Andres Garcia
CS 349 Java Program Application
Professor Baffour
May 5, 2023
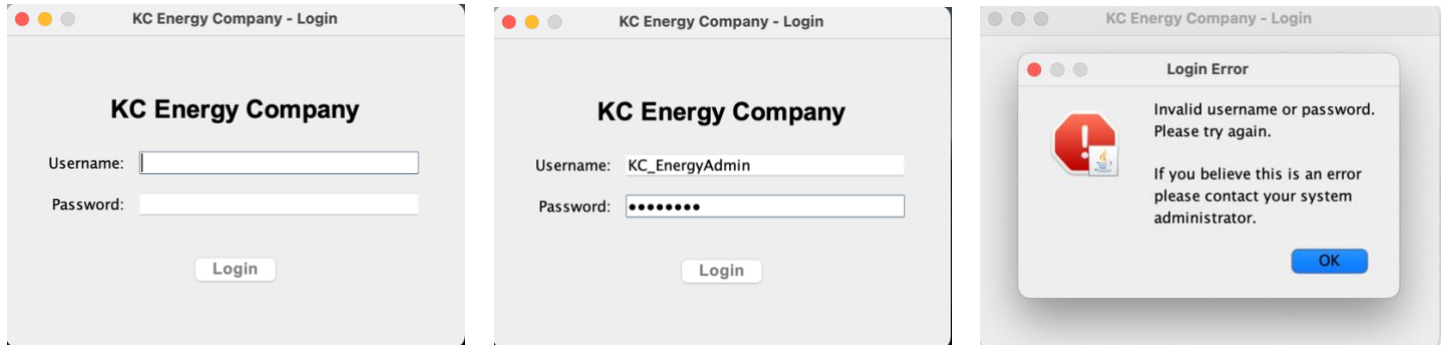
GUI Energy Database Project
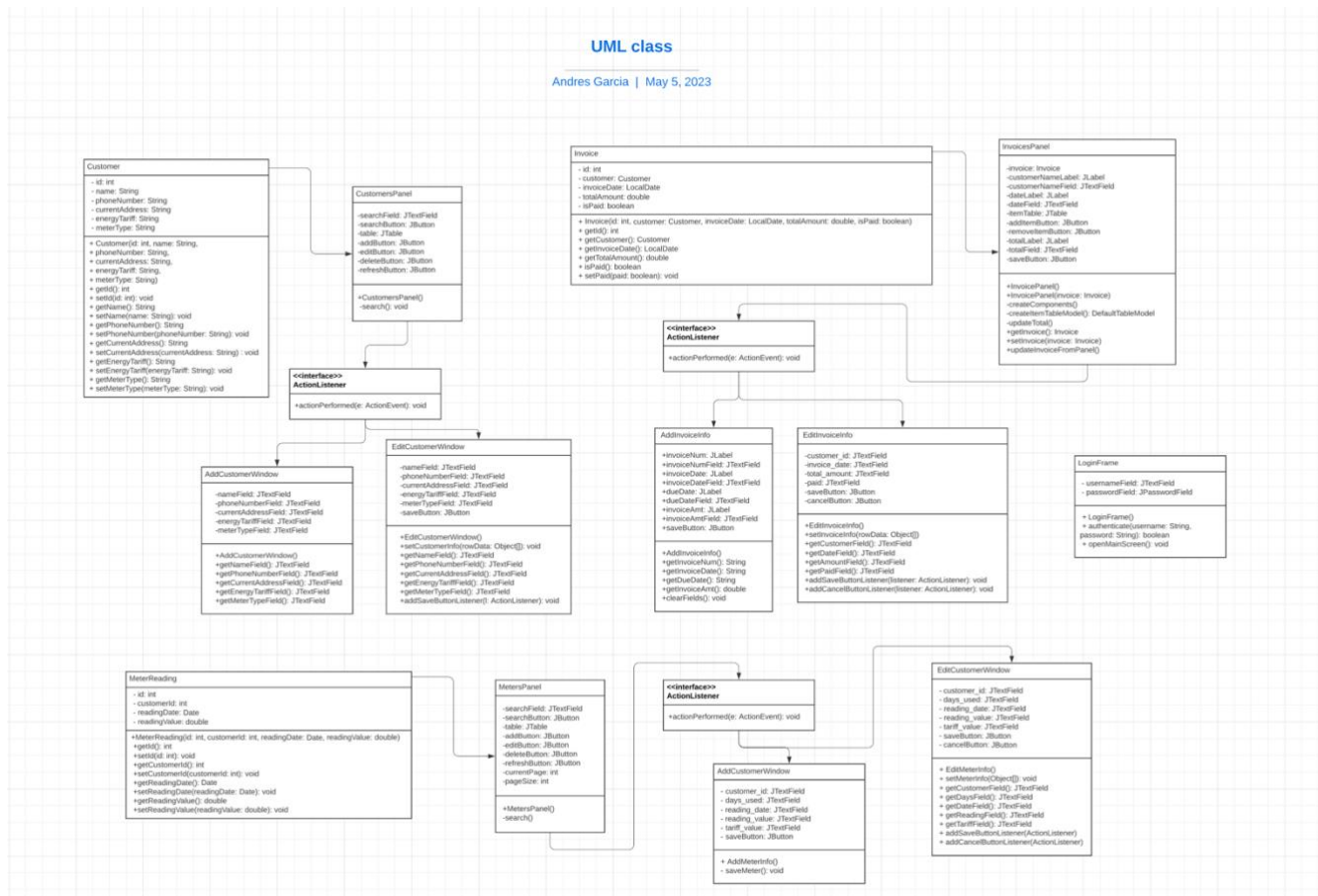
Link to Video Demo: https://youtu.be/SP1SyMjq1vw

