService.checkMenuStatus\(\)](#), and [es.ul.esit.app.middleware.service.ProductS](#)

Here is the caller graph for this function:



#### 7.4.3.8 getAllCashiers()

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

GET all cashiers.

Retrieves a list of all cashiers from the backend.

##### Returns

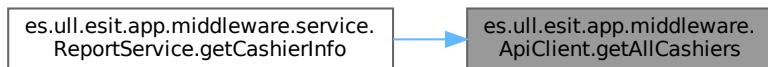
`[List<Cashier>]` List of all cashiers.

Definition at line 330 of file [ApiClient.java](#).

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

Referenced by [es.ul.esit.app.middleware.service.ReportService.getCashierInfo\(\)](#).

Here is the caller graph for this function:



#### 7.4.3.9 getAllDrinks()

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

GET all drinks.

Retrieves a list of all drinks from the backend.

**Returns**

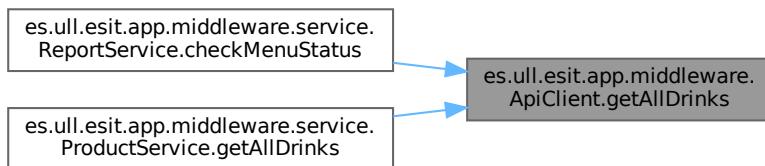
[List<Drink>] List of all drinks.

Definition at line 392 of file [ApiClient.java](#).

```
00392      {
00393         return getList("/api/drinks", new TypeReference<List<Drink>>() {});
00394     }
```

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 ( ) [inline]

GET all maincourses.

Retrieves a list of all maincourses from the backend.

**Returns**

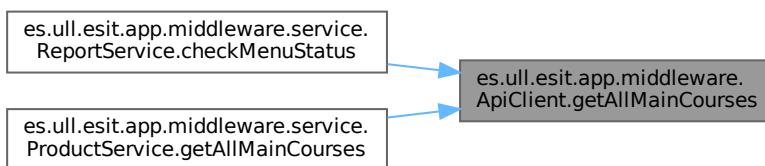
[List<MainCourse>] List of all maincourses.

Definition at line 474 of file [ApiClient.java](#).

```
00474      {
00475         return getList("/api/maincourses", new TypeReference<List<MainCourse>>() {});
00476     }
```

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) [inline]
```

GET appetizer by ID.

Retrieves an appetizer by its ID from the backend.

##### Parameters

<i>id</i>	[Long] ID of the appetizer.
-----------	-----------------------------

##### Returns

[Appetizer] Appetizer object.

Definition at line 263 of file [ApiClient.java](#).

```
00263             {
00264     return get(API_APPETIZERS + id, Appetizer.class);
00265 }
```

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) [inline]
```

GET cashier by ID.

Retrieves a cashier by its ID from the backend.

##### Parameters

<i>id</i>	[Long] ID of the cashier.
-----------	---------------------------

**Returns**

[Cashier] Cashier object.

Definition at line 341 of file [ApiClient.java](#).

```
00341     {
00342         return get("/api/cashiers/" + id, Cashier.class);
00343     }
```

#### 7.4.3.13 getCashierByName()

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

GET cashier by name.

Retrieves a cashier by its username from the backend.

**Parameters**

<i>name</i>	[String] username of the cashier.
-------------	-----------------------------------

**Returns**

[Cashier] Cashier object.

Definition at line 352 of file [ApiClient.java](#).

```
00352     {
00353         return get("/api/cashiers/name/" + name, Cashier.class);
00354     }
```

#### 7.4.3.14 getDrinkById()

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

GET drink by ID.

Retrieves a drink by its ID from the backend.

**Parameters**

<i>id</i>	[Long] ID of the drink.
-----------	-------------------------

**Returns**

[Drink] Drink object.

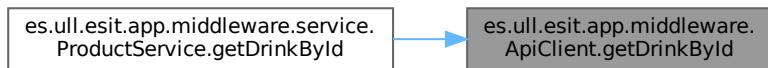
Definition at line 404 of file [ApiClient.java](#).

```
00404     {
00405         return get(API_DRINKS + id, Drink.class);
```

```
00406 }
```

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 (\n    Long id ) [inline]
```

GET maincourse by ID.

Retrieves a maincourse by its ID from the backend.

##### Parameters

<i>id</i>	[Long] ID of the maincourse.
-----------	------------------------------

##### Returns

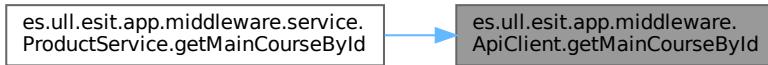
[MainCourse] MainCourse object.

Definition at line 486 of file [ApiClient.java](#).

```
00486\n00487     return get(API_MAINCOURSES + id, MainCourse.class);\n00488 }
```

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) [inline]
```

Legacy login method kept for compatibility.

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

##### Parameters

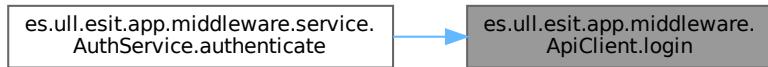
<i>ignored</i>	[String] Ignored parameter.
----------------	-----------------------------

Definition at line 556 of file [ApiClient.java](#).

```
00556
00557     // No-op: kept for compatibility.
00558 }
```

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) [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 ApiClientException.

##### Parameters

<i>username</i>	[String] Username entered by the user.
<i>password</i>	[String] Password entered by the user.

**Returns**

[User] Authenticated User object.

**Definition at line 571 of file ApiClient.java.**

```

00571
00572     String path = API_LOGIN;
00573     return execute("login for user=" + username, () -> {
00574         String jsonBody = mapper.writeValueAsString(Map.of(
00575             "username", username,
00576             "password", password));
00577
00578         HttpRequest request = HttpRequest.newBuilder().uri(URI.create(baseUrl + path))
00579             .header(CONTENT_TYPE,
00580                 APPLICATION_JSON).POST(HttpRequest.BodyPublishers.ofString(jsonBody)).build();
00581
00582         return sendAndHandle("POST " + path, request, (status, body) -> {
00583             if (status == 200) {
00584                 return mapper.readValue(body, User.class);
00585             } else {
00586                 throw new ApiClientException("Login failed with status: " + status + " body: " + body);
00587             }
00588         });
00589     });
00589 }
```

**7.4.3.18 updateAppetizer()**

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

PUT update appetizer by ID.

Updates an existing appetizer in the backend.

**Parameters**

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

**Returns**

[Appetizer] Updated Appetizer object returned from the backend.

**Definition at line 286 of file ApiClient.java.**

```

00286
00287     String path = API_APPETIZERS + id;
00288     return execute("updateAppetizer id=" + id, () -> {
00289         String json = mapper.writeValueAsString(appetizer);
00290         HttpRequest req = HttpRequest.newBuilder().uri(URI.create(baseUrl + path))
00291             .PUT(HttpRequest.BodyPublishers.ofString(json)).header(CONTENT_TYPE,
00292                 APPLICATION_JSON).build();
00293
00294         return sendAndHandle("PUT " + path, req, (status, body) -> {
00295             if (status < 200 || status >= 300) {
00296                 throw new ApiClientException("PUT " + path + " failed with HTTP " + status + " body: " +
00297                     body);
00298             }
00299             return mapper.readValue(body, Appetizer.class);
00300         });
00300 }
```

Referenced by [es.ul.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 ) [inline]
```

PUT update cashier by ID.

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

#### Parameters

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

#### Returns

[Cashier] Updated Cashier object returned from the backend.

Definition at line 365 of file [ApiClient.java](#).

```
00365
00366     String path = "/api/cashiers/" + id;
00367     return execute("updateCashier id=" + id, () -> {
00368         String json = mapper.writeValueAsString(cashier);
00369         HttpRequest req = HttpRequest.newBuilder().uri(URI.create(baseUrl + path))
00370             .PUT(HttpRequest.BodyPublishers.ofString(json)).header(CONTENT_TYPE,
00371             APPLICATION_JSON).build();
00372
00373         return sendAndHandle("PUT " + path, req, (status, body) -> {
00374             if (status < 200 || status >= 300) {
00375                 throw new ApiClientException("PUT " + path + " failed with HTTP " + status + " body: " +
00376                     body);
00377             }
00378         });
00379     }
```

#### 7.4.3.20 updateDrink()

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

PUT update drink by ID.

Updates an existing drink in the backend by its ID.

**Parameters**

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

**Returns**

[Drink] Updated Drink object returned from the backend.

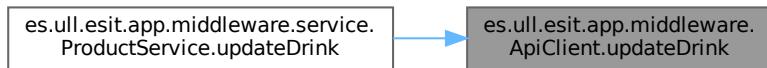
Definition at line 429 of file [ApiClient.java](#).

```

00429             {
00430     String path = API_DRINKS + id;
00431     return execute("updateDrink id=" + id, () -> {
00432         String json = mapper.writeValueAsString(drink);
00433         HttpRequest req = HttpRequest.newBuilder().uri(URI.create(baseUrl + path))
00434             .PUT(HttpRequest.BodyPublishers.ofString(json)).header(CONTENT_TYPE,
00435             APPLICATION_JSON).build();
00436         return sendAndHandle("PUT " + path, req, (status, body) -> {
00437             if (status < 200 || status >= 300) {
00438                 throw new ApiClientException("PUT " + path + " failed with HTTP " + status + " body: " +
00439                     body);
00440             }
00441             return mapper.readValue(body, Drink.class);
00442         });
00443     });
00444 }
```

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 ) [inline]
```

PUT update maincourse by ID.

Updates an existing maincourse in the backend.

**Parameters**

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

**Returns**

[MainCourse] Updated MainCourse object returned from the backend.

Definition at line 511 of file [ApiClient.java](#).

```

00511
00512     String path = API_MAINCOURSES + id;
00513     return execute("updateMainCourse id=" + id, () -> {
00514         String json = mapper.writeValueAsString(mainCourse);
00515         HttpRequest req = HttpRequest.newBuilder().uri(URI.create(baseUrl + path))
00516             .PUT(HttpRequest.BodyPublishers.ofString(json)).header(CONTENT_TYPE,
00517             APPLICATION_JSON).build();
00518         return sendAndHandle("PUT " + path, req, (status, body) -> {
00519             if (status < 200 || status >= 300) {
00520                 throw new ApiClientException("PUT " + path + " failed with HTTP " + status + " body: " +
00521                     body);
00522             }
00523             return mapper.readValue(body, MainCourse.class);
00524         });
00525     });

```

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 API\_APPETIZERS

```
final String es.ull.esit.app.middleware.ApiClient.API_APPETIZERS = "/api/appetizers/" [static], [private]
```

API endpoint for appetizers.

Definition at line 43 of file [ApiClient.java](#).

### 7.4.4.2 API\_DRINKS

```
final String es.ull.esit.app.middleware.ApiClient.API_DRINKS = "/api/drinks/" [static], [private]
```

API endpoint for drinks.

Definition at line 46 of file [ApiClient.java](#).

### 7.4.4.3 API\_LOGIN

```
final String es.ull.esit.app.middleware.ApiClient.API_LOGIN = "/api/login" [static], [private]
```

Definition at line 51 of file [ApiClient.java](#).

#### 7.4.4.4 API\_MAINCOURSES

```
final String es.ull.esit.app.middleware.ApiClient.API_MAINCOURSES = "/api/maincourses/" [static],  
[private]
```

API endpoint for main courses.

Definition at line 49 of file [ApiClient.java](#).

#### 7.4.4.5 APPLICATION\_JSON

```
final String es.ull.esit.app.middleware.ApiClient.APPLICATION_JSON = "application/json" [static],  
[private]
```

Constant for JSON content type.

Definition at line 34 of file [ApiClient.java](#).

#### 7.4.4.6 baseUrl

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

Base URL of the REST API.

Definition at line 57 of file [ApiClient.java](#).

Referenced by [es.ull.esit.app.middleware.ApiClient.ApiClient\(\)](#).

#### 7.4.4.7 CONTENT\_TYPE

```
final String es.ull.esit.app.middleware.ApiClient.CONTENT_TYPE = "Content-Type" [static],  
[private]
```

HTTP header for content type.

Definition at line 40 of file [ApiClient.java](#).

#### 7.4.4.8 http

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

Low-level HTTP client to send requests.

Definition at line 54 of file [ApiClient.java](#).

#### 7.4.4.9 LOGGER

```
final Logger es.ull.esit.app.middleware.ApiClient.LOGGER = LoggerFactory.getLogger(ApiClient.class) [static], [private]
```

Logger for logging events and errors.

Definition at line 37 of file [ApiClient.java](#).

#### 7.4.4.10 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 63 of file [ApiClient.java](#).

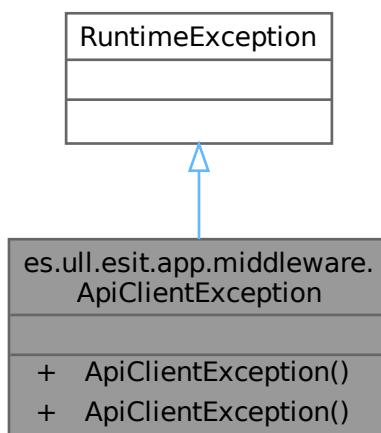
The documentation for this class was generated from the following file:

- [ApiClient.java](#)

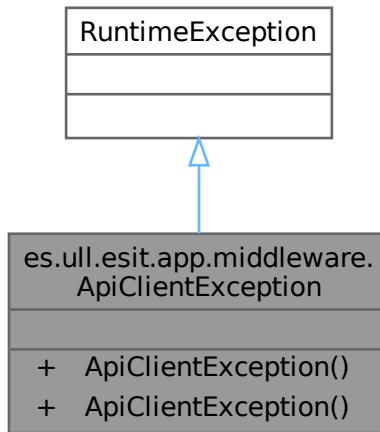
## 7.5 es.ull.esit.app.middleware.ApiClientException Class Reference

Custom exception class for API client errors.

Inheritance diagram for es.ull.esit.app.middleware.ApiClientException:



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



## Public Member Functions

- [ApiClientException](#) (String message)  
*Constructs a new ApiClientException with the specified detail message.*
- [ApiClientException](#) (String message, Throwable cause)  
*Constructs a new ApiClientException with the specified detail message and cause.*

### 7.5.1 Detailed Description

Custom exception class for API client errors.

Inherits from RuntimeException to represent exceptions that occur during API client operations.

Definition at line [9](#) of file [ApiClientException.java](#).

### 7.5.2 Constructor & Destructor Documentation

#### 7.5.2.1 ApiClientException() [1/2]

```
es.ull.esit.app.middleware.ApiClientException.ApiClientException (
    String message )  [inline]
```

Constructs a new ApiClientException with the specified detail message.

##### Parameters

<i>message</i>	[String] Detail message for the exception.
----------------	--

Definition at line 16 of file [ApiClientException.java](#).

```
00016             {
00017     super(message);
00018 }
```

### 7.5.2.2 ApiClientException() [2/2]

```
es.ull.esit.app.middleware.ApiClientException.ApiClientException (
    String message,
    Throwable cause) [inline]
```

Constructs a new ApiClientException with the specified detail message and cause.

#### Parameters

<i>message</i>	[String] Detail message for the exception.
<i>cause</i>	[Throwable] The cause of the exception.

Definition at line 26 of file [ApiClientException.java](#).

```
00026             {
00027     super(message, cause);
00028 }
```

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

- [ApiClientException.java](#)

## 7.6 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:

<pre>es.ull.esit.app.middleware.model. Appetizer</pre>
<ul style="list-style-type: none"> <li>- appetizersId</li> <li>- itemAppetizers</li> <li>- appetizersPrice</li> <li>- receiptId</li> </ul>
<ul style="list-style-type: none"> <li>+ Appetizer()</li> <li>+ Appetizer()</li> <li>+ getAppetizersId()</li> <li>+ setAppetizersId()</li> <li>+ getItemAppetizers()</li> <li>+ setItemAppetizers()</li> <li>+ getAppetizersPrice()</li> <li>+ setAppetizersPrice()</li> <li>+ getReceiptId()</li> <li>+ setReceiptId()</li> </ul>

## 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").*

### **7.6.1 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`.

### 7.6.2 Constructor & Destructor Documentation

### 7.6.2.1 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 }

### 7.6.2.2 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

<i>appetizersId</i>	[Long] Unique identifier of the appetizer.
<i>itemAppetizers</i>	[String] Name of the appetizer item.
<i>appetizersPrice</i>	[Integer] Price of the appetizer.
<i>receiptId</i>	[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](#).

### 7.6.3 Member Function Documentation

#### 7.6.3.1 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](#).

#### 7.6.3.2 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](#).

#### 7.6.3.3 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](#).

### 7.6.3.4 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](#).

### 7.6.3.5 setAppetizersId()

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

Sets the appetizer identifier.

#### Parameters

<i>appetizersId</i>	[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](#).

### 7.6.3.6 setAppetizersPrice()

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

Sets the appetizer price.

#### Parameters

<i>appetizersPrice</i>	[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](#).

### 7.6.3.7 setItemAppetizers()

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

Sets the appetizer item name.

#### Parameters

<i>itemAppetizers</i>	[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](#).

### 7.6.3.8 setReceiptId()

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

Sets the identifier of the related receipt.

#### Parameters

<i>receiptId</i>	[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](#).

## 7.6.4 Member Data Documentation

### 7.6.4.1 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\(\)](#).

#### 7.6.4.2 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\(\)](#).

#### 7.6.4.3 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\(\)](#).

#### 7.6.4.4 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](#)

## 7.7 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:

es.ull.esit.server.middleware.model. Appetizer	
-	appetizersId
-	itemAppetizers
-	appetizersPrice
+	Appetizer()
+	Appetizer()
+	getAppetizersId()
+	setAppetizersId()
+	getItemAppetizers()
+	setItemAppetizers()
+	getAppetizersPrice()
+	setAppetizersPrice()

### 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").*

### 7.7.1 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](#).

### 7.7.2 Constructor & Destructor Documentation

#### 7.7.2.1 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 }
```

#### 7.7.2.2 Appetizer() [2/2]

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

Full constructor.

#### Parameters

<code>appetizersId</code>	[Long] Identifier of the appetizer.
<code>itemAppetizers</code>	[String] Name of the appetizer.
<code>appetizersPrice</code>	[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 }
```

### 7.7.3 Member Function Documentation

#### 7.7.3.1 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     }
```

#### 7.7.3.2 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     }
```

#### 7.7.3.3 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     }
```

#### 7.7.3.4 setAppetizersId()

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

Sets the appetizer ID.

Generated by the database.

**Parameters**

<code>appetizersId</code>	[Long] Appetizer id.
---------------------------	----------------------

Definition at line 65 of file [server/src/main/java/es/ull/esit/server/middleware/model/Appetizer.java](#).

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

### 7.7.3.5 setAppetizersPrice()

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

Sets the appetizer price.

**Parameters**

<code>appetizersPrice</code>	[Integer] Appetizer price.
------------------------------	----------------------------

Definition at line 101 of file [server/src/main/java/es/ull/esit/server/middleware/model/Appetizer.java](#).

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

### 7.7.3.6 setItemAppetizers()

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

Sets the appetizer name.

**Parameters**

<code>itemAppetizers</code>	[String] Appetizer name.
-----------------------------	--------------------------

Definition at line 83 of file [server/src/main/java/es/ull/esit/server/middleware/model/Appetizer.java](#).

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

## 7.7.4 Member Data Documentation

### 7.7.4.1 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](#).

#### 7.7.4.2 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](#).

#### 7.7.4.3 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](#)

## 7.8 es.ull.esit.server.controller.AppetizerController Class Reference

REST controller for managing appetizers.

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

es.ull.esit.server.controller. AppetizerController
- appetizerRepository
+ getAllAppetizers()
+ getAppetizerById()
+ createAppetizer()
+ updateAppetizer()
+ deleteAppetizer()

### 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.*

### 7.8.1 Detailed Description

REST controller for managing appetizers.

Provides 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](#).

### 7.8.2 Member Function Documentation

#### 7.8.2.1 createAppetizer()

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

Creates a new appetizer.

HTTP POST /api/appetizers

#### Parameters

<code>appetizer</code>	[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 }
```

#### 7.8.2.2 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**

<i>id</i>	[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 }
```

**7.8.2.3 getAllAppetizers()**

(ResponseEntity< List< Appetizer > > es.ull.esit.server.controller.AppetizerController.getAllAppetizers ( ) [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 }
```

**7.8.2.4 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**

<i>id</i>	[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 }
```

### 7.8.2.5 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**

<i>id</i>	[Long] Identifier of the appetizer to update.
<i>appetizer</i>	[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 }
```

## 7.8.3 Member Data Documentation

### 7.8.3.1 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.**

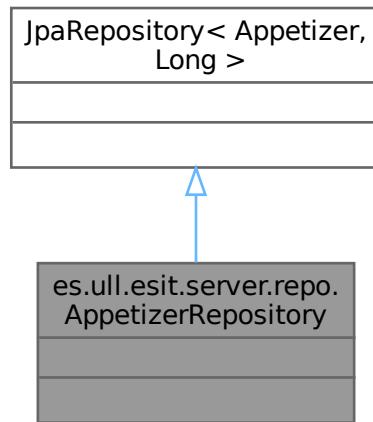
The documentation for this class was generated from the following file:

- [AppetizerController.java](#)

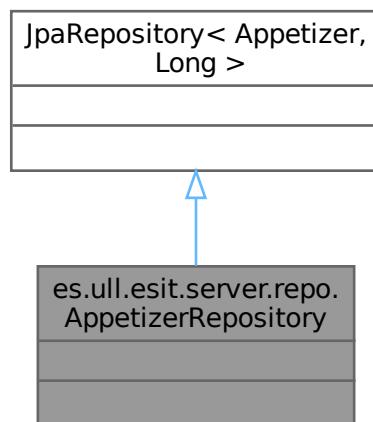
## 7.9 es.ull.esit.server.repo.AppetizerRepository Interface Reference

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:



### 7.9.1 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](#)

## 7.10 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:

es.ull.esit.app.Application Launcher
- loginFactory
+ main()
~ setLoginFactory()
~ getLoginFactory()

### Static Public Member Functions

- static void [main](#) (String[ ] args)  
*Alternate entry point used to launch the Login window.*

### Static Private Attributes

- static java.util.function.Supplier< [Login](#) > [loginFactory](#) = [Login::new](#)  
*Factory (test seam) used to create Login instances.*

### 7.10.1 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](#).

## 7.10.2 Member Function Documentation

### 7.10.2.1 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 using an injectable factory (test seam) so tests can avoid creating real UI components.

#### Parameters

<i>args</i>	[String[]] Command-line arguments (unused).
-------------	---

Definition at line 21 of file [ApplicationLauncher.java](#).

```
00021      {
00022          java.awt.EventQueue.invokeLater(() -> getLoginFactory().get().setVisible(true));
00023      }
```

## 7.10.3 Member Data Documentation

### 7.10.3.1 loginFactory

```
java.util.function.Supplier<Login> es.ull.esit.app.ApplicationLauncher.loginFactory = Login::new [static], [private]
```

Factory (test seam) used to create Login instances.

Tests can override this supplier to inject a stub/mocked Login and avoid creating real Swing components (e.g., in headless CI).

Definition at line 31 of file [ApplicationLauncher.java](#).

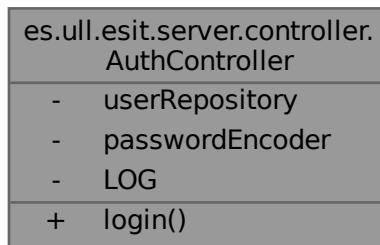
The documentation for this class was generated from the following file:

- [ApplicationLauncher.java](#)

## 7.11 es.ul.esit.server.controller.AuthController Class Reference

Rest controller for handling authentication-related requests.

Collaboration diagram for es.ul.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.*

#### 7.11.1 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](#).

## 7.11.2 Member Function Documentation

### 7.11.2.1 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

<i>request</i>	[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());
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 }
```

## 7.11.3 Member Data Documentation

### 7.11.3.1 LOG

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

Logger for logging authentication events.

Definition at line 26 of file [AuthController.java](#).

### 7.11.3.2 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](#).

### 7.11.3.3 userRepository

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

Repository for accessing user data.

Definition at line 30 of file [AuthController.java](#).

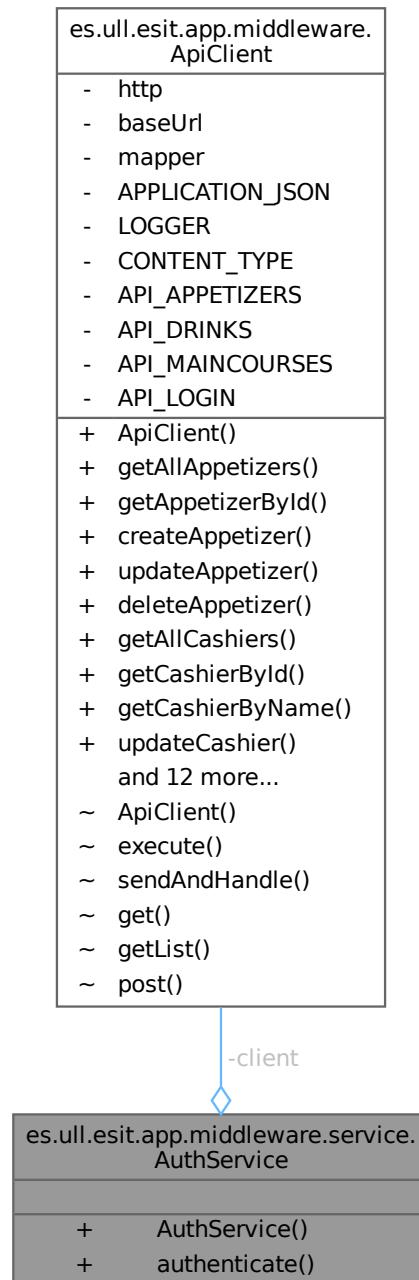
The documentation for this class was generated from the following file:

- [AuthController.java](#)

## 7.12 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)`  
*Attempts to authenticate a user against the backend.*

### Private Attributes

- final [ApiClient client](#)  
*REST API client used to communicate with the backend login endpoint.*

### 7.12.1 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](#).

### 7.12.2 Constructor & Destructor Documentation

#### 7.12.2.1 AuthService()

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

Constructs an AuthService with the given API client.

##### Parameters

<i>client</i>	[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](#).

### 7.12.3 Member Function Documentation

#### 7.12.3.1 authenticate()

```
User es.ull.esit.app.middleware.service.AuthService.authenticate (
    String username,
    String password )  [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**

<i>username</i>	[String] Username entered by the user.
<i>password</i>	[String] Password entered by the user.

**Returns**

[User] Authenticated user object returned by the backend.

Definition at line 37 of file [AuthService.java](#).

```
00037
00038     // 1. Local Validation.
00039     if (username == null || username.trim().isEmpty()) {
00040         throw new IllegalArgumentException("Username cannot be empty.");
00041     }
00042     if (password == null || password.isEmpty()) {
00043         throw new IllegalArgumentException("Password cannot be empty.");
00044     }
00045
00046     // 2. Call API.
00047     return client.login(username, password);
00048 }
```

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:



## 7.12.4 Member Data Documentation

### 7.12.4.1 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.AuthService](#).

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

- [AuthService.java](#)

## 7.13 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:

es.ull.esit.app.middleware.model.BillResult	
-	subTotal
-	vat
-	total
+	BillResult()
+	getSubTotal()
+	getVat()
+	getTotal()

### 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.*

### 7.13.1 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](#).

## 7.13.2 Constructor & Destructor Documentation

### 7.13.2.1 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

<i>subTotal</i>	[double] Subtotal amount before VAT.
<i>vat</i>	[double] VAT amount.
<i>total</i>	[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](#).

## 7.13.3 Member Function Documentation

### 7.13.3.1 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](#).

### 7.13.3.2 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](#).

### 7.13.3.3 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](#).

## 7.13.4 Member Data Documentation

### 7.13.4.1 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\(\)](#).

### 7.13.4.2 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\(\)](#).

### 7.13.4.3 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](#)

## 7.14 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:

es.ull.esit.app.middleware.model.Cashier	
-	id
-	name
-	salary
+	Cashier()
+	Cashier()
+	getId()
+	setId()
+	getName()
+	setName()
+	getSalary()
+	setSalary()

### 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").*

### 7.14.1 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](#).

### 7.14.2 Constructor & Destructor Documentation

#### 7.14.2.1 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 }
```

#### 7.14.2.2 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

<code>id</code>	[Long] Unique identifier of the cashier.
<code>name</code>	[String] Name of the cashier.
<code>salary</code>	[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](#).

### 7.14.3 Member Function Documentation

#### 7.14.3.1 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](#).

#### 7.14.3.2 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](#).

#### 7.14.3.3 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](#).

#### 7.14.3.4 setId()

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

Sets the cashier identifier.

**Parameters**

<i>id</i>	[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](#).

#### 7.14.3.5 setName()

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

Sets the cashier name.

**Parameters**

<i>name</i>	[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](#).

#### 7.14.3.6 setSalary()

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

Sets the cashier salary.

**Parameters**

<i>salary</i>	[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](#).

### 7.14.4 Member Data Documentation

#### 7.14.4.1 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\(\)](#).

#### 7.14.4.2 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\(\)](#).

#### 7.14.4.3 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](#)

## 7.15 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:

es.ull.esit.server.middleware.model.Cashier	
-	id
-	name
-	salary
+	Cashier()
+	Cashier()
+	getId()
+	setId()
+	getName()
+	setName()
+	getSalary()
+	setSalary()

## 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").*

### 7.15.1 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](#).

### 7.15.2 Constructor & Destructor Documentation

#### 7.15.2.1 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 }

#### 7.15.2.2 Cashier() [2/2]

```
es.ull.esit.server.middleware.model.Cashier.Cashier (   

    Long id,  

    String name,  

    Integer salary ) [inline]
```

Full constructor.

**Parameters**

<i>id</i>	[Long] Identifier of the cashier.
<i>name</i>	[String] Username of the cashier.
<i>salary</i>	[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 }
```

### 7.15.3 Member Function Documentation

#### 7.15.3.1 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 }
```

#### 7.15.3.2 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 }
```

#### 7.15.3.3 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 }
```

#### 7.15.3.4 setId()

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

Sets the cashier identifier.

Must be unique.

##### Parameters

<i>id</i>	[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 }
```

#### 7.15.3.5 setName()

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

Sets the cashier name (username).

##### Parameters

<i>name</i>	[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 }
```

#### 7.15.3.6 setSalary()

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

Sets the cashier salary.

##### Parameters

<i>salary</i>	[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 }
```

## 7.15.4 Member Data Documentation

### 7.15.4.1 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](#).

### 7.15.4.2 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](#).

### 7.15.4.3 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](#).

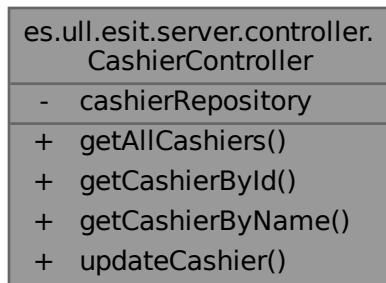
The documentation for this class was generated from the following file:

- [server/src/main/java/es/ull/esit/server/middleware/model/Cashier.java](#)

## 7.16 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](#)

### 7.16.1 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](#).

### 7.16.2 Member Function Documentation

#### 7.16.2.1 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 }
```

#### 7.16.2.2 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**

<i>id</i>	[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 }
```

**7.16.2.3 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**

<i>name</i>	[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 }
```

**7.16.2.4 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**

<i>id</i>	[Long] Identifier of the cashier to update.
<i>cashier</i>	[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     }
```

## 7.16.3 Member Data Documentation

### 7.16.3.1 `cashierRepository`

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

Definition at line 29 of file [CashierController.java](#).

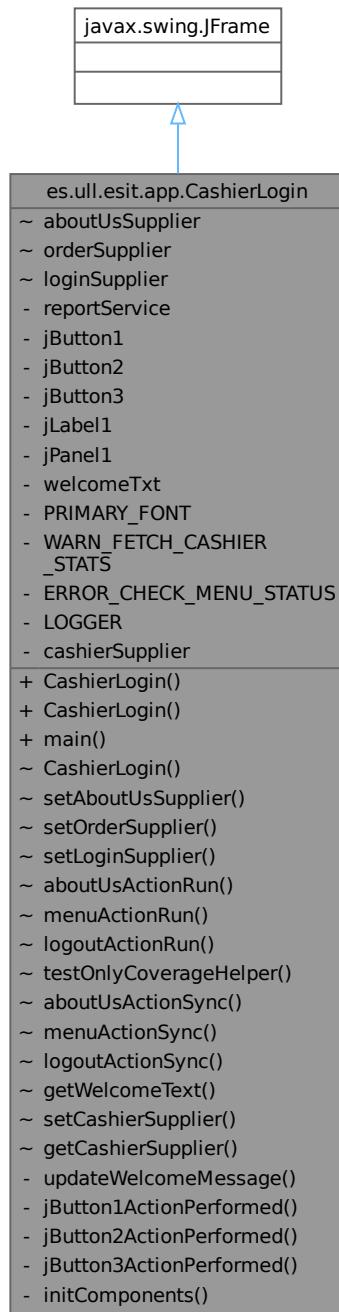
The documentation for this class was generated from the following file:

- [CashierController.java](#)

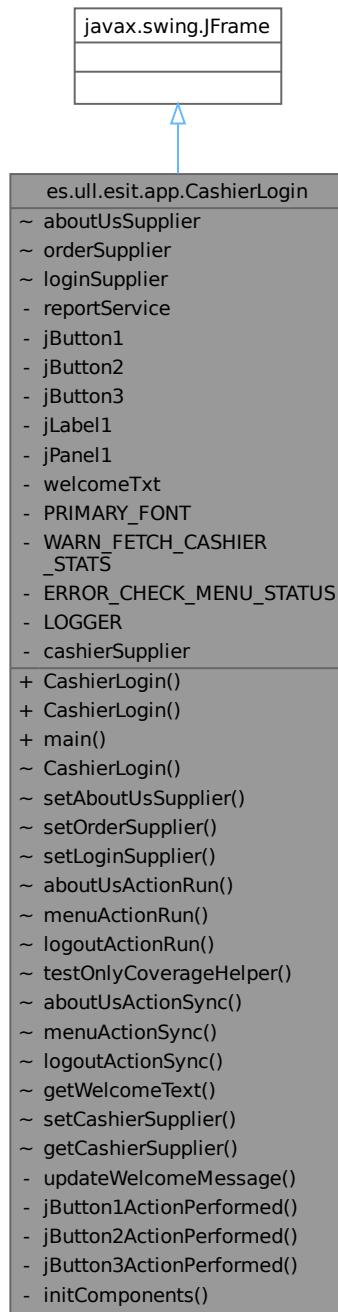
## 7.17 es.ull.esit.app.CashierLogin Class Reference

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 transient 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.*

## Static Private Attributes

- static final String `PRIMARY_FONT` = "Yu Gothic UI"  
*Primary font used in the UI components.*
- static final String `WARN_FETCH_CASHIER_STATS` = "Warning: Could not fetch cashier stats: {}"  
*Log message used when fetching cashier stats fails.*
- static final String `ERROR_CHECK_MENU_STATUS` = "Error checking menu status: {}"  
*Log message used when checking menu status fails.*
- static final Logger `LOGGER` = LoggerFactory.getLogger(CashierLogin.class)  
*Logger instance for logging events and errors.*
- static Supplier<? extends javax.swing.JFrame > `cashierSupplier` = CashierLogin::new  
*Factory (test seam) used by main() to create instances of the top-level CashierLogin window.*

### 7.17.1 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 26 of file [CashierLogin.java](#).

### 7.17.2 Constructor & Destructor Documentation

#### 7.17.2.1 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 50 of file [CashierLogin.java](#).

```
00050      {
00051      this((String) null);
00052 }
```

#### 7.17.2.2 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

<i>name</i>	[String] Name of the cashier returned by the backend (or null for a generic welcome message).
-------------	---

Definition at line 64 of file [CashierLogin.java](#).

```
00064 {
```

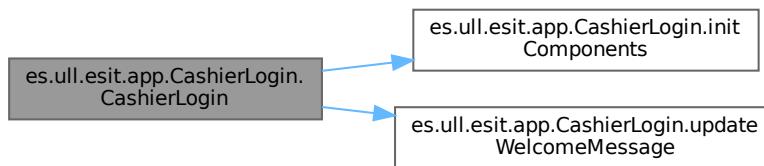
```

00065     initComponents();
00066
00067     ApiClient client = new ApiClient("http://localhost:8080");
00068     this.reportService = new ReportService(client);
00069
00070     updateWelcomeMessage(name);
00071 }

```

References [es.ull.esit.app.CashierLogin.initComponents\(\)](#), and [es.ull.esit.app.CashierLogin.updateWelcomeMessage\(\)](#).

Here is the call graph for this function:



### 7.17.3 Member Function Documentation

#### 7.17.3.1 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 486 of file [CashierLogin.java](#).

```

00486 {
00487
00488     jPanell = new javax.swing.JPanel();
00489     welcomeTxt = new javax.swing.JLabel();
00490     jButton1 = new javax.swing.JButton();
00491     jButton2 = new javax.swing.JButton();
00492     jLabel1 = new javax.swing.JLabel();
00493     jButton3 = new javax.swing.JButton();
00494
00495     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00496     setTitle("Cashier");
00497     setResizable(false);
00498
00499     jPanell.setBackground(new java.awt.Color(248, 244, 230));
00500
00501     welcomeTxt.setFont(new java.awt.Font(PRIMARY_FONT, 1, 24)); // NOI18N
00502     welcomeTxt.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00503     welcomeTxt.setText("Welcome Cashier");
00504
00505     jButton1.setBackground(new java.awt.Color(153, 153, 153));
00506     jButton1.setFont(new java.awt.Font(PRIMARY_FONT, 1, 18)); // NOI18N
00507     jButton1.setText("About us");
00508     jButton1.addActionListener(this::jButton1ActionPerformed);
00509
00510     jButton2.setBackground(new java.awt.Color(153, 153, 153));
00511     jButton2.setFont(new java.awt.Font(PRIMARY_FONT, 1, 18)); // NOI18N
00512     jButton2.setText("Menu");
00513     jButton2.addActionListener(this::jButton2ActionPerformed);
00514

```

```

00515     jLabel1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/ui/screenshot.png"))); // NOI18N
00516
00517     jButton3.setBackground(new java.awt.Color(255, 255, 255));
00518     jButton3.setFont(new java.awt.Font(PRIMARY_FONT, 1, 18)); // NOI18N
00519     jButton3.setText("LogOut");
00520     jButton3.addActionListener(this::jButton3ActionPerformed);
00521
00522     javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
00523     jPanel1.setLayout(jPanel1Layout);
00524     jPanel1Layout.setHorizontalGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00525         .addGroup(jPanel1Layout.createSequentialGroup()
00526             .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00527                 .addGroup(jPanel1Layout.createSequentialGroup()
00528                     .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00529                         .addComponent(welcomeTxt, javax.swing.GroupLayout.PREFERRED_SIZE, 299,
00530                             javax.swing.GroupLayout.PREFERRED_SIZE)
00531
00532                         .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00533                             .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 320,
00534                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00535                             .addGroup(jPanel1Layout.createSequentialGroup()
00536                                 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING,
00537                                     .addComponent(jLabel1)
00538                                     .addGroup(jPanel1Layout.createSequentialGroup()
00539                                         .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
00540                                             .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 141,
00541                                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00542                                             .addGroup(jPanel1Layout.createSequentialGroup()
00543                                                 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00544                                                     .addGroup(jPanel1Layout.createSequentialGroup()
00545                                         .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00546                                             .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 320,
00547                                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00548                                             .addGroup(jPanel1Layout.createSequentialGroup()
00549                                                 .addComponent(jPanel1Layout.setVerticalGroup(
00550                                                     .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00551                                         .addGroup(jPanel1Layout.createSequentialGroup()
00552                                             .addComponent(jPanel1Layout.createSequentialGroup()
00553                                                 .addComponent(welcomeTxt, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
00554                                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00555                                                 .addGroup(jPanel1Layout.createSequentialGroup()
00556                                         .addComponent(jLabel1)
00557                                         .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING,
00558                                             .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00559                                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00560                                             .addGroup(jPanel1Layout.createSequentialGroup()
00561                                                 .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00562                                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00563                                                 .addGroup(jPanel1Layout.createSequentialGroup()
00564                                                     .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00565                                                         .addGroup(jPanel1Layout.createSequentialGroup()
00566                                             .addComponent(jPanel1Layout.createSequentialGroup()
00567                                                 .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00568                                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00569                                                 .addGroup(jPanel1Layout.createSequentialGroup()
00570
00571                                     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00572                                     getContentPane().setLayout(layout);
00573                                     layout.setHorizontalGroup(
00574                                         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00575                                         .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
00576                                         javax.swing.GroupLayout.DEFAULT_SIZE,
00577                                         Short.MAX_VALUE));
00578                                         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00579                                         .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
00580                                         javax.swing.GroupLayout.DEFAULT_SIZE,
00581                                         Short.MAX_VALUE));
00582                                     pack();
00583                                     setLocationRelativeTo(null);
00584     } // </editor-fold> //GEN-END:initComponents

```

References [es.ul.esit.app.CashierLogin.JButton1](#), [es.ul.esit.app.CashierLogin.JButton2](#), [es.ul.esit.app.CashierLogin.JButton3](#), [es.ul.esit.app.CashierLogin.jLabel1](#), [es.ul.esit.app.CashierLogin.JPanel1](#), [es.ul.esit.app.CashierLogin.PRIMARY\\_FONT](#), and [es.ul.esit.app.CashierLogin.welcomeTxt](#).

Referenced by [es.ull.esit.app.CashierLogin.CashierLogin\(\)](#).

Here is the caller graph for this function:



### 7.17.3.2 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

<code>evt</code>	[ <code>java.awt.event.ActionEvent</code> ] Action event triggered by button click.
------------------	---

Definition at line 310 of file [CashierLogin.java](#).

```
00310
00311     new Thread(() -> {
00312         try {
00313             List<es.ull.esit.app.middleware.model.Cashier> cashiers = reportService.getCashierInfo();
00314             LOGGER.info("Found {} cashiers in DB.", cashiers.size());
00315
00316         } catch (Exception ex) {
00317             LOGGER.warn(WARN_FETCH_CASHIER_STATS, ex.getMessage());
00318         }
00319         SwingUtilities.invokeLater(() -> {
00320             aboutUsSupplier.get().setVisible(true);
00321             dispose();
00322         });
00323     }).start();
00324 }
```

References [es.ull.esit.app.CashierLogin.LOGGER](#), [es.ull.esit.app.CashierLogin.reportService](#), and [es.ull.esit.app.CashierLogin.WARN\\_FETCH\\_CASHIER\\_STATS](#)

### 7.17.3.3 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

<code>evt</code>	[ <code>java.awt.event.ActionEvent</code> ] Action event triggered by button click.
------------------	---

Definition at line 366 of file [CashierLogin.java](#).

```
00366
00367     new Thread(() -> {
00368         try {
00369             String status = reportService.checkMenuStatus();
00370             LOGGER.info("Menu status: {}", status);
00371
00372         } catch (Exception ex) {
00373             LOGGER.error(ERROR_CHECK_MENU_STATUS, ex.getMessage());
00374         }
00375
00376         SwingUtilities.invokeLater(() -> {
00377             orderSupplier.get().setVisible(true);
00378             dispose();
00379         });
00380     }).start();
00381 }
```

References [es.ull.esit.app.CashierLogin.ERROR\\_CHECK\\_MENU\\_STATUS](#), [es.ull.esit.app.CashierLogin.LOGGER](#), and [es.ull.esit.app.CashierLogin.reportService](#).

#### 7.17.3.4 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

<code>evt</code>	[ <code>java.awt.event.ActionEvent</code> ] Action event triggered by button click.
------------------	---

Definition at line 416 of file [CashierLogin.java](#).

```
00416
00417     loginSupplier.get().setVisible(true);
00418     dispose();
00419 }
```

### 7.17.3.5 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

<code>args</code>	[String[]] Command line arguments (not used).
-------------------	---

Definition at line 459 of file [CashierLogin.java](#).

```
00459
00460     try {
00461         for (javax.swing.UIManager.LookAndFeelInfo info :
00462             javax.swing.UIManager.getInstalledLookAndFeels()) {
00463             if ("Nimbus".equals(info.getName())) {
00464                 javax.swing.UIManager.setLookAndFeel(info.getClassName());
00465                 break;
00466             }
00467         } catch (ClassNotFoundException | InstantiationException | IllegalAccessException
00468             | javax.swing.UnsupportedLookAndFeel ex) {
00469             java.util.logging.Logger.getLogger(CashierLogin.class.getName()).log(java.util.logging.Level.SEVERE,
00470             null, ex);
00471         }
00472         java.awt.EventQueue.invokeLater(() -> getCashierSupplier().get().setVisible(true));
00473     }
```

### 7.17.3.6 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

<code>name</code>	[String] Cashier name or null.
-------------------	--------------------------------

Definition at line 286 of file [CashierLogin.java](#).

```
00286
00287     if (name == null || name.trim().isEmpty()) {
00288         welcomeTxt.setText("Welcome Cashier");
00289     } else {
00290         welcomeTxt.setText("Welcome " + name);
00291     }
00292 }
```

References [es.ull.esit.app.CashierLogin.welcomeTxt](#).

Referenced by [es.ull.esit.app.CashierLogin.CashierLogin\(\)](#).

Here is the caller graph for this function:



## 7.17.4 Member Data Documentation

### 7.17.4.1 cashierSupplier

```
Supplier<? extends javax.swing.JFrame> es.ull.esit.app.CashierLogin.cashierSupplier = CashierLogin::new [static], [private]
```

Factory (test seam) used by main() to create instances of the top-level CashierLogin window.

Tests can override this supplier to avoid creating real Swing components (e.g., in headless CI environments).

Definition at line 130 of file [CashierLogin.java](#).

### 7.17.4.2 ERROR\_CHECK\_MENU\_STATUS

```
final String es.ull.esit.app.CashierLogin.ERROR_CHECK_MENU_STATUS = "Error checking menu status : {}" [static], [private]
```

Log message used when checking menu status fails.

Definition at line 38 of file [CashierLogin.java](#).

Referenced by [es.ull.esit.app.CashierLogin.jButton2ActionPerformed\(\)](#).

### 7.17.4.3 jButton1

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

Button that opens the "About us" information window.

Definition at line 590 of file [CashierLogin.java](#).

Referenced by [es.ull.esit.app.CashierLogin.initComponents\(\)](#).

#### 7.17.4.4 **jButton2**

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

Button that opens the general menu window (Frame1).

Definition at line 592 of file [CashierLogin.java](#).

Referenced by [es.ull.esit.app.CashierLogin.initComponents\(\)](#).

#### 7.17.4.5 **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 594 of file [CashierLogin.java](#).

Referenced by [es.ull.esit.app.CashierLogin.initComponents\(\)](#).

#### 7.17.4.6 **jLabel1**

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

Label that displays the logo or application image.

Definition at line 596 of file [CashierLogin.java](#).

Referenced by [es.ull.esit.app.CashierLogin.initComponents\(\)](#).

#### 7.17.4.7 **jPanel1**

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

Main container panel for all UI elements.

Definition at line 598 of file [CashierLogin.java](#).

Referenced by [es.ull.esit.app.CashierLogin.initComponents\(\)](#).

#### 7.17.4.8 **LOGGER**

```
final Logger es.ull.esit.app.CashierLogin.LOGGER = LoggerFactory.getLogger(CashierLogin.class)
[static], [private]
```

Logger instance for logging events and errors.

Definition at line 41 of file [CashierLogin.java](#).

Referenced by [es.ull.esit.app.CashierLogin.jButton1ActionPerformed\(\)](#), and [es.ull.esit.app.CashierLogin.jButton2ActionPerformed\(\)](#).

#### 7.17.4.9 PRIMARY\_FONT

```
final String es.ull.esit.app.CashierLogin.PRIMARY_FONT = "Yu Gothic UI" [static], [private]
```

Primary font used in the UI components.