Username and Password: **(Case Sensitive)**

KC_EnergyAdmin
Admin123

UML Class Diagrams

Implemented Features

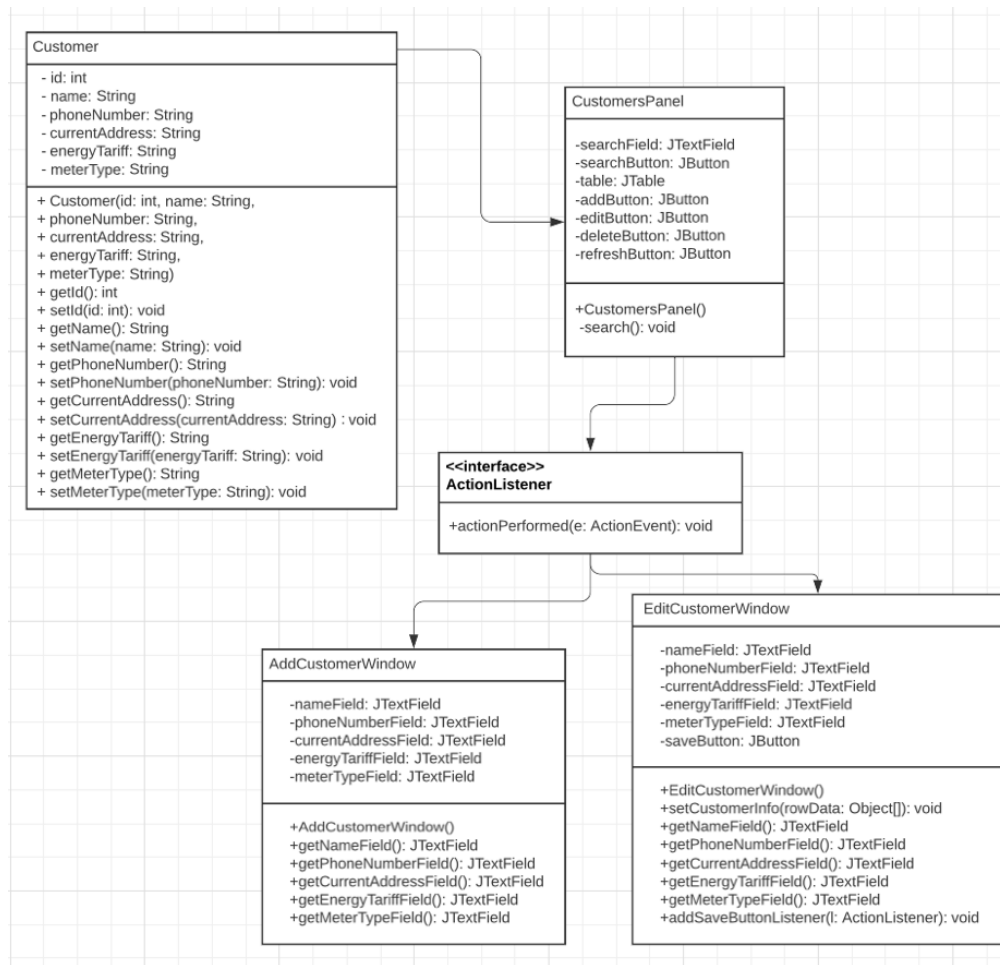| Feature	Implemented (Partial/Full)	Comments
Login w/ Validation	Full	Error Login if login is not recognized
Customer Tab	Full	Show all data within the customer table and display it on a JTable
Search Customer	Full	Case Sensitive search that will search all rows with the inputted text in the search box
Add Customer	Full	Implemented Error if no info is inputted and save is clicked
Edit Customer	Full	Must click refresh button to see the changes made
Delete Customer	Full	Deleted Customer is auto updated to reflect those changes
Refresh Tab	Full	Refresh the table for any changes made after (Add, Edit, Delete)
Invoice Tab	Full	Show all data within the invoice table and display it on a JTable
Search Invoice	Full	Case Sensitive search that will search all rows with the inputted text in the search box
Add Invoice	Full	Implemented Error if no info is inputted and save is clicked
Edit Invoice	Full	Must click refresh button to see the changes made
Delete Invoice	Full	Delete selected row
Generate Invoice	Full	Create an Invoice with each customers unique data
Make PDF	Partial	The PDF button is not yet implemented to create a pdf version of the invoice
Refresh Tab	Full	Refresh the table for any changes made after (Add, Edit, Delete)
Customer Energy Consumption Tab	Full	Show all data within the customer_consumption table and display it on a JTable
Search Energy Consumption	Full	Case Sensitive search that will search all rows with the inputted text in the search box
Add Consumption	Full	Implemented Error if no info is inputted and save is clicked
Edit Consumption	Full	Must click refresh button to see the changes made
Delete Consumption	Full	Delete selected row
Refresh Energy Consumption Tab	Full	Refresh the table for any changes made after (Add, Edit, Delete)