Definition at line 32 of file [CashierLogin.java](#).

Referenced by [es.ull.esit.app.CashierLogin.initComponents\(\)](#).

#### 7.17.4.10 reportService

```
final transient ReportService es.ull.esit.app.CashierLogin.reportService [private]
```

Service used to retrieve reports and basic status from the backend.

Definition at line 29 of file [CashierLogin.java](#).

Referenced by [es.ull.esit.app.CashierLogin.jButton1ActionPerformed\(\)](#), and [es.ull.esit.app.CashierLogin.jButton2ActionPerformed\(\)](#).

#### 7.17.4.11 WARN\_FETCH\_CASHIER\_STATS

```
final String es.ull.esit.app.CashierLogin.WARN_FETCH_CASHIER_STATS = "Warning: Could not  
fetch cashier stats: {}" [static], [private]
```

Log message used when fetching cashier stats fails.

Definition at line 35 of file [CashierLogin.java](#).

Referenced by [es.ull.esit.app.CashierLogin.jButton1ActionPerformed\(\)](#).

#### 7.17.4.12 welcomeTxt

```
javax.swing.JLabel es.ull.esit.app.CashierLogin.welcomeTxt [private]
```

Label that displays the welcome message for the cashier.

Definition at line 600 of file [CashierLogin.java](#).

Referenced by [es.ull.esit.app.CashierLogin.initComponents\(\)](#), and [es.ull.esit.app.CashierLogin.updateWelcomeMessage\(\)](#).

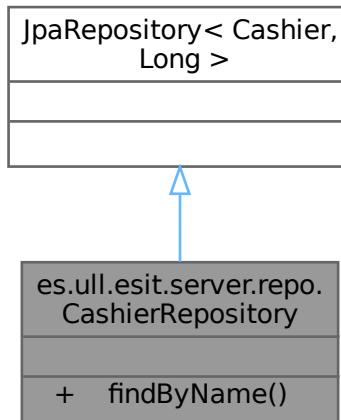
The documentation for this class was generated from the following file:

- [CashierLogin.java](#)

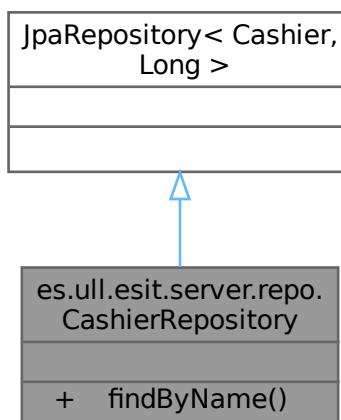
## 7.18 es.ull.esit.server.repo.CashierRepository Interface Reference

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.*

### 7.18.1 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](#).

### 7.18.2 Member Function Documentation

#### 7.18.2.1 `findByName()`

```
Optional< Cashier > es.ull.esit.server.repo.CashierRepository.findByName ( String name )
```

Finds a cashier by their name.

##### Parameters

<code>name</code>	[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](#)

## 7.19 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:

es.ull.esit.app.middleware.model.Drink	
-	drinksId
-	itemDrinks
-	drinksPrice
-	receiptId
+	Drink()
+	Drink()
+	getDrinksId()
+	setDrinksId()
+	getItemDrinks()
+	setItemDrinks()
+	getDrinksPrice()
+	setDrinksPrice()
+	getReceiptId()
+	setReceiptId()

## 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").*

### 7.19.1 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](#).

### 7.19.2 Constructor & Destructor Documentation

#### 7.19.2.1 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       }

#### 7.19.2.2 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

<code>drinksId</code>	[Long] Unique identifier of the drink.
<code>itemDrinks</code>	[String] Name of the drink item.
<code>drinksPrice</code>	[Integer] Price of the drink.
<code>receiptId</code>	[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     this.drinksId = drinksId;
00047     this.itemDrinks = itemDrinks;
00048     this.drinksPrice = drinksPrice;
00049     this.receiptId = receiptId;
00050 }
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](#).

### 7.19.3 Member Function Documentation

#### 7.19.3.1 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](#).

#### 7.19.3.2 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](#).

#### 7.19.3.3 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](#).

#### 7.19.3.4 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](#).

#### 7.19.3.5 setDrinksId()

```
void es.ull.esit.app.middleware.model.Drink.setDrinksId (
    Long drinksId ) [inline]
```

Sets the drink identifier.

##### Parameters

<i>drinksId</i>	[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](#).

#### 7.19.3.6 setDrinksPrice()

```
void es.ull.esit.app.middleware.model.Drink.setDrinksPrice (
    Integer drinksPrice ) [inline]
```

Sets the drink price.

##### Parameters

<i>drinksPrice</i>	[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](#).

### 7.19.3.7 setItemDrinks()

```
void es.ull.esit.app.middleware.model.Drink.setItemDrinks (
    String itemDrinks ) [inline]
```

Sets the drink item name.

#### Parameters

<i>itemDrinks</i>	[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](#).

### 7.19.3.8 setReceiptId()

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

Sets the identifier of the related receipt.

#### Parameters

<i>receiptId</i>	[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](#).

## 7.19.4 Member Data Documentation

### 7.19.4.1 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\(\)](#).

#### 7.19.4.2 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\(\)](#).

#### 7.19.4.3 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\(\)](#).

#### 7.19.4.4 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](#)

## 7.20 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:

es.ull.esit.server.middleware.model.Drink	
-	drinksId
-	itemDrinks
-	drinksPrice
+	Drink()
+	Drink()
+	getDrinksId()
+	setDrinksId()
+	getItemDrinks()
+	setItemDrinks()
+	getDrinksPrice()
+	setDrinksPrice()

### 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").*

## 7.20.1 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](#).

## 7.20.2 Constructor & Destructor Documentation

### 7.20.2.1 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 }
```

### 7.20.2.2 Drink() [2/2]

```
es.ull.esit.server.middleware.model.Drink.Drink (
    Long drinksId,
    String itemDrinks,
    Integer drinksPrice ) [inline]
```

Full constructor.

#### Parameters

<code>drinksId</code>	[Long] Identifier of the drink.
<code>itemDrinks</code>	[String] Name of the drink.
<code>drinksPrice</code>	[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 }
```

### 7.20.3 Member Function Documentation

#### 7.20.3.1 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     }
```

#### 7.20.3.2 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     }
```

#### 7.20.3.3 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     }
```

#### 7.20.3.4 setDrinksId()

```
void es.ull.esit.server.middleware.model.Drink.setDrinksId (
    Long drinksId) [inline]
```

Sets the drink identifier.

Generated by the database.

**Parameters**

<i>drinksId</i>	[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 }
```

**7.20.3.5 setDrinksPrice()**

```
void es.ull.esit.server.middleware.model.Drink.setDrinksPrice (
    Integer drinksPrice) [inline]
```

Sets the drink price.

**Parameters**

<i>drinksPrice</i>	[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 }
```

**7.20.3.6 setItemDrinks()**

```
void es.ull.esit.server.middleware.model.Drink.setItemDrinks (
    String itemDrinks) [inline]
```

Sets the drink name.

**Parameters**

<i>itemDrinks</i>	[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 }
```

**7.20.4 Member Data Documentation****7.20.4.1 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](#).

#### 7.20.4.2 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](#).

#### 7.20.4.3 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](#)

## 7.21 es.ull.esit.server.controller.DrinkController Class Reference

REST controller for managing drinks.

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

es.ull.esit.server.controller.DrinkController
- drinkRepository
+ getAllDrinks()
+ getDrinkById()
+ createDrink()
+ updateDrink()
+ deleteDrink()

### 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.*

### 7.21.1 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](#).

### 7.21.2 Member Function Documentation

#### 7.21.2.1 createDrink()

```
ResponseEntity< Drink > es.ull.esit.server.controller.DrinkController.createDrink ( @RequestBody Drink drink ) [inline]
```

Creates a new drink.

HTTP POST /api/drinks

##### Parameters

<code>drink</code>	[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 }
```

#### 7.21.2.2 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**

<i>id</i>	[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 }
```

**7.21.2.3 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 }
```

**7.21.2.4 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**

<i>id</i>	[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 }
```

### 7.21.2.5 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**

<i>id</i>	[Long] Identifier of the drink to update.
<i>drink</i>	[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 }
```

## 7.21.3 Member Data Documentation

### 7.21.3.1 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.**

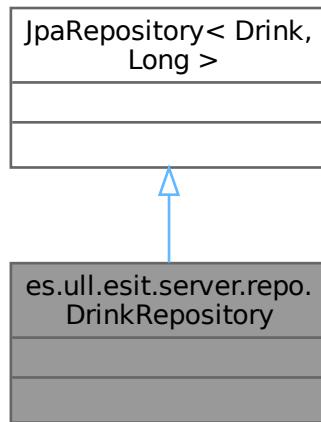
The documentation for this class was generated from the following file:

- [DrinkController.java](#)

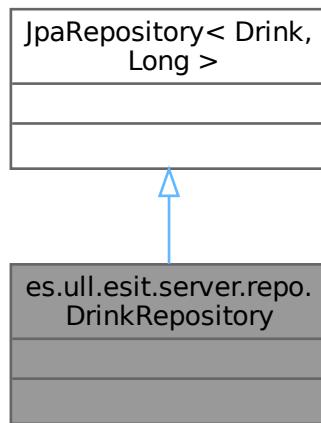
## 7.22 es.ull.esit.server.repo.DrinkRepository Interface Reference

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:



### 7.22.1 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](#)

## 7.23 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:

es.ull.esit.server.controller. HealthController
- dataSource
+ health()
+ checkDatabase()

### 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.*

### 7.23.1 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](#).

## 7.23.2 Member Function Documentation

### 7.23.2.1 checkDatabase()

```
ResponseEntity< Map< String, Object > > es.ullesit.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 }
```

### 7.23.2.2 health()

```
ResponseEntity< Map< String, Object > > es.ullesit.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 }
```

## 7.23.3 Member Data Documentation

### 7.23.3.1 dataSource

DataSource es.ul.esit.server.controller.HealthController.dataSource [private]

DataSource used to verify connectivity with the database.

Definition at line 31 of file [HealthController.java](#).

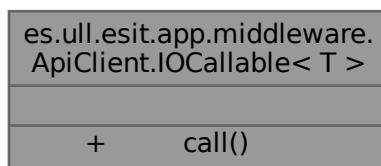
The documentation for this class was generated from the following file:

- [HealthController.java](#)

## 7.24 es.ul.esit.app.middleware.ApiClient.IOCallable< T > Interface Template Reference

Functional interface for lambdas that may throw InterruptedException or IOException.

Collaboration diagram for es.ul.esit.app.middleware.ApiClient.IOCallable< T >:



### Public Member Functions

- T `call()` throws InterruptedException, IOException

### 7.24.1 Detailed Description

Functional interface for lambdas that may throw InterruptedException or IOException.

Definition at line 96 of file [ApiClient.java](#).

### 7.24.2 Member Function Documentation

#### 7.24.2.1 call()

```
T es.ullesit.app.middleware.ApiClient.IOCallable< T >.call ( ) throws InterruptedException,  
IOException
```

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

- [ApiClient.java](#)

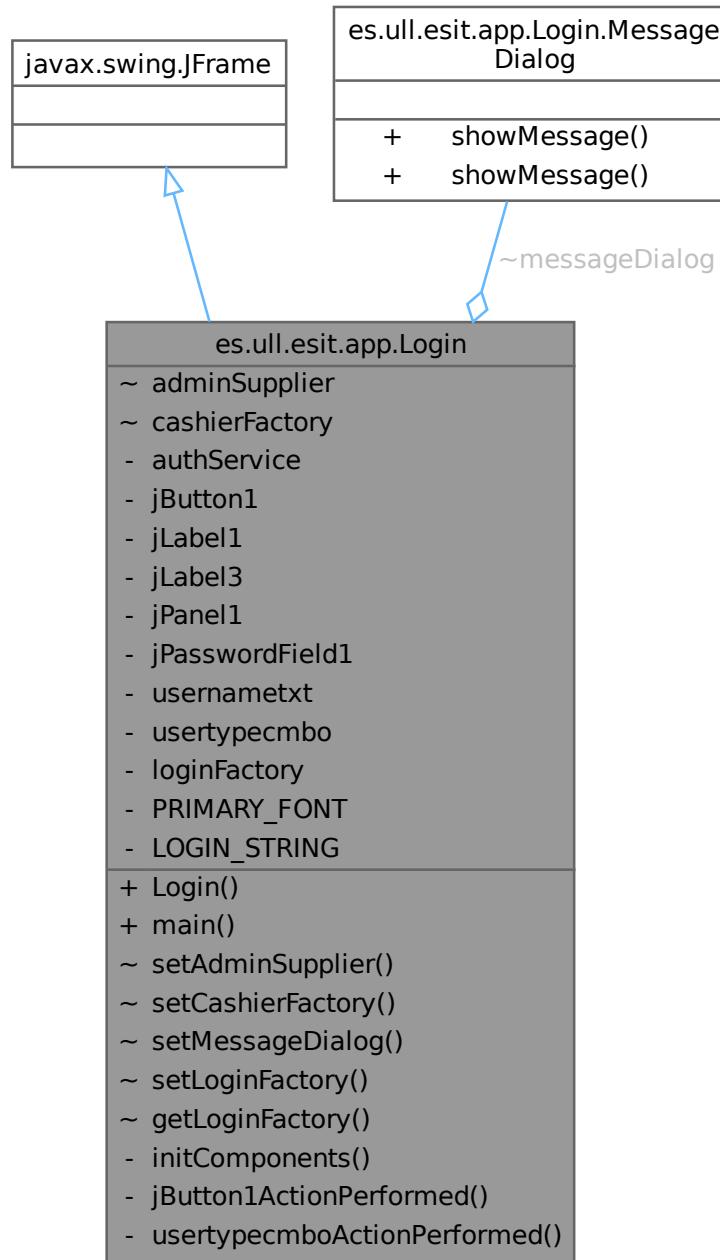
## 7.25 es.ullesit.app.Login Class Reference

Login window for the Restaurant System.

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



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



## Classes

- interface **MessageDialog**

*Abstraction for message dialogs (test seam).*

## 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 transient 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 `usernameetxt`  
*Input field for the username.*
- javax.swing.JComboBox< String > `usertypecmbo`  
*Combo box for user type (admin/cashier).*

## Static Private Attributes

- static java.util.function.Supplier< `Login` > `loginFactory` = `Login::new`  
*Factory (test seam) to create Login instances for main().*
- static final String `PRIMARY_FONT` = "Yu Gothic UI"  
*Primary font used in the GUI.*
- static final String `LOGIN_STRING` = "Log in"  
*String constant for "Log in" button text.*

### 7.25.1 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](#).

### 7.25.2 Constructor & Destructor Documentation

#### 7.25.2.1 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 161 of file [Login.java](#).

```
00161      {
00162      initComponents();
00163      ApiClient client = new ApiClient("http://localhost:8080");
00164      this.authService = new AuthService(client);
00165  }
```

References [es.ull.esit.app.Login.initComponents\(\)](#).

Here is the call graph for this function:



## 7.25.3 Member Function Documentation

### 7.25.3.1 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 button for "Log In", along with the layout and configuration of the window.

Definition at line 178 of file [Login.java](#).

```

00239         .addComponent(jPasswordField1, javax.swing.GroupLayout.PREFERRED_SIZE,
00240             311,
00241                 javax.swing.GroupLayout.PREFERRED_SIZE)
00242         .addComponent(usernametxt, javax.swing.GroupLayout.PREFERRED_SIZE, 311,
00243             javax.swing.GroupLayout.PREFERRED_SIZE)
00244         .addComponent(usertypecmbo, javax.swing.GroupLayout.PREFERRED_SIZE, 311,
00245             javax.swing.GroupLayout.PREFERRED_SIZE)))
00246     .addGroup(jPanellLayout.createSequentialGroup()
00247         .addGap(146, 146, 146)
00248         .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 141,
00249             javax.swing.GroupLayout.PREFERRED_SIZE)))
00250     .addContainerGap(86, Short.MAX_VALUE)));
00251 jPanellLayout.setVerticalGroup(
00252     jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00253         .addGroup(jPanellLayout.createSequentialGroup()
00254             .addGap(39, 39, 39)
00255             .addComponent(jLabel1)
00256             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
00257             .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
00258                 javax.swing.GroupLayout.PREFERRED_SIZE)
00259             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
00260             .addComponent(usernametxt, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00261                 javax.swing.GroupLayout.PREFERRED_SIZE)
00262             .addGap(18, 18, 18)
00263             .addComponent(jPasswordField1, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00264                 javax.swing.GroupLayout.PREFERRED_SIZE)
00265             .addGap(18, 18, 18)
00266             .addComponent(usertypecmbo, javax.swing.GroupLayout.PREFERRED_SIZE, 46,
00267                 javax.swing.GroupLayout.PREFERRED_SIZE)
00268             .addGap(18, 18, 18)
00269             .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00270                 javax.swing.GroupLayout.PREFERRED_SIZE)
00271             .addContainerGap(39, Short.MAX_VALUE)));
00272 javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00273 getContentPane().setLayout(layout);
00274 layout.setHorizontalGroup(
00275     layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00276         .addComponent(jPanell, javax.swing.GroupLayout.PREFERRED_SIZE,
00277             javax.swing.GroupLayout.DEFAULT_SIZE,
00278                 javax.swing.GroupLayout.PREFERRED_SIZE));
00279     layout.setVerticalGroup(
00280         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00281             .addComponent(jPanell, javax.swing.GroupLayout.DEFAULT_SIZE,
00282                 javax.swing.GroupLayout.DEFAULT_SIZE,
00283                 Short.MAX_VALUE));
00284     pack();
00285     setLocationRelativeTo(null);
00286 } // </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.LOGIN\\_STRING](#), [es.ull.esit.app.Login.PRIMARY\\_FONT](#), [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:



### 7.25.3.2 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

<code>evt</code>	[ <code>java.awt.event.ActionEvent</code> ] Action event triggered by button click.
------------------	---

Definition at line 300 of file `Login.java`.

```

00300
00301     GEN-FIRST:event_jButton1ActionPerformed // //
00302     // Get username and password from input fields.
00303     String usernameInput = usernametxt.getText();
00304     String passwordInput = new String(jPasswordField1.getPassword());
00305
00306     jButton1.setEnabled(false);
00307     jButton1.setText("Loading...");
00308
00309     new Thread(() -> {
00310         try {
00311             // Authenticate user through AuthService.
00312             User loggedInUser = authService.authenticate(usernameInput, passwordInput);
00313
00314             SwingUtilities.invokeLater(() -> {
00315                 // Open the corresponding window based on user role. Use the
00316                 // injectable factories so tests can substitute frames.
00317                 if ("ADMIN".equalsIgnoreCase(loggedInUser.getRole())) {
00318                     adminSupplier.get().setVisible(true);
00319                 } else if ("CASHIER".equalsIgnoreCase(loggedInUser.getRole())) {
00320                     cashierFactory.apply(loggedInUser.getUsername()).setVisible(true);
00321                 } else {
00322                     messageDialog.showMessage(this, "Unknown Role: " + loggedInUser.getRole());
00323                     jButton1.setEnabled(true);
00324                     jButton1.setText(LOGIN_STRING);
00325                     return;
00326                 }
00327
00328                 this.dispose();
00329             });
00330
00331         } catch (Exception e) {
00332             SwingUtilities.invokeLater(() -> {
00333                 messageDialog.showMessage(this, "Login failed: " + e.getMessage(), "Login Error",
00334                 JOptionPane.ERROR_MESSAGE);
00335
00336                 jButton1.setEnabled(true);
00337                 jButton1.setText(LOGIN_STRING);
00338             });
00339         }).start();
00340     } // GEN-LAST:event_jButton1ActionPerformed

```

References `es.ul.esit.app.Login.authService`, `es.ul.esit.app.Login.JButton1`, `es.ul.esit.app.Login.jPasswordField1`, `es.ul.esit.app.Login.LOGIN_STRING`, and `es.ul.esit.app.Login.usernametxt`.

7.25.3.3 `main()`

```

static void es.ul.esit.app.Login.main (
    String[] args ) [inline], [static]

```

Main entry point to start the login window application.

## Parameters

<code>args</code>	[ <code>String[]</code> ] Command line arguments (not used).
-------------------	--

Definition at line 359 of file [Login.java](#).

```

00359         {
00360     try {
00361         for (javax.swing.UIManager.LookAndFeelInfo info :
00362             javax.swing.UIManager.getInstalledLookAndFeels()) {
00363             if ("Nimbus".equals(info.getName())) {
00364                 javax.swing.UIManager.setLookAndFeel(info.getClassName());
00365                 break;
00366             }
00367         } catch (ClassNotFoundException | InstantiationException | IllegalAccessException
00368             | javax.swing.UnsupportedLookAndFeelException ex) {
00369             java.util.logging.Logger.getLogger(Login.class.getName()).log(java.util.logging.Level.SEVERE,
00370             null, ex);
00371         }
00372         java.awt.EventQueue.invokeLater(() -> getLoginFactory().get().setVisible(true));
00373     }

```

#### 7.25.3.4 usertypecmbActionPerformed()

```

void es.ull.esit.app.Login.usertypecmbActionPerformed (
    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

<code>evt</code>	[ <code>java.awt.event.ActionEvent</code> ] Action event triggered by combo box selection.
------------------	--

Definition at line 350 of file [Login.java](#).

```

00350 //////////////////////////////////////////////////////////////////
00351 // GEN-FIRST:event_usertypecmbActionPerformed
00352 // No action needed here.
00353 //////////////////////////////////////////////////////////////////
00354 // GEN-LAST:event_usertypecmbActionPerformed

```

### 7.25.4 Member Data Documentation

#### 7.25.4.1 authService

```
final transient AuthService es.ull.esit.app.Login.authService [private]
```

Service to handle authentication logic.

Definition at line 26 of file [Login.java](#).

Referenced by [es.ull.esit.app.Login.jButton1ActionPerformed\(\)](#).

#### 7.25.4.2 jButton1

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

Button to perform login ("Log In").

Definition at line 379 of file [Login.java](#).

Referenced by [es.ull.esit.app.Login.initComponents\(\)](#), and [es.ull.esit.app.Login.jButton1ActionPerformed\(\)](#).

#### 7.25.4.3 jLabel1

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

Label for the image/logo.

Definition at line 381 of file [Login.java](#).

Referenced by [es.ull.esit.app.Login.initComponents\(\)](#).

#### 7.25.4.4 jLabel3

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

Label for the title ("User Login").

Definition at line 383 of file [Login.java](#).

Referenced by [es.ull.esit.app.Login.initComponents\(\)](#).

#### 7.25.4.5 jPanel1

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

Main panel container.

Definition at line 385 of file [Login.java](#).

Referenced by [es.ull.esit.app.Login.initComponents\(\)](#).

#### 7.25.4.6 jPasswordField1

```
javax.swing.JPasswordField es.ull.esit.app.Login.jPasswordField1 [private]
```

Input field for the password.

Definition at line 387 of file [Login.java](#).

Referenced by [es.ull.esit.app.Login.initComponents\(\)](#), and [es.ull.esit.app.Login.jButton1ActionPerformed\(\)](#).

#### 7.25.4.7 LOGIN\_STRING

```
final String es.ull.esit.app.Login.LOGIN_STRING = "Log in" [static], [private]
```

String constant for "Log in" button text.

Definition at line 150 of file [Login.java](#).

Referenced by [es.ull.esit.app.Login.initComponents\(\)](#), and [es.ull.esit.app.Login.jButton1ActionPerformed\(\)](#).

#### 7.25.4.8 loginFactory

```
java.util.function.Supplier<Login> es.ull.esit.app.Login.loginFactory = Login::new [static],  
[private]
```

Factory (test seam) to create Login instances for main().

Tests can override it to prevent creating real windows.

Definition at line 50 of file [Login.java](#).

#### 7.25.4.9 PRIMARY\_FONT

```
final String es.ull.esit.app.Login.PRIMARY_FONT = "Yu Gothic UI" [static], [private]
```

Primary font used in the GUI.

Definition at line 147 of file [Login.java](#).

Referenced by [es.ull.esit.app.Login.initComponents\(\)](#).

#### 7.25.4.10 usernametxt

```
javax.swing.JTextField es.ull.esit.app.Login.usernametxt [private]
```

Input field for the username.

Definition at line 389 of file [Login.java](#).

Referenced by [es.ull.esit.app.Login.initComponents\(\)](#), and [es.ull.esit.app.Login.jButton1ActionPerformed\(\)](#).

#### 7.25.4.11 usertypecmb

```
javax.swing.JComboBox<String> es.ull.esit.app.Login.usertypecmb [private]
```

Combo box for user type (admin/cashier).

Currently not trusted for security decisions.

Definition at line 394 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](#)

## 7.26 es.ul.esit.server.controller.AuthController.LoginRequest Class Reference

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

Collaboration diagram for es.ul.esit.server.controller.AuthController.LoginRequest:



### Public Attributes

- String [username](#)
- String [password](#)

#### 7.26.1 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](#).

#### 7.26.2 Member Data Documentation

##### 7.26.2.1 password

`String es.ul.esit.server.controller.AuthController.LoginRequest.password`

Definition at line [44](#) of file [AuthController.java](#).

##### 7.26.2.2 username

`String es.ul.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](#)

## 7.27 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:

es.ull.esit.app.middleware.model. MainCourse	
-	foodId
-	itemFood
-	foodPrice
-	receiptId
+	MainCourse()
+	MainCourse()
+	getFoodId()
+	setFoodId()
+	getItemFood()
+	setItemFood()
+	getFoodPrice()
+	setFoodPrice()
+	getReceiptId()
+	setReceiptId()

### 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").*

### 7.27.1 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](#).

### 7.27.2 Constructor & Destructor Documentation

#### 7.27.2.1 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       }

#### 7.27.2.2 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

<code>foodId</code>	[Long] Unique identifier of the dish.
<code>itemFood</code>	[String] Name of the dish.
<code>foodPrice</code>	[Integer] Price of the dish.
<code>receiptId</code>	[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     this.foodId = foodId;
00047     this.itemFood = itemFood;
00048     this.foodPrice = foodPrice;
00049     this.receiptId = receiptId;
00050 }
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](#).

### 7.27.3 Member Function Documentation

#### 7.27.3.1 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](#).

#### 7.27.3.2 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](#).

#### 7.27.3.3 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](#).

### 7.27.3.4 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](#).

### 7.27.3.5 setFoodId()

```
void es.ull.esit.app.middleware.model.MainCourse.setFoodId (
    Long foodId ) [inline]
```

Sets the dish identifier.

#### Parameters

<i>foodId</i>	[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](#).

### 7.27.3.6 setFoodPrice()

```
void es.ull.esit.app.middleware.model.MainCourse.setFoodPrice (
    Integer foodPrice ) [inline]
```

Sets the price of the dish.

#### Parameters

<i>foodPrice</i>	[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](#).

### 7.27.3.7 setItemFood()

```
void es.ull.esit.app.middleware.model.MainCourse.setItemFood (
    String itemFood) [inline]
```

Sets the name of the dish.

#### Parameters

<i>itemFood</i>	[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](#).

### 7.27.3.8 setReceiptId()

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

Sets the identifier of the related receipt.

#### Parameters

<i>receiptId</i>	[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](#).

## 7.27.4 Member Data Documentation

### 7.27.4.1 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.setFoodId\(\)](#), and [es.ull.esit.app.middleware.model.MainCourse.getFoodId\(\)](#).

#### 7.27.4.2 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.MainCourse](#) and [es.ull.esit.app.middleware.model.MainCourse.setFoodPrice\(\)](#).

#### 7.27.4.3 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.MainCourse](#) and [es.ull.esit.app.middleware.model.MainCourse.setItemFood\(\)](#).

#### 7.27.4.4 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.MainCourse](#) 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](#)

## 7.28 es.ul.esit.server.middleware.model.MainCourse Class Reference

JPA entity that represents a main course in the menu.

Collaboration diagram for es.ul.esit.server.middleware.model.MainCourse:

es.ul.esit.server.middleware.model. MainCourse	
-	foodId
-	itemFood
-	foodPrice
+	MainCourse()
+	MainCourse()
+	getFoodId()
+	setFoodId()
+	getItemFood()
+	setItemFood()
+	getFoodPrice()
+	setFoodPrice()

### 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").*

## 7.28.1 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](#).

## 7.28.2 Constructor & Destructor Documentation

### 7.28.2.1 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 }

### 7.28.2.2 MainCourse() [2/2]

```
es.ull.esit.server.middleware.model.MainCourse.MainCourse (   

    Long foodId,  

    String itemFood,  

    Integer foodPrice ) [inline]
```

Full constructor.

#### Parameters

<code>foodId</code>	[Long] Identifier of the main course.
<code>itemFood</code>	[String] Name of the main course.
<code>foodPrice</code>	[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 }
```

### 7.28.3 Member Function Documentation

#### 7.28.3.1 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     }
```

#### 7.28.3.2 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     }
```

#### 7.28.3.3 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     }
```

#### 7.28.3.4 setFoodId()

```
void es.ull.esit.server.middleware.model.MainCourse.setFoodId (
    Long foodId ) [inline]
```

Sets the main course ID.

Generated by the database.

**Parameters**

<i>foodId</i>	[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 }
```

### 7.28.3.5 setFoodPrice()

```
void es.ull.esit.server.middleware.model.MainCourse.setFoodPrice (
    Integer foodPrice ) [inline]
```

Sets the main course price.

**Parameters**

<i>foodPrice</i>	[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 }
```

### 7.28.3.6 setItemFood()

```
void es.ull.esit.server.middleware.model.MainCourse.setItemFood (
    String itemFood ) [inline]
```

Sets the main course name.

**Parameters**

<i>itemFood</i>	[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 }
```

## 7.28.4 Member Data Documentation

### 7.28.4.1 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](#).

#### 7.28.4.2 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](#).

#### 7.28.4.3 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](#)

## 7.29 es.ull.esit.server.controller.MainCourseController Class Reference

REST controller for managing main courses.

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

es.ull.esit.server.controller. MainCourseController
- mainCourseRepository
+ getAllMainCourses()
+ getMainCourseById()
+ createMainCourse()
+ updateMainCourse()
+ deleteMainCourse()

### 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.*

### 7.29.1 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](#).

### 7.29.2 Member Function Documentation

#### 7.29.2.1 createMainCourse()

```
ResponseEntity< MainCourse > es.ull.esit.server.controller.MainCourseController.createMainCourse ( @RequestBody MainCourse mainCourse ) [inline]
```

Creates a new main course entry.

Endpoint: POST /api/maincourses

#### Parameters

<i>mainCourse</i>	[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 }
```

{

#### 7.29.2.2 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**

<i>id</i>	[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 }
```

**7.29.2.3 getAllMainCourses()**

ResponseEntity< List< MainCourse > > es.ull.esit.server.controller.MainCourseController.getAllMainCourses () [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 }
```

**7.29.2.4 getMainCourseById()**

ResponseEntity< MainCourse > es.ull.esit.server.controller.MainCourseController.getMainCourseById (
 @PathVariable Long id ) [inline]

Returns a specific main course by its ID.

Endpoint: GET /api/maincourses/{id}

**Parameters**

<i>id</i>	[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 }
```

### 7.29.2.5 updateMainCourse()

```
ResponseEntity< MainCourse > es.ull.esit.server.controller.MainCourseController.updateMainCourse (
    @PathVariable Long id,
    @RequestBody MainCourse mainCourse ) [inline]
```

Updates an existing main course entry.

Endpoint: PUT /api/maincourses/{id}

**Parameters**

<code>id</code>	[Long] Identifier of the main course to update.
<code>mainCourse</code>	[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     }
```

## 7.29.3 Member Data Documentation

### 7.29.3.1 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](#).

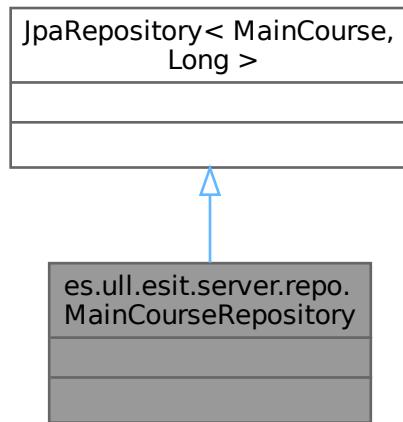
The documentation for this class was generated from the following file:

- [MainCourseController.java](#)

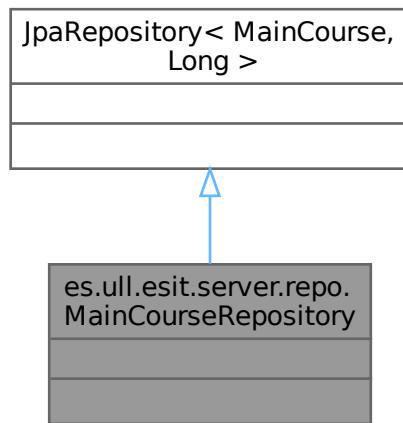
## 7.30 es.ull.esit.server.repo.MainCourseRepository Interface Reference

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:



### 7.30.1 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](#).

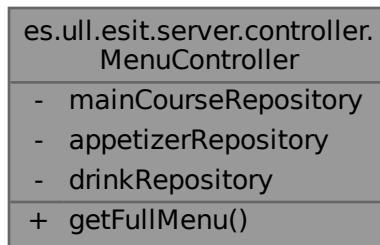
The documentation for this interface was generated from the following file:

- [MainCourseRepository.java](#)

## 7.31 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.*

### 7.31.1 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](#).

### 7.31.2 Member Function Documentation

#### 7.31.2.1 getFullMenu()

```
ResponseEntity< Map< String, Object > > es.ull.esit.server.controller.MenuController.getFullMenu ( ) [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 }
```

### 7.31.3 Member Data Documentation

#### 7.31.3.1 appetizerRepository

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

Repository for appetizer entities.

Definition at line 37 of file [MenuController.java](#).

### 7.31.3.2 drinkRepository

```
DrinkRepository es.ull.esit.server.controller.MenuController.drinkRepository [private]
```

Repository for drink entities.

Definition at line 41 of file [MenuController.java](#).

### 7.31.3.3 mainCourseRepository

```
MainCourseRepository es.ull.esit.server.controller.MenuController.mainCourseRepository [private]
```

Repository for main course entities.

Definition at line 33 of file [MenuController.java](#).

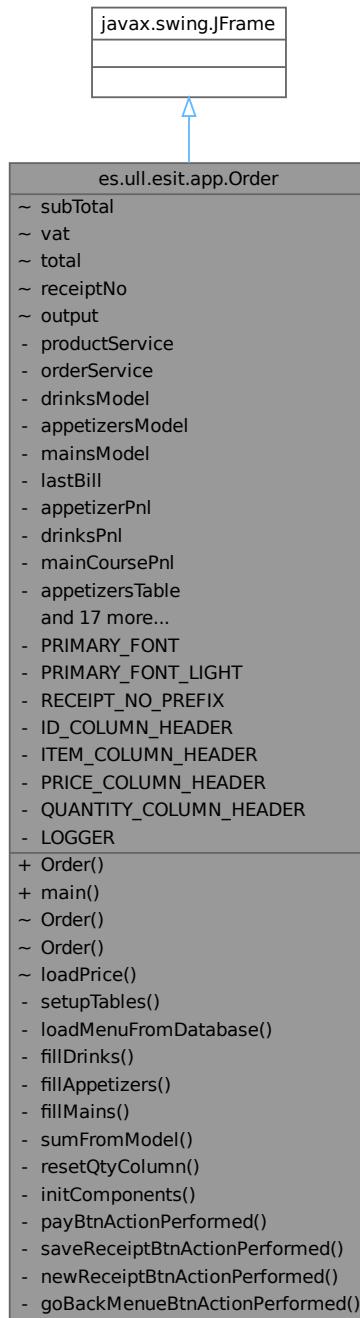
The documentation for this class was generated from the following file:

- [MenuController.java](#)

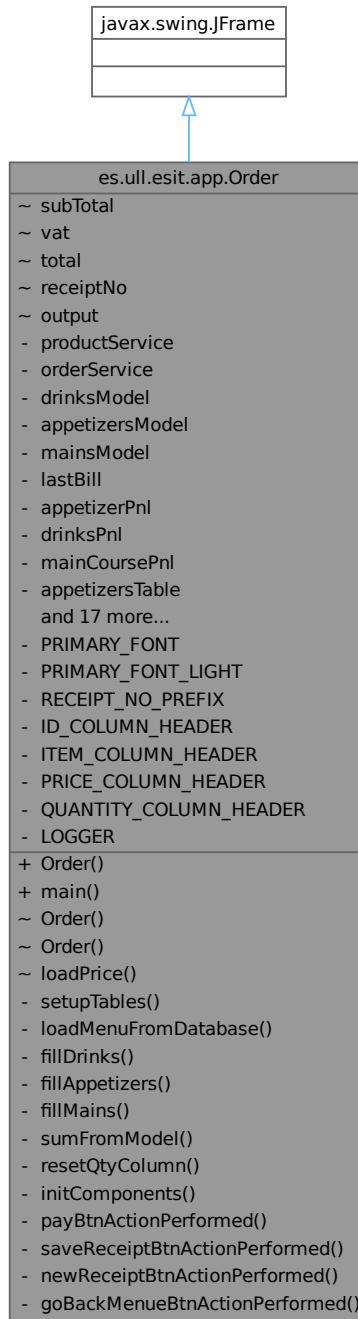
## 7.32 es.ull.esit.app.Order Class Reference

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 transient ProductService [productService](#)

*Service used to load products from the backend.*
- final transient OrderService [orderService](#) = new OrderService()
 

*Service used to calculate bill totals and generate receipt files.*
- DefaultTableModel [drinksModel](#)

*Table model for the category drinks of the menu.*
- DefaultTableModel [appetizersModel](#)

*Table model for the category appetizers of the menu.*
- DefaultTableModel [mainsModel](#)

*Table model for the category main courses of the menu.*
- transient 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.*

### Static Private Attributes

- static final String **PRIMARY\_FONT** = "Yu Gothic UI"
 

*Primary font used in the GUI.*
- static final String **PRIMARY\_FONT\_LIGHT** = "Yu Gothic UI Light"
 

*Lighter variant of the primary font.*
- static final String **RECEIPT\_NO\_PREFIX** = "Receipt No. : "
 

*Prefix used for the receipt number label to avoid duplicated literals.*
- static final String **ID\_COLUMN\_HEADER** = "ID"
 

*Column header used for ID across tables.*
- static final String **ITEM\_COLUMN\_HEADER** = "Item"
 

*Column header used for item name across tables.*
- static final String **PRICE\_COLUMN\_HEADER** = "Price (SR)"
 

*Column header used for price across tables.*
- static final String **QUANTITY\_COLUMN\_HEADER** = "Qty"
 

*Column header used for quantity across tables.*
- static final Logger **LOGGER** = LoggerFactory.getLogger(Order.class)
 

*Logger for this class.*

### 7.32.1 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 31 of file [Order.java](#).

### 7.32.2 Constructor & Destructor Documentation

#### 7.32.2.1 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.

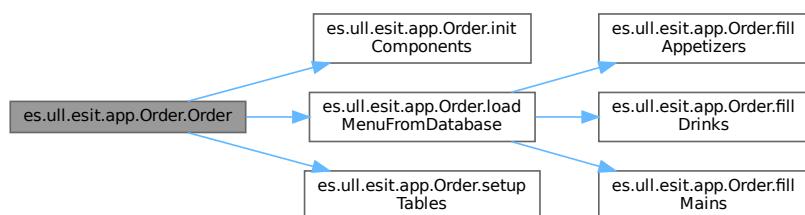
Definition at line 95 of file [Order.java](#).

```
00095         {
00096     initComponents();
00097
00098     // Initialize the Service Layer.
00099     ApiClient client = new ApiClient("http://localhost:8080");
00100     this.productService = new ProductService(client);
00101
00102     // Initialize receipt number label.
00103     receiptNoLbl.setText(RECEIPT_NO_PREFIX + receiptNo);
00104
00105     // Configure tables and models.
00106     setupTables();
00107
00108     // Load menu items from database via REST API.
00109     loadMenuFromDatabase();
00110 }
```

References [es.ull.esit.app.Order.initComponents\(\)](#), [es.ull.esit.app.Order.loadMenuFromDatabase\(\)](#), [es.ull.esit.app.Order.RECEIPT\\_NO\\_PREFIX](#), [es.ull.esit.app.Order.receiptNoLbl](#), and [es.ull.esit.app.Order.setupTables\(\)](#).

Referenced by [es.ull.esit.app.Order.main\(\)](#).

Here is the call graph for this function:



Here is the caller graph for this function:



### 7.32.3 Member Function Documentation

#### 7.32.3.1 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

<i>appetizers</i>	[java.util.List<Appetizer>] List of appetizers retrieved from the backend.
-------------------	--

Definition at line 276 of file [Order.java](#).

```
00276
00277     appetizersModel.setRowCount(0);
00278     for (Appetizer a : appetizers) {
00279         appetizersModel.addRow(new Object[] {
00280             a.getAppetizersId(),
00281             a.getItemAppetizers(),
00282             a.getAppetizersPrice(),
00283             0
00284         });
00285     }
00286 }
```

References [es.ull.esit.app.Order.appetizersModel](#).

Referenced by [es.ull.esit.app.Order.loadMenuFromDatabase\(\)](#).

Here is the caller graph for this function:



#### 7.32.3.2 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**

<code>drinks</code>	<code>[java.util.List&lt;Drink&gt;]</code> List of drinks retrieved from the backend.
---------------------	---

Definition at line 258 of file [Order.java](#).

```

00258     drinksModel.setRowCount(0);
00259     for (Drink d : drinks) {
00260         drinksModel.addRow(new Object[] {
00261             d.getDrinksId(),
00262             d.getItemDrinks(),
00263             d.getDrinksPrice(),
00264             0 // initial quantity
00265         });
00266     }
00267 }
00268 }
```

References [es.ul.esit.app.Order.drinksModel](#).

Referenced by [es.ul.esit.app.Order.loadMenuFromDatabase\(\)](#).

Here is the caller graph for this function:

**7.32.3.3 fillMains()**

```

void es.ul.esit.app.Order.fillMains (
    java.util.List< MainCourse > mains ) [inline], [private]
```

Fills the main courses table model with data from the backend.

**Parameters**

<code>mains</code>	<code>[java.util.List&lt;MainCourse&gt;]</code> List of main courses retrieved from the backend.
--------------------	--

Definition at line 294 of file [Order.java](#).

```

00294
00295     mainsModel.setRowCount(0);
00296     for (MainCourse m : mains) {
00297         mainsModel.addRow(new Object[] {
00298             m.getFoodId(),
00299             m.getItemFood(),
00300             m.getFoodPrice(),
00301             0
00302         });
00303     }
00304 }
```

References [es.ul.esit.app.Order.mainsModel](#).

Referenced by [es.ul.esit.app.Order.loadMenuFromDatabase\(\)](#).

Here is the caller graph for this function:



#### 7.32.3.4 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

<code>evt</code>	[ <code>java.awt.event.ActionEvent</code> ] Event triggered by clicking the "Go Back" button.
------------------	---

Definition at line 784 of file [Order.java](#).

```
00784
00785     GEN-FIRST:event_goBackMenueBtnActionPerformed
00786     this.dispose();
00787     new Login().setVisible(true);
00788 } // GEN-LAST:event_goBackMenueBtnActionPerformed //
```

#### 7.32.3.5 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 374 of file [Order.java](#).