Explanation of Code with Diagrams
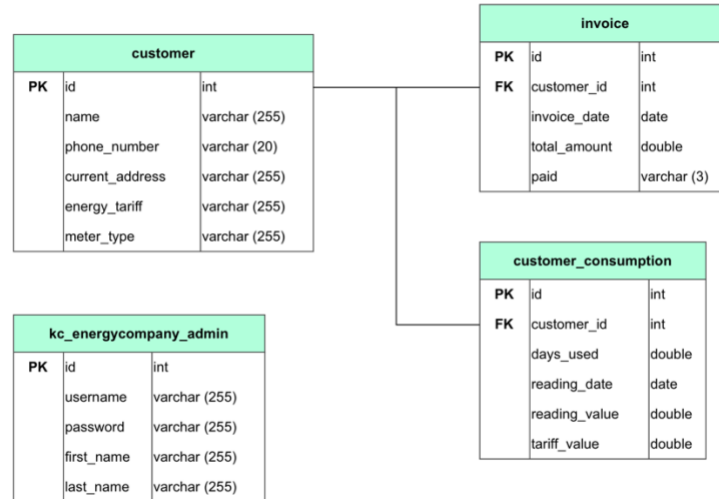
(ERD Diagram for KC_EnergyCompany Database)

The database schema consists of four tables: customer, customer_consumption, invoice, and kc_energycompany_admin.

❖ The **customer table** stores information about the company's customers. It has five columns: **id, name, phone_number, current_address, energy_tariff,** and **meter_type.**
 ➢ The id column is the primary key, and it is auto-incremented. The name, phone_number, current_address, energy_tariff, and meter_type columns store the customer's name, phone number, address, energy tariff, and meter type, respectively.
 ➢ All these columns are mandatory and cannot be null.

❖ The **customer_consumption table** tracks the customers' energy consumption. It has six columns: **id, customer_id, days_used, reading_date, reading_value,** and **tariff_value**.
 ➢ The id column is the primary key, and it is auto-incremented. The customer_id column is a foreign key that references the id column in the customer table.
 ➢ The days_used, reading_date, reading_value, and tariff_value columns store the number of days the energy was used, the date of the reading, the reading value, and the tariff value, respectively.
 ➢ All these columns are mandatory and cannot be null.

❖ The **invoice table** stores information about the customer's invoices. It has five columns: **id, customer_id, invoice_date, total_amount**, and **paid**.
 ➢ The id column is the primary key, and it is auto-incremented. The customer_id column is a foreign key that references the id column in the customer table.
 ➢ The invoice_date, total_amount, and paid columns store the date of the invoice, the total amount of the invoice, and whether the invoice has been paid or not, respectively.
 ➢ All these columns are mandatory and cannot be null.

❖ The **kc_energycompany_admin** table stores the company's administrator's login credentials. It has five columns**: id, username, password, first_name,** and **last_name**.
 ➢ The id column is the primary key, and it is auto-incremented.
 ➢ The username and password columns store the administrator's login credentials, and the first_name and last_name columns store the administrator's first and last names, respectively.
 ➢ All these columns are mandatory and cannot be null.

Customer

- id: int
- name: String
- phoneNumber: String
- currentAddress: String
- energyTariff: String
- meterType: String

+ Customer(id: int, name: String,
+ phoneNumber: String,
+ currentAddress: String,
+ energyTariff: String,
+ meterType: String)
+ getId(): int
+ setId(id: int): void
+ getName(): String
+ setName(name: String): void
+ getPhoneNumber(): String
+ setPhoneNumber(phoneNumber: String): void
+ getCurrentAddress(): String
+ setCurrentAddress(currentAddress: String) : void
+ getEnergyTariff(): String
+ setEnergyTariff(energyTariff: String): void
+ getMeterType(): String
+ setMeterType(meterType: String): void

CustomersPanel

-searchField: JTextField
-searchButton: JButton
-table: JTable
-addButton: JButton
-editButton: JButton
-deleteButton: JButton
-refreshButton: JButton

+CustomersPanel()
-search(): void

**<<interface>>
ActionListener**

+actionPerformed(e: ActionEvent): void

AddCustomerWindow

-nameField: JTextField
-phoneNumberField: JTextField
-currentAddressField: JTextField
-energyTariffField: JTextField
-meterTypeField: JTextField

+AddCustomerWindow()
+getNameField(): JTextField
+getPhoneNumberField(): JTextField
+getCurrentAddressField(): JTextField
+getEnergyTariffField(): JTextField
+getMeterTypeField(): JTextField

EditCustomerWindow

-nameField: JTextField
-phoneNumberField: JTextField
-currentAddressField: JTextField
-energyTariffField: JTextField
-meterTypeField: JTextField
-saveButton: JButton

+EditCustomerWindow()
+setCustomerInfo(rowData: Object[]): void
+getNameField(): JTextField
+getPhoneNumberField(): JTextField
+getCurrentAddressField(): JTextField
+getEnergyTariffField(): JTextField
+getMeterTypeField(): JTextField
+addSaveButtonListener(l: ActionListener): void

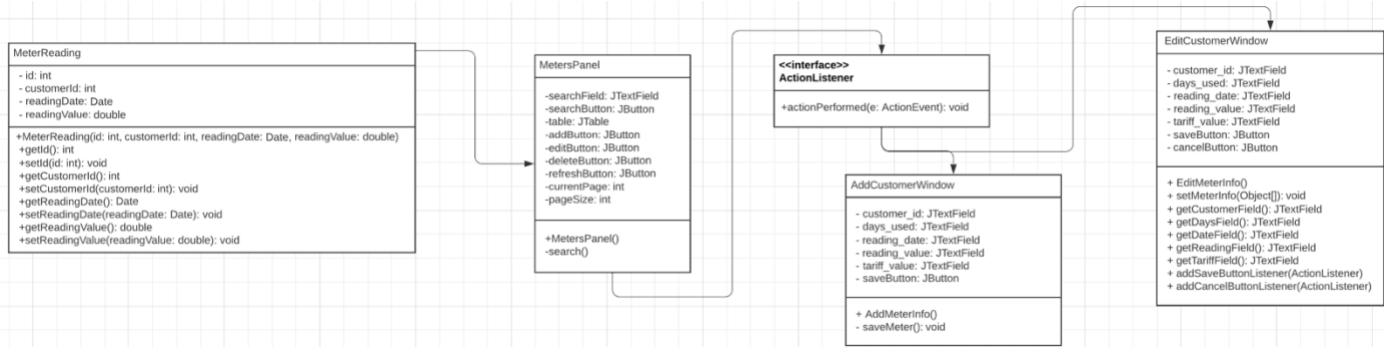*(UML Diagram associated with Customer Functionality)*