```
00374
00375
00376     jPanel2 = new javax.swing.JPanel();
00377     drinksPnl = new javax.swing.JPanel();
00378     drinksScroll = new javax.swing.JScrollPane();
00379     drinksTable = new javax.swing.JTable();
00380     appetizerPnl = new javax.swing.JPanel();
00381     appetizersScroll = new javax.swing.JScrollPane();
00382     appetizersTable = new javax.swing.JTable();
00383     mainCoursePnl = new javax.swing.JPanel();
00384     mainsScroll = new javax.swing.JScrollPane();
00385     mainsTable = new javax.swing.JTable();
00386     jPanel11 = new javax.swing.JPanel();
00387     subTotalLbl = new javax.swing.JLabel();
00388     vatLbl = new javax.swing.JLabel();
00389     totalLbl = new javax.swing.JLabel();
```

```
00390 receiptNoLbl = new javax.swing.JLabel();
00391 payBtn = new javax.swing.JButton();
00392 newReceiptBtn = new javax.swing.JButton();
00393 saveReceiptBtn = new javax.swing.JButton();
00394 jLabel1 = new javax.swing.JLabel();
00395 goBackMenueBtn = new javax.swing.JButton();
00396 jLabel2 = new javax.swing.JLabel();
00397
00398 setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00399 setTitle("Menu");
00400 setBackground(new java.awt.Color(204, 204, 204));
00401 setResizable(false);
00402
00403 jPanel2.setBackground(new java.awt.Color(248, 244, 230));
00404
00405 drinksPnl.setBackground(new java.awt.Color(248, 244, 230));
00406 drinksPnl.setBorder(javax.swing.BorderFactory.createTitledBorder(
00407     new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Drinks",
00408     javax.swing.border.TitledBorder.CENTER, javax.swing.border.TitledBorder.TOP,
00409     new java.awt.Font(PRIMARY_FONT, 0, 18))); // NOI18N
00410 drinksPnl.setPreferredSize(new java.awt.Dimension(260, 278));
00411
00412 drinksTable.setFont(new java.awt.Font(PRIMARY_FONT, 0, 14)); // NOI18N
00413 drinksTable.setModel(new javax.swing.table.DefaultTableModel(
00414     new Object[][] {
00415         },
00416         new String[] {
00417             });
00418     ));
00419
00420 drinksTable.getTableHeader().setReorderingAllowed(false);
00421 drinksScroll.setViewportView(drinksTable);
00422
00423 javax.swing.GroupLayout drinksPnlLayout = new javax.swing.GroupLayout(drinksPnl);
00424 drinksPnl.setLayout(drinksPnlLayout);
00425 drinksPnlLayout.setHorizontalGroup(
00426     drinksPnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00427         .addGroup(drinksPnlLayout.createSequentialGroup()
00428             .addGroup(drinksPnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00429                 .addComponent(drinksScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 281, Short.MAX_VALUE)
00430                 .addComponent(ContainerGap)));
00431     drinksPnlLayout.setVerticalGroup(
00432         drinksPnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00433             .addGroup(drinksPnlLayout.createSequentialGroup()
00434                 .addGroup(drinksPnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00435                     .addComponent(drinksScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 186, Short.MAX_VALUE)
00436                     .addComponent(ContainerGap)));
00437
00438 appetizerPnl.setBackground(new java.awt.Color(248, 244, 230));
00439 appetizerPnl.setBorder(javax.swing.BorderFactory.createTitledBorder(
00440     new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Appetizers",
00441     javax.swing.border.TitledBorder.CENTER, javax.swing.border.TitledBorder.TOP,
00442     new java.awt.Font(PRIMARY_FONT, 0, 18))); // NOI18N
00443 appetizerPnl.setPreferredSize(new java.awt.Dimension(260, 278));
00444
00445 appetizersTable.setFont(new java.awt.Font(PRIMARY_FONT, 0, 14)); // NOI18N
00446 appetizersTable.setModel(new javax.swing.table.DefaultTableModel(
00447     new Object[][] {
00448         },
00449         new String[] {
00450             });
00451     ));
00452 appetizersTable.getTableHeader().setReorderingAllowed(false);
00453 appetizersScroll.setViewportView(appetizersTable);
00454
00455 javax.swing.GroupLayout appetizerPnlLayout = new javax.swing.GroupLayout(appetizerPnl);
00456 appetizerPnl.setLayout(appetizerPnlLayout);
00457 appetizerPnlLayout.setHorizontalGroup(
00458     appetizerPnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00459         .addGroup(appetizerPnlLayout.createSequentialGroup()
00460             .addGroup(appetizerPnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00461                 .addComponent(appetizersScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 368, Short.MAX_VALUE)
00462                 .addComponent(ContainerGap)));
00463     appetizerPnlLayout.setVerticalGroup(
00464         appetizerPnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00465             .addGroup(appetizerPnlLayout.createSequentialGroup()
00466                 .addGroup(appetizerPnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00467                     .addComponent(appetizersScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 186, Short.MAX_VALUE)
00468                     .addComponent(ContainerGap)));
00469
00470 mainCoursePnl.setBackground(new java.awt.Color(248, 244, 230));
00471 mainCoursePnl.setBorder(javax.swing.BorderFactory.createTitledBorder(
00472     new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Main Course",
00473     javax.swing.border.TitledBorder.CENTER, javax.swing.border.TitledBorder.TOP,
00474     new java.awt.Font(PRIMARY_FONT, 0, 18))); // NOI18N
00475 mainCoursePnl.setPreferredSize(new java.awt.Dimension(260, 278));
00476
00477 mainCoursesTable.setFont(new java.awt.Font(PRIMARY_FONT, 0, 14)); // NOI18N
00478 mainCoursesTable.setModel(new javax.swing.table.DefaultTableModel(
00479     new Object[][] {
00480         },
00481         new String[] {
00482             });
00483     ));
00484 mainCoursesTable.getTableHeader().setReorderingAllowed(false);
00485 mainCoursesScroll.setViewportView(mainCoursesTable);
00486
00487 javax.swing.GroupLayout mainCoursePnlLayout = new javax.swing.GroupLayout(mainCoursePnl);
00488 mainCoursePnl.setLayout(mainCoursePnlLayout);
00489 mainCoursePnlLayout.setHorizontalGroup(
00490     mainCoursePnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00491         .addGroup(mainCoursePnlLayout.createSequentialGroup()
00492             .addGroup(mainCoursePnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00493                 .addComponent(mainCoursesScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 368, Short.MAX_VALUE)
00494                 .addComponent(ContainerGap)));
00495     mainCoursePnlLayout.setVerticalGroup(
00496         mainCoursePnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00497             .addGroup(mainCoursePnlLayout.createSequentialGroup()
00498                 .addGroup(mainCoursePnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00499                     .addComponent(mainCoursesScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 186, Short.MAX_VALUE)
00500                     .addComponent(ContainerGap)));
00501
00502 dessertsPnl.setBackground(new java.awt.Color(248, 244, 230));
00503 dessertsPnl.setBorder(javax.swing.BorderFactory.createTitledBorder(
00504     new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Desserts",
00505     javax.swing.border.TitledBorder.CENTER, javax.swing.border.TitledBorder.TOP,
00506     new java.awt.Font(PRIMARY_FONT, 0, 18))); // NOI18N
00507 dessertsPnl.setPreferredSize(new java.awt.Dimension(260, 278));
00508
00509 dessertsTable.setFont(new java.awt.Font(PRIMARY_FONT, 0, 14)); // NOI18N
00510 dessertsTable.setModel(new javax.swing.table.DefaultTableModel(
00511     new Object[][] {
00512         },
00513         new String[] {
00514             });
00515     ));
00516 dessertsTable.getTableHeader().setReorderingAllowed(false);
00517 dessertsScroll.setViewportView(dessertsTable);
00518
00519 javax.swing.GroupLayout dessertsPnlLayout = new javax.swing.GroupLayout(dessertsPnl);
00520 dessertsPnl.setLayout(dessertsPnlLayout);
00521 dessertsPnlLayout.setHorizontalGroup(
00522     dessertsPnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00523         .addGroup(dessertsPnlLayout.createSequentialGroup()
00524             .addGroup(dessertsPnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00525                 .addComponent(dessertsScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 368, Short.MAX_VALUE)
00526                 .addComponent(ContainerGap)));
00527     dessertsPnlLayout.setVerticalGroup(
00528         dessertsPnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00529             .addGroup(dessertsPnlLayout.createSequentialGroup()
00530                 .addGroup(dessertsPnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00531                     .addComponent(dessertsScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 186, Short.MAX_VALUE)
00532                     .addComponent(ContainerGap)));
00533
00534
00535
00536
00537
00538
00539
00540
00541
00542
00543
00544
00545
00546
00547
00548
00549
00550
00551
00552
00553
00554
00555
00556
00557
00558
00559
00560
00561
00562
00563
00564
00565
00566
00567
00568
00569
00570
00571
00572
00573
00574
00575
00576
00577
00578
00579
00580
00581
00582
00583
00584
00585
00586
00587
00588
00589
00590
00591
00592
00593
00594
00595
00596
00597
00598
00599
00600
00601
00602
00603
00604
00605
00606
00607
00608
00609
00610
00611
00612
00613
00614
00615
00616
00617
00618
00619
00620
00621
00622
00623
00624
00625
00626
00627
00628
00629
00630
00631
00632
00633
00634
00635
00636
00637
00638
00639
00640
00641
00642
00643
00644
00645
00646
00647
00648
00649
00650
00651
00652
00653
00654
00655
00656
00657
00658
00659
00660
00661
00662
00663
00664
00665
00666
00667
00668
00669
00670
00671
00672
00673
00674
00675
00676
00677
00678
00679
00680
00681
00682
00683
00684
00685
00686
00687
00688
00689
00690
00691
00692
00693
00694
00695
00696
00697
00698
00699
00700
00701
00702
00703
00704
00705
00706
00707
00708
00709
00710
00711
00712
00713
00714
00715
00716
00717
00718
00719
00720
00721
00722
00723
00724
00725
00726
00727
00728
00729
00730
00731
00732
00733
00734
00735
00736
00737
00738
00739
00740
00741
00742
00743
00744
00745
00746
00747
00748
00749
00750
00751
00752
00753
00754
00755
00756
00757
00758
00759
00760
00761
00762
00763
00764
00765
00766
00767
00768
00769
00770
00771
00772
00773
00774
00775
00776
00777
00778
00779
00780
00781
00782
00783
00784
00785
00786
00787
00788
00789
00790
00791
00792
00793
00794
00795
00796
00797
00798
00799
00800
00801
00802
00803
00804
00805
00806
00807
00808
00809
00810
00811
00812
00813
00814
00815
00816
00817
00818
00819
00820
00821
00822
00823
00824
00825
00826
00827
00828
00829
00830
00831
00832
00833
00834
00835
00836
00837
00838
00839
00840
00841
00842
00843
00844
00845
00846
00847
00848
00849
00850
00851
00852
00853
00854
00855
00856
00857
00858
00859
00860
00861
00862
00863
00864
00865
00866
00867
00868
00869
00870
00871
00872
00873
00874
00875
00876
00877
00878
00879
00880
00881
00882
00883
00884
00885
00886
00887
00888
00889
00890
00891
00892
00893
00894
00895
00896
00897
00898
00899
00900
00901
00902
00903
00904
00905
00906
00907
00908
00909
00910
00911
00912
00913
00914
00915
00916
00917
00918
00919
00920
00921
00922
00923
00924
00925
00926
00927
00928
00929
00930
00931
00932
00933
00934
00935
00936
00937
00938
00939
00940
00941
00942
00943
00944
00945
00946
00947
00948
00949
00950
00951
00952
00953
00954
00955
00956
00957
00958
00959
00960
00961
00962
00963
00964
00965
00966
00967
00968
00969
00970
00971
00972
00973
00974
00975
00976
00977
00978
00979
00980
00981
00982
00983
00984
00985
00986
00987
00988
00989
00990
00991
00992
00993
00994
00995
00996
00997
00998
00999
00999
```

```
00473     new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Main Course",
00474     javax.swing.border.TitledBorder.CENTER, javax.swing.border.TitledBorder.TOP,
00475     new java.awt.Font(PRIMARY_FONT, 0, 18))); // NOI18N
00476 mainCoursePnl.setPreferredSize(new java.awt.Dimension(260, 278));
00477
00478 mainsTable.setFont(new java.awt.Font(PRIMARY_FONT, 0, 14)); // NOI18N
00479 mainsTable.setModel(new javax.swing.table.DefaultTableModel(
00480     new Object[][] {
00481         },
00482         new String[] {
00483             });
00484     ));
00485 mainsTable.getTableHeader().setReorderingAllowed(false);
00486 mainsScroll.setViewportView(mainsTable);
00487
00488 javax.swing.GroupLayout mainCoursePnlLayout = new javax.swing.GroupLayout(mainCoursePnl);
00489 mainCoursePnl.setLayout(mainCoursePnlLayout);
00490 mainCoursePnlLayout.setHorizontalGroup(
00491     mainCoursePnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00492         .addGroup(mainCoursePnlLayout.createSequentialGroup()
00493             .addGroup(mainCoursePnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00494                 .addComponent(mainsScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 427, Short.MAX_VALUE)
00495                 .addGap(0, 0, 0)))
00496         .addGap(0, 0, 0))
00497 mainCoursePnlLayout.setVerticalGroup(
00498     mainCoursePnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00499         .addGroup(mainCoursePnlLayout.createSequentialGroup()
00500             .addComponent(mainsScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 186, Short.MAX_VALUE)
00501             .addGap(0, 0, 0)))
00502         .addGap(0, 0, 0))
00503
00504 jPanel1.setBackground(new java.awt.Color(248, 244, 230));
00505 jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(
00506     new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Receipt",
00507     javax.swing.border.TitledBorder.CENTER, javax.swing.border.TitledBorder.TOP,
00508     new java.awt.Font(PRIMARY_FONT, 0, 14))); // NOI18N
00509
00510 subTotalLbl.setFont(new java.awt.Font(PRIMARY_FONT_LIGHT, 0, 18)); // NOI18N
00511 subTotalLbl.setText("SubTotal: 0.0 SR");
00512
00513 vatLbl.setFont(new java.awt.Font(PRIMARY_FONT_LIGHT, 0, 18)); // NOI18N
00514 vatLbl.setText("VAT included: 0.0 SR");
00515
00516 totalLbl.setFont(new java.awt.Font(PRIMARY_FONT_LIGHT, 0, 18)); // NOI18N
00517 totalLbl.setText("Total: 0.0 SR");
00518
00519 receiptNoLbl.setFont(new java.awt.Font(PRIMARY_FONT_LIGHT, 0, 14)); // NOI18N
00520 receiptNoLbl.setText(RECEIPT_NO_PREFIX + "0");
00521
00522 javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
00523 jPanel1.setLayout(jPanel1Layout);
00524 jPanel1Layout.setHorizontalGroup(
00525     jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00526         .addGroup(jPanel1Layout.createSequentialGroup()
00527             .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00528                 .addGroup(jPanel1Layout.createSequentialGroup()
00529                     .addComponent(subTotalLbl)
00530                     .addComponent(vatLbl)
00531                     .addComponent(totalLbl)
00532                     .addComponent(receiptNoLbl)
00533                     .addGap(0, 0, 0)))
00534             .addGap(0, 0, 0))
00535 jPanel1Layout.setVerticalGroup(
00536     jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00537         .addGroup(jPanel1Layout.createSequentialGroup()
00538             .addComponent(subTotalLbl)
00539             .addGap(0, 0, 0)
00540             .addComponent(vatLbl)
00541             .addGap(0, 0, 0)
00542             .addComponent(totalLbl)
00543             .addGap(0, 0, 0))
00544             .addComponent(receiptNoLbl)));
00545
00546 payBtn.setFont(new java.awt.Font("Lucida Grande", 1, 18)); // NOI18N
00547 payBtn.setText("Pay");
00548 payBtn.addActionListener(this::payBtnActionPerformed);
00549
00550 newReceiptBtn.setFont(new java.awt.Font(PRIMARY_FONT, 1, 14)); // NOI18N
00551 newReceiptBtn.setText("New Receipt");
00552 newReceiptBtn.addActionListener(this::newReceiptBtnActionPerformed);
00553
00554 saveReceiptBtn.setFont(new java.awt.Font(PRIMARY_FONT, 1, 14)); // NOI18N
00555 saveReceiptBtn.setText("Save Receipt");
00556 saveReceiptBtn.addActionListener(this::saveReceiptBtnActionPerformed);
00557
00558 jPanel1.setBackground(new java.awt.Color(255, 153, 0));
```

```

00559     jLabel1.setFont(new java.awt.Font(PRIMARY_FONT, 1, 36)); // NOI18N
00560     jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00561     jLabel1.setText("BLACK PLATE MENU");
00562
00563     goBackMenueBtn.setFont(new java.awt.Font(PRIMARY_FONT, 1, 14)); // NOI18N
00564     goBackMenueBtn.setText("Go Back");
00565     goBackMenueBtn.addActionListener(this::goBackMenueBtnActionPerformed);
00566
00567     jLabel12.setIcon(new javax.swing.ImageIcon(getClass().getResource("/ui/screenshot.png"))); // NOI18N
00568
00569     javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);
00570     jPanel2.setLayout(jPanel2Layout);
00571     jPanel2Layout.setHorizontalGroup(
00572         jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00573             .addGroup(jPanel2Layout.createSequentialGroup()
00574                 .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00575                     .addGroup(jPanel2Layout.createSequentialGroup()
00576                         .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00577                             .addComponent(drinksPnl, javax.swing.GroupLayout.PREFERRED_SIZE, 305,
00578                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00579                             .addGroup(jPanel2Layout.createSequentialGroup()
00580                                 .addComponent(appetizerPnl, javax.swing.GroupLayout.PREFERRED_SIZE, 392,
00581                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00582                                 .addGap(18, 18, 18)
00583                                 .addComponent(mainCoursePnl, javax.swing.GroupLayout.PREFERRED_SIZE, 451,
00584                                     javax.swing.GroupLayout.PREFERRED_SIZE))
00585                             .addGroup(jPanel2Layout.createSequentialGroup()
00586                                 .addGap(251, 251, 251)
00587                                 .addComponent(jLabel12)
00588                                 .addGap(18, 18, 18)
00589                                 .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 433,
00590                                     javax.swing.GroupLayout.PREFERRED_SIZE))
00591                             .addGroup(jPanel2Layout.createSequentialGroup()
00592                                 .addGap(16, 16, 16)
00593                                 .addComponent(goBackMenueBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 130,
00594                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00595                                 .addGap(147, 147, 147)
00596                                 .addComponent(payBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 180,
00597                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00598                                 .addGap(18, 18, 18)
00599                             )
00600                             .addGroup(jPanel2Layout.createSequentialGroup()
00601                                 .addComponent(newReceiptBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 220,
00602                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00603                                 .addComponent(saveReceiptBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 220,
00604                                     javax.swing.GroupLayout.PREFERRED_SIZE))
00605                             .addGroup(jPanel2Layout.createSequentialGroup()
00606                                 .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
00607                                     javax.swing.GroupLayout.DEFAULT_SIZE,
00608                                     javax.swing.GroupLayout.PREFERRED_SIZE))
00609                             .addGap(30, Short.MAX_VALUE)))
00610             .addGroup(jPanel2Layout.createSequentialGroup()
00611                 .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00612                     .addGroup(jPanel2Layout.createSequentialGroup()
00613                         .addComponent(jLabel12, javax.swing.GroupLayout.Alignment.TRAILING)
00614                         .addComponent(jLabel1, javax.swing.GroupLayout.Alignment.TRAILING,
00615                             javax.swing.GroupLayout.PREFERRED_SIZE, 60,
00616                             javax.swing.GroupLayout.PREFERRED_SIZE)
00617                         .addGap(40, 40, 40)
00618                         .addComponent(appetizerPnl, javax.swing.GroupLayout.DEFAULT_SIZE, 230,
00619                             Short.MAX_VALUE)
00620                         .addComponent(mainCoursePnl, javax.swing.GroupLayout.DEFAULT_SIZE, 230,
00621                             Short.MAX_VALUE)
00622                         .addComponent(drinksPnl, javax.swing.GroupLayout.DEFAULT_SIZE, 230,
00623                             Short.MAX_VALUE)
00624                         .addGroup(jPanel2Layout.createSequentialGroup()
00625                             .addComponent(goBackMenueBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 40,
00626                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00627                             .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00628                                 .addComponent(newReceiptBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 30,
00629                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00630                                 .addComponent(saveReceiptBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 29,
00631                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00632                                 .addComponent(payBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 70,
00633                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00634                                 .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
00635                                     javax.swing.GroupLayout.DEFAULT_SIZE,
00636                                     javax.swing.GroupLayout.PREFERRED_SIZE))
00637                             .addGap(30, 30, 30)))
00638             )
00639         )
00640     )
00641     jPanel2.setVerticalGroup(
00642         jPanel2.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00643             .addGroup(jPanel2.createSequentialGroup()
00644                 .addGroup(jPanel2.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00645                     .addGroup(jPanel2.createSequentialGroup()
00646                         .addGroup(jPanel2.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00647                             .addComponent(jLabel12, javax.swing.GroupLayout.PREFERRED_SIZE, 60,
00648                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00649                             .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 433,
00650                                 javax.swing.GroupLayout.PREFERRED_SIZE))
00651                             .addGroup(jPanel2.createSequentialGroup()
00652                                 .addComponent(goBackMenueBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 130,
00653                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00654                                 .addGap(147, 147, 147)
00655                                 .addComponent(payBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 180,
00656                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00657                                 .addGap(18, 18, 18)
00658                                 .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
00659                                     javax.swing.GroupLayout.DEFAULT_SIZE,
00660                                     javax.swing.GroupLayout.PREFERRED_SIZE))
00661                             .addGroup(jPanel2.createSequentialGroup()
00662                                 .addComponent(newReceiptBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 220,
00663                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00664                                 .addComponent(saveReceiptBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 220,
00665                                     javax.swing.GroupLayout.PREFERRED_SIZE))
00666                             .addGap(30, Short.MAX_VALUE)))
00667             .addGroup(jPanel2.createSequentialGroup()
00668                 .addGroup(jPanel2.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00669                     .addGroup(jPanel2.createSequentialGroup()
00670                         .addComponent(appetizerPnl, javax.swing.GroupLayout.DEFAULT_SIZE, 230,
00671                             Short.MAX_VALUE)
00672                         .addComponent(mainCoursePnl, javax.swing.GroupLayout.DEFAULT_SIZE, 230,
00673                             Short.MAX_VALUE)
00674                         .addComponent(drinksPnl, javax.swing.GroupLayout.DEFAULT_SIZE, 230,
00675                             Short.MAX_VALUE)
00676                         .addGroup(jPanel2.createSequentialGroup()
00677                             .addComponent(goBackMenueBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 40,
00678                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00679                             .addGroup(jPanel2.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00680                                 .addComponent(newReceiptBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 30,
00681                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00682                                 .addComponent(saveReceiptBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 29,
00683                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00684                                 .addComponent(payBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 70,
00685                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00686                                 .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
00687                                     javax.swing.GroupLayout.DEFAULT_SIZE,
00688                                     javax.swing.GroupLayout.PREFERRED_SIZE))
00689                             .addGap(30, 30, 30)))
00690             )
00691         )
00692     )
00693 
```

```

00637     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00638     getContentPane().setLayout(layout);
00639     layout.setHorizontalGroup(
00640         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00641             .addComponent(jPanel12, javax.swing.GroupLayout.DEFAULT_SIZE,
00642                     javax.swing.GroupLayout.DEFAULT_SIZE,
00643                     Short.MAX_VALUE));
00644     layout.setVerticalGroup(
00645         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00646             .addGroup(layout.createSequentialGroup()
00647                 .addGap(0, 0, 0)
00648                 .addComponent(jPanel12, javax.swing.GroupLayout.DEFAULT_SIZE,
00649                     javax.swing.GroupLayout.DEFAULT_SIZE,
00650                     Short.MAX_VALUE))
00651             .addGap(0, 0, 0)));
00652 } // </editor-fold> //GEN-END:initComponents

```

References [es.ul.esit.app.Order.appetizerPnl](#), [es.ul.esit.app.Order.appetizersScroll](#), [es.ul.esit.app.Order.appetizersTable](#), [es.ul.esit.app.Order.drinksPnl](#), [es.ul.esit.app.Order.drinksScroll](#), [es.ul.esit.app.Order.drinksTable](#), [es.ul.esit.app.Order.goBackMenu](#), [es.ul.esit.app.Order.jLabel1](#), [es.ul.esit.app.Order.jLabel2](#), [es.ul.esit.app.Order.JPanel1](#), [es.ul.esit.app.Order.JPanel2](#), [es.ul.esit.app.Order.mainCoursePnl](#), [es.ul.esit.app.Order.mainsScroll](#), [es.ul.esit.app.Order.mainsTable](#), [es.ul.esit.app.Order.newRec](#), [es.ul.esit.app.Order.payBtn](#), [es.ul.esit.app.Order.PRIMARY\\_FONT](#), [es.ul.esit.app.Order.PRIMARY\\_FONT\\_LIGHT](#), [es.ul.esit.app.Order.RECEIPT\\_NO\\_PREFIX](#), [es.ul.esit.app.Order.receiptNoLbl](#), [es.ul.esit.app.Order.saveReceiptBtn](#), [es.ul.esit.app.Order.subTotalLbl](#), [es.ul.esit.app.Order.totalLbl](#), and [es.ul.esit.app.Order.vatLbl](#).

Referenced by [es.ul.esit.app.Order.Order\(\)](#).

Here is the caller graph for this function:



### 7.32.3.6 loadMenuFromDatabase()

```
void es.ul.esit.app.Order.loadMenuFromDatabase ( ) [inline], [private]
```

Loads menu data (drinks, appetizers, main courses) from the backend in a background thread.

Definition at line 228 of file [Order.java](#).

```

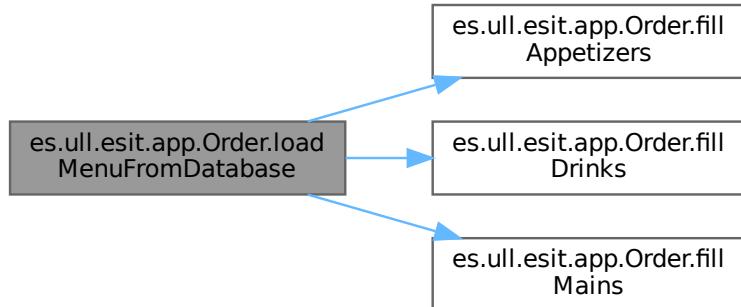
00228     new Thread(() -> {
00229         try {
00230             java.util.List<Drink> drinks = productService.getAllDrinks();
00231             java.util.List<Appetizer> appetizers = productService.getAllAppetizers();
00232             java.util.List<MainCourse> mains = productService.getAllMainCourses();
00233
00234             SwingUtilities.invokeLater(() -> {
00235                 fillDrinks(drinks);
00236                 fillAppetizers(appetizers);
00237                 fillMains(mains);
00238             });
00239         } catch (Exception ex) {
00240             LOGGER.error("Error loading menu from backend", ex);
00241             SwingUtilities.invokeLater(() -> JOptionPane.showMessageDialog(
00242                 this,
00243                 "Error loading menu from backend:\n" + ex.getMessage(),
00244                 "Menu loading error",
00245                 JOptionPane.ERROR_MESSAGE));
00246         }
00247     }).start();
00248 }
00249 }
00250 }

```

References [es.ull.esit.app.Order.fillAppetizers\(\)](#), [es.ull.esit.app.Order.fillDrinks\(\)](#), [es.ull.esit.app.Order.fillMains\(\)](#), [es.ull.esit.app.Order.LOGGER](#), 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:



### 7.32.3.7 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

<i>args</i>	[String[]] Command line arguments (not used).
-------------	---

Definition at line 798 of file [Order.java](#).

```
00798
00799     /* Set the Nimbus look and feel */ {
```

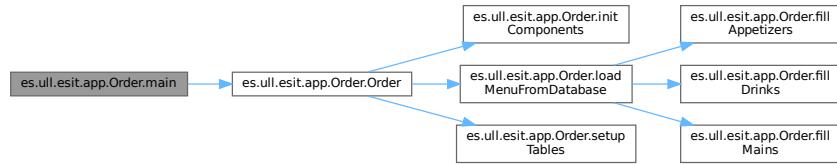
```

00800 // <editor-fold defaultstate="collapsed" desc=" Look and feel setting code
00801 // (optional) ">
00802 /*
00803 * If Nimbus (introduced in Java SE 6) is not available, stay with the default
00804 * look and feel.
00805 * For details see
00806 * http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
00807 */
00808 try {
00809     for (javax.swing.UIManager.LookAndFeelInfo info :
00810         javax.swing.UIManager.getInstalledLookAndFeels()) {
00811         if ("Nimbus".equals(info.getName())) {
00812             javax.swing.UIManager.setLookAndFeel(info.getClassName());
00813             break;
00814         }
00815     } catch (ClassNotFoundException | javax.swing.UnsupportedLookAndFeelException |
00816             InstantiationException
00817             | IllegalAccessException ex) {
00818     java.util.logging.Logger.getLogger(Order.class.getName()).log(java.util.logging.Level.SEVERE,
00819     null, ex);
00820     // </editor-fold>
00821     /* Create and display the form */
00822     java.awt.EventQueue.invokeLater(() -> new Order().setVisible(true));
00823 }

```

References [es.ul.esit.app.Order.Order\(\)](#).

Here is the call graph for this function:



### 7.32.3.8 newReceiptBtnActionPerformed()

```

void es.ul.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

<code>evt</code>	[[ <code>java.awt.event.ActionEvent</code> ]] Event triggered by clicking the "New Receipt" button.
------------------	---

Definition at line 752 of file [Order.java](#).

```

00752                                     { //
GEN-FIRST:event_newReceiptBtnActionPerformed

```

```

00753     if (total == 0.0) {
00754         // Nothing to reset
00755         return;
00756     }
00757
00758     resetQtyColumn(drinksModel);
00759     resetQtyColumn(appetizersModel);
00760     resetQtyColumn(mainsModel);
00761
00762     subTotal = 0.0;
00763     vat = 0.0;
00764     total = 0.0;
00765     lastBill = null;
00766
00767     subTotalLbl.setText("SubTotal: 0.0 SR");
00768     vatLbl.setText("VAT included: 0.0 SR");
00769     totalLbl.setText("Total: 0.0 SR");
00770
00771     receiptNo++;
00772     receiptNoLbl.setText(RECEIPT_NO_PREFIX + receiptNo);
00773 } // GEN-LAST:event_newReceiptBtnActionPerformed

```

References `es.ul.esit.app.Order.appetizersModel`, `es.ul.esit.app.Order.drinksModel`, `es.ul.esit.app.Order.lastBill`, `es.ul.esit.app.Order.mainsModel`, `es.ul.esit.app.Order.RECEIPT_NO_PREFIX`, `es.ul.esit.app.Order.receiptNoLbl`, `es.ul.esit.app.Order.resetQtyColumn()`, `es.ul.esit.app.Order.subTotalLbl`, `es.ul.esit.app.Order.totalLbl`, and `es.ul.esit.app.Order.vatLbl`.

Here is the call graph for this function:



### 7.32.3.9 payBtnActionPerformed()

```

void es.ul.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

<code>evt</code>	[[ <code>java.awt.event.ActionEvent</code> ]] Event triggered by clicking the "Pay" button.
------------------	---

Definition at line 667 of file `Order.java`.

```

00667
00668     GEN-FIRST:event_payBtnActionPerformed
00669     double itemsTotal = 0.0;
00669
00669

```

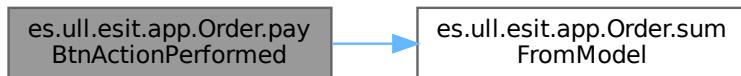
```

00670     itemsTotal += sumFromModel(drinksModel);
00671     itemsTotal += sumFromModel(appetizersModel);
00672     itemsTotal += sumFromModel(mainsModel);
00673
00674     if (itemsTotal == 0.0) {
00675         JOptionPane.showMessageDialog(
00676             this,
00677             "Please select at least one item (Qty > 0).",
00678             "No items selected",
00679             JOptionPane.INFORMATION_MESSAGE);
00680     }
00681 }
00682
00683 // Calculate bill using the dedicated service
00684 lastBill = orderService.calculateBill(itemsTotal);
00685
00686 subTotal = lastBill.getSubTotal();
00687 vat = lastBill.getVat();
00688 total = lastBill.getTotal();
00689
00690 subTotalLbl.setText("SubTotal: " + subTotal + " SR");
00691 vatLbl.setText("VAT included: " + vat + " SR");
00692 totalLbl.setText("Total: " + total + " SR");
00693
00694 JOptionPane.showMessageDialog(this, "Paid successfully");
00695 } // GEN-LAST:event_payBtnActionPerformed

```

References [es.ull.esit.app.Order.appetizersModel](#), [es.ull.esit.app.Order.drinksModel](#), [es.ull.esit.app.Order.lastBill](#), [es.ull.esit.app.Order.mainsModel](#), [es.ull.esit.app.Order.orderService](#), [es.ull.esit.app.Order.subTotalLbl](#), [es.ull.esit.app.Order.sumFromModel](#), [es.ull.esit.app.Order.totalLbl](#), and [es.ull.esit.app.Order.vatLbl](#).

Here is the call graph for this function:



### 7.32.3.10 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 347 of file [Order.java](#).

```

00347
00348     for (int row = 0; row < model.getRowCount(); row++) {
00349         model.setValueAt(0, row, 3); // column 3 is Qty
00350     }
00351 }

```

Referenced by [es.ull.esit.app.Order.newReceiptBtnActionPerformed\(\)](#).

Here is the caller graph for this function:



### 7.32.3.11 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

<code>evt</code>	[java.awt.event.ActionEvent] Event triggered by clicking the "Save Receipt" button.
------------------	---

Definition at line 710 of file [Order.java](#).

```
00710                                     { // 
00711     GEN-FIRST:event_saveReceiptBtnActionPerformed
00712     try {
00713         if (lastBill == null || lastBill.getTotal() == 0.0) {
00714             JOptionPane.showMessageDialog(
00715                 this,
00716                 "There is no paid bill to save.\nPlease press Pay first.",
00717                 "No bill",
00718                 JOptionPane.INFORMATION_MESSAGE);
00719         }
00720     }
00721     orderService.generateReceiptFile(receiptNo, lastBill);
00722
00723     JOptionPane.showMessageDialog(
00724         this,
00725         "Receipt number: " + receiptNo + " has been saved successfully.",
00726         "Receipt saved",
00727         JOptionPane.INFORMATION_MESSAGE);
00728
00729 } catch (Exception ex) {
00730     LOGGER.error("Error saving receipt", ex);
00731     JOptionPane.showMessageDialog(
00732         this,
00733         "Error saving receipt:\n" + ex.getMessage(),
00734         "Error",
00735         JOptionPane.ERROR_MESSAGE);
00736 }
00737 } // GEN-LAST:event_saveReceiptBtnActionPerformed
```

References [es.ull.esit.app.Order.lastBill](#), [es.ull.esit.app.Order.LOGGER](#), and [es.ull.esit.app.Order.orderService](#).

### 7.32.3.12 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 164 of file [Order.java](#).

```

00164             {
00165         // DRINKS
00166     drinksModel = new DefaultTableModel(
00167         new Object[] { ID_COLUMN_HEADER, ITEM_COLUMN_HEADER, PRICE_COLUMN_HEADER,
00168         QUANTITY_COLUMN_HEADER }, 0) {
00169     @Override
00170     public boolean isCellEditable(int row, int column) {
00171         // Only Qty column is editable
00172         return column == 3;
00173     }
00174     @Override
00175     public Class<?> getColumnClass(int columnIndex) {
00176         return switch (columnIndex) {
00177             case 0 -> Long.class; // ID
00178             case 2, 3 -> Integer.class; // Price, Qty
00179             default -> String.class;
00180         };
00181     }
00182 };
00183 drinksTable.setModel(drinksModel);
00184
// APPETIZERS
00185 appetizersModel = new DefaultTableModel(
00186     new Object[] { ID_COLUMN_HEADER, ITEM_COLUMN_HEADER, PRICE_COLUMN_HEADER,
00187         QUANTITY_COLUMN_HEADER }, 0) {
00188     @Override
00189     public boolean isCellEditable(int row, int column) {
00190         return column == 3;
00191     }
00192     @Override
00193     public Class<?> getColumnClass(int columnIndex) {
00194         return switch (columnIndex) {
00195             case 0 -> Long.class;
00196             case 2, 3 -> Integer.class;
00197             default -> String.class;
00198         };
00199     }
00200 };
00201 appetizersTable.setModel(appetizersModel);
00202
// MAIN COURSES
00203 mainsModel = new DefaultTableModel(
00204     new Object[] { ID_COLUMN_HEADER, ITEM_COLUMN_HEADER, PRICE_COLUMN_HEADER,
00205         QUANTITY_COLUMN_HEADER }, 0) {
00206     @Override
00207     public boolean isCellEditable(int row, int column) {
00208         return column == 3;
00209     }
00210     @Override
00211     public Class<?> getColumnClass(int columnIndex) {
00212         return switch (columnIndex) {
00213             case 0 -> Long.class;
00214             case 2, 3 -> Integer.class;
00215             default -> String.class;
00216         };
00217     }
00218 };
00219 };
00220 mainsTable.setModel(mainsModel);
00221 }
```

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.ID\\_COLUMN\\_HEADER](#), [es.ull.esit.app.Order.ITEM\\_COLUMN\\_HEADER](#), [es.ull.esit.app.Order.mainsModel](#), [es.ull.esit.app.Order.mainsTable](#), [es.ull.esit.app.Order.PRICE\\_COLUMN\\_HEADER](#), and [es.ull.esit.app.Order.QUANTITY\\_COLUMN\\_HEADER](#).

Referenced by [es.ull.esit.app.Order.Order\(\)](#).

Here is the caller graph for this function:



### 7.32.3.13 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

<i>model</i>	[javax.swing.table.DefaultTableModel] Table model (drinksModel, appetizersModel or mainsModel).
--------------	---

#### Returns

[double] Sum of the line totals in that model.

Definition at line 317 of file [Order.java](#).

```
00317
00318     double sum = 0.0;
00319     for (int row = 0; row < model.getRowCount(); row++) {
00320         Object qtyObj = model.getValueAt(row, 3); // Qty
00321         Object priceObj = model.getValueAt(row, 2); // Price
00322
00323         if (qtyObj == null || priceObj == null)
00324             continue;
00325
00326         int qty;
00327         int price;
00328         try {
00329             qty = ((Number) qtyObj).intValue();
00330             price = ((Number) priceObj).intValue();
00331         } catch (ClassCastException e) {
00332             // Fallback in case something comes as String
00333             qty = Integer.parseInt(qtyObj.toString());
00334             price = Integer.parseInt(priceObj.toString());
00335         }
00336
00337         if (qty > 0) {
00338             sum += qty * price;
00339         }
00340     }
00341     return sum;
00342 }
```

Referenced by [es.ull.esit.app.Order.payBtnActionPerformed\(\)](#).

Here is the caller graph for this function:



### 7.32.4 Member Data Documentation

#### 7.32.4.1 appetizerPnl

```
javax.swing.JPanel es.ull.esit.app.Order.appetizerPnl [private]
```

Panel that groups all appetizer items and controls.

Definition at line 829 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#).

#### 7.32.4.2 appetizersModel

```
DefaultTableModel es.ull.esit.app.Order.appetizersModel [private]
```

Table model for the category appetizers of the menu.

Definition at line 66 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.fillAppetizers\(\)](#), [es.ull.esit.app.Order.newReceiptBtnActionPerformed\(\)](#), [es.ull.esit.app.Order.payBtnActionPerformed\(\)](#), and [es.ull.esit.app.Order.setupTables\(\)](#).

#### 7.32.4.3 appetizersScroll

```
javax.swing.JScrollPane es.ull.esit.app.Order.appetizersScroll [private]
```

Scroll pane for appetizers table.

Definition at line 837 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#).

#### 7.32.4.4 appetizersTable

```
javax.swing.JTable es.ull.esit.app.Order.appetizersTable [private]
```

Table for appetizers items.

Definition at line 835 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#), and [es.ull.esit.app.Order.setupTables\(\)](#).

#### 7.32.4.5 drinksModel

```
DefaultTableModel es.ull.esit.app.Order.drinksModel [private]
```

Table model for the category drinks of the menu.

Definition at line 64 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.fillDrinks\(\)](#), [es.ull.esit.app.Order.newReceiptBtnActionPerformed\(\)](#), [es.ull.esit.app.Order.payBtnActionPerformed\(\)](#), and [es.ull.esit.app.Order.setupTables\(\)](#).

#### 7.32.4.6 **drinksPnl**

```
javax.swing.JPanel es.ull.esit.app.Order.drinksPnl [private]
```

Panel that groups all drink items and controls.

Definition at line 831 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#).

#### 7.32.4.7 **drinksScroll**

```
javax.swing.JScrollPane es.ull.esit.app.Order.drinksScroll [private]
```

Scroll pane for drinks table.

Definition at line 841 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#).

#### 7.32.4.8 **drinksTable**

```
javax.swing.JTable es.ull.esit.app.Order.drinksTable [private]
```

Table for drinks items.

Definition at line 839 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#), and [es.ull.esit.app.Order.setupTables\(\)](#).

#### 7.32.4.9 **goBackMenueBtn**

```
javax.swing.JButton es.ull.esit.app.Order.goBackMenueBtn [private]
```

Button to go back to the Login window.

Definition at line 843 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#).

#### 7.32.4.10 **ID\_COLUMN\_HEADER**

```
final String es.ull.esit.app.Order.ID_COLUMN_HEADER = "ID" [static], [private]
```

Column header used for ID across tables.

Definition at line 43 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.setupTables\(\)](#).

#### 7.32.4.11 ITEM\_COLUMN\_HEADER

```
final String es.ull.esit.app.Order.ITEM_COLUMN_HEADER = "Item" [static], [private]
```

Column header used for item name across tables.

Definition at line 46 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.setupTables\(\)](#).

#### 7.32.4.12 jLabel1

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

Title label "BLACK PLATE MENU".

Definition at line 845 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#).

#### 7.32.4.13 jLabel2

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

Logo / screenshot label on the top left.

Definition at line 847 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#).

#### 7.32.4.14 jPanel1

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

Panel that shows subtotal, VAT, total and receipt number.

Definition at line 849 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#).

#### 7.32.4.15 jPanel2

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

Main container panel of the Order window.

Definition at line 851 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#).

#### 7.32.4.16 lastBill

```
transient BillResult es.ull.esit.app.Order.lastBill [private]
```

Last calculated bill (after pressing Pay).

Definition at line 71 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.newReceiptBtnActionPerformed\(\)](#), [es.ull.esit.app.Order.payBtnActionPerformed\(\)](#), and [es.ull.esit.app.Order.saveReceiptBtnActionPerformed\(\)](#).

#### 7.32.4.17 LOGGER

```
final Logger es.ull.esit.app.Order.LOGGER = LoggerFactory.getLogger(Order.class) [static],  
[private]
```

Logger for this class.

Replaces printStackTrace() debug output.

Definition at line 61 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.loadMenuFromDatabase\(\)](#), and [es.ull.esit.app.Order.saveReceiptBtnActionPerformed\(\)](#).

#### 7.32.4.18 mainCoursePnl

```
javax.swing.JPanel es.ull.esit.app.Order.mainCoursePnl [private]
```

Panel that groups all main course items and controls.

Definition at line 833 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#).

#### 7.32.4.19 mainsModel

```
DefaultTableModel es.ull.esit.app.Order.mainsModel [private]
```

Table model for the category main courses of the menu.

Definition at line 68 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.fillMains\(\)](#), [es.ull.esit.app.Order.newReceiptBtnActionPerformed\(\)](#), [es.ull.esit.app.Order.payBtnActionPerformed\(\)](#), and [es.ull.esit.app.Order.setupTables\(\)](#).

#### 7.32.4.20 mainsScroll

```
javax.swing.JScrollPane es.ull.esit.app.Order.mainsScroll [private]
```

Scroll pane for main course table.

Definition at line 855 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#).

#### 7.32.4.21 mainsTable

```
javax.swing.JTable es.ull.esit.app.Order.mainsTable [private]
```

Table for main course items.

Definition at line 853 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#), and [es.ull.esit.app.Order.setupTables\(\)](#).

#### 7.32.4.22 newReceiptBtn

```
javax.swing.JButton es.ull.esit.app.Order.newReceiptBtn [private]
```

Button to start a new receipt.

Definition at line 857 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#).

#### 7.32.4.23 orderService

```
final transient OrderService es.ull.esit.app.Order.orderService = new OrderService() [private]
```

Service used to calculate bill totals and generate receipt files.

Definition at line 58 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.payBtnActionPerformed\(\)](#), and [es.ull.esit.app.Order.saveReceiptBtnActionPerformed\(\)](#).

#### 7.32.4.24 payBtn

```
javax.swing.JButton es.ull.esit.app.Order.payBtn [private]
```

Button to calculate and confirm payment.

Definition at line 859 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#).

#### 7.32.4.25 PRICE\_COLUMN\_HEADER

```
final String es.ull.esit.app.Order.PRICE_COLUMN_HEADER = "Price (SR)" [static], [private]
```

Column header used for price across tables.

Definition at line 49 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.setupTables\(\)](#).

#### 7.32.4.26 PRIMARY\_FONT

```
final String es.ull.esit.app.Order.PRIMARY_FONT = "Yu Gothic UI" [static], [private]
```

Primary font used in the GUI.

Definition at line 34 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#).

#### 7.32.4.27 PRIMARY\_FONT\_LIGHT

```
final String es.ull.esit.app.Order.PRIMARY_FONT_LIGHT = "Yu Gothic UI Light" [static], [private]
```

Lighter variant of the primary font.

Definition at line 37 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#).

#### 7.32.4.28 productService

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

Service used to load products from the backend.

Definition at line 55 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.loadMenuFromDatabase\(\)](#).

#### 7.32.4.29 QUANTITY\_COLUMN\_HEADER

```
final String es.ull.esit.app.Order.QUANTITY_COLUMN_HEADER = "Qty" [static], [private]
```

Column header used for quantity across tables.

Definition at line 52 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.setupTables\(\)](#).

#### 7.32.4.30 RECEIPT\_NO\_PREFIX

```
final String es.ull.esit.app.Order.RECEIPT_NO_PREFIX = "Receipt No. : " [static], [private]
```

Prefix used for the receipt number label to avoid duplicated literals.

Definition at line 40 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#), [es.ull.esit.app.Order.newReceiptBtnActionPerformed\(\)](#), and [es.ull.esit.app.Order.Order\(\)](#).

#### 7.32.4.31 receiptNoLbl

```
javax.swing.JLabel es.ull.esit.app.Order.receiptNoLbl [private]
```

Label that shows the current receipt number.

Definition at line 861 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#), [es.ull.esit.app.Order.newReceiptBtnActionPerformed\(\)](#), and [es.ull.esit.app.Order.Order\(\)](#).

#### 7.32.4.32 saveReceiptBtn

```
javax.swing.JButton es.ull.esit.app.Order.saveReceiptBtn [private]
```

Button to save the current receipt to a file.

Definition at line 863 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#).

#### 7.32.4.33 subTotalLbl

```
javax.swing.JLabel es.ull.esit.app.Order.subTotalLbl [private]
```

Label that displays the subtotal amount.

Definition at line 865 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#), [es.ull.esit.app.Order.newReceiptBtnActionPerformed\(\)](#), and [es.ull.esit.app.Order.payBtnActionPerformed\(\)](#).

#### 7.32.4.34 totalLbl

```
javax.swing.JLabel es.ull.esit.app.Order.totalLbl [private]
```

Label that displays the total amount.

Definition at line 867 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#), [es.ull.esit.app.Order.newReceiptBtnActionPerformed\(\)](#), and [es.ull.esit.app.Order.payBtnActionPerformed\(\)](#).

#### 7.32.4.35 vatLbl

```
javax.swing.JLabel es.ull.esit.app.Order.vatLbl [private]
```

Label that displays the VAT amount.

Definition at line 869 of file [Order.java](#).

Referenced by [es.ull.esit.app.Order.initComponents\(\)](#), [es.ull.esit.app.Order.newReceiptBtnActionPerformed\(\)](#), and [es.ull.esit.app.Order.payBtnActionPerformed\(\)](#).

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

- [Order.java](#)

## 7.33 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:

es.ull.esit.app.middleware.service. OrderService	
-	VAT_RATE
+	calculateBill()
+	generateReceiptFile()

### Public Member Functions

- BillResult [calculateBill](#) (double itemsTotalSum)
- void [generateReceiptFile](#) (int receiptNo, BillResult bill) throws FileNotFoundException  
*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%).*

### 7.33.1 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 18 of file [OrderService.java](#).

### 7.33.2 Member Function Documentation

#### 7.33.2.1 calculateBill()

```
BillResult es.ull.esit.app.middleware.service.OrderService.calculateBill (
    double itemsTotalSum ) [inline]
```

Definition at line 23 of file [OrderService.java](#).

```
00023
00024     double vat = itemsTotalSum * VAT_RATE;
00025     double total = itemsTotalSum + vat;
00026
00027     return new BillResult(
00028         Math.round(itemsTotalSum * 100.0) / 100.0,
00029         Math.round(vat * 100.0) / 100.0,
00030         Math.round(total * 100.0) / 100.0);
00031 }
```

References [es.ull.esit.app.middleware.service.OrderService.VAT\\_RATE](#).

### 7.33.2.2 generateReceiptFile()

```
void es.ul.esit.app.middleware.service.OrderService.generateReceiptFile (
    int receiptNo,
    BillResult bill ) throws FileNotFoundException [inline]
```

Generates a local text file representing a receipt.

#### Parameters

<i>receiptNo</i>	[int] Numeric identifier of the receipt.
<i>bill</i>	[BillResult] Calculated bill result to print.

#### Exceptions

<i>FileNotFoundException</i>	If the receipt file cannot be created or opened.
------------------------------	--

Definition at line 40 of file [OrderService.java](#).

```
00040
00041
00042     try {
00043         Files.createDirectories(Path.of("receipts"));
00044
00045         Path file = Path.of("receipts", "billNo." + receiptNo + ".txt");
00046         try (PrintWriter output =
00047              new PrintWriter(Files.newBufferedWriter(file, StandardCharsets.UTF_8))) {
00048
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     } catch (IOException e) {
00060         FileNotFoundException fnfe = new FileNotFoundException(
00061             "Could not create/write receipt file for receiptNo=" + receiptNo);
00062         fnfe.initCause(e);
00063         throw fnfe;
00064     }
00065 }
```

### 7.33.3 Member Data Documentation

#### 7.33.3.1 VAT\_RATE

```
final double es.ul.esit.app.middleware.service.OrderService.VAT_RATE = 0.15 [static], [private]
```

VAT rate applied to the subtotal (15%).

Definition at line 21 of file [OrderService.java](#).

Referenced by [es.ul.esit.app.middleware.service.OrderService.calculateBill\(\)](#).

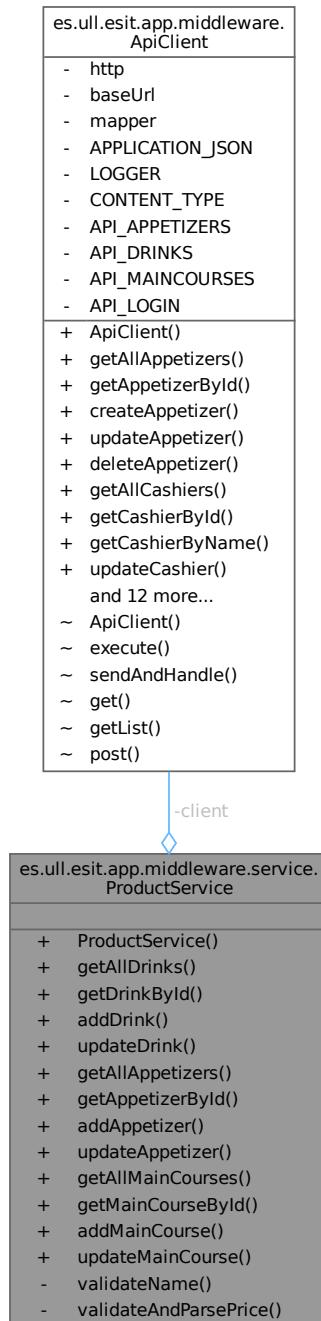
The documentation for this class was generated from the following file:

- [OrderService.java](#)

## 7.34 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 \(\)](#)  
*Retrieves all drinks from the backend.*
  - Drink [getDrinkById \(Long id\)](#)  
*Retrieves a single drink by its identifier.*
  - void [addDrink \(String name, String priceStr\)](#)  
*Adds a new drink to the backend menu.*
  - void [updateDrink \(Long id, String name, String priceStr\)](#)  
*Updates an existing drink in the backend menu.*
  - List< Appetizer > [getAllAppetizers \(\)](#)  
*Retrieves all appetizers from the backend.*
  - Appetizer [getAppetizerById \(Long id\)](#)  
*Retrieves a single appetizer by its identifier.*
  - void [addAppetizer \(String name, String priceStr\)](#)  
*Adds a new appetizer to the backend menu.*
  - void [updateAppetizer \(Long id, String name, String priceStr\)](#)  
*Updates an existing appetizer in the backend menu.*
  - List< MainCourse > [getAllMainCourses \(\)](#)  
*Retrieves all main courses from the backend.*
  - MainCourse [getMainCourseById \(Long id\)](#)  
*Retrieves a single main course by its identifier.*
  - void [addMainCourse \(String name, String priceStr\)](#)  
*Adds a new main course to the backend menu.*
  - void [updateMainCourse \(Long id, String name, String priceStr\)](#)  
*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.*

## 7.34.1 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](#).

## 7.34.2 Constructor & Destructor Documentation

### 7.34.2.1 ProductService()

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

Constructs a ProductService with the given API client.

### Parameters

<code>client</code>	[ApiClient] REST client used to perform HTTP requests.
---------------------	--

Definition at line 26 of file [ProductService.java](#).