The UML diagrams provided show the design of a software application that manages customer data for an energy company. The application is designed using a Model-View-Controller (MVC) architecture, with the data model represented by the Customer class.

The Customer class has several attributes that store customer data, including id, name, phone number, current address, energy tariff, and meter type. It has getter and setter methods for each attribute to manipulate customer data.

The CustomersPanel class represents the view, which provides a graphical user interface for the user to interact with customer data. It has various UI components, including a search field and button, table, add, edit, delete, and refresh buttons. The ActionListener interface is implemented to listen to events triggered by UI components and update the view accordingly.

The AddCustomerWindow and EditCustomerWindow classes represent modal windows that allow the user to add or edit customer data. They have text fields that allow the user to enter customer data, and a save button that triggers an ActionListener to update the customer data.

Overall, the UML diagrams provide a clear representation of the design of the software application, showing how the various components interact with each other to manage customer data for the energy company.

MeterReading

- id: int
- customerId: int
- readingDate: Date
- readingValue: double

+MeterReading(id: int, customerId: int, readingDate: Date, readingValue: double)
+getId(): int
+setId(id: int): void
+getCustomerId(): int
+setCustomerId(customerId: int): void
+getReadingDate(): Date
+setReadingDate(readingDate: Date): void
+getReadingValue(): double
+setReadingValue(readingValue: double): void

MetersPanel

-searchField: JTextField
-searchButton: JButton
-table: JTable
-addButton: JButton
-editButton: JButton
-deleteButton: JButton
-refreshButton: JButton
-currentPage: int
-pageSize: int

+MetersPanel()
-search()

<<interface>>
ActionListener

+actionPerformed(e: ActionEvent): void

AddCustomerWindow

- customer_id: JTextField
- days_used: JTextField
- reading_date: JTextField
- reading_value: JTextField
- tariff_value: JTextField
- saveButton: JButton

+ AddMeterInfo()
- saveMeter(): void

EditCustomerWindow

- customer_id: JTextField
- days_used: JTextField
- reading_date: JTextField
- reading_value: JTextField
- tariff_value: JTextField
- saveButton: JButton
- cancelButton: JButton

+ EditMeterInfo()
+ setMeterInfo(Object[]): void
+ getCustomerField(): JTextField
+ getDaysField(): JTextField
+ getDateField(): JTextField
+ getReadingField(): JTextField
+ getTariffField(): JTextField
+ addSaveButtonListener(ActionListener)
+ addCancelButtonListener(ActionListener)

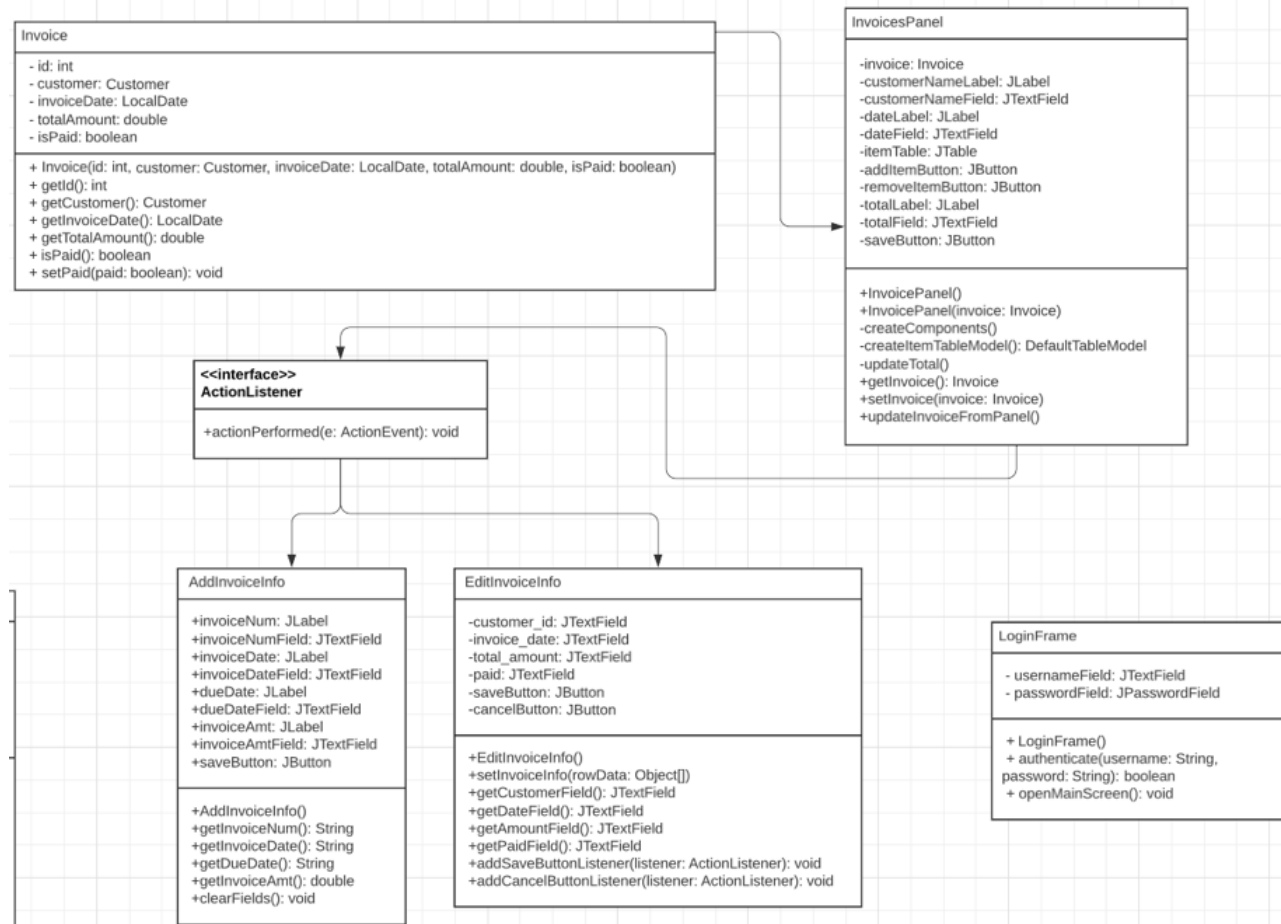*(UML Diagram associated with Customer Consumption Functionality)*

The UML diagrams provided represent the design of a software application that manages meter data for an energy company. The application follows the Model-View-Controller (MVC) architecture and consists of several components that work together to manage meter data.

The MetersPanel class represents the view, providing a graphical user interface for the user to interact with meter data. It includes a search field and button, table, add, edit, delete, and refresh buttons. The table displays meter data in a tabular format, with columns for customer ID, days used, reading date, reading value, and tariff value.

The AddMeterInfo and EditMeterInfo classes represent modal windows that allow the user to add or edit meter data. They have text fields that allow the user to enter meter data, including customer ID, days used, reading date, reading value, and tariff value. The Save button triggers an ActionListener to save the meter data, while the Cancel button allows the user to cancel the operation.

The ActionListener interface is implemented to listen to events triggered by UI components and update the view accordingly. It includes a single method, actionPerformed, which is called when an action event occurs.

Overall, the UML diagrams provide a clear representation of the design of the software application, showing how the various components interact with each other to manage meter data for the energy company. The application provides an intuitive interface for the user to add, edit, and delete meter data, as well as search for specific meter data.