```
00026     this.client = client;
00027 }
00028 }
```

References [es.ul.esit.app.middleware.service.ProductService.client](#).

## 7.34.3 Member Function Documentation

### 7.34.3.1 addAppetizer()

```
void es.ul.esit.app.middleware.service.ProductService.addAppetizer (
    String name,
    String priceStr) [inline]
```

Adds a new appetizer to the backend menu.

Validates user input and sends a POST request using the ApiClient.

### Parameters

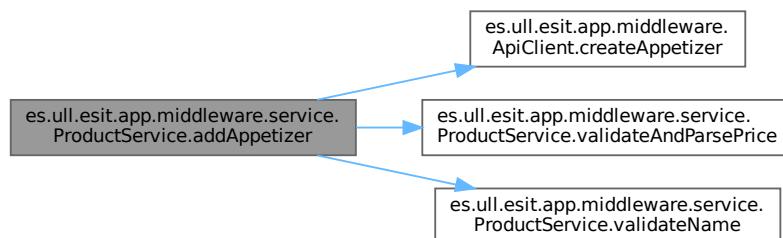
<code>name</code>	[String] Name of the new appetizer.
<code>priceStr</code>	[String] Price entered as a string.

Definition at line 115 of file [ProductService.java](#).

```
00115
00116     int price = validateAndParsePrice(priceStr);
00117     validateName(name);
00118
00119     Appetizer newAppetizer = new Appetizer(null, name, price, null);
00120     client.createAppetizer(newAppetizer);
00121 }
```

References [es.ul.esit.app.middleware.service.ProductService.client](#), [es.ul.esit.app.middleware.ApiClient.createAppetizer\(\)](#), [es.ul.esit.app.middleware.service.ProductService.validateAndParsePrice\(\)](#), and [es.ul.esit.app.middleware.service.ProductService.validateName\(\)](#).

Here is the call graph for this function:



### 7.34.3.2 addDrink()

```
void es.ull.esit.app.middleware.service.ProductService.addDrink (
    String name,
    String priceStr ) [inline]
```

Adds a new drink to the backend menu.

Validates the user input and sends a POST request using the ApiClient.

#### Parameters

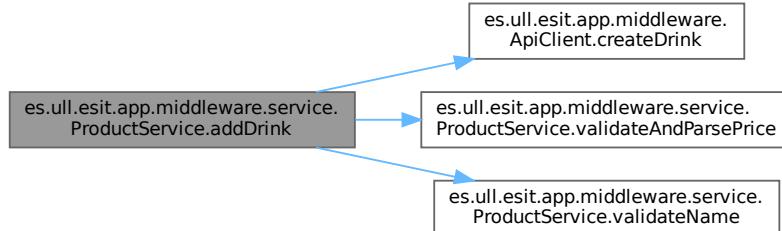
<code>name</code>	[String] Name of the new drink.
<code>priceStr</code>	[String] Price entered as a string.

Definition at line 59 of file [ProductService.java](#).

```
00059
00060     int price = validateAndParsePrice(priceStr);
00061     validateName(name);
00062
00063     Drink newDrink = new Drink(null, name, price, null);
00064     client.createDrink(newDrink);
00065 }
```

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.validateName\(\)](#)

Here is the call graph for this function:



### 7.34.3.3 addMainCourse()

```
void es.ull.esit.app.middleware.service.ProductService.addMainCourse (
    String name,
    String priceStr ) [inline]
```

Adds a new main course to the backend menu.

Validates user input and sends a POST request using the ApiClient.

### Parameters

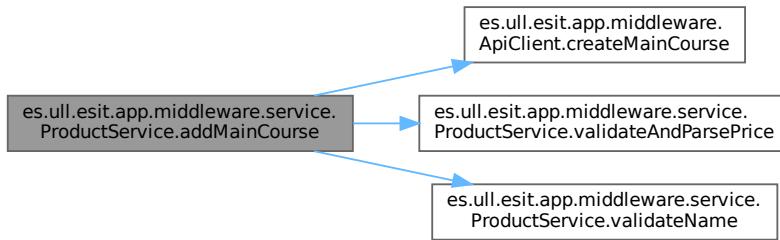
<code>name</code>	[String] Name of the new main course.
<code>priceStr</code>	[String] Price entered as a string.

Definition at line 171 of file [ProductService.java](#).

```
00171
00172     int price = validateAndParsePrice(priceStr);
00173     validateName(name);
00174
00175     MainCourse newMainCourse = new MainCourse(null, name, price, null);
00176     client.createMainCourse(newMainCourse);
00177 }
```

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.validateName\(\)](#)

Here is the call graph for this function:



### 7.34.3.4 getAllAppetizers()

```
List< Appetizer > es.ull.esit.app.middleware.service.ProductService.getAllAppetizers ( )
[inline]
```

Retrieves all appetizers from the backend.

#### Returns

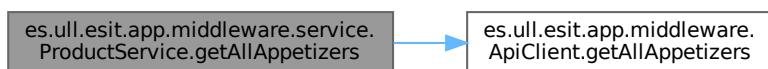
[List<Appetizer>] List of all appetizers in the menu.

Definition at line 93 of file [ProductService.java](#).

```
00093
00094     return client.getAllAppetizers();
00095 }
```

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:



### 7.34.3.5 getAllDrinks()

```
List< Drink > es.ull.esit.app.middleware.service.ProductService.getAllDrinks ( ) [inline]
```

Retrieves all drinks from the backend.

#### Returns

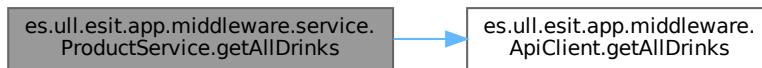
[List<Drink>] List of all drinks available in the menu.

Definition at line 37 of file [ProductService.java](#).

```
00037     {
00038         return client.getAllDrinks();
00039     }
```

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:



### 7.34.3.6 getAllMainCourses()

```
List< MainCourse > es.ull.esit.app.middleware.service.ProductService.getAllMainCourses ( ) [inline]
```

Retrieves all main courses from the backend.

#### Returns

[List<MainCourse>] List of all main courses in the menu.

Definition at line 149 of file [ProductService.java](#).

```
00149     {
00150         return client.getAllMainCourses();
00151     }
```

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:



### 7.34.3.7 getAppetizerById()

```
Appetizer es.ull.esit.app.middleware.service.ProductService.getAppetizerById (
    Long id ) [inline]
```

Retrieves a single appetizer by its identifier.

**Parameters**

<i>id</i>	[Long] Unique identifier of the appetizer.
-----------	--

**Returns**

[Appetizer] Appetizer object corresponding to the given id.

Definition at line 103 of file [ProductService.java](#).

```
00103
00104     return client.getAppetizerById(id);
00105 }
```

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:

**7.34.3.8 getDrinkById()**

```
Drink es.ull.esit.app.middleware.service.ProductService.getDrinkById (
    Long id ) [inline]
```

Retrieves a single drink by its identifier.

**Parameters**

<i>id</i>	[Long] Unique identifier of the drink.
-----------	--

**Returns**

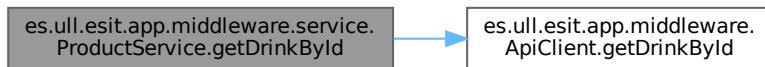
[Drink] Drink object corresponding to the given id.

Definition at line 47 of file [ProductService.java](#).

```
00047
00048     return client.getDrinkById(id);
00049 }
```

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:



### 7.34.3.9 getMainCourseById()

```
MainCourse es.ull.esit.app.middleware.service.ProductService.getMainCourseById (
    Long id ) [inline]
```

Retrieves a single main course by its identifier.

#### Parameters

<i>id</i>	[Long] Unique identifier of the main course.
-----------	--

#### Returns

[MainCourse] MainCourse object corresponding to the given id.

Definition at line 159 of file [ProductService.java](#).

```
00159
00160     return client.getMainCourseById(id);
00161 }
```

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:



### 7.34.3.10 updateAppetizer()

```
void es.ull.esit.app.middleware.service.ProductService.updateAppetizer (
    Long id,
    String name,
    String priceStr ) [inline]
```

Updates an existing appetizer in the backend menu.

### Parameters

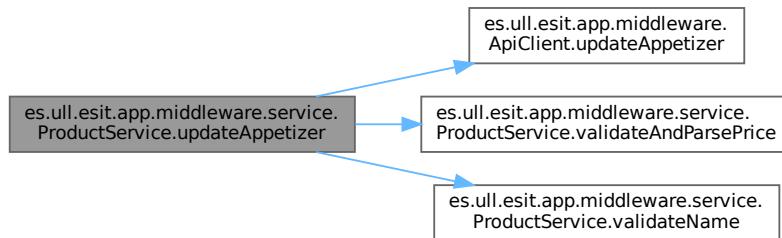
<i>id</i>	[Long] Identifier of the appetizer to update.
<i>name</i>	[String] New name for the appetizer.
<i>priceStr</i>	[String] New price entered as a string.

Definition at line 130 of file [ProductService.java](#).

```
00130
00131     if (id == null) {
00132         throw new IllegalArgumentException("No appetizer selected!");
00133     }
00134
00135     int price = validateAndParsePrice(priceStr);
00136     validateName(name);
00137
00138     Appetizer updatedAppetizer = new Appetizer(id, name, price, null);
00139     client.updateAppetizer(id, updatedAppetizer);
00140 }
```

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.validateName\(\)](#).

Here is the call graph for this function:



### 7.34.3.11 updateDrink()

```
void es.ull.esit.app.middleware.service.ProductService.updateDrink (
    Long id,
    String name,
    String priceStr ) [inline]
```

Updates an existing drink in the backend menu.

### Parameters

<i>id</i>	[Long] Identifier of the drink to update.
<i>name</i>	[String] New name for the drink.
<i>priceStr</i>	[String] New price entered as a string.

Definition at line 74 of file [ProductService.java](#).

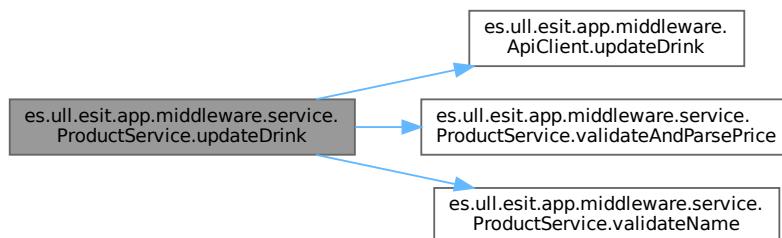
```
00074
00075     if (id == null) {
```

```

00076     throw new IllegalArgumentException("No drink selected!");
00077 }
00078
00079 int price = validateAndParsePrice(priceStr);
00080 validateName(name);
00081
00082 Drink updatedDrink = new Drink(id, name, price, null);
00083 client.updateDrink(id, updatedDrink);
00084 }
```

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.validateName\(\)](#)

Here is the call graph for this function:



### 7.34.3.12 updateMainCourse()

```

void es.ull.esit.app.middleware.service.ProductService.updateMainCourse (
    Long id,
    String name,
    String priceStr ) [inline]
```

Updates an existing main course in the backend menu.

#### Parameters

<i>id</i>	[Long] Identifier of the main course to update.
<i>name</i>	[String] New name for the main course.
<i>priceStr</i>	[String] New price entered as a string.

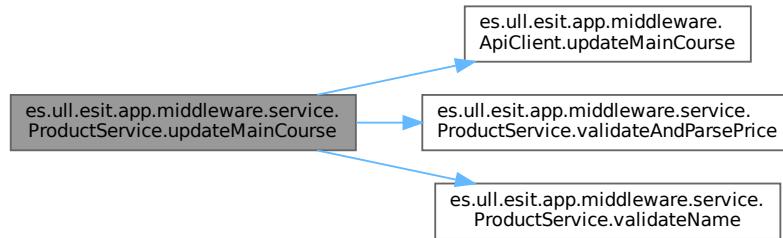
Definition at line 186 of file [ProductService.java](#).

```

00186
00187     if (id == null) {
00188         throw new IllegalArgumentException("No main course selected!");
00189     }
00190
00191     int price = validateAndParsePrice(priceStr);
00192     validateName(name);
00193
00194     MainCourse updatedMainCourse = new MainCourse(id, name, price, null);
00195     client.updateMainCourse(id, updatedMainCourse);
00196 }
```

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.validateName\(\)](#)

Here is the call graph for this function:



### 7.34.3.13 validateAndParsePrice()

```
int es.ul.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

<i>priceStr</i>	[String] Price entered as a string.
-----------------	-------------------------------------

#### Returns

[int] Parsed price value.

#### Exceptions

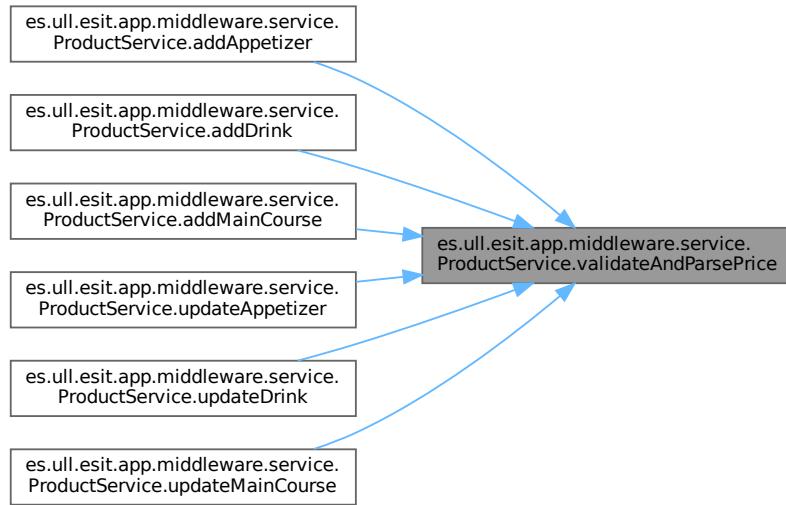
<i>IllegalArgumentException</i>	If the price is empty, negative or not a valid number.
---------------------------------	--

Definition at line 221 of file [ProductService.java](#).

```
00221
00222     if (priceStr == null || priceStr.trim().isEmpty()) {
00223         throw new IllegalArgumentException("Price cannot be empty.");
00224     }
00225     try {
00226         int price = Integer.parseInt(priceStr.trim());
00227         if (price < 0) {
00228             throw new IllegalArgumentException("Price cannot be negative.");
00229         }
00230         return price;
00231     } catch (NumberFormatException e) {
00232         throw new IllegalArgumentException("Price must be a valid whole number.");
00233     }
00234 }
```

Referenced by [es.ul.esit.app.middleware.service.ProductService.addAppetizer\(\)](#), [es.ul.esit.app.middleware.service.ProductService.addMainCourse\(\)](#), [es.ul.esit.app.middleware.service.ProductService.updateAppetizer\(\)](#), [es.ul.esit.app.middleware.service.ProductService.updateDrink\(\)](#), and [es.ul.esit.app.middleware.service.ProductService.updateMainCourse\(\)](#).

Here is the caller graph for this function:



### 7.34.3.14 validateName()

```
void es.ul.esit.app.middleware.service.ProductService.validateName (
    String name) [inline], [private]
```

Validates that the item name is not null or empty.

#### Parameters

<i>name</i>	[String] Name of the item to validate.
-------------	--

#### Exceptions

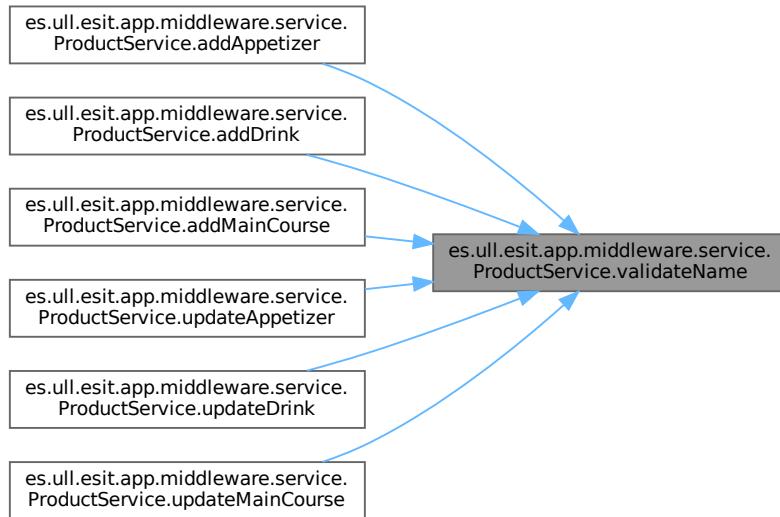
<i>IllegalArgumentException</i>	If the name is null or empty.
---------------------------------	-------------------------------

Definition at line 206 of file [ProductService.java](#).

```
00206
00207     if (name == null || name.trim().isEmpty()) {
00208         throw new IllegalArgumentException("Item name cannot be empty.");
00209     }
00210 }
```

Referenced by [es.ul.esit.app.middleware.service.ProductService.addAppetizer\(\)](#), [es.ul.esit.app.middleware.service.ProductService.addDrink\(\)](#), [es.ul.esit.app.middleware.service.ProductService.addMainCourse\(\)](#), [es.ul.esit.app.middleware.service.ProductService.updateAppetizer\(\)](#), [es.ul.esit.app.middleware.service.ProductService.updateDrink\(\)](#), and [es.ul.esit.app.middleware.service.ProductService.updateMainCourse\(\)](#).

Here is the caller graph for this function:



## 7.34.4 Member Data Documentation

### 7.34.4.1 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.addMainCourse\(\)](#), [es.ull.esit.app.middleware.service.ProductService.getAllAppetizers\(\)](#), [es.ull.esit.app.middleware.service.ProductService.getAllDrinks\(\)](#), [es.ull.esit.app.middleware.service.ProductService.getAllMainCourses\(\)](#), [es.ull.esit.app.middleware.service.ProductService.getAppetizerById\(\)](#), [es.ull.esit.app.middleware.service.ProductService.getDrinkById\(\)](#), [es.ull.esit.app.middleware.service.ProductService.getMainCourseById\(\)](#), [es.ull.esit.app.middleware.service.ProductService.getProductByName\(\)](#), [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\(\)](#).

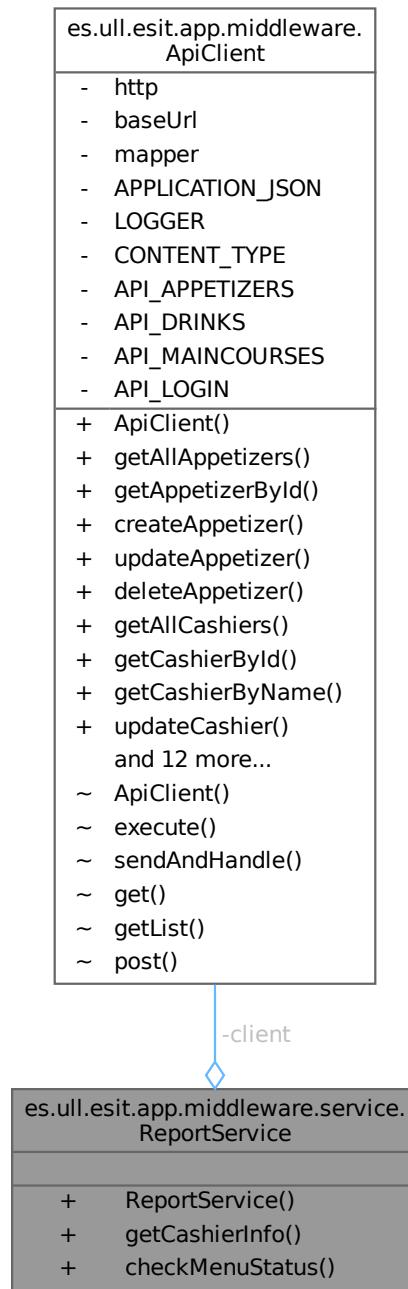
The documentation for this class was generated from the following file:

- [ProductService.java](#)

## 7.35 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 \(\)](#)  
*Loads information about all registered cashiers.*
  - String [checkMenuStatus \(\)](#)  
*Checks backend connectivity and counts menu items.*

### Private Attributes

- final [ApiClient client](#)  
*REST API client used to communicate with the backend.*

## 7.35.1 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](#).

## 7.35.2 Constructor & Destructor Documentation

### 7.35.2.1 ReportService()

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

Constructs a ReportService with the given API client.

#### Parameters

<code>client</code>	[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](#).

## 7.35.3 Member Function Documentation

### 7.35.3.1 checkMenuStatus()

```
String es.ull.esit.app.middleware.service.ReportService.checkMenuStatus () [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

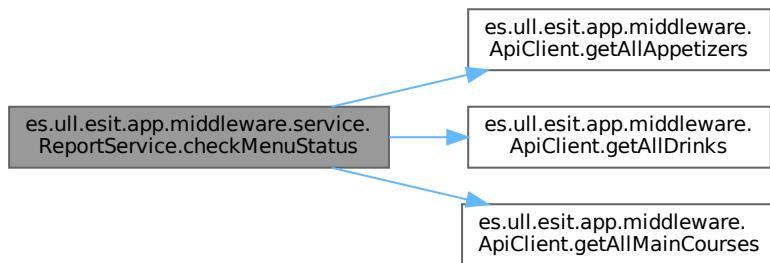
[String] Human-readable status message about the menu system.

Definition at line 49 of file [ReportService.java](#).

```
00049     {
00050         // Fetch lists to verify connectivity.
00051         List<Appetizer> appetizers = client.getAllAppetizers();
00052         List<Drink> drinks = client.getAllDrinks();
00053         List<MainCourse> mainCourses = client.getAllMainCourses();
00054
00055         return String.format(
00056             "Menu System Online: %d appetizers, %d drinks, %d main courses available.",
00057             appetizers.size(), drinks.size(), mainCourses.size());
00058     }
```

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:



#### 7.35.3.2 getCashierInfo()

List< Cashier > es.ull.esit.app.middleware.service.ReportService.getCashierInfo ( ) [inline]

Loads information about all registered cashiers.

Uses the "/api/cashiers" endpoint to obtain the list.

**Returns**

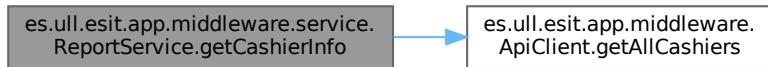
[List<Cashier>] List of cashiers returned by the backend.

Definition at line 37 of file [ReportService.java](#).

```
00037     {
00038     return client.getAllCashiers();
00039 }
```

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:



## 7.35.4 Member Data Documentation

### 7.35.4.1 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.ReportService.getAllCashiers\(\)](#) and [es.ull.esit.app.middleware.service.ReportService.getReport\(\)](#).

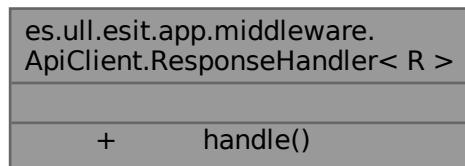
The documentation for this class was generated from the following file:

- [ReportService.java](#)

## 7.36 es.ull.esit.app.middleware.ApiClient.ResponseHandler< R > Interface Template Reference

Functional interface for processing HTTP responses and possibly throwing IOException from JSON parsing.

Collaboration diagram for es.ull.esit.app.middleware.ApiClient.ResponseHandler< R >:



## Public Member Functions

- R [handle](#) (int status, String body) throws IOException

### 7.36.1 Detailed Description

Functional interface for processing HTTP responses and possibly throwing IOException from JSON parsing.

Definition at line [125](#) of file [ApiClient.java](#).

### 7.36.2 Member Function Documentation

#### 7.36.2.1 handle()

```
R es.ull.esit.app.middleware.ApiClient.ResponseHandler< R >.handle (
    int status,
    String body ) throws IOException
```

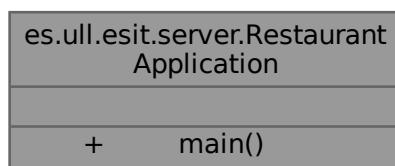
The documentation for this interface was generated from the following file:

- [ApiClient.java](#)

## 7.37 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.*

### 7.37.1 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](#).

### 7.37.2 Member Function Documentation

#### 7.37.2.1 main()

```
static void es.ul.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

<code>args</code>	[String[]] Command-line arguments (not used).
-------------------	---

Definition at line 30 of file [RestaurantApplication.java](#).

```
00030 {
00031     SpringApplication.run(RestaurantApplication.class, args);
00032 }
```

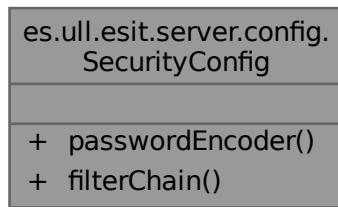
The documentation for this class was generated from the following file:

- [RestaurantApplication.java](#)

## 7.38 es.ul.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:*

### 7.38.1 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](#).

### 7.38.2 Member Function Documentation

#### 7.38.2.1 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**

<i>http</i>	[HttpSecurity] The HttpSecurity builder to define security policies.
-------------	--

**Returns**

[SecurityFilterChain] The configured security filter chain.

**Exceptions**

<i>Exception</i>	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 }
```

### 7.38.2.2 passwordEncoder()

PasswordEncoder es.uill.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](#)

## 7.39 es.uill.esit.app.middleware.model.User Class Reference

Client-side model representing an authenticated user.

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

es.ull.esit.app.middleware.model.User	
-	username
-	role
+	User()
+	User()
+	getUsername()
+	setUsername()
+	getRole()
+	setRole()
+	isAdmin()

## 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.*

### 7.39.1 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](#).

## 7.39.2 Constructor & Destructor Documentation

### 7.39.2.1 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       }

### 7.39.2.2 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

<i>username</i>	[String] Username of the user.
<i>role</i>	[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     }
```

## 7.39.3 Member Function Documentation

### 7.39.3.1 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     }
```

### 7.39.3.2 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      }
```

### 7.39.3.3 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      }
```

### 7.39.3.4 setRole()

```
void es.ull.esit.app.middleware.model.User.setRole (
    String role ) [inline]
```

Sets the user role.

#### Parameters

<i>role</i>	[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      }
```

### 7.39.3.5 setUsername()

```
void es.ull.esit.app.middleware.model.User.setUsername (
    String username ) [inline]
```

Sets the username.

#### Parameters

<i>username</i>	[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     }
```

### 7.39.4 Member Data Documentation

#### 7.39.4.1 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](#).

#### 7.39.4.2 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](#)

## 7.40 es.ull.esit.server.middleware.model.User Class Reference

JPA entity that represents a user in the system.

Collaboration diagram for es.ul.esit.server.middleware.model.User:

es.ul.esit.server.middleware.model.User	
-	id
-	username
-	passwordHash
-	role
+	getId()
+	setId()
+	getUsername()
+	setUsername()
+	getPasswordHash()
+	setPasswordHash()
+	getRole()
+	setRole()

## 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.*

### 7.40.1 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](#).

### 7.40.2 Member Function Documentation

#### 7.40.2.1 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 }
```

#### 7.40.2.2 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 }
```

#### 7.40.2.3 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 }
```

#### 7.40.2.4 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     }
```

#### 7.40.2.5 setId()

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

Sets the database identifier of the user.

##### Parameters

<i>id</i>	[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     }
```

#### 7.40.2.6 setPasswordHash()

```
void es.ull.esit.server.middleware.model.User.setPasswordHash (
    String passwordHash ) [inline]
```

Sets the BCrypt hashed password of the user.

##### Parameters

<i>passwordHash</i>	[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     }
```

#### 7.40.2.7 setRole()

```
void es.ull.esit.server.middleware.model.User.setRole (
    String role ) [inline]
```

Sets the role of the user.

**Parameters**

<i>role</i>	[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 }
```

#### 7.40.2.8 setUsername()

```
void es.ull.esit.server.middleware.model.User.setUsername (
    String username ) [inline]
```

Sets the user's username.

**Parameters**

<i>username</i>	[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 }
```

### 7.40.3 Member Data Documentation

#### 7.40.3.1 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](#).

#### 7.40.3.2 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](#).

#### 7.40.3.3 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](#).

#### 7.40.3.4 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](#).

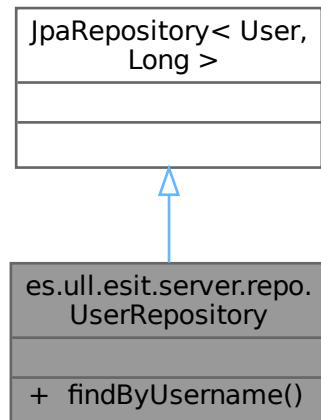
The documentation for this class was generated from the following file:

- [server/src/main/java/es/ull/esit/server/middleware/model/User.java](#)

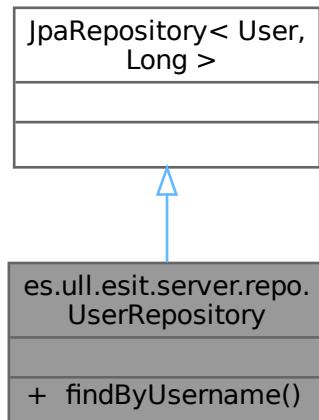
## 7.41 es.ull.esit.server.repo.UserRepository Interface Reference

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.*

#### 7.41.1 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](#).

#### 7.41.2 Member Function Documentation

##### 7.41.2.1 [findByUsername\(\)](#)

```
Optional< User > es.ull.esit.server.repo.UserRepository.findByUsername (
    String username )
```

Finds a user by their unique username.

###### Parameters

<i>username</i>	[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 -- password: admin
00063 INSERT INTO users (id, username, password_hash, role)
00064 VALUES (1, 'admin', '$2b$10$TAxhf.ziuvw7ynX1SQYjMeftWo47ft8M4JfdXjLoBPdERwuo7zD06', 'ADMIN');
00065
00066 -- CAJEROS (generan automáticamente filas en cashier gracias al trigger)
00067 -- password: cashier123 (hash reutilizado)
00068 INSERT INTO users (id, username, password_hash, role)
00069 VALUES
00070 (2, 'laura.mendez', '$2b$10$HLeNcr00JL6qpGJUd.klqud4PfSRci2qvukr7fnklGjtMO06OhwLC', 'CASHIER'),
00071 (3, 'diego.santos', '$2b$10$HLeNcr00JL6qpGJUd.klqud4PfSRci2qvukr7fnklGjtMO06OhwLC', 'CASHIER');
00072
00073 -- En este punto, gracias al trigger:
00074 --   cashier tendrá:
00075 --   (2, 'laura.mendez', 1500) y (3, 'diego.santos', 1500)
00076
00077 -- =====
00078 -- DEPENDENTS DE LOS CASHIERS
00079 -- =====
00080 INSERT INTO dependent_cashier (dependent_name, dependent_relation, dependent_sex, cashier_id)
00081 VALUES
00082 ('emma', 'daughter', 'F', 2),
00083 ('alex', 'son', 'M', 3);
00084
00085 -- =====
00086 -- ITEMS DEL TICKET (combinaciones reales del menú Black Plate)
00087 -- =====
00088 INSERT INTO item (item_food, food_price, food_id,
00089           item_appetizers, appetizers_price, appetizers_id,
00090           item_drinks, drinks_price, drinks_id,
00091           receipt_id)
00092 VALUES
00093 ('Grilled Salmon',     18, 1, 'Truffle Fries',      6, 1, 'Coca Cola',        2, 1, 1),
00094 ('Black Angus Burger', 16, 2, 'Garlic Bread',       4, 2, 'Sparkling Water', 2, 2, 2),
00095 ('Chicken Teriyaki',   14, 3, 'Hummus Bowl',       5, 3, 'Iced Tea',          3, 3, 3),
00096 ('Vegan Buddha Bowl',  13, 4, 'Sweet Potato Chips', 4, 4, 'Lemonade',         3, 4, 4),
00097 ('Pasta Alfredo',     11, 5, 'Bruschetta',        4, 5, 'Coca Cola',        2, 1, 5);
00098
00099 -- =====
00100 -- CHEF PREPARA PLATOS
00101 -- =====
00102 INSERT INTO chef_prepare_item (chef_id, food_id, sweet_id, drinks_id)
00103 VALUES
00104 (3001, 1, 1, 1),
00105 (3002, 2, 2, 2),
00106 (3003, 3, 3, 3);
00107
00108 -- =====
00109 -- RECEIPTS TOMADOS POR CASHIERS (usa IDs 2 y 3)
00110 -- =====
00111 INSERT INTO receipt_taken_by_cashier (receipt_id, cashier_id)
00112 VALUES
00113 (1, 2),
00114 (2, 2),
00115 (3, 3),
00116 (4, 3),
00117 (5, 2);
00118
00119 -- =====
00120 -- Tabla receipt_copy (pruebas)
00121 -- =====
00122 INSERT INTO receipt_copy (receipt_id, receipt_time, receipt_date, receipt_total)
00123 VALUES
00124 (1, '12:45:51', '2024-10-01', 42);
00125
00126 DELETE FROM receipt_copy WHERE receipt_id = 1;
00127
00128 -- =====
00129 -- Productos para la API (Black Plate)
00130 -- =====
00131 INSERT INTO appetizers (name, price) VALUES
00132 ('Truffle Fries',    6),
00133 ('Garlic Bread',     4),
00134 ('Hummus Bowl',      5),
00135 ('Sweet Potato Chips', 4),
00136 ('Bruschetta',       4);
00137
00138 INSERT INTO drinks (name, price) VALUES
00139 ('Coca Cola',        2),
00140 ('Sparkling Water',  2),
00141 ('Iced Tea',         3),

```

```

00142 ('Lemonade',      3),
00143 ('Red Fruit Juice', 4);
00144
00145 INSERT INTO maincourse (name, price) VALUES
00146 ('Grilled Salmon',     18),
00147 ('Black Angus Burger', 16),
00148 ('Chicken Teriyaki',   14),
00149 ('Vegan Buddha Bowl',  13),
00150 ('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_restaurant','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,'abdullah','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_Prepares_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: Cheffname --
00232 DELIMITER $$ 
00233 CREATE PROCEDURE Cheffname (IN Cheffname VARCHAR(10))
00234 BEGIN
00235   SELECT * FROM Chef WHERE Chef_name = Cheffname;
00236 END $$ 
00237 DELIMITER ;

```

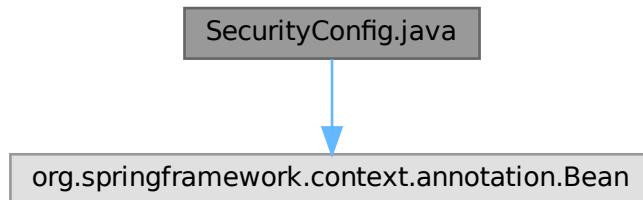
```
00238
00239 CALL Chefname('mohammad');
00240 CALL Chefname('abduallah');
00241
00242 -- TRIGGER: uppercase names in Manager --
00243 CREATE TRIGGER uppercase BEFORE INSERT ON Manager
00244 FOR EACH ROW
00245 SET NEW.Manager_Fname = UPPER(NEW.Manager_Fname);
00246
00247 INSERT INTO Manager VALUES (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](#)

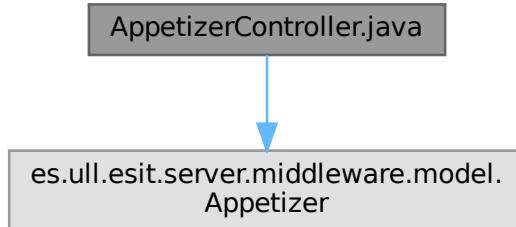
## 8.17 SecurityConfig.java

[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.](#)

```

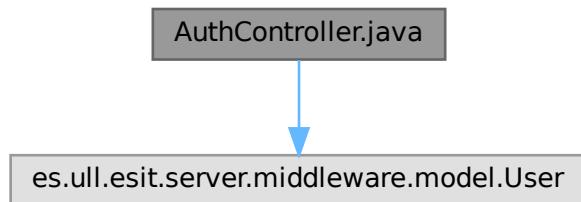
00001 package es.ull.esit.server.controller;
00002
00003 import es.ull.esit.server.middleware.model.Appetizer;
00004 import es.ull.esit.server.repo.AppetizerRepository;
00005 import org.springframework.beans.factory.annotation.Autowired;
00006 import org.springframework.http.*;
00007 import org.springframework.web.bind.annotation.*;
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](#)

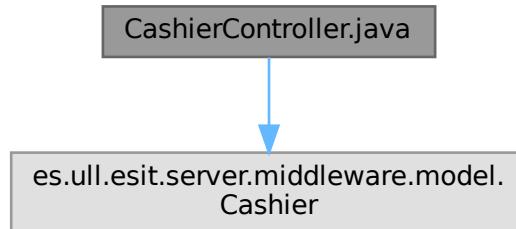
## 8.21 AuthController.java

[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
00032     @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());
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.](#)

```

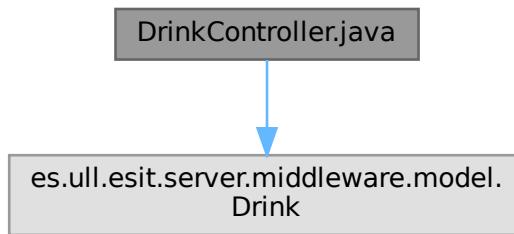
00001 package es.ull.esit.server.controller;
00002
00003 import es.ull.esit.server.middleware.model.Cashier;
00004 import es.ull.esit.server.repo.CashierRepository;
00005 import org.springframework.beans.factory.annotation.Autowired;
00006 import org.springframework.http.HttpStatus;
00007 import org.springframework.http.ResponseEntity;
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

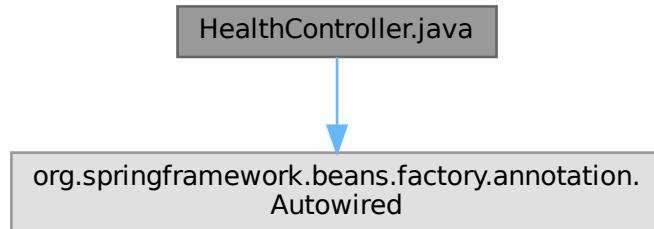
[Go to the documentation of this file.](#)

```
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.](#)

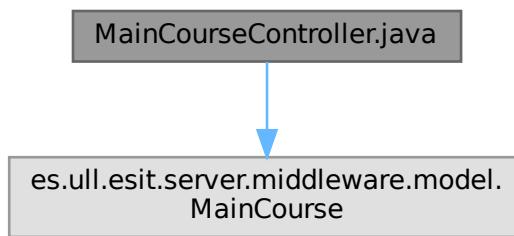
```
00001 package es.ull.esit.server.controller;
00002
00003 import org.springframework.beans.factory.annotation.Autowired;
00004 import org.springframework.http.HttpStatus;
00005 import org.springframework.http.ResponseEntity;
00006 import org.springframework.web.bind.annotation.GetMapping;
00007 import 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
00029     @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
00052     @GetMapping("/db-check")
00053     public ResponseEntity<Map<String, Object>> checkDatabase() {
00054         Map<String, Object> response = new HashMap<>();
00055
00056         try (Connection conn = dataSource.getConnection()) {
00057             boolean isValid = conn.isValid(2);
00058
00059             if (isValid) {
00060                 response.put("status", "UP");
00061                 response.put("database", "Connected");
00062                 response.put("catalog", conn.getCatalog());
00063                 response.put("url", conn.getMetaData().getURL());
00064                 response.put("timestamp", System.currentTimeMillis());
00065                 return ResponseEntity.ok(response);
00066             } else {
00067                 response.put("status", "DOWN");
00068                 response.put("database", "Connection not valid");
00069                 return ResponseEntity.status(HttpStatus.SERVICE_UNAVAILABLE).body(response);
00070             }
00071         } catch (Exception e) {
00072             response.put("status", "DOWN");
00073             response.put("database", "Connection failed");
00074             response.put("error", e.getMessage());
00075             response.put("timestamp", System.currentTimeMillis());
00076             return ResponseEntity.status(HttpStatus.SERVICE_UNAVAILABLE).body(response);
00077         }
00078     }
00079 }
```

## 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

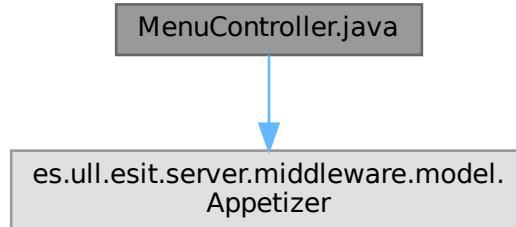
[Go to the documentation of this file.](#)

```
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.](#)

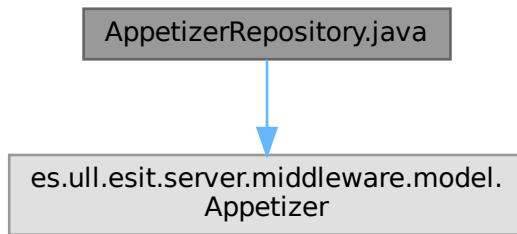
```
00001 package es.ull.esit.server.controller;
00002
00003 import es.ull.esit.server.middleware.model.Appetizer;
00004 import es.ull.esit.server.middleware.model.Drink;
00005 import es.ull.esit.server.middleware.model.MainCourse;
00006 import es.ull.esit.server.repo.AppetizerRepository;
00007 import 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
00019 @RestController
00020 @RequestMapping("/api/menu")
00021 public class MenuController {
00022
00023     @Autowired
00024     private MainCourseRepository mainCourseRepository;
00025
00026     @Autowired
00027     private AppetizerRepository appetizerRepository;
00028
00029     @Autowired
00030 }
```

```

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.](#)

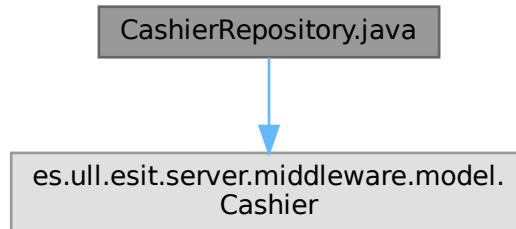
```

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](#)

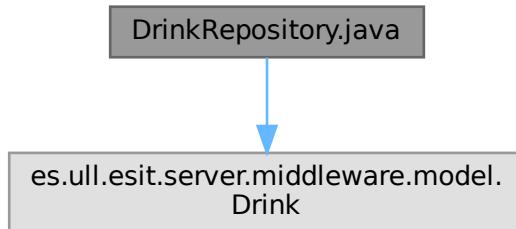
## 8.35 CashierRepository.java

[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
0014 public interface CashierRepository extends JpaRepository<Cashier, Long> {
00015     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](#)

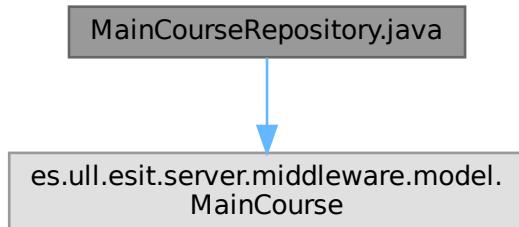
## 8.37 DrinkRepository.java

[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

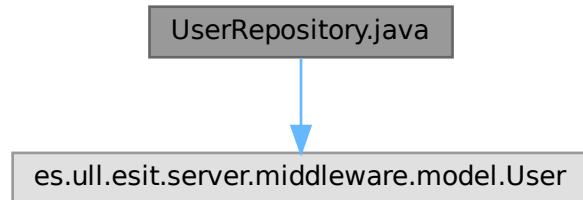
[Go to the documentation of this file.](#)

```
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

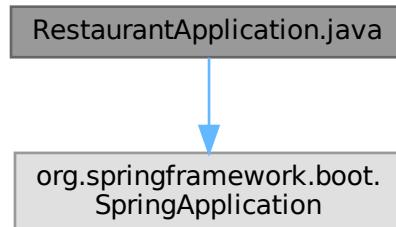
[Go to the documentation of this file.](#)

```
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.ulb.esit.app

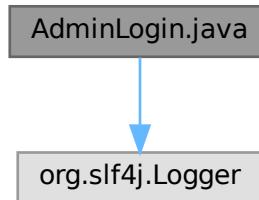
## 8.45 AboutUs.java

[Go to the documentation of this file.](#)

```
00099         .addContainerGap(71, Short.MAX_VALUE)
00100         .addComponent(jLabel1)
00101         .addGap(67, 67, 67)
00102         .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 203,
00103             javax.swing.GroupLayout.PREFERRED_SIZE)
00104         .addGap(26, 26, 26)
00105         .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00106             javax.swing.GroupLayout.PREFERRED_SIZE)
00107         .addGap(30, 30, 30)));
00108
00109     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00110     getContentPane().setLayout(layout);
00111     layout.setHorizontalGroup(
00112         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00113             .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
00114                 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,
00119                 Short.MAX_VALUE));
00120
00121     pack();
00122     setLocationRelativeTo(null);
00123 // </editor-fold>//GEN-END:initComponents
00124
00125 private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {//
00126     GEN-FIRST:event_jButton3ActionPerformed
00127     new CashierLogin().setVisible(true);
00128     this.dispose(); // Close current window
00129 }// GEN-LAST:event_jButton3ActionPerformed
00130
00131 public static void main(String[] args) {
00132     try {
00133         for (javax.swing.UIManager.LookAndFeelInfo info :
00134             javax.swing.UIManager.getInstalledLookAndFeels()) {
00135             if ("Nimbus".equals(info.getName())) {
00136                 javax.swing.UIManager.setLookAndFeel(info.getClassName());
00137                 break;
00138             }
00139         }
00140     } catch (ClassNotFoundException | InstantiationException | IllegalAccessException
00141         | javax.swing.UnsupportedLookAndFeelException ex) {
00142         java.util.logging.Logger.getLogger(AboutUs.class.getName())
00143             .log(java.util.logging.Level.SEVERE, null, ex);
00144     }
00145 // </editor-fold>
00146
00147     // Create and display the form
00148     java.awt.EventQueue.invokeLater(() -> new AboutUs().setVisible(true));
00149 }
00150
00151 // Variables declaration - do not modify
00152 // GEN-BEGIN:variables
00153 // Sonarqube rule java:S1450 must be ignored here as these variables are
00154 // auto-generated by the Form Editor and need to remain as instance variables.
00155 private javax.swing.JButton jButton3;
00156 private javax.swing.JLabel jLabel1;
00157 private javax.swing.JPanel jPanel1;
00158 private javax.swing.JScrollPane jScrollPane1;
00159 private javax.swing.JTextArea ourStory;
00160 // End of variables declaration//GEN-END:variables
00161 }
```

## 8.46 AdminLogin.java File Reference

```
import org.slf4j.Logger;
Include dependency graph for AdminLogin.java:
```



### 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
00003 import org.slf4j.Logger;
00004 import org.slf4j.LoggerFactory;
00005 import java.util.function.Supplier;
00006
00016 public class AdminLogin extends javax.swing.JFrame {
00017
00019     private static final String PRIMARY_FONT = "Yu Gothic UI";
00020
00022     private static final Logger LOGGER = LoggerFactory.getLogger(AdminLogin.class);
00023
00030     static Supplier<java.awt.Window> adminProductsSupplier = AdminProducts::new;
00031
00038     static Supplier<java.awt.Window> orderSupplier = Order::new;
00039
00045     static Supplier<java.awt.Window> loginSupplier = Login::new;
00046
00052     public AdminLogin() {
00053         initComponents();
00054     }
00055
00063     @SuppressWarnings("unchecked")
00064     // <editor-fold defaultstate="collapsed" desc="Generated Code">
00065     // GEN-BEGIN:initComponents
00066     private void initComponents() {
00067
00068         jPanel1 = new javax.swing.JPanel();
00069         jLabel3 = new javax.swing.JLabel();
00070         jLabel1 = new javax.swing.JLabel();
00071         jButton2 = new javax.swing.JButton();
00072
00073     }
00074
00075     // </editor-fold>
00076
00077 }
```

```
00072     jButton1 = new javax.swing.JButton();
00073     jButton3 = new javax.swing.JButton();
00074 
00075     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00076     setTitle("Admin ");
00077     setResizable(false);
00078 
00079     jPanell.setBackground(new java.awt.Color(248, 244, 230));
00080 
00081     jLabel13.setFont(new java.awt.Font(PRIMARY_FONT, 1, 24)); // NOI18N
00082     jLabel13.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00083     jLabel13.setText("Welcome Admin");
00084 
00085     jLabel11.setIcon(new javax.swing.ImageIcon(getClass().getResource("/ui/screenshot.png"))); // NOI18N
00086 
00087     jButton2.setBackground(new java.awt.Color(153, 153, 153));
00088     jButton2.setFont(new java.awt.Font(PRIMARY_FONT, 1, 18)); // NOI18N
00089     jButton2.setText("Update Prices");
00090     jButton2.addActionListener(this::jButton2ActionPerformed);
00091 
00092     jButton1.setBackground(new java.awt.Color(153, 153, 153));
00093     jButton1.setFont(new java.awt.Font(PRIMARY_FONT, 1, 18)); // NOI18N
00094     jButton1.setText("Menu");
00095     jButton1.addActionListener(this::jButton1ActionPerformed);
00096 
00097     jButton3.setFont(new java.awt.Font(PRIMARY_FONT, 1, 18)); // NOI18N
00098     jButton3.setText("LogOut");
00099     jButton3.addActionListener(this::jButton3ActionPerformed);
00100 
00101     javax.swing.GroupLayout jPanellLayout = new javax.swing.GroupLayout(jPanell);
00102     jPanell.setLayout(jPanellLayout);
00103     jPanellLayout.setHorizontalGroup(
00104         jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00105             .addGroup(jPanellLayout.createSequentialGroup()
00106                 .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00107                     .addGroup(jPanellLayout.createSequentialGroup()
00108                         .addGroup(jPanellLayout.createSequentialGroup()
00109                             .addComponent(jLabel11)
00110                             .addGroup(jPanellLayout.createSequentialGroup()
00111                                 .addGap(66, 66, 66)
00112                                 .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00113                                     .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 320,
00114                                         javax.swing.GroupLayout.PREFERRED_SIZE)
00115                                     .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 320,
00116                                         javax.swing.GroupLayout.PREFERRED_SIZE)
00117                                     .addComponent(jLabel13, javax.swing.GroupLayout.PREFERRED_SIZE, 306,
00118                                         javax.swing.GroupLayout.PREFERRED_SIZE))
00119                                     .addGroup(jPanellLayout.createSequentialGroup()
00120                                         .addGap(152, 152, 152)
00121                                         .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 141,
00122                                             javax.swing.GroupLayout.PREFERRED_SIZE))
00123                                     .addGap(69, Short.MAX_VALUE)))
00124             .addGroup(jPanellLayout.createSequentialGroup()
00125                 .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00126                     .addGroup(jPanellLayout.createSequentialGroup()
00127                         .addGap(31, 31, 31)
00128                         .addComponent(jLabel13, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
00129                             javax.swing.GroupLayout.PREFERRED_SIZE)
00130                         .addGap(18, 18, 18)
00131                         .addComponent(jLabel11)
00132                         .addGap(29, 29, 29)
00133                         .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00134                             javax.swing.GroupLayout.PREFERRED_SIZE)
00135                         .addGap(18, 18, 18)
00136                         .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00137                             javax.swing.GroupLayout.PREFERRED_SIZE)
00138                         .addGap(29, 29, 29)
00139                         .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00140                             javax.swing.GroupLayout.PREFERRED_SIZE)
00141                         .addGap(115, Short.MAX_VALUE)))
00142 
00143     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00144     getContentPane().setLayout(layout);
00145     layout.setHorizontalGroup(
00146         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00147             .addComponent(jPanell, javax.swing.GroupLayout.PREFERRED_SIZE,
00148                 javax.swing.GroupLayout.PREFERRED_SIZE));
00149     layout.setVerticalGroup(
00150         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00151             .addComponent(jPanell, javax.swing.GroupLayout.DEFAULT_SIZE,
00152                 Short.MAX_VALUE));
00153 
00154     pack();
```

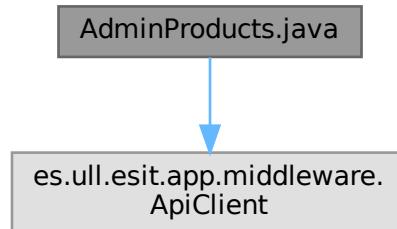
```

00155     setLocationRelativeTo(null);
00156 } // </editor-fold> //GEN-END:initComponents
00157
00167 private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {//
00168     GEN-FIRST:event_jButton2ActionPerformed
00169     try {
00170         adminProductsSupplier.get().setVisible(true);
00171         this.dispose();
00172     } catch (Exception ex) {
00173         LOGGER.error("Error opening product admin window", ex);
00174         javax.swing.JOptionPane.showMessageDialog(
00175             this,
00176             "Error opening product admin window:\n" + ex.getMessage(),
00177             "Error",
00178             javax.swing.JOptionPane.ERROR_MESSAGE);
00179 } // GEN-LAST:event_jButton2ActionPerformed
00180
00189 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//
00190     GEN-FIRST:event_jButton1ActionPerformed
00191     try {
00192         orderSupplier.get().setVisible(true);
00193         this.dispose();
00194     } catch (Exception ex) {
00195         LOGGER.error("Error opening menu window", ex);
00196         javax.swing.JOptionPane.showMessageDialog(
00197             this,
00198             "Error opening menu window:\n" + ex.getMessage(),
00199             "Error",
00200             javax.swing.JOptionPane.ERROR_MESSAGE);
00201 } // GEN-LAST:event_jButton1ActionPerformed
00202
00211 private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {//
00212     GEN-FIRST:event_jButton3ActionPerformed
00213     loginSupplier.get().setVisible(true);
00214 } // GEN-LAST:event_jButton3ActionPerformed
00215
00225 public static void main(String[] args) {
00226     try {
00227         for (javax.swing.UIManager.LookAndFeelInfo info :
00228             javax.swing.UIManager.getInstalledLookAndFeels()) {
00229             if ("Nimbus".equals(info.getName())) {
00230                 javax.swing.UIManager.setLookAndFeel(info.getClassName());
00231                 break;
00232             }
00233         } catch (ClassNotFoundException | InstantiationException | IllegalAccessException
00234             | javax.swing.UnsupportedLookAndFeel ex) {
00235             java.util.logging.Logger.getLogger(AdminLogin.class.getName()).log(java.util.logging.Level.SEVERE,
00236             null, ex);
00237         } // </editor-fold>
00238
00239         java.awt.EventQueue.invokeLater(() -> new AdminLogin().setVisible(true));
00240     }
00241
00242 // Variables declaration - do not modify
00243 // GEN-BEGIN:variables
00244 // Sonarqube rule java:S1450 must be ignored here as these variables are
00245 // auto-generated by the Form Editor and need to remain as instance variables.
00247 private javax.swing.JButton jButton1;
00249 private javax.swing.JButton jButton2;
00251 private javax.swing.JButton jButton3;
00253 private javax.swing.JLabel jLabel1;
00255 private javax.swing.JLabel jLabel3;
00257 private javax.swing.JPanel jPanel1;
00258 // End of variables declaration//GEN-END:variables
00259 }

```

## 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 import org.slf4j.Logger;
00013 import org.slf4j.LoggerFactory;
00014
00026 public class AdminProducts extends javax.swing.JFrame {
00027
00029     private final transient ProductService productService;
00030
00032     private static final Logger LOGGER = LoggerFactory.getLogger(AdminProducts.class);
00033
00035     private static final String PRIMARY_FONT = "Yu Gothic UI";
00037     private static final String SECONDARY_FONT = "Thonburi";
00038
00040     private static final String FIRST_COLUMN_HEADER = "ID";
00042     private static final String SECOND_COLUMN_HEADER = "Item Name";
00044     private static final String THIRD_COLUMN_HEADER = "Item Price";
00045
00047     private static final String UPDATE_STRING = "Update";
00048 }
```

```

00050 DefaultTableModel modelDrink;
00052 DefaultTableModel modelAppetizers;
00054 DefaultTableModel modelMainCourse;
00055
00057 String[] columnNames = { FIRST_COLUMN_HEADER, SECOND_COLUMN_HEADER, THIRD_COLUMN_HEADER };
00058
00060 Long selectedDrinkID;
00062 Long selectedAppetizerID;
00064 Long selectedMainCourseID;
00065
00073 public AdminProducts() {
00074     initComponents();
00075
00076     // Initialize the Service Layer.
00077     ApiClient client = new ApiClient("http://localhost:8080");
00078     this.productService = new ProductService(client);
00079
00080     // Initialize table models and set shared headers.
00081     modelDrink = new DefaultTableModel();
00082     modelDrink.setColumnIdentifiers(columnNames);
00083     modelAppetizers = new DefaultTableModel();
00084     modelAppetizers.setColumnIdentifiers(columnNames);
00085     modelMainCourse = new DefaultTableModel();
00086     modelMainCourse.setColumnIdentifiers(columnNames);
00087
00088     // Link models to tables.
00089     jTable1.setModel(modelDrink);
00090     jTable2.setModel(modelAppetizers);
00091     jTable3.setModel(modelMainCourse);
00092
00093     // Load initial data from backend.
00094     refreshAllTables();
00095 }
00096
00110 AdminProducts(ProductService productService, boolean startLoading) {
00111     initComponents();
00112
00113     // Use injected service instead of creating a new ApiClient.
00114     this.productService = productService;
00115
00116     // Initialize table models and set shared headers.
00117     modelDrink = new DefaultTableModel();
00118     modelDrink.setColumnIdentifiers(columnNames);
00119     modelAppetizers = new DefaultTableModel();
00120     modelAppetizers.setColumnIdentifiers(columnNames);
00121     modelMainCourse = new DefaultTableModel();
00122     modelMainCourse.setColumnIdentifiers(columnNames);
00123
00124     // Link models to tables.
00125     jTable1.setModel(modelDrink);
00126     jTable2.setModel(modelAppetizers);
00127     jTable3.setModel(modelMainCourse);
00128
00129     if (startLoading) {
00130         refreshAllTables();
00131     }
00132 }
00133
00140 private void refreshAllTables() {
00141     loadDrinks();
00142     loadAppetizer();
00143     loadMainCourse();
00144 }
00145
00153 void loadDrinks() {
00154     new Thread(() -> {
00155         try {
00156             List<Drink> drinks = productService.getAllDrinks();
00157             SwingUtilities.invokeLater(() -> {
00158                 modelDrink.setRowCount(0);
00159                 for (Drink drink : drinks) {
00160                     modelDrink.addRow(new Object[] {
00161                         drink.getDrinksId(),
00162                         drink.getItemDrinks(),
00163                         drink.getDrinksPrice()
00164                     });
00165                 }
00166             });
00167         } catch (Exception ex) {
00168             SwingUtilities
00169                 .invokeLater(() -> JOptionPane.showMessageDialog(null, "Error loading drinks: " +
00170                         ex.getMessage()));
00171         }
00172     }).start();
00173 }
00183 void loadDrinksSync() {

```

```

00184     try {
00185         List<Drink> drinks = productService.getAllDrinks();
00186         modelDrink.setRowCount(0);
00187         for (Drink drink : drinks) {
00188             modelDrink.addRow(new Object[] { drink.getDrinksId(), drink.getItemDrinks(),
00189                 drink.getDrinksPrice() });
00190         }
00191     } catch (Exception ex) {
00192         throw new IllegalStateException("Failed to load drinks", ex);
00193     }
00194 }
00195 void loadAppetizer() {
00196     new Thread(() -> {
00197         try {
00198             List<Appetizer> appetizers = productService.getAllAppetizers();
00199             SwingUtilities.invokeLater(() -> {
00200                 modelappetizers.setRowCount(0);
00201                 for (Appetizer appetizer : appetizers) {
00202                     modelappetizers.addRow(new Object[] {
00203                         appetizer.getAppetizersId(),
00204                         appetizer.getItemAppetizers(),
00205                         appetizer.getAppetizersPrice()
00206                     });
00207                 }
00208             });
00209         } catch (Exception ex) {
00210             SwingUtilities
00211                 .invokeLater(() -> JOptionPane.showMessageDialog(null, "Error loading appetizers: " +
00212                 ex.getMessage()));
00213         }
00214     }).start();
00215 }
00216 void loadAppetizerSync() {
00217     try {
00218         List<Appetizer> appetizers = productService.getAllAppetizers();
00219         modelappetizers.setRowCount(0);
00220         for (Appetizer appetizer : appetizers) {
00221             modelappetizers.addRow(new Object[] { appetizer.getAppetizersId(),
00222                 appetizer.getItemAppetizers(),
00223                 appetizer.getAppetizersPrice() });
00224         }
00225     } catch (Exception ex) {
00226         throw new IllegalStateException("Failed to load appetizers", ex);
00227     }
00228 }
00229 void loadmainCourse() {
00230     new Thread(() -> {
00231         try {
00232             List<MainCourse> mainCourses = productService.getAllMainCourses();
00233             SwingUtilities.invokeLater(() -> {
00234                 modelmaincourse.setRowCount(0);
00235                 for (MainCourse mainCourse : mainCourses) {
00236                     modelmaincourse.addRow(new Object[] {
00237                         mainCourse.getFoodId(),
00238                         mainCourse.getItemFood(),
00239                         mainCourse.getFoodPrice()
00240                     });
00241                 }
00242             });
00243         } catch (Exception ex) {
00244             SwingUtilities
00245                 .invokeLater(() -> JOptionPane.showMessageDialog(null, "Error loading main courses: " +
00246                 ex.getMessage()));
00247         }
00248     }).start();
00249 }
00250 void loadmainCourseSync() {
00251     try {
00252         List<MainCourse> mainCourses = productService.getAllMainCourses();
00253         modelmaincourse.setRowCount(0);
00254         for (MainCourse mainCourse : mainCourses) {
00255             modelmaincourse
00256                 .addRow(new Object[] { mainCourse.getFoodId(), mainCourse.getItemFood(),
00257                     mainCourse.getFoodPrice() });
00258         }
00259     } catch (Exception ex) {
00260         throw new IllegalStateException("Failed to load main courses", ex);
00261     }
00262 }
00263 void selectDrinkByRowSync(int row) {
00264     if (row < 0 || row >= modelDrink.getRowCount())
00265         return;
00266 }
```

```

00303     String idStr = modelDrink.getValueAt(row, 0).toString();
00304     Long id = Long.valueOf(idStr);
00305     Drink drink = productService.getDrinkById(id);
00306     itemname.setText(drink.getItemDrinks());
00307     itemprice.setText(String.valueOf(drink.getDrinksPrice()));
00308 }
00309
00320 void selectAppetizerByRowSync(int row) {
00321     if (row < 0 || row >= modelappetizers.getRowCount())
00322         return;
00323     String idStr = modelappetizers.getValueAt(row, 0).toString();
00324     Long id = Long.valueOf(idStr);
00325     Appetizer appetizer = productService.getAppetizerById(id);
00326     itemname1.setText(appetizer.getItemAppetizers());
00327     itemprice1.setText(String.valueOf(appetizer.getAppetizersPrice()));
00328 }
00329
00340 void selectMainCourseByRowSync(int row) {
00341     if (row < 0 || row >= modelmaincourse.getRowCount())
00342         return;
00343     String idStr = modelmaincourse.getValueAt(row, 0).toString();
00344     Long id = Long.valueOf(idStr);
00345     MainCourse mainCourse = productService.getMainCourseById(id);
00346     itemname2.setText(mainCourse.getItemFood());
00347     itemprice2.setText(String.valueOf(mainCourse.getFoodPrice()));
00348 }
00349
00358 @SuppressWarnings("unchecked")
00359 // <editor-fold defaultstate="collapsed" desc="Generated
00360 // Code">/GEN-BEGIN:initComponents
00361 private void initComponents() {
00362
00363     jPanell = new javax.swing.JPanel();
00364     jLabel1 = new javax.swing.JLabel();
00365     jTabbedPane1 = new javax.swing.JTabbedPane();
00366     jPanel12 = new javax.swing.JPanel();
00367     itemname = new javax.swing.JTextField();
00368     itemprice = new javax.swing.JTextField();
00369     jScrollPane1 = new javax.swing.JScrollPane();
00370     jTable1 = new javax.swing.JTable();
00371     jButton1 = new javax.swing.JButton();
00372     jButton2 = new javax.swing.JButton();
00373     jLabel12 = new javax.swing.JLabel();
00374     jLabel13 = new javax.swing.JLabel();
00375     jLabel14 = new javax.swing.JLabel();
00376     jPanel13 = new javax.swing.JPanel();
00377     jLabel15 = new javax.swing.JLabel();
00378     jLabel16 = new javax.swing.JLabel();
00379     jLabel17 = new javax.swing.JLabel();
00380     jScrollPane2 = new javax.swing.JScrollPane();
00381     jTable2 = new javax.swing.JTable();
00382     itemname1 = new javax.swing.JTextField();
00383     itemprice1 = new javax.swing.JTextField();
00384     jButton4 = new javax.swing.JButton();
00385     jButton5 = new javax.swing.JButton();
00386     jPanel14 = new javax.swing.JPanel();
00387     jLabel18 = new javax.swing.JLabel();
00388     jLabel19 = new javax.swing.JLabel();
00389     jLabel10 = new javax.swing.JLabel();
00390     jScrollPane3 = new javax.swing.JScrollPane();
00391     jTable3 = new javax.swing.JTable();
00392     itemname2 = new javax.swing.JTextField();
00393     itemprice2 = new javax.swing.JTextField();
00394     jButton6 = new javax.swing.JButton();
00395     jButton7 = new javax.swing.JButton();
00396     jButton3 = new javax.swing.JButton();
00397
00398     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00399     setTitle("Admin");
00400     setResizable(false);
00401
00402     jPanell.setBackground(new java.awt.Color(248, 244, 230));
00403
00404     jLabel11.setBackground(new java.awt.Color(255, 153, 0));
00405     jLabel11.setFont(new java.awt.Font(PRIMARY_FONT, 1, 36)); // NOI18N
00406     jLabel11.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00407     jLabel11.setText("Items Prices Update Portal");
00408
00409     jTabbedPane1.setBackground(new java.awt.Color(248, 244, 230));
00410     jTabbedPane1.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));
00411     jTabbedPane1.setCursor(new java.awt.Cursor(java.awt.Cursor.DEFAULT_CURSOR));
00412     jTabbedPane1.setFont(new java.awt.Font(PRIMARY_FONT, 1, 18)); // NOI18N
00413
00414     jPanel12.setBackground(new java.awt.Color(248, 244, 230));
00415
00416     itemname.setFont(new java.awt.Font(PRIMARY_FONT, 0, 18)); // NOI18N
00417     itemname.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);

```



```

00501         .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED,
00502                         javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
00503
00504     .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00505         .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 586,
00506                     javax.swing.GroupLayout.PREFERRED_SIZE)
00507         .addComponent(jLabel4, javax.swing.GroupLayout.PREFERRED_SIZE, 408,
00508                     javax.swing.GroupLayout.PREFERRED_SIZE))
00509     .addGroup(jPanel2Layout.createSequentialGroup()
00510         .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE, 1111,
00511                     javax.swing.GroupLayout.PREFERRED_SIZE)
00512         .addGap(18, 18, 18))))))
00513 jPanel2Layout.setVerticalGroup(
00514     jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00515         .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
00516             jPanel2Layout.createSequentialGroup()
00517                 .addComponent(jLabel2)
00518                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 21,
00519                             Short.MAX_VALUE)
00520             .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00521                 .addComponent(jLabel3)
00522                 .addComponent(jLabel4)
00523                 .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00524                     .addGroup(jPanel2Layout.createSequentialGroup()
00525                         .addComponent(itemname, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00526                         javax.swing.GroupLayout.PREFERRED_SIZE)
00527                         .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
00528                         .addComponent(itemprice, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00529                             javax.swing.GroupLayout.PREFERRED_SIZE)
00530                         .addGap(37, 37, 37)
00531
00532             .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00533                 .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00534                             javax.swing.GroupLayout.PREFERRED_SIZE)
00535                 .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00536                             javax.swing.GroupLayout.PREFERRED_SIZE)))
00537             .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 203,
00538                             javax.swing.GroupLayout.PREFERRED_SIZE))
00539             .addGap(48, 48, 48));
00540
00541 jPanel1.addTab("Drinks", jPanel2);
00542
00543 jPanel3.setBackground(new java.awt.Color(248, 244, 230));
00544
00545 jLabel5.setFont(new java.awt.Font(PRIMARY_FONT, 0, 14)); // NOI18N
00546 jLabel5.setText("Enter new appetizer information carefully to add.");
00547
00548 jLabel6.setBackground(new java.awt.Color(255, 153, 0));
00549 jLabel6.setFont(new java.awt.Font(PRIMARY_FONT, 1, 25)); // NOI18N
00550 jLabel6.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00551 jLabel6.setText("Available Appetizer");
00552
00553 jLabel7.setFont(new java.awt.Font(PRIMARY_FONT, 0, 14)); // NOI18N
00554 jLabel7.setText("Please select the appetizer to update the price.");
00555
00556 jTable2.setFont(new java.awt.Font(PRIMARY_FONT, 0, 14)); // NOI18N
00557 jTable2.setModel(new javax.swing.table.DefaultTableModel(
00558     new Object[][] {
00559         {},
00560         {},
00561         {}
00562     },
00563     new String[] {
00564
00565     }));
00566 jTable2.addMouseListener(new java.awt.event.MouseAdapter() {
00567     @Override
00568     public void mouseClicked(java.awt.event.MouseEvent evt) {
00569         jTable2MouseClicked(evt);
00570     }
00571 });
00572 jScrollPane2.setViewportView(jTable2);
00573
00574 itemname1.setFont(new java.awt.Font(PRIMARY_FONT, 0, 18)); // NOI18N
00575 itemname1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00576 itemname1.setBorder(javax.swing.BorderFactory.createTitledBorder(
00577     new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), SECOND_COLUMN_HEADER,
00578     javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00579     new java.awt.Font(PRIMARY_FONT, 0, 18))); // NOI18N
00580
00581 itemprice1.setFont(new java.awt.Font(PRIMARY_FONT, 0, 18)); // NOI18N
00582 itemprice1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);

```

```

00583     itemprice1.setBorder(javax.swing.BorderFactory.createTitledBorder(
00584         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), THIRD_COLUMN_HEADER,
00585         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00586         new java.awt.Font(PRIMARY_FONT, 0, 18))); // NOI18N
00587
00588     jButton4.setBackground(new java.awt.Color(255, 255, 255));
00589     jButton4.setFont(new java.awt.Font(SECONDARY_FONT, 0, 18)); // NOI18N
00590     jButton4.setText(UPDATE_STRING);
00591     jButton4.addActionListener(this::jButton4ActionPerformed);
00592
00593     jButton5.setBackground(new java.awt.Color(255, 255, 255));
00594     jButton5.setFont(new java.awt.Font(SECONDARY_FONT, 0, 18)); // NOI18N
00595     jButton5.setText("Add");
00596     jButton5.addActionListener(this::jButton5ActionPerformed);
00597
00598     javax.swing.GroupLayout jPanel3Layout = new javax.swing.GroupLayout(jPanel3);
00599     jPanel3.setLayout(jPanel3Layout);
00600     jPanel3Layout.setHorizontalGroup(
00601         jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00602             .addGroup(jPanel3Layout.createSequentialGroup()
00603                 .addGap(27, 27, Short.MAX_VALUE)
00604                 .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00605                     .addGroup(jPanel3Layout.createSequentialGroup()
00606                         .addGap(24, 24, 24)
00607                         .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00608                             .addGroup(jPanel3Layout.createSequentialGroup()
00609                                 .addGap(8, 8, 8)
00610                                 .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00611                                     .addComponent(jButton5, javax.swing.GroupLayout.PREFERRED_SIZE, 91,
00612                                         javax.swing.GroupLayout.PREFERRED_SIZE)
00613                                     .addComponent(jButton4, javax.swing.GroupLayout.PREFERRED_SIZE, 103,
00614                                         javax.swing.GroupLayout.PREFERRED_SIZE)
00615                                     .addGroup(jPanel3Layout.createSequentialGroup()
00616                                         .addComponent(itemprice1)
00617                                         .addComponent(itemname1)
00618                                         .addComponent(jLabel5, javax.swing.GroupLayout.PREFERRED_SIZE,
00619                                             javax.swing.GroupLayout.PREFERRED_SIZE, Short.MAX_VALUE)
00620                                         .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
00621                                             .addComponent(jLabel5, javax.swing.GroupLayout.PREFERRED_SIZE, Short.MAX_VALUE)
00622
00623                                         .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00624                                             .addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED_SIZE, 586,
00625                                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00626                                             .addComponent(jLabel7, javax.swing.GroupLayout.PREFERRED_SIZE, 408,
00627                                                 javax.swing.GroupLayout.PREFERRED_SIZE))
00628                                         .addGroup(jPanel3Layout.createSequentialGroup()
00629                                             .addComponent(jLabel6, javax.swing.GroupLayout.PREFERRED_SIZE, 1111,
00630                                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00631                                             .addGap(18, 18, 18)))
00632                                         .addGap(115, 115, 115))
00633                                         .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00634                                             .addGroup(jPanel3Layout.createSequentialGroup()
00635                                                 .addGap(21, 21, Short.MAX_VALUE)
00636                                                 .addComponent(jLabel6)
00637                                                 .addGap(37, 37, 37)
00638
00639                                         .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00640                                             .addComponent(jLabel5)
00641                                             .addComponent(jLabel7)
00642                                             .addComponent(jLabel15, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00643                                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00644                                             .addComponent(itemname1, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00645                                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00646                                             .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00647                                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00648                                             .addGap(37, 37, 37)
00649                                         .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00650                                             .addComponent(jButton4, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00651                                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00652                                             .addComponent(jButton5, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00653                                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00654                                             .addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED_SIZE, 203,
00655                                                 javax.swing.GroupLayout.PREFERRED_SIZE))
00656                                         .addGap(48, 48, 48)))
00657                                         .addGap(48, 48, 48)));
00658
00659     jTabbedPane1.addTab("Appetizers", jPanel3);
00660
00661     jPanel4.setBackground(new java.awt.Color(248, 244, 230));
00662     jPanel4.setFont(new java.awt.Font(PRIMARY_FONT, 0, 18)); // NOI18N

```

```
00664 jLabel8.setBackground(new java.awt.Color(248, 244, 230));  
00665 jLabel8.setFont(new java.awt.Font(PRIMARY_FONT, 0, 14)); // NOI18N  
00666 jLabel8.setText("Enter new item information carefully to add.");  
00667  
00668 jLabel9.setBackground(new java.awt.Color(255, 153, 0));  
00669 jLabel9.setFont(new java.awt.Font(PRIMARY_FONT, 1, 25)); // NOI18N  
00670 jLabel9.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);  
00671 jLabel9.setText("Available MainCourse");  
00672  
00673 jLabel10.setFont(new java.awt.Font(PRIMARY_FONT, 0, 14)); // NOI18N  
00674 jLabel10.setText("Please select the item to update the price.");  
00675  
00676 jTable3.setFont(new java.awt.Font(PRIMARY_FONT, 0, 14)); // NOI18N  
00677 jTable3.setModel(new javax.swing.table.DefaultTableModel(  
00678     new Object[][] {  
00679         {},  
00680         {},  
00681         {},  
00682         {}  
00683     },  
00684     new String[] {  
00685         ""});  
00686 jTable3.addMouseListener(new java.awt.event.MouseAdapter() {  
00687     @Override  
00688     public void mouseClicked(java.awt.event.MouseEvent evt) {  
00689         jTable3MouseClicked(evt);  
00690     }  
00691 };  
00692 );  
00693 jScrollPane3.setViewportView(jTable3);  
00694  
00695 itemname2.setFont(new java.awt.Font(PRIMARY_FONT, 0, 18)); // NOI18N  
00696 itemname2.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);  
00697 itemname2.setBorderStyle(javax.swing.BorderFactory.createTitledBorder(  
00698     new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), SECOND_COLUMN_HEADER,  
00699     javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,  
00700     new java.awt.Font(PRIMARY_FONT, 0, 18))); // NOI18N  
00701  
00702 itemprice2.setFont(new java.awt.Font(PRIMARY_FONT, 0, 18)); // NOI18N  
00703 itemprice2.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);  
00704 itemprice2.setBorderStyle(javax.swing.BorderFactory.createTitledBorder(  
00705     new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), THIRD_COLUMN_HEADER,  
00706     javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,  
00707     new java.awt.Font(PRIMARY_FONT, 0, 18))); // NOI18N  
00708  
00709 jButton6.setBackground(new java.awt.Color(255, 255, 255));  
00710 jButton6.setFont(new java.awt.Font(SECONDARY_FONT, 0, 18)); // NOI18N  
00711 jButton6.setText(UPDATE_STRING);  
00712 jButton6.addActionListener(this::jButton6ActionPerformed);  
00713  
00714 jButton7.setBackground(new java.awt.Color(255, 255, 255));  
00715 jButton7.setFont(new java.awt.Font(SECONDARY_FONT, 0, 18)); // NOI18N  
00716 jButton7.setText("Add");  
00717 jButton7.addActionListener(this::jButton7ActionPerformed);  
00718  
00719 javax.swing.GroupLayout jPanel4Layout = new javax.swing.GroupLayout(jPanel4);  
00720 jPanel4.setLayout(jPanel4Layout);  
00721 jPanel4Layout.setHorizontalGroup(  
00722     jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
00723         .addGroup(jPanel4Layout.createSequentialGroup()  
00724             .createSequentialGroup()  
00725                 .addGap(0, 27, Short.MAX_VALUE)  
00726                 .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
00727                     .addGroup(jPanel4Layout.createSequentialGroup()  
00728                         .addGroup(jPanel4Layout.createSequentialGroup()  
00729                             .addGroup(jPanel4Layout.createSequentialGroup()  
00730                                 .addComponent(jButton7, javax.swing.GroupLayout.PREFERRED_SIZE, 91,  
00731                                     javax.swing.GroupLayout.PREFERRED_SIZE)  
00732                                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,  
00733                                     Short.MAX_VALUE)  
00734                                 .addComponent(jButton6, javax.swing.GroupLayout.PREFERRED_SIZE, 103,  
00735                                     javax.swing.GroupLayout.PREFERRED_SIZE)  
00736                                 .addGap(8, 8, 8))  
00737                                 .addComponent(itemprice2)  
00738                                 .addComponent(itemname2, javax.swing.GroupLayout.DEFAULT_SIZE, 311,  
00739                                     Short.MAX_VALUE)  
00740                                 .addComponent(jLabel8, javax.swing.GroupLayout.DEFAULT_SIZE,  
00741                                     javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))  
00742                                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED,  
00743                                     javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)  
00744                                 .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
00745                                     .addComponent(jScrollPane3, javax.swing.GroupLayout.PREFERRED_SIZE, 586,
```

```

00745             javax.swing.GroupLayout.PREFERRED_SIZE)
00746             .addComponent(jLabel10, javax.swing.GroupLayout.PREFERRED_SIZE, 408,
00747                         javax.swing.GroupLayout.PREFERRED_SIZE))
00748             .addGap(115, 115, 115))
00749             .addGroup(jPanel4Layout.createSequentialGroup()
00750                 .addComponent(jLabel19, javax.swing.GroupLayout.PREFERRED_SIZE, 1111,
00751                             javax.swing.GroupLayout.PREFERRED_SIZE)
00752                 .addGap(18, 18, 18))));}
00753     jPanel4Layout.setVerticalGroup(
00754         jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00755             .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
00756                 jPanel4Layout.createSequentialGroup()
00757                     .addComponent(jLabel19)
00758                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 21,
00759                         Short.MAX_VALUE)
00760                     .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00761                         .addComponent(jLabel18)
00762                         .addComponent(jLabel10))
00763                         .addGap(18, 18, 18)
00764                         .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00765                             .addGroup(jPanel4Layout.createSequentialGroup()
00766                                 .addComponent(itemname2, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00767                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00768                                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
00769                                 .addComponent(itemprice2, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00770                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00771                                 .addGap(37, 37, 37)
00772                         .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00773                             .addComponent(jButton6, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00774                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00775                             .addComponent(jButton7, javax.swing.GroupLayout.PREFERRED_SIZE, 42,
00776                                 javax.swing.GroupLayout.PREFERRED_SIZE)))
00777                         .addComponent(jScrollPane3, javax.swing.GroupLayout.PREFERRED_SIZE, 203,
00778                             javax.swing.GroupLayout.PREFERRED_SIZE))
00779                         .addGap(48, 48, 48));
00780     jTabbedPane.addTab("MainCourse", jPanel4);
00781
00782     jButton3.setBackground(new java.awt.Color(255, 255, 255));
00783     jButton3.setFont(new java.awt.Font(PRIMARY_FONT, 1, 18)); // NOI18N
00784     jButton3.setText("Go Back");
00785     jButton3.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));
00786     jButton3.addActionListener(this::jButton3ActionPerformed);
00787
00788     javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
00789     jPanel1.setLayout(jPanel1Layout);
00790     jPanel1Layout.setHorizontalGroup(
00791         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00792             .addGroup(jPanel1Layout.createSequentialGroup()
00793                 .addGroup(jPanel1Layout.createSequentialGroup()
00794                     .addGroup(jPanel1Layout.createSequentialGroup()
00795                         .addGroup(jPanel1Layout.createSequentialGroup()
00796                             .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 122,
00797                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00798                             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
00799                                 javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
00800                             .addComponent(jLabel11, javax.swing.GroupLayout.PREFERRED_SIZE, 689,
00801                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00802                             .addGap(218, 218, 218)
00803                             .addGroup(jPanel1Layout.createSequentialGroup()
00804                                 .addComponent(jTabbedPane, javax.swing.GroupLayout.PREFERRED_SIZE,
00805                                     javax.swing.GroupLayout.DEFAULT_SIZE,
00806                                     javax.swing.GroupLayout.PREFERRED_SIZE)
00807                                     .addContainerGap(29, Short.MAX_VALUE)))));
00808         jPanel1Layout.setVerticalGroup(
00809             jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00810                 .addGroup(jPanel1Layout.createSequentialGroup()
00811                     .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
00812                         .addComponent(jLabel11, javax.swing.GroupLayout.PREFERRED_SIZE, 60,
00813                             javax.swing.GroupLayout.PREFERRED_SIZE)
00814                         .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 37,
00815                             javax.swing.GroupLayout.PREFERRED_SIZE))
00816                         .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED,
00817                             javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
00818                         .addComponent(jTabbedPane, javax.swing.GroupLayout.PREFERRED_SIZE, 392,
00819                             javax.swing.GroupLayout.PREFERRED_SIZE)
00820                         .addGap(29, 29, 29)));
00821
00822     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00823     getContentPane().setLayout(layout);
00824     layout.setHorizontalGroup(
00825         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

```

```

00826         .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
00827             javax.swing.GroupLayout.DEFAULT_SIZE,
00828             Short.MAX_VALUE));
00829     layout.setVerticalGroup(
00830         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00831             .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
00832                 .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
00833                     javax.swing.GroupLayout.PREFERRED_SIZE)
00834                 .addGap(0, 0, Short.MAX_VALUE)));
00835     pack();
00836     setLocationRelativeTo(null);
00837 } // </editor-fold> //GEN-END:initComponents
00838
00849 private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {//
00850     GEN-FIRST:event_jButton2ActionPerformed
00851     String name = itemname.getText();
00852     String price = itemprice.getText();
00853
00854     new Thread(() -> {
00855         try {
00856             productService.addDrink(name, price);
00857             SwingUtilities.invokeLater(() -> {
00858                 JOptionPane.showMessageDialog(this, "Drink Added Successfully.");
00859                 itemname.setText("");
00860                 itemprice.setText("");
00861                 loadDrinks();
00862             });
00863         } catch (Exception ex) {
00864             SwingUtilities.invokeLater(() -> JOptionPane.showMessageDialog(this, "Error adding drink: " +
00865                 ex.getMessage()));
00866         }
00867     }).start();
00868 } // GEN-LAST:event_jButton2ActionPerformed
00869
00876 private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {//
00877     GEN-FIRST:event_jButton3ActionPerformed
00878     new AdminLogin().setVisible(true);
00879     this.dispose();
00880 } // GEN-LAST:event_jButton3ActionPerformed
00881
00889 private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {//
00890     GEN-FIRST:event_jButton5ActionPerformed
00891     String name = itemnamel.getText();
00892     String price = itempricel.getText();
00893
00894     new Thread(() -> {
00895         try {
00896             productService.addAppetizer(name, price);
00897             SwingUtilities.invokeLater(() -> {
00898                 JOptionPane.showMessageDialog(this, "Appetizer Added Successfully.");
00899                 itemnamel.setText("");
00900                 itempricel.setText("");
00901                 loadAppetizer();
00902             });
00903         } catch (Exception ex) {
00904             SwingUtilities
00905                 .invokeLater(() -> JOptionPane.showMessageDialog(this, "Error adding appetizer: " +
00906                 ex.getMessage()));
00907         }
00908     }).start();
00909 } // GEN-LAST:event_jButton5ActionPerformed
00910
00917 private void jButton7ActionPerformed(java.awt.event.ActionEvent evt) {//
00918     GEN-FIRST:event_jButton7ActionPerformed
00919     String name = itemname2.getText();
00920     String price = itemprice2.getText();
00921
00922     new Thread(() -> {
00923         try {
00924             productService.addMainCourse(name, price);
00925             SwingUtilities.invokeLater(() -> {
00926                 JOptionPane.showMessageDialog(this, "Main Course Added Successfully.");
00927                 itemname2.setText("");
00928                 itemprice2.setText("");
00929                 loadmainCourse();
00930             });
00931         } catch (Exception ex) {
00932             SwingUtilities
00933                 .invokeLater(() -> JOptionPane.showMessageDialog(this, "Error adding main course: " +
00934                 ex.getMessage()));
00935         }
00936     }).start();
00937 } // GEN-LAST:event_jButton7ActionPerformed
00938
00945 private void jTable1KeyPressed(java.awt.event.KeyEvent evt) {//
00946     GEN-FIRST:event_jTable1KeyPressed

```

```

00946     // No action needed for key press in this version.
00947 } // GEN-LAST:event_jTable1KeyPressed
00948
00959 private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {//
GEN-FIRST:event_jTable1MouseClicked
00960     int row = jTable1.getSelectedRow();
00961     if (row == -1)
00962         return;
00963
00964     String idStr = jTable1.getValueAt(row, 0).toString();
00965     selectedDrinkID = Long.valueOf(idStr);
00966     LOGGER.info("Selected Drink ID: {}", selectedDrinkID);
00967
00968     new Thread(() -> {
00969         try {
00970             Drink drink = productService.getDrinkById(selectedDrinkID);
00971             SwingUtilities.invokeLater(() -> {
00972                 itemname.setText(drink.getItemDrinks());
00973                 itemprice.setText(String.valueOf(drink.getDrinksPrice()));
00974             });
00975         } catch (Exception ex) {
00976             SwingUtilities
00977                 .invokeLater(() -> JOptionPane.showMessageDialog(null, "Error loading drink details: " +
ex.getMessage()));
00978         }
00979     }).start();
00980 } // GEN-LAST:event_jTable1MouseClicked
00981
00990 private void jTable2MouseClicked(java.awt.event.MouseEvent evt) {//
GEN-FIRST:event_jTable2MouseClicked
00991     int row = jTable2.getSelectedRow();
00992     if (row == -1)
00993         return;
00994
00995     String idStr = jTable2.getValueAt(row, 0).toString();
00996     selectedAppetizerID = Long.valueOf(idStr);
00997     LOGGER.info("Selected Appetizer ID: {}", selectedAppetizerID);
00998
00999     new Thread(() -> {
01000         try {
01001             Appetizer appetizer = productService.getAppetizerById(selectedAppetizerID);
01002             SwingUtilities.invokeLater(() -> {
01003                 itemname1.setText(appetizer.getItemAppetizers());
01004                 itemprice1.setText(String.valueOf(appetizer.getAppetizersPrice()));
01005             });
01006         } catch (Exception ex) {
01007             SwingUtilities.invokeLater(
01008                 () -> JOptionPane.showMessageDialog(null, "Error loading appetizer details: " +
ex.getMessage()));
01009         }
01010     }).start();
01011 } // GEN-LAST:event_jTable2MouseClicked
01012
01021 private void jTable3MouseClicked(java.awt.event.MouseEvent evt) {//
GEN-FIRST:event_jTable3MouseClicked
01022     int row = jTable3.getSelectedRow();
01023     if (row == -1)
01024         return;
01025
01026     String idStr = jTable3.getValueAt(row, 0).toString();
01027     selectedMainCourseID = Long.valueOf(idStr);
01028     LOGGER.info("Selected Main Course ID: {}", selectedMainCourseID);
01029
01030     new Thread(() -> {
01031         try {
01032             MainCourse mainCourse = productService.getMainCourseById(selectedMainCourseID);
01033             SwingUtilities.invokeLater(() -> {
01034                 itemname2.setText(mainCourse.getItemFood());
01035                 itemprice2.setText(String.valueOf(mainCourse.getFoodPrice()));
01036             });
01037         } catch (Exception ex) {
01038             SwingUtilities.invokeLater(
01039                 () -> JOptionPane.showMessageDialog(null, "Error loading main course details: " +
ex.getMessage()));
01040         }
01041     }).start();
01042 } // GEN-LAST:event_jTable3MouseClicked
01043
01052 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//
GEN-FIRST:event_jButton1ActionPerformed
01053     String name = itemname.getText();
01054     String price = itemprice.getText();
01055
01056     new Thread(() -> {
01057         try {
01058             productService.updateDrink(selectedDrinkID, name, price);
01059             SwingUtilities.invokeLater(() -> {

```

```

01060         JOptionPane.showMessageDialog(this, "Drink updated successfully.");
01061         itemname.setText("");
01062         itemprice.setText("");
01063         selectedDrinkID = null;
01064         loadDrinks();
01065     });
01066 } catch (Exception ex) {
01067     SwingUtilities
01068         .invokeLater(() -> JOptionPane.showMessageDialog(this, "Error updating drink: " +
01069             ex.getMessage()));
01070 }
01071 }).start();
01072 // GEN-LAST:event_jButton1ActionPerformed
01073
01081 private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {//
01082     GEN-FIRST:event_jButton4ActionPerformed
01083     String name = itemname1.getText();
01084     String price = itemprice1.getText();
01085
01086     new Thread(() -> {
01087         try {
01088             productService.updateAppetizer(selectedAppetizerID, name, price);
01089             SwingUtilities.invokeLater(() -> {
01090                 JOptionPane.showMessageDialog(this, "Appetizer updated successfully.");
01091                 itemname1.setText("");
01092                 itemprice1.setText("");
01093                 selectedAppetizerID = null;
01094                 loadAppetizer();
01095             });
01096         } catch (Exception ex) {
01097             SwingUtilities
01098                 .invokeLater(() -> JOptionPane.showMessageDialog(this, "Error updating appetizer: " +
01099                     ex.getMessage()));
01100         }
01101     }).start();
01102 // GEN-LAST:event_jButton4ActionPerformed
01103
01104 private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {//
01105     GEN-FIRST:event_jButton6ActionPerformed
01106     String name = itemname2.getText();
01107     String price = itemprice2.getText();
01108
01109     new Thread(() -> {
01110         try {
01111             productService.updateMainCourse(selectedMainCourseID, name, price);
01112             SwingUtilities.invokeLater(() -> {
01113                 JOptionPane.showMessageDialog(this, "Main course updated successfully.");
01114                 itemname2.setText("");
01115                 itemprice2.setText("");
01116                 selectedMainCourseID = null;
01117                 loadmainCourse();
01118             });
01119         } catch (Exception ex) {
01120             SwingUtilities
01121                 .invokeLater(() -> JOptionPane.showMessageDialog(this, "Error updating main course: " +
01122                     ex.getMessage()));
01123         }
01124     }).start();
01125 // GEN-LAST:event_jButton6ActionPerformed
01126
01140 public static void main(String[] args) {
01141     try {
01142         for (javax.swing.UIManager.LookAndFeelInfo info :
01143             javax.swing.UIManager.getInstalledLookAndFeels()) {
01144             if ("Nimbus".equals(info.getName())) {
01145                 javax.swing.UIManager.setLookAndFeel(info.getClassName());
01146                 break;
01147             }
01148         } catch (ClassNotFoundException | InstantiationException | IllegalAccessException
01149             | javax.swing.UnsupportedLookAndFeel ex) {
01150             java.util.logging.Logger.getLogger(AdminProducts.class.getName()).log(java.util.logging.Level.SEVERE,
01151                 null, ex);
01152         // </editor-fold>
01153
01154         /* Create and display the form */
01155         java.awt.EventQueue.invokeLater(() -> new AdminProducts().setVisible(true));
01156     }
01157
01158 // Variables declaration - do not modify//GEN-BEGIN:variables
01159 // Sonarqube rule java:S1450 must be ignored here as these variables are
01160 // auto-generated by the Form Editor and need to remain as instance variables.
01161 javax.swing.JTextField itemname;
01162 javax.swing.JTextField itemname1;
01163 javax.swing.JTextField itemname2;
01164
01165
01166
01167
01168
01169
01170
01171
01172
01173
01174
01175
01176
01177
01178
01179
01180
01181
01182
01183
01184
01185
01186
01187
01188
01189
01190
01191
01192
01193
01194
01195
01196
01197
01198
01199
01200
01201
01202
01203
01204
01205
01206
01207
01208
01209
01210
01211
01212
01213
01214
01215
01216
01217
01218
01219
01220
01221
01222
01223
01224
01225
01226
01227
01228
01229
01230
01231
01232
01233
01234
01235
01236
01237
01238
01239
01240
01241
01242
01243
01244
01245
01246
01247
01248
01249
01250
01251
01252
01253
01254
01255
01256
01257
01258
01259
01260
01261
01262
01263
01264
01265
01266
01267
01268
01269
01270
01271
01272
01273
01274
01275
01276
01277
01278
01279
01280
01281
01282
01283
01284
01285
01286
01287
01288
01289
01290
01291
01292
01293
01294
01295
01296
01297
01298
01299
01300
01301
01302
01303
01304
01305
01306
01307
01308
01309
01310
01311
01312
01313
01314
01315
01316
01317
01318
01319
01320
01321
01322
01323
01324
01325
01326
01327
01328
01329
01330
01331
01332
01333
01334
01335
01336
01337
01338
01339
01340
01341
01342
01343
01344
01345
01346
01347
01348
01349
01350
01351
01352
01353
01354
01355
01356
01357
01358
01359
01360
01361
01362
01363
01364
01365
01366
01367
01368
01369
01370
01371
01372
01373
01374
01375
01376
01377
01378
01379
01380
01381
01382
01383
01384
01385
01386
01387
01388
01389
01390
01391
01392
01393
01394
01395
01396
01397
01398
01399
01400
01401
01402
01403
01404
01405
01406
01407
01408
01409
01410
01411
01412
01413
01414
01415
01416
01417
01418
01419
01420
01421
01422
01423
01424
01425
01426
01427
01428
01429
01430
01431
01432
01433
01434
01435
01436
01437
01438
01439
01440
01441
01442
01443
01444
01445
01446
01447
01448
01449
01450
01451
01452
01453
01454
01455
01456
01457
01458
01459
01460
01461
01462
01463
01464
01465
01466
01467
01468
01469
01470
01471
01472
01473
01474
01475
01476
01477
01478
01479
01480
01481
01482
01483
01484
01485
01486
01487
01488
01489
01490
01491
01492
01493
01494
01495
01496
01497
01498
01499
01500
01501
01502
01503
01504
01505
01506
01507
01508
01509
01510
01511
01512
01513
01514
01515
01516
01517
01518
01519
01520
01521
01522
01523
01524
01525
01526
01527
01528
01529
01530
01531
01532
01533
01534
01535
01536
01537
01538
01539
01540
01541
01542
01543
01544
01545
01546
01547
01548
01549
01550
01551
01552
01553
01554
01555
01556
01557
01558
01559
01560
01561
01562
01563
01564
01565
01566
01567
01568
01569
01570
01571
01572
01573
01574
01575
01576
01577
01578
01579
01580
01581
01582
01583
01584
01585
01586
01587
01588
01589
01590
01591
01592
01593
01594
01595
01596
01597
01598
01599
01600
01601
01602
01603
01604
01605
01606
01607
01608
01609
01610
01611
01612
01613
01614
01615
01616
01617
01618
01619
01620
01621
01622
01623
01624
01625
01626
01627
01628
01629
01630
01631
01632
01633
01634
01635
01636
01637
01638
01639
01640
01641
01642
01643
01644
01645
01646
01647
01648
01649
01650
01651
01652
01653
01654
01655
01656
01657
01658
01659
01660
01661
01662
01663
01664
01665
01666
01667
01668
01669
01670
01671
01672
01673
01674
01675
01676
01677
01678
01679
01680
01681
01682
01683
01684
01685
01686
01687
01688
01689
01690
01691
01692
01693
01694
01695
01696
01697
01698
01699
01700
01701
01702
01703
01704
01705
01706
01707
01708
01709
01710
01711
01712
01713
01714
01715
01716
01717
01718
01719
01720
01721
01722
01723
01724
01725
01726
01727
01728
01729
01730
01731
01732
01733
01734
01735
01736
01737
01738
01739
01740
01741
01742
01743
01744
01745
01746
01747
01748
01749
01750
01751
01752
01753
01754
01755
01756
01757
01758
01759
01760
01761
01762
01763
01764
01765
01766
01767
01768
01769
01770
01771
01772
01773
01774
01775
01776
01777
01778
01779
01780
01781
01782
01783
01784
01785
01786
01787
01788
01789
01790
01791
01792
01793
01794
01795
01796
01797
01798
01799
01800
01801
01802
01803
01804
01805
01806
01807
01808
01809
01810
01811
01812
01813
01814
01815
01816
01817
01818
01819
01820
01821
01822
01823
01824
01825
01826
01827
01828
01829
01830
01831
01832
01833
01834
01835
01836
01837
01838
01839
01840
01841
01842
01843
01844
01845
01846
01847
01848
01849
01850
01851
01852
01853
01854
01855
01856
01857
01858
01859
01860
01861
01862
01863
01864
01865
01866
01867
01868
01869
01870
01871
01872
01873
01874
01875
01876
01877
01878
01879
01880
01881
01882
01883
01884
01885
01886
01887
01888
01889
01890
01891
01892
01893
01894
01895
01896
01897
01898
01899
01900
01901
01902
01903
01904
01905
01906
01907
01908
01909
01910
01911
01912
01913
01914
01915
01916
01917
01918
01919
01920
01921
01922
01923
01924
01925
01926
01927
01928
01929
01930
01931
01932
01933
01934
01935
01936
01937
01938
01939
01940
01941
01942
01943
01944
01945
01946
01947
01948
01949
01950
01951
01952
01953
01954
01955
01956
01957
01958
01959
01960
01961
01962
01963
01964
01965
01966
01967
01968
01969
01970
01971
01972
01973
01974
01975
01976
01977
01978
01979
01980
01981
01982
01983
01984
01985
01986
01987
01988
01989
01990
01991
01992
01993
01994
01995
01996
01997
01998
01999
01999
02000
02001
02002
02003
02004
02005
02006
02007
02008
02009
020010
020011
020012
020013
020014
020015
020016
020017
020018
020019
020020
020021
020022
020023
020024
020025
020026
020027
020028
020029
020030
020031
020032
020033
020034
020035
020036
020037
020038
020039
020040
020041
020042
020043
020044
020045
020046
020047
020048
020049
020050
020051
020052
020053
020054
020055
020056
020057
020058
020059
020060
020061
020062
020063
020064
020065
020066
020067
020068
020069
020070
020071
020072
020073
020074
020075
020076
020077
020078
020079
020080
020081
020082
020083
020084
020085
020086
020087
020088
020089
020090
020091
020092
020093
020094
020095
020096
020097
020098
020099
0200100
0200101
0200102
0200103
0200104
0200105
0200106
0200107
0200108
0200109
0200110
0200111
0200112
0200113
0200114
0200115
0200116
0200117
0200118
0200119
0200120
0200121
0200122
0200123
0200124
0200125
0200126
0200127
0200128
0200129
0200130
0200131
0200132
0200133
0200134
0200135
0200136
0200137
0200138
0200139
0200140
0200141
0200142
0200143
0200144
0200145
0200146
0200147
0200148
0200149
0200150
0200151
0200152
0200153
0200154
0200155
0200156
0200157
0200158
0200159
0200160
0200161
0200162
0200163
0200164
0200165
0200166
0200167
0200168
0200169
0200170
0200171
0200172
0200173
0200174
0200175
0200176
0200177
0200178
0200179
0200180
0200181
0200182
0200183
0200184
0200185
0200186
0200187
0200188
0200189
0200190
0200191
0200192
0200193
0200194
0200195
0200196
0200197
0200198
0200199
0200200
0200201
0200202
0200203
0200204
0200205
0200206
0200207
0200208
0200209
0200210
0200211
0200212
0200213
0200214
0200215
0200216
0200217
0200218
0200219
0200220
0200221
0200222
0200223
0200224
0200225
0200226
0200227
0200228
0200229
0200230
0200231
0200232
0200233
0200234
0200235
0200236
0200237
0200238
0200239
0200240
0200241
0200242
0200243
0200244
0200245
0200246
0200247
0200248
0200249
0200250
0200251
0200252
0200253
0200254
0200255
0200256
0200257
0200258
0200259
0200260
0200261
0200262
0200263
0200264
0200265
0200266
0200267
0200268
0200269
0200270
0200271
0200272
0200273
0200274
0200275
0200276
0200277
0200278
0200279
0200280
0200281
0200282
0200283
0200284
0200285
0200286
0200287
0200288
0200289
0200290
0200291
0200292
0200293
0200294
0200295
0200296
0200297
0200298
0200299
0200300
0200301
0200302
0200303
0200304
0200305
0200306
0200307
0200308
0200309
0200310
0200311
0200312
0200313
0200314
0200315
0200316
0200317
0200318
0200319
0200320
0200321
0200322
0200323
0200324
0200325
0200326
0200327
0200328
0200329
0200330
0200331
0200332
0200333
0200334
0200335
0200336
0200337
0200338
0200339
0200340
0200341
0200342
0200343
0200344
0200345
0200346
0200347
0200348
0200349
0200350
0200351
0200352
0200353
0200354
0200355
0200356
0200357
0200358
0200359
0200360
0200361
0200362
0200363
0200364
0200365
0200366
0200367
0200368
0200369
0200370
0200371
0200372
0200373
0200374
0200375
0200376
0200377
0200378
0200379
0200380
0200381
0200382
0200383
0200384
0200385
0200386
0200387
0200388
0200389
0200390
0200391
0200392
0200393
0200394
0200395
0200396
0200397
0200398
0200399
0200400
0200401
0200402
0200403
0200404
0200405
0200406
0200407
0200408
0200409
0200410
0200411
0200412
0200413
0200414
0200415
0200416
0200417
0200418
0200419
0200420
0200421
0200422
0200423
0200424
0200425
0200426
0200427
0200428
0200429
0200430
0200431
0200432
0200433
0200434
0200435
0200436
0200437
0200438
0200439
0200440
0200441
0200442
0200443
0200444
0200445
0200446
0200447
0200448
0200449
0200450
0200451
0200452
0200453
0200454
0200455
0200456
0200457
0200458
0200459
0200460
0200461
0200462
0200463
0200464
0200465
0200466
0200467
0200468
0200469
0200470
0200471
0200472
0200473
0200474
0200475
0200476
0200477
0200478
0200479
0200480
0200481
0200482
0200483
0200484
0200485
0200486
0200487
0200488
0200489
0200490
0200491
0200492
0200493
0200494
0200495
0200496
0200497
0200498
0200499
0200500
0200501
0200502
0200503
0200504
0200505
0200506
0200507
0200508
0200509
0200510
0200511
0200512
0200513
0200514
0200515
0200516
0200517
0200518
0200519
0200520
0200521
0200522
0200523
0200524
0200525
0200526
0200527
0200528
0200529
0200530
0200531
0200532
0200533
0200534
0200535
0200536
0200537
0200538
0200539
0200540
0200541
0200542
0200543
0200544
0200545
0200546
0200547
0200548
0200549
0200550
0200551
0200552
0200553
0200554
0200555
0200556
0200557
0200558
0200559
0200560
0200561
0200562
0200563
0200564
0200565
0200566
0200567
0200568
0200569
0200570
0200571
0200572
0200573
0200574
0200575
0200576
0200577
0200578
0200579
0200580
0200581
0200582
0200583
0200584
0200585
0200586
0200587
0200588
0200589
0200590
0200591
0200592
0200593
0200594
0200595
0200596
0200597
0200598
0200599
0200600
0200601
0200602
0200603
0200604
0200605
0200606
0200607
0200608
0200609
0200610
0200611
0200612
0200613
0200614
0200615
0200616
0200617
0200618
0200619
0200620
0200621
0200622
0200623
0200624
0200625
0200626
0200627
0200628
0200629
0200630
0200631
0200632
0200633
0200634
0200635
0200636
0200637
0200638
0200639
0200640
02006
```

```

01168 javax.swing.JTextField itemprice;
01170 javax.swing.JTextField itemprice1;
01172 javax.swing.JTextField itemprice2;
01174 javax.swing.JButton jButton1;
01176 javax.swing.JButton jButton2;
01178 javax.swing.JButton jButton3;
01180 javax.swing.JButton jButton4;
01182 javax.swing.JButton jButton5;
01184 javax.swing.JButton jButton6;
01186 javax.swing.JButton jButton7;
01188 javax.swing.JLabel jLabel1;
01190 javax.swing.JLabel jLabel10;
01192 javax.swing.JLabel jLabel2;
01194 javax.swing.JLabel jLabel3;
01196 javax.swing.JLabel jLabel4;
01198 javax.swing.JLabel jLabel5;
01200 javax.swing.JLabel jLabel6;
01202 javax.swing.JLabel jLabel7;
01204 javax.swing.JLabel jLabel8;
01206 javax.swing.JLabel jLabel9;
01208 javax.swing.JPanel jPanel1;
01210 javax.swing.JPanel jPanel2;
01212 javax.swing.JPanel jPanel3;
01214 javax.swing.JPanel jPanel4;
01216 javax.swing.JScrollPane jScrollPane1;
01218 javax.swing.JScrollPane jScrollPane2;
01220 javax.swing.JScrollPane jScrollPane3;
01222 javax.swing.JTabbedPane jTabbedPane1;
01224 javax.swing.JTable jTable1;
01226 javax.swing.JTable jTable2;
01228 javax.swing.JTable jTable3;
01229 // End of variables declaration//GEN-END:variables
01230 }

```

## 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.](#)

```

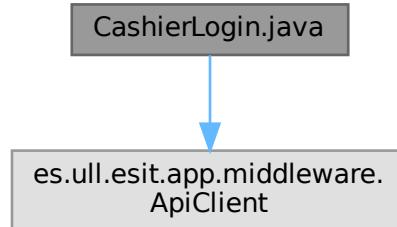
00001 package es.ull.esit.app;
00002
00010 public class ApplicationLauncher {
00011
00021     public static void main(String[] args) {
00022         java.awt.EventQueue.invokeLater(() -> getLoginFactory().get().setVisible(true));
00023     }
00024
00031     private static java.util.function.Supplier<Login> loginFactory = Login::new;
00032
00040     static void setLoginFactory(java.util.function.Supplier<Login> factory) {
00041         loginFactory = factory;
00042     }
00043
00051     static java.util.function.Supplier<Login> getLoginFactory() {
00052         return loginFactory;
00053     }
00054 }

```

## 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 java.util.function.Supplier;
00007 import javax.swing.SwingUtilities;
00008 import org.slf4j.Logger;
00009 import org.slf4j.LoggerFactory;
0010
0026 public class CashierLogin extends javax.swing.JFrame {
0027
0029     private final transient ReportService reportService;
0030
0032     private static final String PRIMARY_FONT = "Yu Gothic UI";
0033
0035     private static final String WARN_FETCH_CASHIER_STATS = "Warning: Could not fetch cashier stats: {}";
0036
0038     private static final String ERROR_CHECK_MENU_STATUS = "Error checking menu status: {}";
0039
0041     private static final Logger LOGGER = LoggerFactory.getLogger(CashierLogin.class);
0042
0050     public CashierLogin() {
0051         this((String) null);
0052     }
0053
0064     public CashierLogin(String name) {
0065         initComponents();
0066     }
  
```

```
00067     ApiClient client = new ApiClient("http://localhost:8080");
00068     this.reportService = new ReportService(client);
00069
00070     updateWelcomeMessage(name);
00071 }
00072
00087     CashierLogin(ReportService reportService, String name, boolean startBackground) {
00088         initComponents();
00089         this.reportService = reportService;
00090         updateWelcomeMessage(name);
00091         // do not start background threads in tests; if requested, callers may
00092         // manually invoke the async handlers or startBackground could be used
00093         // in future to mimic the default constructor behaviour.
00094         if (startBackground) {
00095             // no-op for now; left intentionally minimal to avoid side-effects in tests.
00096         }
00097     }
00098
00105     transient Supplier<? extends javax.swing.JFrame> aboutUsSupplier = AboutUs::new;
00106
00113     transient Supplier<? extends javax.swing.JFrame> orderSupplier = Order::new;
00114
00121     transient Supplier<? extends javax.swing.JFrame> loginSupplier = Login::new;
00122
00130     private static Supplier<? extends javax.swing.JFrame> cashierSupplier = CashierLogin::new;
00131
00141     static void setCashierSupplier(Supplier<? extends javax.swing.JFrame> s) {
00142         cashierSupplier = s;
00143     }
00144
00155     static Supplier<? extends javax.swing.JFrame> getCashierSupplier() {
00156         return cashierSupplier;
00157     }
00158
00167     void setAboutUsSupplier(Supplier<? extends javax.swing.JFrame> s) {
00168         this.aboutUsSupplier = s;
00169     }
00170
00180     void setOrderSupplier(Supplier<? extends javax.swing.JFrame> s) {
00181         this.orderSupplier = s;
00182     }
00183
00192     void setLoginSupplier(Supplier<? extends javax.swing.JFrame> s) {
00193         this.loginSupplier = s;
00194     }
00195
00205     void aboutUsActionRun() {
00206         try {
00207             List<es.ull.esit.app.middleware.model.Cashier> cashiers = reportService.getCashierInfo();
00208             LOGGER.info("Found {} cashiers in DB.", cashiers == null ? 0 : cashiers.size());
00209         } catch (Exception ex) {
00210             LOGGER.warn(WARN_FETCH_CASHIER_STATS, ex.getMessage());
00211         }
00212         // Use supplier so tests can inject a no-op frame
00213         javax.swing.JFrame f = aboutUsSupplier.get();
00214         f.setVisible(true);
00215         dispose();
00216     }
00217
00227     void menuActionRun() {
00228         try {
00229             String status = reportService.checkMenuStatus();
00230             LOGGER.info("Menu status: {}", status);
00231         } catch (Exception ex) {
00232             LOGGER.error(ERROR_CHECK_MENU_STATUS, ex.getMessage());
00233         }
00234         javax.swing.JFrame f = orderSupplier.get();
00235         f.setVisible(true);
00236         dispose();
00237     }
00238
00249     void logoutActionRun() {
00250         javax.swing.JFrame f = loginSupplier.get();
00251         f.setVisible(true);
00252         dispose();
00253     }
00254
00262     void testOnlyCoverageHelper() {
00263         int a = (int) (System.currentTimeMillis() % 2);
00264         if (a == 0) {
00265             a = 1;
00266         } else {
00267             a = -1;
00268         }
00269         if (a == 1) {
00270             a++;
00271         } else {
```

```

00272     a--;
00273 }
String tmp = "ok" + a;
LOGGER.debug("coverage helper: {}", tmp);
}
00277
00286 private void updateWelcomeMessage(String name) {
00287     if (name == null || name.trim().isEmpty()) {
00288         welcomeTxt.setText("Welcome Cashier");
00289     } else {
00290         welcomeTxt.setText("Welcome " + name);
00291     }
00293
00310 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
00311     new Thread(() -> {
00312         try {
00313             List<es.ull.esit.app.middleware.model.Cashier> cashiers = reportService.getCashierInfo();
00314             LOGGER.info("Found {} cashiers in DB.", cashiers.size());
00315
00316         } catch (Exception ex) {
00317             LOGGER.warn(WARN_FETCH_CASHIER_STATS, ex.getMessage());
00318         }
00319         SwingUtilities.invokeLater(() -> {
00320             aboutUsSupplier.get().setVisible(true);
00321             dispose();
00322         });
00323     }).start();
00324 }
00325
00339 int aboutUsActionSync() {
00340     try {
00341         List<es.ull.esit.app.middleware.model.Cashier> cashiers = reportService.getCashierInfo();
00342         return cashiers == null ? 0 : cashiers.size();
00343     } catch (Exception ex) {
00344         LOGGER.warn(WARN_FETCH_CASHIER_STATS, ex.getMessage());
00345         return -1;
00346     }
00347 }
00348
00366 private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
00367     new Thread(() -> {
00368         try {
00369             String status = reportService.checkMenuStatus();
00370             LOGGER.info("Menu status: {}", status);
00371
00372         } catch (Exception ex) {
00373             LOGGER.error(ERROR_CHECK_MENU_STATUS, ex.getMessage());
00374         }
00375
00376         SwingUtilities.invokeLater(() -> {
00377             orderSupplier.get().setVisible(true);
00378             dispose();
00379         });
00380     }).start();
00381 }
00382
00394 String menuActionSync() {
00395     try {
00396         return reportService.checkMenuStatus();
00397     } catch (Exception ex) {
00398         LOGGER.error(ERROR_CHECK_MENU_STATUS, ex.getMessage());
00399         return null;
00400     }
00401 }
00402
00416 private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
00417     loginSupplier.get().setVisible(true);
00418     dispose();
00419 }
00420
00432 boolean logoutActionSync() {
00433     // In production this would open the Login window and dispose(). For tests
00434     // we simply indicate the action was executed.
00435     return true;
00436 }
00437
00445 String getWelcomeText() {
00446     return welcomeTxt.getText();
00447 }
00448
00459 public static void main(String[] args) {
00460     try {
00461         for (javax.swing.UIManager.LookAndFeelInfo info :
00462             javax.swing.UIManager.getInstalledLookAndFeels()) {
00463             if ("Nimbus".equals(info.getName())) {
00464                 javax.swing.UIManager.setLookAndFeel(info.getClassName());
00465             }
00466         }
00467     }
00468 }
00469
00470 }
00471
00472 }
00473
00474 }
00475
00476 }
00477
00478 }
00479
00480 }
00481
00482 }
00483
00484 }
00485
00486 }
00487
00488 }
00489
00490 }
00491
00492 }
00493
00494 }
00495
00496 }
00497
00498 }
00499
00500 }
00501
00502 }
00503
00504 }
00505
00506 }
00507
00508 }
00509
00510 }
00511
00512 }
00513
00514 }
00515
00516 }
00517
00518 }
00519
00520 }
00521
00522 }
00523
00524 }
00525
00526 }
00527
00528 }
00529
00530 }
00531
00532 }
00533
00534 }
00535
00536 }
00537
00538 }
00539
00540 }
00541
00542 }
00543
00544 }
00545
00546 }
00547
00548 }
00549
00550 }
00551
00552 }
00553
00554 }
00555
00556 }
00557
00558 }
00559
00560 }
00561
00562 }
00563
00564 }
00565
00566 }
00567
00568 }
00569
00570 }
00571
00572 }
00573
00574 }
00575
00576 }
00577
00578 }
00579
00580 }
00581
00582 }
00583
00584 }
00585
00586 }
00587
00588 }
00589
00590 }
00591
00592 }
00593
00594 }
00595
00596 }
00597
00598 }
00599
00600 }
00601
00602 }
00603
00604 }
00605
00606 }
00607
00608 }
00609
00610 }
00611
00612 }
00613
00614 }
00615
00616 }
00617
00618 }
00619
00620 }
00621
00622 }
00623
00624 }
00625
00626 }
00627
00628 }
00629
00630 }
00631
00632 }
00633
00634 }
00635
00636 }
00637
00638 }
00639
00640 }
00641
00642 }
00643
00644 }
00645
00646 }
00647
00648 }
00649
00650 }
00651
00652 }
00653
00654 }
00655
00656 }
00657
00658 }
00659
00660 }
00661
00662 }
00663
00664 }
00665
00666 }
00667
00668 }
00669
00670 }
00671
00672 }
00673
00674 }
00675
00676 }
00677
00678 }
00679
00680 }
00681
00682 }
00683
00684 }
00685
00686 }
00687
00688 }
00689
00690 }
00691
00692 }
00693
00694 }
00695
00696 }
00697
00698 }
00699
00700 }
00701
00702 }
00703
00704 }
00705
00706 }
00707
00708 }
00709
00710 }
00711
00712 }
00713
00714 }
00715
00716 }
00717
00718 }
00719
00720 }
00721
00722 }
00723
00724 }
00725
00726 }
00727
00728 }
00729
00730 }
00731
00732 }
00733
00734 }
00735
00736 }
00737
00738 }
00739
00740 }
00741
00742 }
00743
00744 }
00745
00746 }
00747
00748 }
00749
00750 }
00751
00752 }
00753
00754 }
00755
00756 }
00757
00758 }
00759
00760 }
00761
00762 }
00763
00764 }
00765
00766 }
00767
00768 }
00769
00770 }
00771
00772 }
00773
00774 }
00775
00776 }
00777
00778 }
00779
00780 }
00781
00782 }
00783
00784 }
00785
00786 }
00787
00788 }
00789
00790 }
00791
00792 }
00793
00794 }
00795
00796 }
00797
00798 }
00799
00800 }
00801
00802 }
00803
00804 }
00805
00806 }
00807
00808 }
00809
00810 }
00811
00812 }
00813
00814 }
00815
00816 }
00817
00818 }
00819
00820 }
00821
00822 }
00823
00824 }
00825
00826 }
00827
00828 }
00829
00830 }
00831
00832 }
00833
00834 }
00835
00836 }
00837
00838 }
00839
00840 }
00841
00842 }
00843
00844 }
00845
00846 }
00847
00848 }
00849
00850 }
00851
00852 }
00853
00854 }
00855
00856 }
00857
00858 }
00859
00860 }
00861
00862 }
00863
00864 }
00865
00866 }
00867
00868 }
00869
00870 }
00871
00872 }
00873
00874 }
00875
00876 }
00877
00878 }
00879
00880 }
00881
00882 }
00883
00884 }
00885
00886 }
00887
00888 }
00889
00890 }
00891
00892 }
00893
00894 }
00895
00896 }
00897
00898 }
00899
00900 }
00901
00902 }
00903
00904 }
00905
00906 }
00907
00908 }
00909
00910 }
00911
00912 }
00913
00914 }
00915
00916 }
00917
00918 }
00919
00920 }
00921
00922 }
00923
00924 }
00925
00926 }
00927
00928 }
00929
00930 }
00931
00932 }
00933
00934 }
00935
00936 }
00937
00938 }
00939
00940 }
00941
00942 }
00943
00944 }
00945
00946 }
00947
00948 }
00949
00950 }
00951
00952 }
00953
00954 }
00955
00956 }
00957
00958 }
00959
00960 }
00961
00962 }
00963
00964 }
00965
00966 }
00967
00968 }
00969
00970 }
00971
00972 }
00973
00974 }
00975
00976 }
00977
00978 }
00979
00980 }
00981
00982 }
00983
00984 }
00985
00986 }
00987
00988 }
00989
00990 }
00991
00992 }
00993
00994 }
00995
00996 }
00997
00998 }
00999
01000 }
01001
01002 }
01003
01004 }
01005
01006 }
01007
01008 }
01009
01010 }
01011
01012 }
01013
01014 }
01015
01016 }
01017
01018 }
01019
01020 }
01021
01022 }
01023
01024 }
01025
01026 }
01027
01028 }
01029
01030 }
01031
01032 }
01033
01034 }
01035
01036 }
01037
01038 }
01039
01040 }
01041
01042 }
01043
01044 }
01045
01046 }
01047
01048 }
01049
01050 }
01051
01052 }
01053
01054 }
01055
01056 }
01057
01058 }
01059
01060 }
01061
01062 }
01063
01064 }
01065
01066 }
01067
01068 }
01069
01070 }
01071
01072 }
01073
01074 }
01075
01076 }
01077
01078 }
01079
01080 }
01081
01082 }
01083
01084 }
01085
01086 }
01087
01088 }
01089
01090 }
01091
01092 }
01093
01094 }
01095
01096 }
01097
01098 }
01099
01100 }
01101
01102 }
01103
01104 }
01105
01106 }
01107
01108 }
01109
01110 }
01111
01112 }
01113
01114 }
01115
01116 }
01117
01118 }
01119
01120 }
01121
01122 }
01123
01124 }
01125
01126 }
01127
01128 }
01129
01130 }
01131
01132 }
01133
01134 }
01135
01136 }
01137
01138 }
01139
01140 }
01141
01142 }
01143
01144 }
01145
01146 }
01147
01148 }
01149
01150 }
01151
01152 }
01153
01154 }
01155
01156 }
01157
01158 }
01159
01160 }
01161
01162 }
01163
01164 }
01165
01166 }
01167
01168 }
01169
01170 }
01171
01172 }
01173
01174 }
01175
01176 }
01177
01178 }
01179
01180 }
01181
01182 }
01183
01184 }
01185
01186 }
01187
01188 }
01189
01190 }
01191
01192 }
01193
01194 }
01195
01196 }
01197
01198 }
01199
01200 }
01201
01202 }
01203
01204 }
01205
01206 }
01207
01208 }
01209
01210 }
01211
01212 }
01213
01214 }
01215
01216 }
01217
01218 }
01219
01220 }
01221
01222 }
01223
01224 }
01225
01226 }
01227
01228 }
01229
01230 }
01231
01232 }
01233
01234 }
01235
01236 }
01237
01238 }
01239
01240 }
01241
01242 }
01243
01244 }
01245
01246 }
01247
01248 }
01249
01250 }
01251
01252 }
01253
01254 }
01255
01256 }
01257
01258 }
01259
01260 }
01261
01262 }
01263
01264 }
01265
01266 }
01267
01268 }
01269
01270 }
01271
01272 }
01273
01274 }
01275
01276 }
01277
01278 }
01279
01280 }
01281
01282 }
01283
01284 }
01285
01286 }
01287
01288 }
01289
01290 }
01291
01292 }
01293
01294 }
01295
01296 }
01297
01298 }
01299
01300 }
01301
01302 }
01303
01304 }
01305
01306 }
01307
01308 }
01309
01310 }
01311
01312 }
01313
01314 }
01315
01316 }
01317
01318 }
01319
01320 }
01321
01322 }
01323
01324 }
01325
01326 }
01327
01328 }
01329
01330 }
01331
01332 }
01333
01334 }
01335
01336 }
01337
01338 }
01339
01340 }
01341
01342 }
01343
01344 }
01345
01346 }
01347
01348 }
01349
01350 }
01351
01352 }
01353
01354 }
01355
01356 }
01357
01358 }
01359
01360 }
01361
01362 }
01363
01364 }
01365
01366 }
01367
01368 }
01369
01370 }
01371
01372 }
01373
01374 }
01375
01376 }
01377
01378 }
01379
01380 }
01381
01382 }
01383
01384 }
01385
01386 }
01387
01388 }
01389
01390 }
01391
01392 }
01393
01394 }
01395
01396 }
01397
01398 }
01399
01400 }
01401
01402 }
01403
01404 }
01405
01406 }
01407
01408 }
01409
01410 }
01411
01412 }
01413
01414 }
01415
01416 }
01417
01418 }
01419
01420 }
01421
01422 }
01423
01424 }
01425
01426 }
01427
01428 }
01429
01430 }
01431
01432 }
01433
01434 }
01435
01436 }
01437
01438 }
01439
01440 }
01441
01442 }
01443
01444 }
01445
01446 }
01447
01448 }
01449
01450 }
01451
01452 }
01453
01454 }
01455
01456 }
01457
01458 }
01459
01460 }
01461
01462 }
01463
01464 }
01465
01466 }
01467
01468 }
01469
01470 }
01471
01472 }
01473
01474 }
01475
01476 }
01477
01478 }
01479
01480 }
01481
01482 }
01483
01484 }
01485
01486 }
01487
01488 }
01489
01490 }
01491
01492 }
01493
01494 }
01495
01496 }
01497
01498 }
01499
01500 }
01501
01502 }
01503
01504 }
01505
01506 }
01507
01508 }
01509
01510 }
01511
01512 }
01513
01514 }
01515
01516 }
01517
01518 }
01519
01520 }
01521
01522 }
01523
01524 }
01525
01526 }
01527
01528 }
01529
01530 }
01531
01532 }
01533
01534 }
01535
01536 }
01537
01538 }
01539
01540 }
01541
01542 }
01543
01544 }
01545
01546 }
01547
01548 }
01549
01550 }
01551
01552 }
01553
01554 }
01555
01556 }
01557
01558 }
01559
01560 }
01561
01562 }
01563
01564 }
01565
01566 }
01567
01568 }
01569
01570 }
01571
01572 }
01573
01574 }
01575
01576 }
01577
01578 }
01579
01580 }
01581
01582 }
01583
01584 }
01585
01586 }
01587
01588 }
01589
01590 }
01591
01592 }
01593
01594 }
01595
01596 }
01597
01598 }
01599
01600 }
01601
01602 }
01603
01604 }
01605
01606 }
01607
01608 }
01609
01610 }
01611
01612 }
01613
01614 }
01615
01616 }
01617
01618 }
01619
01620 }
01621
01622 }
01623
01624 }
01625
01626 }
01627
01628 }
01629
01630 }
01631
01632 }
01633
01634 }
01635
01636 }
01637
01638 }
01639
01640 }
01641
01642 }
01643
01644 }
01645
01646 }
01647
01648 }
01649
01650 }
01651
01652 }
01653
01654 }
01655
01656 }
01657
01658 }
01659
01660 }
01661
01662 }
01663
01664 }
01665
01666 }
01667
01668 }
01669
01670 }
01671
01672 }
01673
01674 }
01675
01676 }
01677
01678 }
01679
01680 }
01681
01682 }
01683
01684 }
01685
01686 }
01687
01688 }
01689
01690 }
01691
01692 }
01693
01694 }
01695
01696 }
01697
01698 }
01699
01700 }
01701
01702 }
01703
01704 }
01705
01706 }
01707
01708 }
01709
01710 }
01711
01712 }
01713
01714 }
01715
01716 }
01717
01718 }
01719
01720 }
01721
01722 }
01723
01724 }
01725
01726 }
01727
01728 }
01729
01730 }
01731
01732 }
01733
01734 }
01735
01736 }
01737
01738 }
01739
01740 }
01741
01742 }
01743
01744 }
01745
01746 }
01747
01748 }
01749
01750 }
01751
01752 }
01753
01754 }
01755
01756 }
01757
01758 }
01759
01760 }
01761
01762 }
01763
01764 }
01765
01766 }
01767
01768 }
01769
01770 }
01771
01772 }
01773
01774 }
01775
01776 }
01777
01778 }
01779
01780 }
01781
01782 }
01783
01784 }
01785
01786 }
01787
01788 }
01789
01790 }
01791
01792 }
01793
01794 }
01795
01796 }
01797
01798 }
01799
01800 }
01801
01802 }
01803
01804 }
01805
01806 }
01807
01808 }
01809
01810 }
01811
01812 }
01813
01814 }
01815
01816 }
01817
01818 }
01819
01820 }
01821
01822 }
01823
01824 }
01825
01826 }
01827
01828 }
01829
01830 }
01831
01832 }
01833
01834 }
01835
01836 }
01837
01838 }
01839
01840 }
01841
01842 }
01843
01844 }
01845
01846 }
01847
01848 }
01849
01850 }
01851
01852 }
01853
01854 }
01855
01856 }
01857
01858 }
01859
01860 }
01861
01862 }
01863
01864 }
01865
01866 }
01867
01868 }
01869
01870 }
01871
01872 }
01873
01874 }
01875
01876 }
01877
01878 }
01879
01880 }
01881
01882 }
01883
01884 }
01885
01886 }
01887
01888 }
01889
01890 }
01891
01892 }
01893
01894 }
01895
01896 }
01897
01898 }
01899
01900 }
01901
01902 }
01903
01904 }
01905
01906 }
01907
01908 }
01909
01910 }
01911
01912 }
01913
01914 }
01915
01916 }
01917
01918 }
01919
01920 }
01921
01922 }
01923
01924 }
01925
01926 }
01927
01928 }
01929
01930 }
01931
01932 }
01933
01934 }
01935
01936 }
01937
01938 }
01939
01940 }
01941
01942 }
01943
01944 }
01945
01946 }
01947
01948 }
01949
01950 }
01951
01952 }
01953
01954 }
01955
01956 }
01957
01958 }
01959
01960 }
01961
01962 }
01963
01964 }
01965
01966 }
01967
01968 }
01969
01970 }
01971
01972 }
01973
01974 }
01975
01976 }
01977
01978 }
01979
01980 }
01981
01982 }
01983
01984 }
01985
01986 }
01987
01988 }
01989
01990 }
01991
01992 }
01993
01994 }
01995
01996 }
01997
01998 }
01999
02000 }
02001
02002 }
02003
02004 }
02005
02006 }
02007
02008 }
02009
02010 }
02011
02012 }
02013
02014 }
02015
02016 }
02017
02018 }
02019
02020 }
02021
02022 }
02023
02024 }
02025
02026 }
02027
02028 }
02029
02030 }
02031
02032 }
02033
02034 }
02035
02036 }
02037
02038 }
02039
02040 }
02041
02042 }
02043
02044 }
02045
02046 }
02047
02048 }
02049
02050 }
02051
02052 }
02053
02054 }
02055
02056 }
02057
02058 }
02059
02060 }
02061
02062 }
02063
02064 }
02065
02066 }
02067
02068 }
02069
02070 }
02071
02072 }
02073
02074 }
02075
02076 }
02077
02078 }
02079
02080 }
02081
02082 }
02083
02084 }
02085
02086 }
02087
02088 }
02089
02090 }
02091
02092 }
02093
02094 }
02095
02096 }
02097
02098 }
02099
02100 }
02101
02102 }
02103
02104 }
02105
02106 }
02107
02108 }
02109
02110 }
02111
02112 }
02113
02114 }
02115
02116 }
02117
02118 }
02119
02120 }
02121
02122 }
02123
02124 }
02125
02126 }
02127
02128 }
02129
02130 }
02131
02132 }
02133
02134 }
02135
02136 }
02137
02138 }
02139
02140 }
02141
02142 }
02143
02144 }
02145
02146 }
02147
02148 }
02149
02150 }
02151
02152 }
0
```

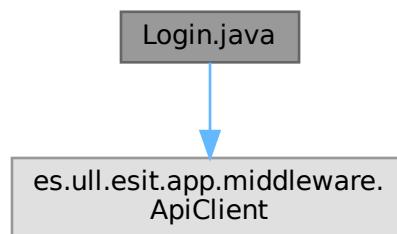


```

00550     jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00551         .addGroup(jPanellLayout.createSequentialGroup()
00552             .addGap(45, 45, 45)
00553             .addComponent(welcomeTxt, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
00554                 javax.swing.GroupLayout.PREFERRED_SIZE)
00555             .addGap(18, 18, 18)
00556             .addComponent(jLabel1)
00557             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 99,
00558                 Short.MAX_VALUE)
00558         .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00559             javax.swing.GroupLayout.PREFERRED_SIZE)
00560         .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00561             .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00562                 javax.swing.GroupLayout.PREFERRED_SIZE)
00563             .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING,
00564                 .addGroup(jPanellLayout.createSequentialGroup()
00565                     .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00566                         .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00567                             javax.swing.GroupLayout.PREFERRED_SIZE)
00568                         .addGroup(jPanellLayout.createSequentialGroup()
00569                             .addGap(242, 242, 242))))))
00570
00571     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00572     getContentPane().setLayout(layout);
00573     layout.setHorizontalGroup(
00574         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00575             .addGroup(layout.createSequentialGroup()
00576                 .addComponent(jPanell, javax.swing.GroupLayout.DEFAULT_SIZE,
00577                     javax.swing.GroupLayout.DEFAULT_SIZE,
00578                     Short.MAX_VALUE));
00577         layout.setVerticalGroup(
00578             layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00579                 .addComponent(jPanell, javax.swing.GroupLayout.DEFAULT_SIZE,
00580                     javax.swing.GroupLayout.DEFAULT_SIZE,
00581                     Short.MAX_VALUE));
00581
00582     pack();
00583     setLocationRelativeTo(null);
00584 } // </editor-fold> //GEN-END:initComponents
00585
00586 // Variables declaration - do not modify //GEN-BEGIN:variables
00587 // Sonarqube rule java:S1450 must be ignored here as these variables are
00588 // auto-generated by the Form Editor and need to remain as instance variables.
00589 private javax.swing.JButton jButton1;
00590 private javax.swing.JButton jButton2;
00591 private javax.swing.JButton jButton3;
00592 private javax.swing.JLabel jLabel1;
00593 private javax.swing.JPanel jPanell;
00594 private javax.swing.JLabel welcomeTxt;
00595 // End of variables declaration //GEN-END:variables
00596 }
00597 }
```

## 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.*
- interface **es.ull.esit.app.Login.MessageDialog**  
*Abstraction for message dialogs (test seam).*

## 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 transient AuthService authService;
00027
00033     transient java.util.function.Supplier<? extends javax.swing.JFrame> adminSupplier = AdminLogin::new;
00034
00043     transient java.util.function.Function<String, ? extends javax.swing.JFrame> cashierFactory =
00044         CashierLogin::new;
00045
00050     private static java.util.function.Supplier<Login> loginFactory = Login::new;
00051
00058     interface MessageDialog {
00065         void showMessage(java.awt.Component parent, Object message);
00066
00076         void showMessage(java.awt.Component parent, Object message, String title, int messageType);
00077     }
00078
00085     transient MessageDialog messageDialog = new MessageDialog() {
00086         @Override
00087             public void showMessage(java.awt.Component parent, Object message) {
00088                 JOptionPane.showMessageDialog(parent, message);
00089             }
00090
00091         @Override
00092             public void showMessage(java.awt.Component parent, Object message, String title, int messageType)
00093             {
00094                 JOptionPane.showMessageDialog(parent, message, title, messageType);
00095             }
00096     };
00103     void setAdminSupplier(java.util.function.Supplier<? extends javax.swing.JFrame> s) {
00104         this.adminSupplier = s;
00105     }
00106
00113     void setCashierFactory(java.util.function.Function<String, ? extends javax.swing.JFrame> f) {
00114         this.cashierFactory = f;
00115     }
00116
00122     void setMessageDialog(MessageDialog md) {
00123         this.messageDialog = md;
00124     }
00125
00132     static void setLoginFactory(java.util.function.Supplier<Login> f) {
00133         loginFactory = f;
00134     }
00135
00142     static java.util.function.Supplier<Login> getLoginFactory() {
00143         return loginFactory;
00144     }
00145
00147     private static final String PRIMARY_FONT = "Yu Gothic UI";
00148
```

```
00150 private static final String LOGIN_STRING = "Log in";
00151
00161 public Login() {
00162     initComponents();
00163     ApiClient client = new ApiClient("http://localhost:8080");
00164     this.authService = new AuthService(client);
00165 }
00166
00175 @SuppressWarnings("unchecked")
00176 // <editor-fold defaultstate="collapsed" desc="Generated
00177 // Code">//GEN-BEGIN:initComponents
00178 private void initComponents() {
00179
00180     jPanel1 = new javax.swing.JPanel();
00181     jLabel1 = new javax.swing.JLabel();
00182     jLabel3 = new javax.swing.JLabel();
00183     usernametxt = new javax.swing.JTextField();
00184     usertypecmb = new javax.swing.JComboBox<>();
00185     jButton1 = new javax.swing.JButton();
00186     jPasswordField1 = new javax.swing.JPasswordField();
00187
00188     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00189     setTitle("Login");
00190     setResizable(false);
00191
00192     jPanel1.setBackground(new java.awt.Color(248, 244, 230));
00193
00194     jLabel1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/ui/screenshot.png"))); // NOI18N
00195
00196     jLabel3.setFont(new java.awt.Font(PRIMARY_FONT, 1, 24)); // NOI18N
00197     jLabel3.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00198     jLabel3.setText("User Login ");
00199
00200     usernametxt.setFont(new java.awt.Font(PRIMARY_FONT, 0, 18)); // NOI18N
00201     usernametxt.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00202     usernametxt.setBorder(javax.swing.BorderFactory.createTitledBorder(
00203         new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Username",
00204         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00205         new java.awt.Font(PRIMARY_FONT, 0, 18))); // NOI18N
00206
00207     usertypecmb.setEditable(true);
00208     usertypecmb.setFont(new java.awt.Font(PRIMARY_FONT, 0, 18)); // NOI18N
00209     usertypecmb.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] { "admin", "cashier" }));
00210     usertypecmb.addActionListener(this::usertypecmbActionPerformed);
00211
00212     jButton1.setFont(new java.awt.Font(PRIMARY_FONT, 1, 18)); // NOI18N
00213     jButton1.setText(LOGIN_STRING);
00214     jButton1.addActionListener(this::jButton1ActionPerformed);
00215
00216     jPasswordField1.setFont(new java.awt.Font("Thonburi", 0, 18)); // NOI18N
00217     jPasswordField1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
00218     jPasswordField1.setBorder(javax.swing.BorderFactory.createTitledBorder(
00219         javax.swing.BorderFactory.createLineBorder(new java.awt.Color(0, 0, 0)), "Password",
00220         javax.swing.border.TitledBorder.LEFT, javax.swing.border.TitledBorder.TOP,
00221         new java.awt.Font("Lucida Grande", 0, 18))); // NOI18N
00222
00223     javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
00224     jPanel1.setLayout(jPanel1Layout);
00225     jPanel1Layout.setHorizontalGroup(
00226         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00227             .addGroup(jPanel1Layout.createSequentialGroup()
00228                 .addGroup(jPanel1Layout.createSequentialGroup()
00229                     .addGroup(jPanel1Layout.createSequentialGroup()
00230                         .addGap(132, 132, 132)
00231                         .addComponent(jLabel1))
00232                     .addGroup(jPanel1Layout.createSequentialGroup()
00233                         .addGap(61, 61, 61)
00234                         .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 309,
00235                             javax.swing.GroupLayout.PREFERRED_SIZE))
00236                     .addGroup(jPanel1Layout.createSequentialGroup()
00237                         .addGap(74, 74, 74)
00238
00239                         .addGroup(jPanel1Layout.createSequentialGroup()
00240                             .addGap(311, 311, 311)
00241                             .addComponent(jPasswordField1, javax.swing.GroupLayout.PREFERRED_SIZE,
00242                                 javax.swing.GroupLayout.PREFERRED_SIZE)
00243                             .addComponent(usertypecmb, javax.swing.GroupLayout.PREFERRED_SIZE, 311,
00244                                 javax.swing.GroupLayout.PREFERRED_SIZE)))
00245                     .addGroup(jPanel1Layout.createSequentialGroup()
00246                         .addGap(146, 146, 146)
00247                         .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE,
00248                             javax.swing.GroupLayout.PREFERRED_SIZE))
00249                     .addGap(86, Short.MAX_VALUE)))
00250     );
00251 }
```

```

00250     jPanel1.setLayout.setVerticalGroup(
00251         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00252             .addGroup(jPanel1Layout.createSequentialGroup()
00253                 .addGap(39, 39, 39)
00254                 .addComponent(jLabel1)
00255                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
00256                 .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
00257                     javax.swing.GroupLayout.PREFERRED_SIZE)
00258                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
00259                 .addComponent(usernameTxt, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00260                     javax.swing.GroupLayout.PREFERRED_SIZE)
00261                 .addGap(18, 18, 18)
00262                 .addComponent(jPasswordField1, javax.swing.GroupLayout.PREFERRED_SIZE, 59,
00263                     javax.swing.GroupLayout.PREFERRED_SIZE)
00264                 .addGap(18, 18, 18)
00265                 .addComponent(userTypeCmbo, javax.swing.GroupLayout.PREFERRED_SIZE, 46,
00266                     javax.swing.GroupLayout.PREFERRED_SIZE)
00267                 .addGap(18, 18, 18)
00268                 .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
00269                     javax.swing.GroupLayout.PREFERRED_SIZE)
00270                 .addContainerGap(39, Short.MAX_VALUE)));
00271
00272     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00273     getContentPane().setLayout(layout);
00274     layout.setHorizontalGroup(
00275         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00276             .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
00277                 javax.swing.GroupLayout.DEFAULT_SIZE,
00278                     javax.swing.GroupLayout.PREFERRED_SIZE));
00279     layout.setVerticalGroup(
00280         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00281             .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
00282                 javax.swing.GroupLayout.DEFAULT_SIZE,
00283                     Short.MAX_VALUE));
00284
00285     pack();
00286     setLocationRelativeTo(null);
00287 // </editor-fold> //GEN-END:initComponents
00288
00289 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) //{
00290     GEN-FIRST:event_jButton1ActionPerformed
00291     // Get username and password from input fields.
00292     String usernameInput = usernameTxt.getText();
00293     String passwordInput = new String(jPasswordField1.getPassword());
00294
00295     jButton1.setEnabled(false);
00296     jButton1.setText("Loading...");
00297
00298     new Thread(() -> {
00299         try {
00300             // Authenticate user through AuthService.
00301             User loggedInUser = authService.authenticate(usernameInput, passwordInput);
00302
00303             SwingUtilities.invokeLater(() -> {
00304
00305                 // Open the corresponding window based on user role. Use the
00306                 // injectable factories so tests can substitute frames.
00307                 if ("ADMIN".equalsIgnoreCase(loggedInUser.getRole())) {
00308                     adminSupplier.get().setVisible(true);
00309                 } else if ("CASHIER".equalsIgnoreCase(loggedInUser.getRole())) {
00310                     cashierFactory.apply(loggedInUser.getUsername()).setVisible(true);
00311                 } else {
00312                     messageDialog.showMessage(this, "Unknown Role: " + loggedInUser.getRole());
00313                     jButton1.setEnabled(true);
00314                     jButton1.setText(LOGIN_STRING);
00315                     return;
00316                 }
00317
00318                 this.dispose();
00319             });
00320
00321         } catch (Exception e) {
00322             SwingUtilities.invokeLater(() -> {
00323                 messageDialog.showMessage(this, "Login failed: " + e.getMessage(), "Login Error",
00324                     JOptionPane.ERROR_MESSAGE);
00325
00326                 jButton1.setEnabled(true);
00327                 jButton1.setText(LOGIN_STRING);
00328             });
00329         }
00330     } // catch (Exception e) {
00331     SwingUtilities.invokeLater(() -> {
00332         messageDialog.showMessage(this, "Login failed: " + e.getMessage(), "Login Error",
00333             JOptionPane.ERROR_MESSAGE);
00334
00335         jButton1.setEnabled(true);
00336         jButton1.setText(LOGIN_STRING);
00337     });
00338 }
00339 }).start();
00340 // GEN-LAST:event_jButton1ActionPerformed
00341
00342 private void userTypeCmboActionPerformed(java.awt.event.ActionEvent evt) //{
00343     GEN-FIRST:event_userTypeCmboActionPerformed
00344     // No action needed here.
00345 // GEN-LAST:event_userTypeCmboActionPerformed

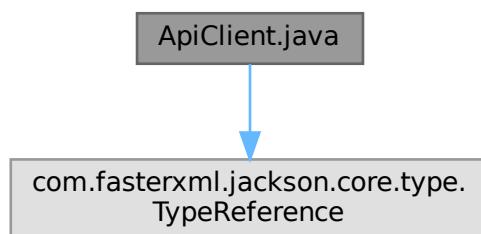
```

```

00353
00359     public static void main(String[] args) {
00360         try {
00361             for (javax.swing.UIManager.LookAndFeelInfo info :
00362                 javax.swing.UIManager.getInstalledLookAndFeels()) {
00363                 if ("Nimbus".equals(info.getName())) {
00364                     javax.swing.UIManager.setLookAndFeel(info.getClassName());
00365                     break;
00366                 }
00367             } catch (ClassNotFoundException | InstantiationException | IllegalAccessException
00368             | javax.swing.UnsupportedLookAndFeelException ex) {
00369                 java.util.logging.Logger.getLogger(Login.class.getName()).log(java.util.logging.Level.SEVERE,
00370                         null, ex);
00371             }
00372             java.awt.EventQueue.invokeLater(() -> getLoginFactory().get().setVisible(true));
00373         }
00374
00375         // Variables declaration - do not modify//GEN-BEGIN:variables
00376         // Sonarqube rule java:S1450 must be ignored here as these variables are
00377         // auto-generated by the Form Editor and need to remain as instance variables.
00378         private javax.swing.JButton jButton1;
00379         private javax.swing.JLabel jLabel1;
00380         private javax.swing.JLabel jLabel13;
00381         private javax.swing.JPanel jPanel1;
00382         private javax.swing.JPasswordField jPasswordField1;
00383         private javax.swing.JTextField usernametxt;
00384         private javax.swing.JComboBox<String> usertypecmbo;
00385
00386         // End of variables declaration//GEN-END:variables
00395     }
00396 }
```

## 8.56 ApiClient.java File Reference

import com.fasterxml.jackson.core.type.TypeReference;  
 Include dependency graph for ApiClient.java:



### Classes

- class [es.ull.esit.app.middleware.ApiClient](#)  
*API client for interacting with the backend REST API.*
- interface [es.ull.esit.app.middleware.ApiClient.IOCallable< T >](#)  
*Functional interface for lambdas that may throw InterruptedException or IOException.*
- interface [es.ull.esit.app.middleware.ApiClient.ResponseHandler< R >](#)  
*Functional interface for processing HTTP responses and possibly throwing IOException from JSON parsing.*

### 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.core.type.TypeReference;
00004 import com.fasterxml.jackson.databind.ObjectMapper;
00005
00006 import java.io.IOException;
00007 import java.net.URI;
00008 import java.net.http.HttpClient;
00009 import java.net.http.HttpRequest;
00010 import java.net.http.HttpResponse;
00011 import java.time.Duration;
00012 import java.util.List;
00013 import java.util.Map;
00014
00015 import es.ull.esit.app.middleware.model.Appetizer;
00016 import es.ull.esit.app.middleware.model.Cashier;
00017 import es.ull.esit.app.middleware.model.Drink;
00018 import es.ull.esit.app.middleware.model.MainCourse;
00019 import es.ull.esit.app.middleware.model.User;
00020
00021 import org.slf4j.Logger;
00022 import org.slf4j.LoggerFactory;
00023
00031 public class ApiClient {
00032
00034     private static final String APPLICATION_JSON = "application/json";
00035
00037     private static final Logger LOGGER = LoggerFactory.getLogger(ApiClient.class);
00038
00040     private static final String CONTENT_TYPE = "Content-Type";
00041
00043     private static final String API_APPETIZERS = "/api/appetizers/";
00044
00046     private static final String API_DRINKS = "/api/drinks/";
00047
00049     private static final String API_MAINCOURSES = "/api/maincourses/";
00050
00051     private static final String API_LOGIN = "/api/login";
00052
00054     private final HttpClient http;
00055
00057     private final String baseUrl;
00058
00063     private final ObjectMapper mapper;
00064
00074     public ApiClient(String baseUrl) {
00075         this.baseUrl = baseUrl.endsWith("/") ? baseUrl.substring(0, baseUrl.length() - 1) : baseUrl;
00076         this.http = HttpClient.newBuilder()
00077             .connectTimeout(Duration.ofSeconds(5))
00078             .build();
00079         this.mapper = new ObjectMapper();
00080     }
00081
00086     ApiClient(String baseUrl, HttpClient http, ObjectMapper mapper) {
00087         this.baseUrl = baseUrl.endsWith("/") ? baseUrl.substring(0, baseUrl.length() - 1) : baseUrl;
00088         this.http = http;
00089         this.mapper = mapper == null ? new ObjectMapper() : mapper;
00090     }
00091
00095     @FunctionalInterface
00096     private interface IOCallable<T> {
00097         T call() throws InterruptedException, IOException;
00098     }
00099
00109     private <T> T execute(String action, IOCallable<T> callable) {
00110         try {
00111             return callable.call();
00112         } catch (InterruptedException e) {
00113             Thread.currentThread().interrupt();
00114             throw new ApiClientException("Thread interrupted during " + action, e);
00115         } catch (IOException e) {
00116             throw new ApiClientException("I/O error during " + action, e);
00117         }
00118     }
00119
00124     @FunctionalInterface
00125     private interface ResponseHandler<R> {
00126         R handle(int status, String body) throws IOException;
00127     }
00134     private <R> R sendAndHandle(String action, HttpRequest request, ResponseHandler<R> handler)
00135         throws InterruptedException, IOException {
```

```

00136     HttpResponse<String> res = http.send(request, HttpResponse.BodyHandlers.ofString());
00137     int status = res.statusCode();
00138     String body = res.body();
00139
00140     LOGGER.info("{} -> HTTP {}", action, status);
00141     LOGGER.info("Response body: '{}'", body);
00142
00143     try {
00144         return handler.handle(status, body);
00145     } catch (IOException e) {
00146         // Wrap parsing IO exceptions with more context (status + body) so the UI
00147         // receives a helpful message instead of a generic one.
00148         throw new ApiClientException("I/O error during " + action + " status: " + status + " body: " +
00149             body, e);
00150     }
00151
00152     // -----
00153     // Helpers genéricos
00154     // -----
00155
00169     private <T> T get(String path, Class<T> responseType) {
00170         return execute("GET " + path, () -> sendAndHandle("GET " + path,
00171             HttpRequest.newBuilder().uri(URI.create(baseUrl + path)).GET().header("Accept",
00172                 APPLICATION_JSON).build(),
00173                 (status, body) -> {
00174                     if (status == 204 || body == null || body.trim().isEmpty()) {
00175                         return null;
00176                     }
00177                     if (status < 200 || status >= 300) {
00178                         throw new ApiClientException("GET " + path + " failed with HTTP " + status + " body: " +
00179                             body);
00180                     }
00181                 });
00182
00196     private <T> List<T> getList(String path, TypeReference<List<T>> typeRef) {
00197         return execute("GET " + path, () -> sendAndHandle("GET " + path,
00198             HttpRequest.newBuilder().uri(URI.create(baseUrl + path)).GET().header("Accept",
00199                 APPLICATION_JSON).build(),
00200                 (status, body) -> {
00201                     if (status == 204 || body == null || body.trim().isEmpty()) {
00202                         return java.util.Collections.emptyList();
00203                     }
00204                     if (status < 200 || status >= 300) {
00205                         throw new ApiClientException("GET " + path + " failed with HTTP " + status + " body: " +
00206                             body);
00207                     }
00208                 });
00209
00225     private <T, R> R post(String path, T body, Class<R> responseType) {
00226         return execute("POST " + path, () -> {
00227             String json = mapper.writeValueAsString(body);
00228             HttpRequest req = HttpRequest.newBuilder().uri(URI.create(baseUrl + path))
00229                 .POST(HttpRequest.BodyPublishers.ofString(json)).header(CONTENT_TYPE,
00230                 APPLICATION_JSON).build();
00231
00232             return sendAndHandle("POST " + path, req, (status, responseBody) -> {
00233                 if (status < 200 || status >= 300) {
00234                     throw new ApiClientException("POST " + path + " failed with HTTP " + status + " body: " +
00235                         responseBody);
00236                 }
00237             });
00238         }
00239
00240     // -----
00241     // CRUD methods for Appetizers
00242     // -----
00243
00251     public List<Appetizer> getAllAppetizers() {
00252         return getList("/api/appetizers", new TypeReference<List<Appetizer>>() {});
00253     }
00254
00263     public Appetizer getAppetizerById(Long id) {
00264         return get(API_APPETIZERS + id, Appetizer.class);
00265     }
00266
00274     public Appetizer createAppetizer(Appetizer appetizer) {
00275         return post("/api/appetizers", appetizer, Appetizer.class);
00276     }
00277
00286     public Appetizer updateAppetizer(Long id, Appetizer appetizer) {

```

```
00287     String path = API_APPETIZERS + id;
00288     return execute("updateAppetizer id=" + id, () -> {
00289         String json = mapper.writeValueAsString(appetizer);
00290         HttpRequest req = HttpRequest.newBuilder().uri(URI.create(baseUrl + path))
00291             .PUT(HttpRequest.BodyPublishers.ofString(json)).header(CONTENT_TYPE,
00292             APPLICATION_JSON).build();
00293     }
00294     return sendAndHandle("PUT " + path, req, (status, body) -> {
00295         if (status < 200 || status >= 300) {
00296             throw new ApiClientException("PUT " + path + " failed with HTTP " + status + " body: " +
00297                 body);
00298         }
00299     });
00300 }
00301
00302 public void deleteAppetizer(Long id) {
00303     String path = API_APPETIZERS + id;
00304     execute("deleteAppetizer id=" + id, () -> sendAndHandle("DELETE " + path,
00305         HttpRequest.newBuilder().uri(URI.create(baseUrl + path)).DELETE().build(), (status, body) -> {
00306         if (status < 200 || status >= 300) {
00307             throw new ApiClientException("DELETE " + path + " failed with HTTP " + status + " body: " +
00308                 body);
00309         }
00310         return null;
00311     }));
00312 }
00313
00314 // -----
00315 // READ / UPDATE methods for Cashiers
00316 // -----
00317
00318 00320 public List<Cashier> getAllCashiers() {
00319     return getList("/api/cashiers", new TypeReference<List<Cashier>>() {});
00321 }
00322
00323 00341 public Cashier getCashierById(Long id) {
00324     return get("/api/cashiers/" + id, Cashier.class);
00325 }
00326
00327 00352 public Cashier getCashierByName(String name) {
00328     return get("/api/cashiers/name/" + name, Cashier.class);
00329 }
00330
00331 00365 public Cashier updateCashier(Long id, Cashier cashier) {
00332     String path = "/api/cashiers/" + id;
00333     return execute("updateCashier id=" + id, () -> {
00334         String json = mapper.writeValueAsString(cashier);
00335         HttpRequest req = HttpRequest.newBuilder().uri(URI.create(baseUrl + path))
00336             .PUT(HttpRequest.BodyPublishers.ofString(json)).header(CONTENT_TYPE,
00337             APPLICATION_JSON).build();
00338     }
00339     return sendAndHandle("PUT " + path, req, (status, body) -> {
00340         if (status < 200 || status >= 300) {
00341             throw new ApiClientException("PUT " + path + " failed with HTTP " + status + " body: " +
00342                 body);
00343         }
00344         return mapper.readValue(body, Cashier.class);
00345     });
00346 }
00347
00348 00381 // -----
00349 // CRUD methods for Drinks
00350 // -----
00351
00352 00392 public List<Drink> getAllDrinks() {
00353     return getList("/api/drinks", new TypeReference<List<Drink>>() {});
00354 }
00355
00356 00404 public Drink getDrinkById(Long id) {
00357     return get(API_DRINKS + id, Drink.class);
00358 }
00359
00360 00416 public Drink createDrink(Drink drink) {
00361     return post("/api/drinks", drink, Drink.class);
00362 }
00363
00364 00429 public Drink updateDrink(Long id, Drink drink) {
00365     String path = API_DRINKS + id;
00366     return execute("updateDrink id=" + id, () -> {
00367         String json = mapper.writeValueAsString(drink);
00368         HttpRequest req = HttpRequest.newBuilder().uri(URI.create(baseUrl + path))
00369             .PUT(HttpRequest.BodyPublishers.ofString(json)).header(CONTENT_TYPE,
00370             APPLICATION_JSON).build();
00371 }
```

```

00436     return sendAndHandle("PUT " + path, req, (status, body) -> {
00437         if (status < 200 || status >= 300) {
00438             throw new ApiClientException("PUT " + path + " failed with HTTP " + status + " body: " +
00439                 body);
00440             return mapper.readValue(body, Drink.class);
00441         });
00442     });
00443 }
00444
00452 public void deleteDrink(Long id) {
00453     String path = API_DRINKS + id;
00454     execute("deleteDrink id=" + id, () -> sendAndHandle("DELETE " + path,
00455         HttpRequest.newBuilder().uri(URI.create(baseUrl + path)).DELETE().build(), (status, body) -> {
00456             if (status < 200 || status >= 300) {
00457                 throw new ApiClientException("DELETE " + path + " failed with HTTP " + status + " body: " +
00458                     body);
00459             }
00460             return null;
00461         }));
00462 }
00463 // -----
00464 // CRUD methods for MainCourses
00465 // -----
00466
00474 public List<MainCourse> getAllMainCourses() {
00475     return getList("/api/maincourses", new TypeReference<List<MainCourse>>() {});
00476 }
00477
00486 public MainCourse getMainCourseById(Long id) {
00487     return get(API_MAINCOURSES + id, MainCourse.class);
00488 }
00489
00498 public MainCourse createMainCourse(MainCourse mainCourse) {
00499     return post("/api/maincourses", mainCourse, MainCourse.class);
00500 }
00501
00511 public MainCourse updateMainCourse(Long id, MainCourse mainCourse) {
00512     String path = API_MAINCOURSES + id;
00513     return execute("updateMainCourse id=" + id, () -> {
00514         String json = mapper.writeValueAsString(mainCourse);
00515         HttpRequest req = HttpRequest.newBuilder().uri(URI.create(baseUrl + path))
00516             .PUT(HttpRequest.BodyPublishers.ofString(json)).header(CONTENT_TYPE,
00517             APPLICATION_JSON).build();
00518
00519         return sendAndHandle("PUT " + path, req, (status, body) -> {
00520             if (status < 200 || status >= 300) {
00521                 throw new ApiClientException("PUT " + path + " failed with HTTP " + status + " body: " +
00522                     body);
00523             }
00524             return mapper.readValue(body, MainCourse.class);
00525         });
00526     });
00534 public void deleteMainCourse(Long id) {
00535     String path = API_MAINCOURSES + id;
00536     execute("deleteMainCourse id=" + id, () -> sendAndHandle("DELETE " + path,
00537         HttpRequest.newBuilder().uri(URI.create(baseUrl + path)).DELETE().build(), (status, body) -> {
00538             if (status < 200 || status >= 300) {
00539                 throw new ApiClientException("DELETE " + path + " failed with HTTP " + status + " body: " +
00540                     body);
00541             }
00542             return null;
00543         }));
00544 }
00545 // -----
00546 // Authentication methods
00547 // -----
00548
00556 public void login(String ignored) {
00557     // No-op: kept for compatibility.
00558 }
00559
00571 public User login(String username, String password) {
00572     String path = API_LOGIN;
00573     return execute("login for user=" + username, () -> {
00574         String jsonBody = mapper.writeValueAsString(Map.of(
00575             "username", username,
00576             "password", password));
00577
00578         HttpRequest request = HttpRequest.newBuilder().uri(URI.create(baseUrl + path))
00579             .header(CONTENT_TYPE,
00580             APPLICATION_JSON).POST(HttpRequest.BodyPublishers.ofString(jsonBody)).build();
00580     });
}

```

```
00581     return sendAndHandle("POST " + path, request, (status, body) -> {
00582         if (status == 200) {
00583             return mapper.readValue(body, User.class);
00584         } else {
00585             throw new ApiClientException("Login failed with status: " + status + " body: " + body);
00586         }
00587     });
00588 );
00589 }
00590 }
```

## 8.58 ApiClientException.java File Reference

### Classes

- class [es.ull.esit.app.middleware.ApiClientException](#)  
*Custom exception class for API client errors.*

### Packages

- package [es.ull.esit.app.middleware](#)

## 8.59 ApiClientException.java

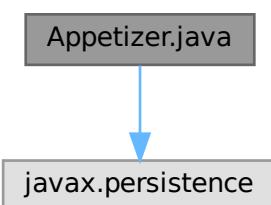
[Go to the documentation of this file.](#)

```
00001 package es.ull.esit.app.middleware;
00002
00009 public class ApiClientException extends RuntimeException {
00010
00016     public ApiClientException(String message) {
00017         super(message);
00018     }
00019
00026     public ApiClientException(String message, Throwable cause) {
00027         super(message, cause);
00028     }
00029 }
```

## 8.60 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](#)

## 8.61 server/src/main/java/es/ull/esit/server/middleware/model/ ← Appetizer.java

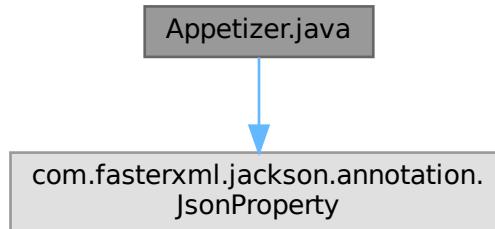
[Go to the documentation of this file.](#)

```

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 }
```

## 8.62 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](#)

## 8.63 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     {
00048         this.appetizersId = appetizersId;
00049         this.itemAppetizers = itemAppetizers;
00050         this.appetizersPrice = appetizersPrice;
00050         this.receiptId = receiptId;
  
```

```

00051    }
00052
00053    public Long getAppetizersId() {
00054        return appetizersId;
00055    }
00056
00057    public void setAppetizersId(Long appetizersId) {
00058        this.appetizersId = appetizersId;
00059    }
00060
00061
00062    public String getItemAppetizers() {
00063        return itemAppetizers;
00064    }
00065
00066    public void setItemAppetizers(String itemAppetizers) {
00067        this.itemAppetizers = itemAppetizers;
00068    }
00069
00070
00071    public Integer getAppetizersPrice() {
00072        return appetizersPrice;
00073    }
00074
00075    public void setAppetizersPrice(Integer appetizersPrice) {
00076        this.appetizersPrice = appetizersPrice;
00077    }
00078
00079
00080    public Long getReceiptId() {
00081        return receiptId;
00082    }
00083
00084    public void setReceiptId(Long receiptId) {
00085        this.receiptId = receiptId;
00086    }
00087
00088
00089    public void setReceiptId() {
00090        receiptId = null;
00091    }
00092
00093
00094
00095
00096
00097
00098
00099
00100
00101
00102
00103
00104
00105
00106
00107
00108
00109
00110
00111
00112
00113
00114
00115
00116
00117
00118
00119
00120
00121
00122
00123
00124 }
```

## 8.64 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](#)

## 8.65 BillResult.java

[Go to the documentation of this file.](#)

```

00001 package es.ull.esit.app.middleware.model;
00002
00003 public class BillResult {
00004
00005     private final double subTotal;
00006
00007     private final double vat;
00008
00009     private final double total;
00010
00011     public BillResult(double subTotal, double vat, double total) {
00012         this.subTotal = subTotal;
00013         this.vat = vat;
00014         this.total = total;
00015     }
00016
00017     public double getSubTotal() {
00018         return subTotal;
00019     }
00020
00021     public double getVat() {
00022         return vat;
00023     }
00024
00025     public double getTotal() {
00026         return total;
00027     }
00028
00029
00030
00031
00032
00033
00034
00035
00036
00037
00038
00039
00040
00041
00042
00043
00044
00045
00046
00047 }
```

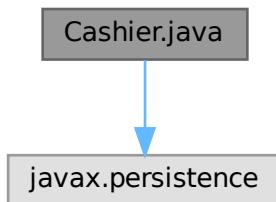
```

00048     return vat;
00049 }
00050
00056 public double getTotal() {
00057     return total;
00058 }
00059 }
```

## 8.66 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](#)

## 8.67 server/src/main/java/es/ull/esit/server/middleware/model/← Cashier.java

[Go to the documentation of this file.](#)

```

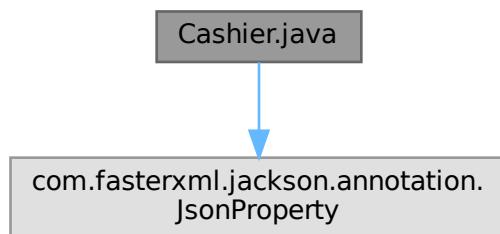
00001 package es.ull.esit.server.middleware.model;
00002
00003 import javax.persistence.*;
00004 import com.fasterxml.jackson.annotation.JsonIgnoreProperties;
00005
00012 @Entity
00013 @Table(name = "cashier")
00014 @JsonIgnoreProperties({"hibernateLazyInitializer", "handler"})
00015 public class Cashier {
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 }
```

## 8.68 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

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

[Go to the documentation of this file.](#)

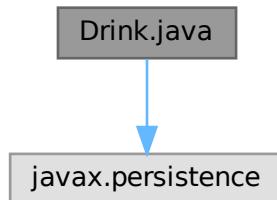
```
00001 package es.ull.esit.app.middleware.model;
00002
00003 import com.fasterxml.jackson.annotation.JsonProperty;
00004
00011 public class Cashier {
00012
00014     @JsonProperty("id")
00015     private Long id;
00016
00018     @JsonProperty("name")
00019     private String name;
00020
00022     @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 }
```

## 8.70 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](#)

## 8.71 server/src/main/java/es/ull/esit/server/middleware/model/Drink.java

[Go to the documentation of this file.](#)

```

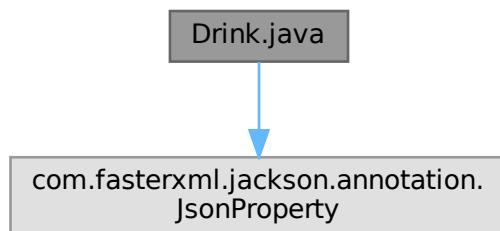
00001 package es.ull.esit.server.middleware.model;
00002
00003 import javax.persistence.*;
00004 import com.fasterxml.jackson.annotation.JsonIgnoreProperties;
00005
00012 @Entity
00013 @Table(name = "drinks")
00014 @JsonIgnoreProperties({ "hibernateLazyInitializer", "handler" })
00015 public class Drink {
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
00050 public Long getDrinksId() {
00051     return drinksId;
00052 }
00053
00054 public void setDrinksId(Long drinksId) {
00055     this.drinksId = drinksId;
00056 }
00057
00058 public String getItemDrinks() {
00059     return itemDrinks;
00060 }
00061
00062 public void setItemDrinks(String itemDrinks) {
00063     this.itemDrinks = itemDrinks;
00064 }
00065
00066 public Integer getDrinksPrice() {
00067     return drinksPrice;
00068 }
00069
00070 public void setDrinksPrice(Integer drinksPrice) {
00071     this.drinksPrice = drinksPrice;
00072 }
00073 }
```

## 8.72 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](#)

## 8.73 src/main/java/es/ull/esit/app/middleware/model/Drink.java

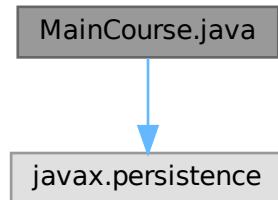
[Go to the documentation of this file.](#)

```
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 }
```

## 8.74 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](#)

## 8.75 server/src/main/java/es/ull/esit/server/middleware/model/MainCourse.java

[Go to the documentation of this file.](#)

```

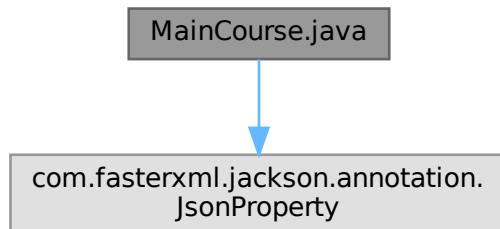
00001 package es.ull.esit.server.middleware.model;
00002
00003 import javax.persistence.*;
00004 import com.fasterxml.jackson.annotation.JsonIgnoreProperties;
00005
00012 @Entity
00013 @Table(name = "maincourse")
00014 @JsonIgnoreProperties({ "hibernateLazyInitializer", "handler" })
00015 public class 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 }
```

## 8.76 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](#)

## 8.77 src/main/java/es/ull/esit/app/middleware/model/MainCourse.java

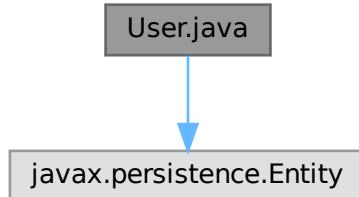
[Go to the documentation of this file.](#)

```
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 }
```

## 8.78 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](#)

## 8.79 server/src/main/java/es/ull/esit/server/middleware/model/User.java

[Go to the documentation of this file.](#)

```

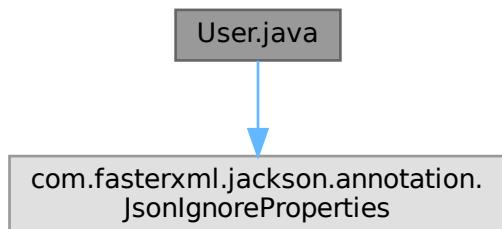
00001 package es.ull.esit.server.middleware.model;
00002
00003 import javax.persistence.Entity;
00004 import javax.persistence.Table;
00005 import javax.persistence.Id;
00006 import javax.persistence.GeneratedValue;
00007 import javax.persistence.GenerationType;
00008 import javax.persistence.Column;
00009
00010 import com.fasterxml.jackson.annotation.JsonIgnoreProperties;
00011 import com.fasterxml.jackson.annotation.JsonIgnore;
00012
00013 @Entity
00014 @Table(name = "users")
00015 @JsonIgnoreProperties({ "hibernateLazyInitializer", "handler" })
00016 public class User {
00017     @Id
00018     @GeneratedValue(strategy = GenerationType.IDENTITY)
00019     private Long id;
00020
00021     @Column(unique = true, nullable = false)
00022     private String username;
00023
00024     @Column(name = "password_hash", nullable = false)
00025     @JsonIgnore
00026     private String passwordHash;
00027
00028
00029
00030
00031
00032
00033
00034
00035
00036
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 }
```

## 8.80 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](#)

## 8.81 src/main/java/es/ull/esit/app/middleware/model/User.java

[Go to the documentation of this file.](#)

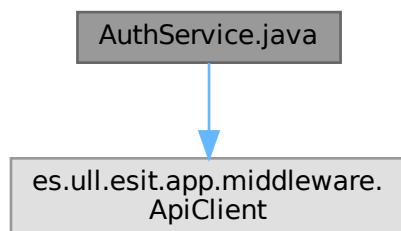
```

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 }
```

## 8.82 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

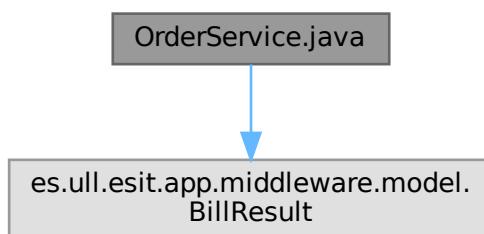
## 8.83 AuthService.java

[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
00037     public User authenticate(String username, String password) {
00038         // 1. Local Validation.
00039         if (username == null || username.trim().isEmpty()) {
00040             throw new IllegalArgumentException("Username cannot be empty.");
00041         }
00042         if (password == null || password.isEmpty()) {
00043             throw new IllegalArgumentException("Password cannot be empty.");
00044         }
00045
00046         // 2. Call API.
00047         return client.login(username, password);
00048     }
00049 }
```

## 8.84 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

## 8.85 OrderService.java

[Go to the documentation of this file.](#)

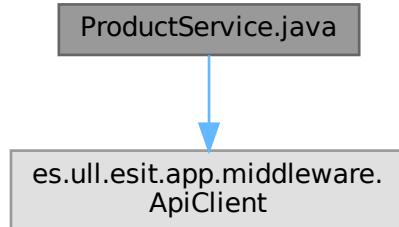
```

00001 package es.ull.esit.app.middleware.service;
00002
00003 import es.ull.esit.app.middleware.model.BillResult;
00004
00005 import java.io.FileNotFoundException;
00006 import java.io.IOException;
00007 import java.io.PrintWriter;
00008 import java.nio.charset.StandardCharsets;
00009 import java.nio.file.Files;
00010 import java.nio.file.Path;
00011
00012
00013 public class OrderService {
00014
00015     private static final double VAT_RATE = 0.15;
00016
00017
00018     public BillResult calculateBill(double itemsTotalSum) {
00019         double vat = itemsTotalSum * VAT_RATE;
00020         double total = itemsTotalSum + vat;
00021
00022         return new BillResult(
00023             Math.round(itemsTotalSum * 100.0) / 100.0,
00024             Math.round(vat * 100.0) / 100.0,
00025             Math.round(total * 100.0) / 100.0);
00026     }
00027
00028
00029     public void generateReceiptFile(int receiptNo, BillResult bill) throws FileNotFoundException {
00030
00031         try {
00032             Files.createDirectories(Path.of("receipts"));
00033
00034             Path file = Path.of("receipts", "billNo." + receiptNo + ".txt");
00035             try (PrintWriter output =
00036                  new PrintWriter(Files.newBufferedWriter(file, StandardCharsets.UTF_8))) {
00037
00038                 output.println(" Bill number is: " + receiptNo);
00039                 output.println("=====");
00040                 output.println("-----");
00041                 output.println("Subtotal is: " + bill.getSubTotal() + " SR");
00042                 output.println("vat: " + bill.getVat() + " SR");
00043                 output.println("Total is: " + bill.getTotal() + " SR");
00044                 output.println();
00045                 output.println("THANK YOU FOR ORDERING");
00046             }
00047
00048         } catch (IOException e) {
00049             FileNotFoundException fnfe = new FileNotFoundException(
00050                 "Could not create/write receipt file for receiptNo=" + receiptNo);
00051             fnfe.initCause(e);
00052             throw fnfe;
00053         }
00054     }
00055 }
00056 }
```

## 8.86 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](#)

## 8.87 ProductService.java

[Go to the documentation of this file.](#)

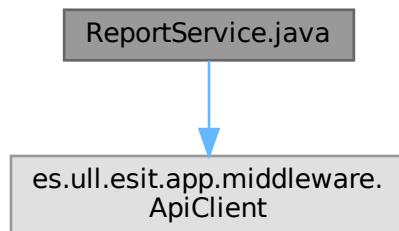
```
00001 package es.ull.esit.app.middleware.service;
00002
00003 import es.ull.esit.app.middleware.ApiClient;
00004 import es.ull.esit.app.middleware.model.Appetizer;
00005 import es.ull.esit.app.middleware.model.Drink;
00006 import es.ull.esit.app.middleware.model.MainCourse;
00007 import java.util.List;
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
00037     public List<Drink> getAllDrinks() {
00038         return client.getAllDrinks();
00039     }
00040
00047     public Drink getDrinkById(Long id) {
00048         return client.getDrinkById(id);
00049     }
00050
00059     public void addDrink(String name, String priceStr) {
00060         int price = validateAndParsePrice(priceStr);
00061         validateName(name);
```

```
00062
00063     Drink newDrink = new Drink(null, name, price, null);
00064     client.createDrink(newDrink);
00065 }
00066
00074 public void updateDrink(Long id, String name, String priceStr) {
00075     if (id == null) {
00076         throw new IllegalArgumentException("No drink selected!");
00077     }
00078
00079     int price = validateAndParsePrice(priceStr);
00080     validateName(name);
00081
00082     Drink updatedDrink = new Drink(id, name, price, null);
00083     client.updateDrink(id, updatedDrink);
00084 }
00085
00086 // ===== APPETIZERS LOGIC =====
00087
00093 public List<Appetizer> getAllAppetizers() {
00094     return client.getAllAppetizers();
00095 }
00096
00103 public Appetizer getAppetizerById(Long id) {
00104     return client.getAppetizerById(id);
00105 }
00106
00115 public void addAppetizer(String name, String priceStr) {
00116     int price = validateAndParsePrice(priceStr);
00117     validateName(name);
00118
00119     Appetizer newAppetizer = new Appetizer(null, name, price, null);
00120     client.createAppetizer(newAppetizer);
00121 }
00122
00130 public void updateAppetizer(Long id, String name, String priceStr) {
00131     if (id == null) {
00132         throw new IllegalArgumentException("No appetizer selected!");
00133     }
00134
00135     int price = validateAndParsePrice(priceStr);
00136     validateName(name);
00137
00138     Appetizer updatedAppetizer = new Appetizer(id, name, price, null);
00139     client.updateAppetizer(id, updatedAppetizer);
00140 }
00141
00142 // ===== MAIN COURSE LOGIC =====
00143
00149 public List<MainCourse> getAllMainCourses() {
00150     return client.getAllMainCourses();
00151 }
00152
00159 public MainCourse getMainCourseById(Long id) {
00160     return client.getMainCourseById(id);
00161 }
00162
00171 public void addMainCourse(String name, String priceStr) {
00172     int price = validateAndParsePrice(priceStr);
00173     validateName(name);
00174
00175     MainCourse newMainCourse = new MainCourse(null, name, price, null);
00176     client.createMainCourse(newMainCourse);
00177 }
00178
00186 public void updateMainCourse(Long id, String name, String priceStr) {
00187     if (id == null) {
00188         throw new IllegalArgumentException("No main course selected!");
00189     }
00190
00191     int price = validateAndParsePrice(priceStr);
00192     validateName(name);
00193
00194     MainCourse updatedMainCourse = new MainCourse(id, name, price, null);
00195     client.updateMainCourse(id, updatedMainCourse);
00196 }
00197
00198 // ===== VALIDATION HELPERS =====
00199
00206 private void validateName(String name) {
00207     if (name == null || name.trim().isEmpty()) {
00208         throw new IllegalArgumentException("Item name cannot be empty.");
00209     }
00210 }
00211
00218 private int validateAndParsePrice(String priceStr) {
00219     if (priceStr == null || priceStr.trim().isEmpty()) {
00220         throw new IllegalArgumentException("Price cannot be empty or null.");
00221     }
00222     return Integer.parseInt(priceStr);
00223 }
```

```
00223     throw new IllegalArgumentException("Price cannot be empty.");
00224 }
00225 try {
00226     int price = Integer.parseInt(priceStr.trim());
00227     if (price < 0) {
00228         throw new IllegalArgumentException("Price cannot be negative.");
00229     }
00230     return price;
00231 } catch (NumberFormatException e) {
00232     throw new IllegalArgumentException("Price must be a valid whole number.");
00233 }
00234 }
00235 }
```

## 8.88 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](#)

## 8.89 ReportService.java

[Go to the documentation of this file.](#)

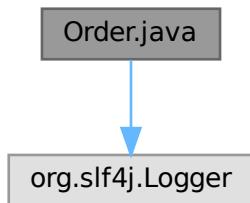
```
00001 package es.ull.esit.app.middleware.service;
00002
00003 import es.ull.esit.app.middleware.ApiClient;
00004 import es.ull.esit.app.middleware.model.Appetizer;
00005 import es.ull.esit.app.middleware.model.Cashier;
00006 import es.ull.esit.app.middleware.model.Drink;
00007 import es.ull.esit.app.middleware.model.MainCourse;
00008 import java.util.List;
00009
00016 public class ReportService {
00017     private final ApiClient client;
```

```

00020
00026     public ReportService(ApiClient client) {
00027         this.client = client;
00028     }
00029
00037     public List<Cashier> getCashierInfo() {
00038         return client.getAllCashiers();
00039     }
00040
00049     public String checkMenuStatus() {
00050         // Fetch lists to verify connectivity.
00051         List<Appetizer> appetizers = client.getAllAppetizers();
00052         List<Drink> drinks = client.getAllDrinks();
00053         List<MainCourse> mainCourses = client.getAllMainCourses();
00054
00055         return String.format(
00056             "Menu System Online: %d appetizers, %d drinks, %d main courses available.",
00057             appetizers.size(), drinks.size(), mainCourses.size());
00058     }
00059 }
```

## 8.90 Order.java File Reference

import org.slf4j.Logger;  
 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

## 8.91 Order.java

[Go to the documentation of this file.](#)

```

00001 package es.ull.esit.app;
00002
00003 import org.slf4j.Logger;
00004 import org.slf4j.LoggerFactory;
00005
00006 import es.ull.esit.app.middleware.ApiClient;
```

```
00007 import es.ull.esit.app.middleware.model.Appetizer;
00008 import es.ull.esit.app.middleware.model.BillResult;
00009 import es.ull.esit.app.middleware.model.Drink;
00010 import es.ull.esit.app.middleware.model.MainCourse;
00011 import es.ull.esit.app.middleware.service.OrderService;
00012 import es.ull.esit.app.middleware.service.ProductService;
00013
00014 import java.io.PrintWriter;
00015 import javax.swing.JOptionPane;
00016 import javax.swing.SwingUtilities;
00017 import javax.swing.table.DefaultTableModel;
00018
00031 public class Order extends javax.swing.JFrame {
00032
00034     private static final String PRIMARY_FONT = "Yu Gothic UI";
00035
00037     private static final String PRIMARY_FONT_LIGHT = "Yu Gothic UI Light";
00038
00040     private static final String RECEIPT_NO_PREFIX = "Receipt No. : ";
00041
00043     private static final String ID_COLUMN_HEADER = "ID";
00044
00046     private static final String ITEM_COLUMN_HEADER = "Item";
00047
00049     private static final String PRICE_COLUMN_HEADER = "Price (SR)";
00050
00052     private static final String QUANTITY_COLUMN_HEADER = "Qty";
00053
00055     private final transient ProductService productService;
00056
00058     private final transient OrderService orderService = new OrderService();
00059
00061     private static final Logger LOGGER = LoggerFactory.getLogger(Order.class);
00062
00064     private DefaultTableModel drinksModel;
00066     private DefaultTableModel appetizersModel;
00068     private DefaultTableModel mainsModel;
00069
00071     private transient BillResult lastBill;
00072
00074     double subTotal;
00076     double vat;
00078     double total;
00079
00081     int receiptNo = 1;
00082
00087     transient PrintWriter output;
00088
00095     public Order() {
00096         initComponents();
00097
00098         // Initialize the Service Layer.
00099         ApiClient client = new ApiClient("http://localhost:8080");
00100         this.productService = new ProductService(client);
00101
00102         // Initialize receipt number label.
00103         receiptNoLbl.setText(RECEIPT_NO_PREFIX + receiptNo);
00104
00105         // Configure tables and models.
00106         setupTables();
00107
00108         // Load menu items from database via REST API.
00109         loadMenuFromDatabase();
00110     }
00111
00125     Order(es.ull.esit.app.middleware.service.ProductService productService, boolean loadMenu) {
00126         initComponents();
00127         this.productService = productService;
00128
00129         // Initialize receipt number label.
00130         receiptNoLbl.setText(RECEIPT_NO_PREFIX + receiptNo);
00131
00132         // Configure tables and models.
00133         setupTables();
00134
00135         if (loadMenu) {
00136             loadMenuFromDatabase();
00137         }
00138     }
00139
00151     Order(es.ull.esit.app.middleware.service.ProductService productService) {
00152         this(productService, true);
00153     }
00154
00164     private void setupTables() {
00165         // DRINKS
00166         drinksModel = new DefaultTableModel(
```

```
00167     new Object[] { ID_COLUMN_HEADER, ITEM_COLUMN_HEADER, PRICE_COLUMN_HEADER,
QUANTITY_COLUMN_HEADER }, 0) {
00168     @Override
00169     public boolean isCellEditable(int row, int column) {
00170         // Only Qty column is editable
00171         return column == 3;
00172     }
00173
00174     @Override
00175     public Class<?> getColumnClass(int columnIndex) {
00176         return switch (columnIndex) {
00177             case 0 -> Long.class; // ID
00178             case 2, 3 -> Integer.class; // Price, Qty
00179             default -> String.class;
00180         };
00181     }
00182 };
00183 drinksTable.setModel(drinksModel);
00184
00185 // APPETIZERS
00186 appetizersModel = new DefaultTableModel(
00187     new Object[] { ID_COLUMN_HEADER, ITEM_COLUMN_HEADER, PRICE_COLUMN_HEADER,
QUANTITY_COLUMN_HEADER }, 0) {
00188     @Override
00189     public boolean isCellEditable(int row, int column) {
00190         return column == 3;
00191     }
00192
00193     @Override
00194     public Class<?> getColumnClass(int columnIndex) {
00195         return switch (columnIndex) {
00196             case 0 -> Long.class;
00197             case 2, 3 -> Integer.class;
00198             default -> String.class;
00199         };
00200     }
00201 };
00202 appetizersTable.setModel(appetizersModel);
00203
00204 // MAIN COURSES
00205 mainsModel = new DefaultTableModel(
00206     new Object[] { ID_COLUMN_HEADER, ITEM_COLUMN_HEADER, PRICE_COLUMN_HEADER,
QUANTITY_COLUMN_HEADER }, 0) {
00207     @Override
00208     public boolean isCellEditable(int row, int column) {
00209         return column == 3;
00210     }
00211
00212     @Override
00213     public Class<?> getColumnClass(int columnIndex) {
00214         return switch (columnIndex) {
00215             case 0 -> Long.class;
00216             case 2, 3 -> Integer.class;
00217             default -> String.class;
00218         };
00219     }
00220 };
00221 mainsTable.setModel(mainsModel);
00222 }
00223
00224 private void loadMenuFromDatabase() {
00225     new Thread(() -> {
00226         try {
00227             java.util.List<Drink> drinks = productService.getAllDrinks();
00228             java.util.List<Appetizer> appetizers = productService.getAllAppetizers();
00229             java.util.List<MainCourse> mains = productService.getAllMainCourses();
00230
00231             SwingUtilities.invokeLater(() -> {
00232                 fillDrinks(drinks);
00233                 fillAppetizers(appetizers);
00234                 fillMains(mains);
00235             });
00236         } catch (Exception ex) {
00237             LOGGER.error("Error loading menu from backend", ex);
00238             SwingUtilities.invokeLater(() -> JOptionPane.showMessageDialog(
00239                     this,
00240                     "Error loading menu from backend:\n" + ex.getMessage(),
00241                     "Menu loading error",
00242                     JOptionPane.ERROR_MESSAGE));
00243         }
00244     }).start();
00245 }
00246
00247 private void fillDrinks(java.util.List<Drink> drinks) {
00248     drinksModel.setRowCount(0);
00249     for (Drink d : drinks) {
00250
00251 }
```

```
00261     drinksModel.addRow(new Object[] {
00262         d.getDrinksId(),
00263         d.getItemDrinks(),
00264         d.getDrinksPrice(),
00265         0 // initial quantity
00266     });
00267 }
00268 }
00269
00270 private void fillAppetizers(java.util.List<Appetizer> appetizers) {
00271     appetizersModel.setRowCount(0);
00272     for (Appetizer a : appetizers) {
00273         appetizersModel.addRow(new Object[] {
00274             a.getAppetizersId(),
00275             a.getItemAppetizers(),
00276             a.getAppetizersPrice(),
00277             0
00278         });
00279     }
00280 }
00281
00282 private void fillMains(java.util.List<MainCourse> mains) {
00283     mainsModel.setRowCount(0);
00284     for (MainCourse m : mains) {
00285         mainsModel.addRow(new Object[] {
00286             m.getFoodId(),
00287             m.getItemFood(),
00288             m.getFoodPrice(),
00289             0
00290         });
00291     }
00292 }
00293
00294 private double sumFromModel(DefaultTableModel model) {
00295     double sum = 0.0;
00296     for (int row = 0; row < model.getRowCount(); row++) {
00297         Object qtyObj = model.getValueAt(row, 3); // Qty
00298         Object priceObj = model.getValueAt(row, 2); // Price
00299
00300         if (qtyObj == null || priceObj == null)
00301             continue;
00302
00303         int qty;
00304         int price;
00305         try {
00306             qty = ((Number) qtyObj).intValue();
00307             price = ((Number) priceObj).intValue();
00308         } catch (ClassCastException e) {
00309             // Fallback in case something comes as String
00310             qty = Integer.parseInt(qtyObj.toString());
00311             price = Integer.parseInt(priceObj.toString());
00312         }
00313
00314         if (qty > 0) {
00315             sum += qty * price;
00316         }
00317     }
00318     return sum;
00319 }
00320
00321 private void resetQtyColumn(DefaultTableModel model) {
00322     for (int row = 0; row < model.getRowCount(); row++) {
00323         model.setValueAt(0, row, 3); // column 3 is Qty
00324     }
00325 }
00326
00327 void loadPrice() {
00328     // No-op
00329 }
00330
00331 @SuppressWarnings("unchecked")
00332 // <editor-fold defaultstate="collapsed" desc="Generated
00333 // Code">/>GEN-BEGIN:initComponents
00334 private void initComponents() {
00335
00336     JPanel2 = new javax.swing.JPanel();
00337     drinksPnl = new javax.swing.JPanel();
00338     drinksScroll = new javax.swing.JScrollPane();
00339     drinksTable = new javax.swing.JTable();
00340     appetizerPnl = new javax.swing.JPanel();
00341     appetizersScroll = new javax.swing.JScrollPane();
00342     appetizersTable = new javax.swing.JTable();
00343     mainCoursePnl = new javax.swing.JPanel();
00344     mainsScroll = new javax.swing.JScrollPane();
00345     mainsTable = new javax.swing.JTable();
00346     JPanel1 = new javax.swing.JPanel();
00347     subTotalLbl = new javax.swing.JLabel();
```



```
00471 mainCoursePnl.setBackground(new java.awt.Color(248, 244, 230));
00472 mainCoursePnl.setBorder(javax.swing.BorderFactory.createTitledBorder(
00473     new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Main Course",
00474     javax.swing.border.TitledBorder.CENTER, javax.swing.border.TitledBorder.TOP,
00475     new java.awt.Font(PRIMARY_FONT, 0, 18))); // NOI18N
00476 mainCoursePnl.setPreferredSize(new java.awt.Dimension(260, 278));
00477
00478 mainsTable.setFont(new java.awt.Font(PRIMARY_FONT, 0, 14)); // NOI18N
00479 mainsTable.setModel(new javax.swing.table.DefaultTableModel(
00480     new Object[][] {
00481         ,
00482         new String[] {
00483             });
00484     });
00485 });
00486 mainsTable.getTableHeader().setReorderingAllowed(false);
00487 mainsScroll.setViewportView(mainsTable);
00488
00489 javax.swing.GroupLayout mainCoursePnlLayout = new javax.swing.GroupLayout(mainCoursePnl);
00490 mainCoursePnl.setLayout(mainCoursePnlLayout);
00491 mainCoursePnlLayout.setHorizontalGroup(
00492     mainCoursePnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00493         .addGroup(mainCoursePnlLayout.createSequentialGroup()
00494             .addGap(10, 10, 10)
00495             .addComponent(mainsScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 427, Short.MAX_VALUE)
00496             .addGap(10, 10, 10))
00497     mainCoursePnlLayout.setVerticalGroup(
00498         mainCoursePnlLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00499             .addGroup(mainCoursePnlLayout.createSequentialGroup()
00500                 .addGap(10, 10, 10)
00501                 .addComponent(mainsScroll, javax.swing.GroupLayout.DEFAULT_SIZE, 186, Short.MAX_VALUE)
00502                 .addGap(10, 10, 10)));
00503
00504 jPanel1.setBackground(new java.awt.Color(248, 244, 230));
00505 jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(
00506     new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true), "Receipt",
00507     javax.swing.border.TitledBorder.CENTER, javax.swing.border.TitledBorder.TOP,
00508     new java.awt.Font(PRIMARY_FONT, 0, 14))); // NOI18N
00509
00510 subTotalLbl.setFont(new java.awt.Font(PRIMARY_FONT_LIGHT, 0, 18)); // NOI18N
00511 subTotalLbl.setText("SubTotal: 0.0 SR");
00512
00513 vatLbl.setFont(new java.awt.Font(PRIMARY_FONT_LIGHT, 0, 18)); // NOI18N
00514 vatLbl.setText("VAT included: 0.0 SR");
00515
00516 totalLbl.setFont(new java.awt.Font(PRIMARY_FONT_LIGHT, 0, 18)); // NOI18N
00517 totalLbl.setText("Total: 0.0 SR");
00518
00519 receiptNoLbl.setFont(new java.awt.Font(PRIMARY_FONT_LIGHT, 0, 14)); // NOI18N
00520 receiptNoLbl.setText(RECEIPT_NO_PREFIX + "0");
00521
00522 javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
00523 jPanel1.setLayout(jPanel1Layout);
00524 jPanel1Layout.setHorizontalGroup(
00525     jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00526         .addGroup(jPanel1Layout.createSequentialGroup()
00527             .addGap(10, 10, 10)
00528             .addGroup(jPanel1Layout.createSequentialGroup()
00529                 .addComponent(subTotalLbl)
00530                 .addComponent(vatLbl)
00531                 .addComponent(totalLbl)
00532                 .addComponent(receiptNoLbl))
00533                 .addGap(10, 10, 10))
00534     jPanel1Layout.setVerticalGroup(
00535         jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00536             .addGroup(jPanel1Layout.createSequentialGroup()
00537                 .addGap(10, 10, 10)
00538                 .addComponent(subTotalLbl)
00539                 .addGap(10, 10, 10)
00540                 .addComponent(vatLbl)
00541                 .addGap(10, 10, 10)
00542                 .addComponent(totalLbl)
00543                 .addGap(10, 10, 10))
00544             .addComponent(receiptNoLbl)));
00545
00546 payBtn.setFont(new java.awt.Font("Lucida Grande", 1, 18)); // NOI18N
00547 payBtn.setText("Pay");
00548 payBtn.addActionListener(this::payBtnActionPerformed);
00549
00550 newReceiptBtn.setFont(new java.awt.Font(PRIMARY_FONT, 1, 14)); // NOI18N
00551 newReceiptBtn.setText("New Receipt");
00552 newReceiptBtn.addActionListener(this::newReceiptBtnActionPerformed);
00553
00554 saveReceiptBtn.setFont(new java.awt.Font(PRIMARY_FONT, 1, 14)); // NOI18N
00555 saveReceiptBtn.setText("Save Receipt");
00556 saveReceiptBtn.addActionListener(this::saveReceiptBtnActionPerformed);
```



```

00635         .addGap(30, 30, 30)));
00636
00637     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
00638     getContentPane().setLayout(layout);
00639     layout.setHorizontalGroup(
00640         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00641             .addComponent(jPanel12, javax.swing.GroupLayout.DEFAULT_SIZE,
00642                 javax.swing.GroupLayout.DEFAULT_SIZE,
00643                 Short.MAX_VALUE));
00644     layout.setVerticalGroup(
00645         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00646             .addGroup(layout.createSequentialGroup()
00647                 .addComponent(jPanel12, javax.swing.GroupLayout.DEFAULT_SIZE,
00648                     javax.swing.GroupLayout.DEFAULT_SIZE,
00649                     Short.MAX_VALUE)
00650             .addGap(0, 0, 0)
00651             .addContainerGap())));
00652 } // </editor-fold> //GEN-END:initComponents
00653
00654 private void payBtnActionPerformed(java.awt.event.ActionEvent evt) {} //GEN-FIRST:event_payBtnActionPerformed
00655 double itemsTotal = 0.0;
00656
00657 itemsTotal += sumFromModel(drinksModel);
00658 itemsTotal += sumFromModel(appetizersModel);
00659 itemsTotal += sumFromModel(mainsModel);
00660
00661 if (itemsTotal == 0.0) {
00662     JOptionPane.showMessageDialog(
00663         this,
00664         "Please select at least one item (Qty > 0).",
00665         "No items selected",
00666         JOptionPane.INFORMATION_MESSAGE);
00667     return;
00668 }
00669
00670 // Calculate bill using the dedicated service
00671 lastBill = orderService.calculateBill(itemsTotal);
00672
00673 subTotal = lastBill.getSubTotal();
00674 vat = lastBill.getVat();
00675 total = lastBill.getTotal();
00676
00677 subTotalLbl.setText("SubTotal: " + subTotal + " SR");
00678 vatLbl.setText("VAT included: " + vat + " SR");
00679 totalLbl.setText("Total: " + total + " SR");
00680
00681 JOptionPane.showMessageDialog(this, "Paid successfully");
00682 } // GEN-LAST:event_payBtnActionPerformed
00683
00684 private void saveReceiptBtnActionPerformed(java.awt.event.ActionEvent evt) {} //GEN-FIRST:event_saveReceiptBtnActionPerformed
00685 try {
00686     if (lastBill == null || lastBill.getTotal() == 0.0) {
00687         JOptionPane.showMessageDialog(
00688             this,
00689             "There is no paid bill to save.\nPlease press Pay first.",
00690             "No bill",
00691             JOptionPane.INFORMATION_MESSAGE);
00692     }
00693 }
00694
00695 orderService.generateReceiptFile(receiptNo, lastBill);
00696
00697 JOptionPane.showMessageDialog(
00698     this,
00699     "Receipt number: " + receiptNo + " has been saved successfully.",
00700     "Receipt saved",
00701     JOptionPane.INFORMATION_MESSAGE);
00702
00703 } catch (Exception ex) {
00704     LOGGER.error("Error saving receipt", ex);
00705     JOptionPane.showMessageDialog(
00706         this,
00707         "Error saving receipt:\n" + ex.getMessage(),
00708         "Error",
00709         JOptionPane.ERROR_MESSAGE);
00710 }
00711 } // GEN-LAST:event_saveReceiptBtnActionPerformed
00712
00713 private void newReceiptBtnActionPerformed(java.awt.event.ActionEvent evt) {} //GEN-FIRST:event_newReceiptBtnActionPerformed
00714 if (total == 0.0) {
00715     // Nothing to reset
00716     return;
00717 }
00718
00719
00720
00721
00722
00723
00724
00725
00726
00727
00728
00729
00730
00731
00732
00733
00734
00735
00736
00737
00738
00739
00740
00741
00742
00743
00744
00745
00746
00747
00748
00749
00750
00751
00752
00753
00754
00755

```

```

00756     }
00757
00758     resetQtyColumn(drinksModel);
00759     resetQtyColumn(appetizersModel);
00760     resetQtyColumn(mainsModel);
00761
00762     subTotal = 0.0;
00763     vat = 0.0;
00764     total = 0.0;
00765     lastBill = null;
00766
00767     subTotalLbl.setText("SubTotal: 0.0 SR");
00768     vatLbl.setText("VAT included: 0.0 SR");
00769     totalLbl.setText("Total: 0.0 SR");
00770
00771     receiptNo++;
00772     receiptNoLbl.setText(RECEIPT_NO_PREFIX + receiptNo);
00773 // GEN-LAST:event_newReceiptBtnActionPerformed
00774
00784     private void goBackMenueBtnActionPerformed(java.awt.event.ActionEvent evt) {//
00785 // GEN-FIRST:event_goBackMenueBtnActionPerformed
00786     this.dispose();
00787     new Login().setVisible(true);
00788 // GEN-LAST:event_goBackMenueBtnActionPerformed
00789
00798     public static void main(String[] args) {
00799         /* Set the Nimbus look and feel */
00800         // <editor-fold defaultstate="collapsed" desc=" Look and feel setting code
00801         // (optional) ">
00802         /*
00803             * If Nimbus (introduced in Java SE 6) is not available, stay with the default
00804             * look and feel.
00805             * For details see
00806             * http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
00807             */
00808         try {
00809             for (javax.swing.UIManager.LookAndFeelInfo info :
00810                 javax.swing.UIManager.getInstalledLookAndFeels()) {
00811                 if ("Nimbus".equals(info.getName())) {
00812                     javax.swing.UIManager.setLookAndFeel(info.getClassName());
00813                     break;
00814                 }
00815             } catch (ClassNotFoundException | javax.swing.UnsupportedLookAndFeelException |
00816 InstantiationException
00817             | IllegalAccessException ex) {
00818                 java.util.logging.Logger.getLogger(Order.class.getName()).log(java.util.logging.Level.SEVERE,
00819                         null, ex);
00820             }
00821         /* Create and display the form */
00822         java.awt.EventQueue.invokeLater(() -> new Order().setVisible(true));
00823     }
00824
00825 // Variables declaration - do not modify//GEN-BEGIN:variables
00826 // Sonarqube rule java:S1450 must be ignored here as these variables are
00827 // auto-generated by the Form Editor and need to remain as instance variables.
00828 private javax.swing.JPanel appetizerPnl;
00829 private javax.swing.JPanel drinksPnl;
00830 private javax.swing.JPanel mainCoursePnl;
00831 private javax.swing.JTable appetizersTable;
00832 private javax.swing.JScrollPane appetizersScroll;
00833 private javax.swing.JTable drinksTable;
00834 private javax.swing.JScrollPane drinksScroll;
00835 private javax.swing.JButton goBackMenueBtn;
00836 private javax.swing.JLabel jLabel1;
00837 private javax.swing.JLabel jLabel2;
00838 private javax.swing.JPanel jPanel1;
00839 private javax.swing.JPanel jPanel2;
00840 private javax.swing.JPanel mainsTable;
00841 private javax.swing.JScrollPane mainsScroll;
00842 private javax.swing.JButton newReceiptBtn;
00843 private javax.swing.JButton payBtn;
00844 private javax.swing.JLabel receiptNoLbl;
00845 private javax.swing.JButton saveReceiptBtn;
00846 private javax.swing.JLabel subTotalLbl;
00847 private javax.swing.JLabel totalLbl;
00848 private javax.swing.JLabel vatLbl;
00849 // End of variables declaration//GEN-END:variables
00871 }

```

# Index

00-create-db.sql, 237  
01-tables.sql, 237  
02-procedures.sql, 240  
03-triggers.sql, 240  
04-privileges.sql, 241  
05-data.sql, 241

AboutUs  
    es.ull.esit.app.AboutUs, 19

AboutUs.java, 263, 264

addAppetizer  
    es.ull.esit.app.middleware.service.ProductService, 206

addDrink  
    es.ull.esit.app.middleware.service.ProductService, 206

addMainCourse  
    es.ull.esit.app.middleware.service.ProductService, 207

AdminLogin  
    es.ull.esit.app.AdminLogin, 28

AdminLogin.java, 266

AdminProducts  
    es.ull.esit.app.AdminProducts, 38

AdminProducts.java, 269

API\_APPETIZERS  
    es.ull.esit.app.middleware.ApiClient, 71

API\_DRINKS  
    es.ull.esit.app.middleware.ApiClient, 71

API\_LOGIN  
    es.ull.esit.app.middleware.ApiClient, 71

API\_MAINCOURSES  
    es.ull.esit.app.middleware.ApiClient, 71

ApiClient  
    es.ull.esit.app.middleware.ApiClient, 57

ApiClient.java, 290, 291

ApiClientException  
    es.ull.esit.app.middleware.ApiClientException, 74, 75

ApiClientException.java, 295

Appetizer  
    es.ull.esit.app.middleware.model.Appetizer, 77  
    es.ull.esit.server.middleware.model.Appetizer, 83

Appetizer.java, 295–297

AppetizerController.java, 248

appetizerPnl  
    es.ull.esit.app.Order, 195

appetizerRepository  
    es.ull.esit.server.controller.AppetizerController, 89  
    es.ull.esit.server.controller.MenuController, 174

AppetizerRepository.java, 258

appetizersId  
    es.ull.esit.app.middleware.model.Appetizer, 80  
    es.ull.esit.server.middleware.model.Appetizer, 85

appetizersModel  
    es.ull.esit.app.Order, 195

appetizersPrice  
    es.ull.esit.app.middleware.model.Appetizer, 80  
    es.ull.esit.server.middleware.model.Appetizer, 85

appetizersScroll  
    es.ull.esit.app.Order, 195

appetizersTable  
    es.ull.esit.app.Order, 195

APPLICATION\_JSON  
    es.ull.esit.app.middleware.ApiClient, 72

ApplicationLauncher.java, 281

AuthController.java, 249, 250

authenticate  
    es.ull.esit.app.middleware.service.AuthService, 97

AuthService  
    es.ull.esit.app.middleware.service.AuthService, 97

authService  
    es.ull.esit.app.Login, 154

AuthService.java, 310, 311

baseUrl  
    es.ull.esit.app.middleware.ApiClient, 72

BillResult  
    es.ull.esit.app.middleware.model.BillResult, 100

BillResult.java, 298

calculateBill  
    es.ull.esit.app.middleware.service.OrderService, 202

call  
    es.ull.esit.app.middleware.ApiClient.IOCallable< T >, 146

Cashier  
    es.ull.esit.app.middleware.model.Cashier, 103  
    es.ull.esit.server.middleware.model.Cashier, 107

Cashier.java, 299–301

CashierController.java, 251

CashierLogin  
    es.ull.esit.app.CashierLogin, 117

CashierLogin.java, 282

cashierRepository  
    es.ull.esit.server.controller.CashierController, 113

CashierRepository.java, 259

cashierSupplier  
    es.ull.esit.app.CashierLogin, 123

checkDatabase  
     es.ull.esit.server.controller.HealthController, 144  
 checkMenuStatus  
     es.ull.esit.app.middleware.service.ReportService,  
         218  
 client  
     es.ull.esit.app.middleware.service.AuthService, 98  
     es.ull.esit.app.middleware.service.ProductService,  
         216  
     es.ull.esit.app.middleware.service.ReportService,  
         220  
 CONTENT\_TYPE  
     es.ull.esit.app.middleware.ApiClient, 72  
 createAppetizer  
     es.ull.esit.app.middleware.ApiClient, 58  
     es.ull.esit.server.controller.AppetizerController, 87  
 createDrink  
     es.ull.esit.app.middleware.ApiClient, 58  
     es.ull.esit.server.controller.DrinkController, 139  
 createMainCourse  
     es.ull.esit.app.middleware.ApiClient, 59  
     es.ull.esit.server.controller.MainCourseController,  
         169  
 dataSource  
     es.ull.esit.server.controller.HealthController, 145  
 deleteAppetizer  
     es.ull.esit.app.middleware.ApiClient, 60  
     es.ull.esit.server.controller.AppetizerController, 87  
 deleteDrink  
     es.ull.esit.app.middleware.ApiClient, 60  
     es.ull.esit.server.controller.DrinkController, 139  
 deleteMainCourse  
     es.ull.esit.app.middleware.ApiClient, 61  
     es.ull.esit.server.controller.MainCourseController,  
         169  
 Drink  
     es.ull.esit.app.middleware.model.Drink, 129  
     es.ull.esit.server.middleware.model.Drink, 135  
 Drink.java, 302–304  
 DrinkController.java, 252, 253  
 drinkRepository  
     es.ull.esit.server.controller.DrinkController, 141  
     es.ull.esit.server.controller.MenuController, 174  
 DrinkRepository.java, 260  
 drinksId  
     es.ull.esit.app.middleware.model.Drink, 132  
     es.ull.esit.server.middleware.model.Drink, 137  
 drinksModel  
     es.ull.esit.app.Order, 195  
 drinksPnl  
     es.ull.esit.app.Order, 195  
 drinksPrice  
     es.ull.esit.app.middleware.model.Drink, 132  
     es.ull.esit.server.middleware.model.Drink, 137  
 drinksScroll  
     es.ull.esit.app.Order, 196  
 drinksTable  
     es.ull.esit.app.Order, 196  
 ERROR\_CHECK\_MENU\_STATUS  
     es.ull.esit.app.CashierLogin, 123  
 es.ull.esit.app, 13  
 es.ull.esit.app.AboutUs, 17  
     AboutUs, 19  
     initComponents, 20  
     jButton3, 23  
     jButton3ActionPerformed, 21  
     jLabel1, 23  
     jPanel1, 23  
     jScrollPane1, 23  
     main, 22  
     ourStory, 23  
     textBlock, 23  
 es.ull.esit.app.AdminLogin, 24  
     AdminLogin, 28  
     initComponents, 28  
     jButton1, 32  
     jButton1ActionPerformed, 30  
     jButton2, 32  
     jButton2ActionPerformed, 30  
     jButton3, 33  
     jButton3ActionPerformed, 31  
     jLabel1, 33  
     jLabel3, 33  
     jPanel1, 33  
     LOGGER, 33  
     main, 31  
     PRIMARY\_FONT, 34  
 es.ull.esit.app.AdminProducts, 34  
     AdminProducts, 38  
     FIRST\_COLUMN\_HEADER, 52  
     initComponents, 39  
     jButton1ActionPerformed, 45  
     jButton2ActionPerformed, 45  
     jButton3ActionPerformed, 46  
     jButton4ActionPerformed, 46  
     jButton5ActionPerformed, 47  
     jButton6ActionPerformed, 48  
     jButton7ActionPerformed, 48  
     jTable1KeyPressed, 49  
     jTable1MouseClicked, 49  
     jTable2MouseClicked, 50  
     jTable3MouseClicked, 50  
     LOGGER, 52  
     main, 51  
     PRIMARY\_FONT, 53  
     productService, 53  
     refreshAllTables, 52  
     SECOND\_COLUMN\_HEADER, 53  
     SECONDARY\_FONT, 53  
     THIRD\_COLUMN\_HEADER, 53  
     UPDATE\_STRING, 54  
 es.ull.esit.app.ApplicationLauncher, 91  
     loginFactory, 92  
     main, 92  
 es.ull.esit.app.CashierLogin, 113  
     CashierLogin, 117

cashierSupplier, 123  
ERROR\_CHECK\_MENU\_STATUS, 123  
initComponents, 118  
jButton1, 123  
jButton1ActionPerformed, 120  
jButton2, 123  
jButton2ActionPerformed, 120  
jButton3, 124  
jButton3ActionPerformed, 121  
jLabel1, 124  
jPanel1, 124  
LOGGER, 124  
main, 121  
PRIMARY\_FONT, 124  
reportService, 125  
updateWelcomeMessage, 122  
WARN\_FETCH\_CASHIER\_STATS, 125  
welcomeTxt, 125  
es.ull.esit.app.Login, 146  
authService, 154  
initComponents, 151  
jButton1, 154  
jButton1ActionPerformed, 152  
jLabel1, 154  
jLabel3, 155  
jPanel1, 155  
jPasswordField1, 155  
Login, 150  
LOGIN\_STRING, 155  
loginFactory, 155  
main, 153  
PRIMARY\_FONT, 156  
usernametxt, 156  
usertypecmb, 156  
usertypecmbActionPerformed, 154  
es.ull.esit.app.middleware, 13  
es.ull.esit.app.middleware.ApiClient, 54  
API\_APPETIZERS, 71  
API\_DRINKS, 71  
API\_LOGIN, 71  
API\_MAINCOURSES, 71  
ApiClient, 57  
APPLICATION\_JSON, 72  
baseUrl, 72  
CONTENT\_TYPE, 72  
createAppetizer, 58  
createDrink, 58  
createMainCourse, 59  
deleteAppetizer, 60  
deleteDrink, 60  
deleteMainCourse, 61  
getAllAppetizers, 61  
getAllCashiers, 62  
getAllDrinks, 62  
getAllMainCourses, 63  
getAppetizerById, 63  
getCashierById, 64  
getCashierByName, 65  
getDrinkById, 65  
getMainCourseById, 66  
http, 72  
LOGGER, 72  
login, 66, 67  
mapper, 73  
updateAppetizer, 68  
updateCashier, 69  
updateDrink, 69  
updateMainCourse, 70  
es.ull.esit.app.middleware.ApiClient.IOCallable< T >, 145  
call, 146  
es.ull.esit.app.middleware.ApiClient.ResponseHandler< R >, 220  
handle, 221  
es.ull.esit.app.middleware.ApiClientException, 73  
ApiClientException, 74, 75  
es.ull.esit.app.middleware.model, 14  
es.ull.esit.app.middleware.model.Appetizer, 75  
Appetizer, 77  
appetizersId, 80  
appetizersPrice, 80  
getAppetizersId, 78  
getAppetizersPrice, 78  
getItemAppetizers, 78  
getReceiptId, 78  
itemAppetizers, 81  
receiptId, 81  
setAppetizersId, 79  
setAppetizersPrice, 79  
setItemAppetizers, 79  
setReceiptId, 80  
es.ull.esit.app.middleware.model.BillResult, 99  
BillResult, 100  
getSubTotal, 100  
getTotal, 100  
getVat, 100  
subTotal, 101  
total, 101  
vat, 101  
es.ull.esit.app.middleware.model.Cashier, 102  
Cashier, 103  
getId, 104  
getName, 104  
getSalary, 104  
id, 105  
name, 106  
salary, 106  
setId, 104  
setName, 105  
setSalary, 105  
es.ull.esit.app.middleware.model.Drink, 127  
Drink, 129  
drinksId, 132  
drinksPrice, 132  
getDrinksId, 130  
getDrinksPrice, 130

getItemDrinks, 130  
 getReceiptId, 130  
 itemDrinks, 133  
 receiptId, 133  
 setDrinksId, 131  
 setDrinksPrice, 131  
 setItemDrinks, 131  
 setReceiptId, 132  
 es.ul.esit.app.middleware.model.MainCourse, 158  
 foodId, 162  
 foodPrice, 162  
 getFoodId, 160  
 getFoodPrice, 160  
 getItemFood, 160  
 getReceiptId, 160  
 itemFood, 163  
 MainCourse, 159  
 receiptId, 163  
 setFoodId, 161  
 setFoodPrice, 161  
 setItemFood, 161  
 setReceiptId, 162  
 es.ul.esit.app.middleware.model.User, 224  
 getRole, 226  
 getUsername, 226  
 isAdmin, 227  
 role, 228  
 setRole, 227  
 setUsername, 227  
 User, 226  
 username, 228  
 es.ul.esit.app.middleware.service, 14  
 es.ul.esit.app.middleware.service.AuthService, 95  
 authenticate, 97  
 AuthService, 97  
 client, 98  
 es.ul.esit.app.middleware.service.OrderService, 202  
 calculateBill, 202  
 generateReceiptFile, 202  
 VAT\_RATE, 203  
 es.ul.esit.app.middleware.service.ProductService, 204  
 addAppetizer, 206  
 addDrink, 206  
 addMainCourse, 207  
 client, 216  
 getAllAppetizers, 208  
 getAllDrinks, 208  
 getAllMainCourses, 209  
 getAppetizerById, 209  
 getDrinkById, 210  
 getMainCourseById, 211  
 ProductService, 205  
 updateAppetizer, 211  
 updateDrink, 212  
 updateMainCourse, 213  
 validateAndParsePrice, 214  
 validateName, 215  
 es.ul.esit.app.middleware.service.ReportService, 217  
 checkMenuStatus, 218  
 client, 220  
 getCashierInfo, 219  
 ReportService, 218  
 es.ul.esit.app.Order, 175  
 appetizerPnl, 195  
 appetizersModel, 195  
 appetizersScroll, 195  
 appetizersTable, 195  
 drinksModel, 195  
 drinksPnl, 195  
 drinksScroll, 196  
 drinksTable, 196  
 fillAppetizers, 181  
 fillDrinks, 181  
 fillMains, 182  
 goBackMenueBtn, 196  
 goBackMenueBtnActionPerformed, 183  
 ID\_COLUMN\_HEADER, 196  
 initComponents, 183  
 ITEM\_COLUMN\_HEADER, 196  
 jLabel1, 197  
 jLabel2, 197  
 jPanel1, 197  
 jPanel2, 197  
 lastBill, 197  
 loadMenuFromDatabase, 187  
 LOGGER, 198  
 main, 188  
 mainCoursePnl, 198  
 mainsModel, 198  
 mainsScroll, 198  
 mainsTable, 198  
 newReceiptBtn, 199  
 newReceiptBtnActionPerformed, 189  
 Order, 180  
 orderService, 199  
 payBtn, 199  
 payBtnActionPerformed, 190  
 PRICE\_COLUMN\_HEADER, 199  
 PRIMARY\_FONT, 199  
 PRIMARY\_FONT\_LIGHT, 200  
 productService, 200  
 QUANTITY\_COLUMN\_HEADER, 200  
 RECEIPT\_NO\_PREFIX, 200  
 receiptNoLbl, 200  
 resetQtyColumn, 191  
 saveReceiptBtn, 201  
 saveReceiptBtnActionPerformed, 191  
 setupTables, 192  
 subTotalLbl, 201  
 sumFromModel, 193  
 totalLbl, 201  
 vatLbl, 201  
 es.ul.esit.server, 14  
 es.ul.esit.server.config, 15  
 es.ul.esit.server.config.SecurityConfig, 222  
 filterChain, 223

passwordEncoder, 224  
es.ull.esit.server.controller, 15  
es.ull.esit.server.controller.AppetizerController, 86  
    appetizerRepository, 89  
    createAppetizer, 87  
    deleteAppetizer, 87  
    getAllAppetizers, 88  
    getAppetizerById, 88  
    updateAppetizer, 89  
es.ull.esit.server.controller.AuthController, 93  
    LOG, 94  
    login, 94  
    passwordEncoder, 94  
    userRepository, 95  
es.ull.esit.server.controller.AuthController.LoginRequest, 157  
    password, 157  
    username, 157  
es.ull.esit.server.controller.CashierController, 110  
    cashierRepository, 113  
    getAllCashiers, 111  
    getCashierById, 111  
    getCashierByName, 112  
    updateCashier, 112  
es.ull.esit.server.controller.DrinkController, 138  
    createDrink, 139  
    deleteDrink, 139  
    drinkRepository, 141  
    getAllDrinks, 140  
    getDrinkById, 140  
    updateDrink, 141  
es.ull.esit.server.controller.HealthController, 143  
    checkDatabase, 144  
    dataSource, 145  
    health, 144  
es.ull.esit.server.controller.MainCourseController, 168  
    createMainCourse, 169  
    deleteMainCourse, 169  
    getAllMainCourses, 170  
    getMainCourseById, 170  
    mainCourseRepository, 171  
    updateMainCourse, 171  
es.ull.esit.server.controller.MenuController, 173  
    appetizerRepository, 174  
    drinkRepository, 174  
    getFullMenu, 174  
    mainCourseRepository, 175  
es.ull.esit.server.middleware.model, 15  
es.ull.esit.server.middleware.model.Appetizer, 82  
    Appetizer, 83  
    appetizersId, 85  
    appetizersPrice, 85  
    getAppetizersId, 84  
    getAppetizersPrice, 84  
    getItemAppetizers, 84  
    itemAppetizers, 86  
    setAppetizersId, 84  
    setAppetizersPrice, 85  
    setItemAppetizers, 85  
es.ull.esit.server.middleware.model.Cashier, 106  
    Cashier, 107  
    getId, 108  
    getName, 108  
    getSalary, 108  
    id, 110  
    name, 110  
    salary, 110  
    setId, 108  
    setName, 109  
    setSalary, 109  
es.ull.esit.server.middleware.model.Drink, 134  
    Drink, 135  
    drinksId, 137  
    drinksPrice, 137  
    getDrinksId, 136  
    getDrinksPrice, 136  
    getItemDrinks, 136  
    itemDrinks, 138  
    setDrinksId, 136  
    setDrinksPrice, 137  
    setItemDrinks, 137  
es.ull.esit.server.middleware.model.MainCourse, 164  
    foodId, 167  
    foodPrice, 167  
    getFoodId, 166  
    getFoodPrice, 166  
    getItemFood, 166  
    itemFood, 168  
    MainCourse, 165  
    setFoodId, 166  
    setFoodPrice, 167  
    setItemFood, 167  
es.ull.esit.server.middleware.model.User, 228  
    getId, 230  
    getPasswordHash, 230  
    getRole, 230  
    getUsername, 230  
    id, 232  
    passwordHash, 232  
    role, 232  
    setId, 231  
    setPasswordHash, 231  
    setRole, 231  
    setUsername, 232  
    username, 232  
es.ull.esit.server.repo, 16  
es.ull.esit.server.repo.AppetizerRepository, 90  
es.ull.esit.server.repo.CashierRepository, 126  
    findByName, 127  
es.ull.esit.server.repo.DrinkRepository, 142  
es.ull.esit.server.repo.MainCourseRepository, 172  
es.ull.esit.server.repo.UserRepository, 233  
    findByUsername, 234  
es.ull.esit.server.RestaurantApplication, 221  
    main, 222  
fillAppetizers

es.ul.esit.app.Order, 181  
 fillDrinks  
     es.ul.esit.app.Order, 181  
 fillMains  
     es.ul.esit.app.Order, 182  
 filterChain  
     es.ul.esit.server.config.SecurityConfig, 223  
 findByName  
     es.ul.esit.server.repo.CashierRepository, 127  
 findByUsername  
     es.ul.esit.server.repo.UserRepository, 234  
 FIRST\_COLUMN\_HEADER  
     es.ul.esit.app.AdminProducts, 52  
 foodId  
     es.ul.esit.app.middleware.model.MainCourse, 162  
     es.ul.esit.server.middleware.model.MainCourse,  
         167  
 foodPrice  
     es.ul.esit.app.middleware.model.MainCourse, 162  
     es.ul.esit.server.middleware.model.MainCourse,  
         167  
 generateReceiptFile  
     es.ul.esit.app.middleware.service.OrderService,  
         202  
 getAllAppetizers  
     es.ul.esit.app.middleware.ApiClient, 61  
     es.ul.esit.app.middleware.service.ProductService,  
         208  
     es.ul.esit.server.controller.AppetizerController, 88  
 getAllCashiers  
     es.ul.esit.app.middleware.ApiClient, 62  
     es.ul.esit.server.controller.CashierController, 111  
 getAllDrinks  
     es.ul.esit.app.middleware.ApiClient, 62  
     es.ul.esit.app.middleware.service.ProductService,  
         208  
     es.ul.esit.server.controller.DrinkController, 140  
 getAllMainCourses  
     es.ul.esit.app.middleware.ApiClient, 63  
     es.ul.esit.app.middleware.service.ProductService,  
         209  
     es.ul.esit.server.controller.MainCourseController,  
         170  
 getAppetizerById  
     es.ul.esit.app.middleware.ApiClient, 63  
     es.ul.esit.app.middleware.service.ProductService,  
         209  
     es.ul.esit.server.controller.AppetizerController, 88  
 getAppetizersId  
     es.ul.esit.app.middleware.model.Appetizer, 78  
     es.ul.esit.server.middleware.model.Appetizer, 84  
 getAppetizersPrice  
     es.ul.esit.app.middleware.model.Appetizer, 78  
     es.ul.esit.server.middleware.model.Appetizer, 84  
 getCashierById  
     es.ul.esit.app.middleware.ApiClient, 64  
     es.ul.esit.server.controller.CashierController, 111  
 getCashierByName  
     es.ul.esit.app.middleware.ApiClient, 65  
     es.ul.esit.server.controller.CashierController, 112  
 getCashierInfo  
     es.ul.esit.app.middleware.service.ReportService,  
         219  
 getDrinkByld  
     es.ul.esit.app.middleware.ApiClient, 65  
     es.ul.esit.app.middleware.service.ProductService,  
         210  
     es.ul.esit.server.controller.DrinkController, 140  
 getDrinksId  
     es.ul.esit.app.middleware.model.Drink, 130  
     es.ul.esit.server.middleware.model.Drink, 136  
 getDrinksPrice  
     es.ul.esit.app.middleware.model.Drink, 130  
     es.ul.esit.server.middleware.model.Drink, 136  
 getFoodId  
     es.ul.esit.app.middleware.model.MainCourse, 160  
     es.ul.esit.server.middleware.model.MainCourse,  
         166  
 getFoodPrice  
     es.ul.esit.app.middleware.model.MainCourse, 160  
     es.ul.esit.server.middleware.model.MainCourse,  
         166  
 getFullMenu  
     es.ul.esit.server.controller.MenuController, 174  
 getId  
     es.ul.esit.app.middleware.model.Cashier, 104  
     es.ul.esit.server.middleware.model.Cashier, 108  
     es.ul.esit.server.middleware.model.User, 230  
 getItemAppetizers  
     es.ul.esit.app.middleware.model.Appetizer, 78  
     es.ul.esit.server.middleware.model.Appetizer, 84  
 getItemDrinks  
     es.ul.esit.app.middleware.model.Drink, 130  
     es.ul.esit.server.middleware.model.Drink, 136  
 getItemFood  
     es.ul.esit.app.middleware.model.MainCourse, 160  
     es.ul.esit.server.middleware.model.MainCourse,  
         166  
 getMainCourseByld  
     es.ul.esit.app.middleware.ApiClient, 66  
     es.ul.esit.app.middleware.service.ProductService,  
         211  
     es.ul.esit.server.controller.MainCourseController,  
         170  
 getName  
     es.ul.esit.app.middleware.model.Cashier, 104  
     es.ul.esit.server.middleware.model.Cashier, 108  
 getPasswordHash  
     es.ul.esit.server.middleware.model.User, 230  
 getReceiptId  
     es.ul.esit.app.middleware.model.Appetizer, 78  
     es.ul.esit.app.middleware.model.Drink, 130  
     es.ul.esit.app.middleware.model.MainCourse, 160  
 getRole  
     es.ul.esit.app.middleware.model.User, 226  
     es.ul.esit.server.middleware.model.User, 230

getSalary  
    es.ul.esit.app.middleware.model.Cashier, 104  
    es.ul.esit.server.middleware.model.Cashier, 108  
getSubTotal  
    es.ul.esit.app.middleware.model.BillResult, 100  
getTotal  
    es.ul.esit.app.middleware.model.BillResult, 100  
getUsername  
    es.ul.esit.app.middleware.model.User, 226  
    es.ul.esit.server.middleware.model.User, 230  
getVat  
    es.ul.esit.app.middleware.model.BillResult, 100  
goBackMenueBtn  
    es.ul.esit.app.Order, 196  
goBackMenueBtnActionPerformed  
    es.ul.esit.app.Order, 183  
  
handle  
    es.ul.esit.app.middleware.ApiClient.ResponseHandler<  
        R >, 221  
health  
    es.ul.esit.server.controller.HealthController, 144  
HealthController.java, 254  
http  
    es.ul.esit.app.middleware.ApiClient, 72  
  
id  
    es.ul.esit.app.middleware.model.Cashier, 105  
    es.ul.esit.server.middleware.model.Cashier, 110  
    es.ul.esit.server.middleware.model.User, 232  
ID\_COLUMN\_HEADER  
    es.ul.esit.app.Order, 196  
init.sql, 243  
initComponents  
    es.ul.esit.app.AboutUs, 20  
    es.ul.esit.app.AdminLogin, 28  
    es.ul.esit.app.AdminProducts, 39  
    es.ul.esit.app.CashierLogin, 118  
    es.ul.esit.app.Login, 151  
    es.ul.esit.app.Order, 183  
isAdmin  
    es.ul.esit.app.middleware.model.User, 227  
ITEM\_COLUMN\_HEADER  
    es.ul.esit.app.Order, 196  
itemAppetizers  
    es.ul.esit.app.middleware.model.Appetizer, 81  
    es.ul.esit.server.middleware.model.Appetizer, 86  
itemDrinks  
    es.ul.esit.app.middleware.model.Drink, 133  
    es.ul.esit.server.middleware.model.Drink, 138  
itemFood  
    es.ul.esit.app.middleware.model.MainCourse, 163  
    es.ul.esit.server.middleware.model.MainCourse,  
        168  
  
jButton1  
    es.ul.esit.app.AdminLogin, 32  
    es.ul.esit.app.CashierLogin, 123  
    es.ul.esit.app.Login, 154  
  
jButton1ActionPerformed  
    es.ul.esit.app.AdminLogin, 30  
    es.ul.esit.app.AdminProducts, 45  
    es.ul.esit.app.CashierLogin, 120  
    es.ul.esit.app.Login, 152  
jButton2  
    es.ul.esit.app.AdminLogin, 32  
    es.ul.esit.app.CashierLogin, 123  
jButton2ActionPerformed  
    es.ul.esit.app.AdminLogin, 30  
    es.ul.esit.app.AdminProducts, 45  
    es.ul.esit.app.CashierLogin, 120  
jButton3  
    es.ul.esit.app.AboutUs, 23  
    es.ul.esit.app.AdminLogin, 33  
    es.ul.esit.app.CashierLogin, 124  
jButton3ActionPerformed  
    es.ul.esit.app.AboutUs, 21  
    es.ul.esit.app.AdminLogin, 31  
    es.ul.esit.app.AdminProducts, 46  
    es.ul.esit.app.CashierLogin, 121  
jButton4ActionPerformed  
    es.ul.esit.app.AdminProducts, 46  
jButton5ActionPerformed  
    es.ul.esit.app.AdminProducts, 47  
jButton6ActionPerformed  
    es.ul.esit.app.AdminProducts, 48  
jButton7ActionPerformed  
    es.ul.esit.app.AdminProducts, 48  
jLabel1  
    es.ul.esit.app.AboutUs, 23  
    es.ul.esit.app.AdminLogin, 33  
    es.ul.esit.app.CashierLogin, 124  
    es.ul.esit.app.Login, 154  
    es.ul.esit.app.Order, 197  
jLabel2  
    es.ul.esit.app.Order, 197  
jLabel3  
    es.ul.esit.app.AdminLogin, 33  
    es.ul.esit.app.Login, 155  
jPanel1  
    es.ul.esit.app.AboutUs, 23  
    es.ul.esit.app.AdminLogin, 33  
    es.ul.esit.app.CashierLogin, 124  
    es.ul.esit.app.Login, 155  
    es.ul.esit.app.Order, 197  
jPanel2  
    es.ul.esit.app.Order, 197  
jPasswordField1  
    es.ul.esit.app.Login, 155  
jScrollPane1  
    es.ul.esit.app.AboutUs, 23  
jTable1KeyPressed  
    es.ul.esit.app.AdminProducts, 49  
jTable1MouseClicked  
    es.ul.esit.app.AdminProducts, 49  
jTable2MouseClicked  
    es.ul.esit.app.AdminProducts, 50

jTable3MouseClicked  
 es.ull.esit.app.AdminProducts, 50

lastBill  
 es.ull.esit.app.Order, 197

loadMenuFromDatabase  
 es.ull.esit.app.Order, 187

LOG  
 es.ull.esit.server.controller.AuthController, 94

LOGGER  
 es.ull.esit.app.AdminLogin, 33  
 es.ull.esit.app.AdminProducts, 52  
 es.ull.esit.app.CashierLogin, 124  
 es.ull.esit.app.middleware.ApiClient, 72  
 es.ull.esit.app.Order, 198

Login  
 es.ull.esit.app.Login, 150

login  
 es.ull.esit.app.middleware.ApiClient, 66, 67  
 es.ull.esit.server.controller.AuthController, 94

Login.java, 286, 287

LOGIN\_STRING  
 es.ull.esit.app.Login, 155

loginFactory  
 es.ull.esit.app.ApplicationLauncher, 92  
 es.ull.esit.app.Login, 155

main  
 es.ull.esit.app.AboutUs, 22  
 es.ull.esit.app.AdminLogin, 31  
 es.ull.esit.app.AdminProducts, 51  
 es.ull.esit.app.ApplicationLauncher, 92  
 es.ull.esit.app.CashierLogin, 121  
 es.ull.esit.app.Login, 153  
 es.ull.esit.app.Order, 188  
 es.ull.esit.server.RestaurantApplication, 222

MainCourse  
 es.ull.esit.app.middleware.model.MainCourse, 159  
 es.ull.esit.server.middleware.model.MainCourse, 165

MainCourse.java, 305–307

MainCourseController.java, 255, 256

mainCoursePnl  
 es.ull.esit.app.Order, 198

mainCourseRepository  
 es.ull.esit.server.controller.MainCourseController, 171  
 es.ull.esit.server.controller.MenuController, 175

MainCourseRepository.java, 261

mainsModel  
 es.ull.esit.app.Order, 198

mainsScroll  
 es.ull.esit.app.Order, 198

mainsTable  
 es.ull.esit.app.Order, 198

mapper  
 es.ull.esit.app.middleware.ApiClient, 73

MenuController.java, 257

name  
 es.ull.esit.app.middleware.model.Cashier, 106  
 es.ull.esit.server.middleware.model.Cashier, 110

newReceiptBtn  
 es.ull.esit.app.Order, 199

newReceiptBtnActionPerformed  
 es.ull.esit.app.Order, 189

Order  
 es.ull.esit.app.Order, 180

Order.java, 316

orderService  
 es.ull.esit.app.Order, 199

OrderService.java, 311, 312

ourStory  
 es.ull.esit.app.AboutUs, 23

password  
 es.ull.esit.server.controller.AuthController.LoginRequest, 157

passwordEncoder  
 es.ull.esit.server.config.SecurityConfig, 224  
 es.ull.esit.server.controller.AuthController, 94

passwordHash  
 es.ull.esit.server.middleware.model.User, 232

payBtn  
 es.ull.esit.app.Order, 199

payBtnActionPerformed  
 es.ull.esit.app.Order, 190

PRICE\_COLUMN\_HEADER  
 es.ull.esit.app.Order, 199

PRIMARY\_FONT  
 es.ull.esit.app.AdminLogin, 34  
 es.ull.esit.app.AdminProducts, 53  
 es.ull.esit.app.CashierLogin, 124  
 es.ull.esit.app.Login, 156  
 es.ull.esit.app.Order, 199

PRIMARY\_FONT\_LIGHT  
 es.ull.esit.app.Order, 200

ProductService  
 es.ull.esit.app.middleware.service.ProductService, 205

productService  
 es.ull.esit.app.AdminProducts, 53  
 es.ull.esit.app.Order, 200

ProductService.java, 313

QUANTITY\_COLUMN\_HEADER  
 es.ull.esit.app.Order, 200

README, 1

README.md, 247

RECEIPT\_NO\_PREFIX  
 es.ull.esit.app.Order, 200

receiptId  
 es.ull.esit.app.middleware.model.Appetizer, 81  
 es.ull.esit.app.middleware.model.Drink, 133  
 es.ull.esit.app.middleware.model.MainCourse, 163

receiptNoLbl

es.ul.esit.app.Order, 200  
refreshAllTables  
    es.ul.esit.app.AdminProducts, 52  
ReportService  
    es.ul.esit.app.middleware.service.ReportService,  
        218  
reportService  
    es.ul.esit.app.CashierLogin, 125  
ReportService.java, 315  
resetQtyColumn  
    es.ul.esit.app.Order, 191  
RestaurantApplication.java, 263  
role  
    es.ul.esit.app.middleware.model.User, 228  
    es.ul.esit.server.middleware.model.User, 232  
salary  
    es.ul.esit.app.middleware.model.Cashier, 106  
    es.ul.esit.server.middleware.model.Cashier, 110  
saveReceiptBtn  
    es.ul.esit.app.Order, 201  
saveReceiptBtnActionPerformed  
    es.ul.esit.app.Order, 191  
SECOND\_COLUMN\_HEADER  
    es.ul.esit.app.AdminProducts, 53  
SECONDARY\_FONT  
    es.ul.esit.app.AdminProducts, 53  
SecurityConfig.java, 247  
setAppetizersId  
    es.ul.esit.app.middleware.model.Appetizer, 79  
    es.ul.esit.server.middleware.model.Appetizer, 84  
setAppetizersPrice  
    es.ul.esit.app.middleware.model.Appetizer, 79  
    es.ul.esit.server.middleware.model.Appetizer, 85  
setDrinksId  
    es.ul.esit.app.middleware.model.Drink, 131  
    es.ul.esit.server.middleware.model.Drink, 136  
setDrinksPrice  
    es.ul.esit.app.middleware.model.Drink, 131  
    es.ul.esit.server.middleware.model.Drink, 137  
setFoodId  
    es.ul.esit.app.middleware.model.MainCourse, 161  
    es.ul.esit.server.middleware.model.MainCourse,  
        166  
setFoodPrice  
    es.ul.esit.app.middleware.model.MainCourse, 161  
    es.ul.esit.server.middleware.model.MainCourse,  
        167  
setId  
    es.ul.esit.app.middleware.model.Cashier, 104  
    es.ul.esit.server.middleware.model.Cashier, 108  
    es.ul.esit.server.middleware.model.User, 231  
setItemAppetizers  
    es.ul.esit.app.middleware.model.Appetizer, 79  
    es.ul.esit.server.middleware.model.Appetizer, 85  
setItemDrinks  
    es.ul.esit.app.middleware.model.Drink, 131  
    es.ul.esit.server.middleware.model.Drink, 137  
setItemFood  
    es.ul.esit.app.middleware.model.MainCourse, 161  
    es.ul.esit.server.middleware.model.MainCourse,  
        167  
setName  
    es.ul.esit.app.middleware.model.Cashier, 105  
    es.ul.esit.server.middleware.model.Cashier, 109  
setPasswordHash  
    es.ul.esit.server.middleware.model.User, 231  
setReceiptId  
    es.ul.esit.app.middleware.model.Appetizer, 80  
    es.ul.esit.app.middleware.model.Drink, 132  
    es.ul.esit.app.middleware.model.MainCourse, 162  
setRole  
    es.ul.esit.app.middleware.model.User, 227  
    es.ul.esit.server.middleware.model.User, 231  
setSalary  
    es.ul.esit.app.middleware.model.Cashier, 105  
    es.ul.esit.server.middleware.model.Cashier, 109  
setupTables  
    es.ul.esit.app.Order, 192  
setUsername  
    es.ul.esit.app.middleware.model.User, 227  
    es.ul.esit.server.middleware.model.User, 232  
subTotal  
    es.ul.esit.app.middleware.model.BillResult, 101  
subTotalLbl  
    es.ul.esit.app.Order, 201  
sumFromModel  
    es.ul.esit.app.Order, 193  
textBlock  
    es.ul.esit.app.AboutUs, 23  
THIRD\_COLUMN\_HEADER  
    es.ul.esit.app.AdminProducts, 53  
total  
    es.ul.esit.app.middleware.model.BillResult, 101  
totalLbl  
    es.ul.esit.app.Order, 201  
UPDATE\_STRING  
    es.ul.esit.app.AdminProducts, 54  
updateAppetizer  
    es.ul.esit.app.middleware.ApiClient, 68  
    es.ul.esit.app.middleware.service.ProductService,  
        211  
    es.ul.esit.server.controller.AppetizerController, 89  
updateCashier  
    es.ul.esit.app.middleware.ApiClient, 69  
    es.ul.esit.server.controller.CashierController, 112  
updateDrink  
    es.ul.esit.app.middleware.ApiClient, 69  
    es.ul.esit.app.middleware.service.ProductService,  
        212  
    es.ul.esit.server.controller.DrinkController, 141  
updateMainCourse  
    es.ul.esit.app.middleware.ApiClient, 70  
    es.ul.esit.app.middleware.service.ProductService,  
        213

es.ull.esit.server.controller.MainCourseController,  
    171  
updateWelcomeMessage  
    es.ull.esit.app.CashierLogin, 122  
User  
    es.ull.esit.app.middleware.model.User, 226  
User.java, 308–310  
username  
    es.ull.esit.app.middleware.model.User, 228  
    es.ull.esit.server.controller.AuthController.LoginRequest,  
        157  
    es.ull.esit.server.middleware.model.User, 232  
usernametxt  
    es.ull.esit.app.Login, 156  
userRepository  
    es.ull.esit.server.controller.AuthController, 95  
UserRepository.java, 262  
usertypecmbo  
    es.ull.esit.app.Login, 156  
usertypecmboActionPerformed  
    es.ull.esit.app.Login, 154  
  
validateAndParsePrice  
    es.ull.esit.app.middleware.service.ProductService,  
        214  
validateName  
    es.ull.esit.app.middleware.service.ProductService,  
        215  
vat  
    es.ull.esit.app.middleware.model.BillResult, 101  
VAT\_RATE  
    es.ull.esit.app.middleware.service.OrderService,  
        203  
vatLbl  
    es.ull.esit.app.Order, 201  
  
WARN\_FETCH\_CASHIER\_STATS  
    es.ull.esit.app.CashierLogin, 125  
welcomeTxt  
    es.ull.esit.app.CashierLogin, 125