Invoice

- id: int
- customer: Customer
- invoiceDate: LocalDate
- totalAmount: double
- isPaid: boolean

+ Invoice(id: int, customer: Customer, invoiceDate: LocalDate, totalAmount: double, isPaid: boolean)
+ getId(): int
+ getCustomer(): Customer
+ getInvoiceDate(): LocalDate
+ getTotalAmount(): double
+ isPaid(): boolean
+ setPaid(paid: boolean): void

InvoicesPanel

-invoice: Invoice
-customerNameLabel: JLabel
-customerNameField: JTextField
-dateLabel: JLabel
-dateField: JTextField
-itemTable: JTable
-addItemButton: JButton
-removeItemButton: JButton
-totalLabel: JLabel
-totalField: JTextField
-saveButton: JButton

+InvoicePanel()
+InvoicePanel(invoice: Invoice)
-createComponents()
-createItemTableModel(): DefaultTableModel
-updateTotal()
+getInvoice(): Invoice
+setInvoice(invoice: Invoice)
+updateInvoiceFromPanel()

<<interface>>
ActionListener

+actionPerformed(e: ActionEvent): void

AddInvoiceInfo

+invoiceNum: JLabel
+invoiceNumField: JTextField
+invoiceDate: JLabel
+invoiceDateField: JTextField
+dueDate: JLabel
+dueDateField: JTextField
+invoiceAmt: JLabel
+invoiceAmtField: JTextField
+saveButton: JButton

+AddInvoiceInfo()
+getInvoiceNum(): String
+getInvoiceDate(): String
+getDueDate(): String
+getInvoiceAmt(): double
+clearFields(): void

EditInvoiceInfo

-customer_id: JTextField
-invoice_date: JTextField
-total_amount: JTextField
-paid: JTextField
-saveButton: JButton
-cancelButton: JButton

+EditInvoiceInfo()
+setInvoiceInfo(rowData: Object[])
+getCustomerField(): JTextField
+getDateField(): JTextField
+getAmountField(): JTextField
+getPaidField(): JTextField
+addSaveButtonListener(listener: ActionListener): void
+addCancelButtonListener(listener: ActionListener): void

LoginFrame

- usernameField: JTextField
- passwordField: JPasswordField

+ LoginFrame()
+ authenticate(username: String, password: String): boolean
+ openMainScreen(): void

(UML Diagram associated with Invoice Functionality)

The provided UML diagrams illustrate the design of a software application that manages invoice data. The application is designed using a Model-View-Controller (MVC) architecture, with the data model represented by the Invoice class.
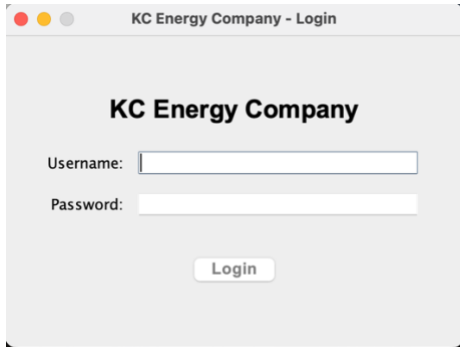
The Invoice class has several attributes that store invoice data, including id, customer, invoiceDate, totalAmount, and isPaid. It has getter and setter methods for each attribute to manipulate invoice data. The isPaid attribute is a boolean value indicating whether the invoice has been paid or not.

The InvoicePanel class represents the view, which provides a graphical user interface for the user to interact with invoice data. It has various UI components, including customerNameLabel and Field, dateLabel and Field, itemTable, addItemButton, removeItemButton, totalLabel and Field, and saveButton. The ActionListener interface is implemented to listen to events triggered by UI components and update the view accordingly.

The EditInvoiceInfo and AddInvoiceInfo classes represent modal windows that allow the user to add or edit invoice data. They have text fields that allow the user to enter invoice data, and a save button that triggers an ActionListener to update the invoice data.

Overall, the UML diagrams provide a clear representation of the design of the software application, showing how the various components interact with each other to manage invoice data. The InvoicePanel provides a user-friendly interface for adding and editing invoices, while the Invoice class provides a structured way to store invoice data.
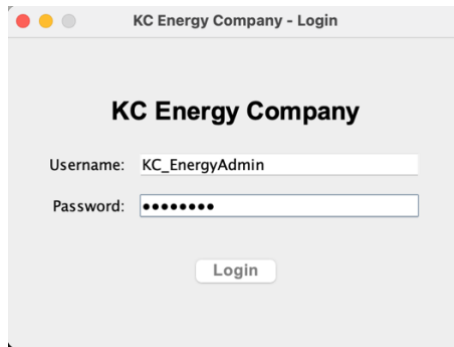
Screenshots / Key Features
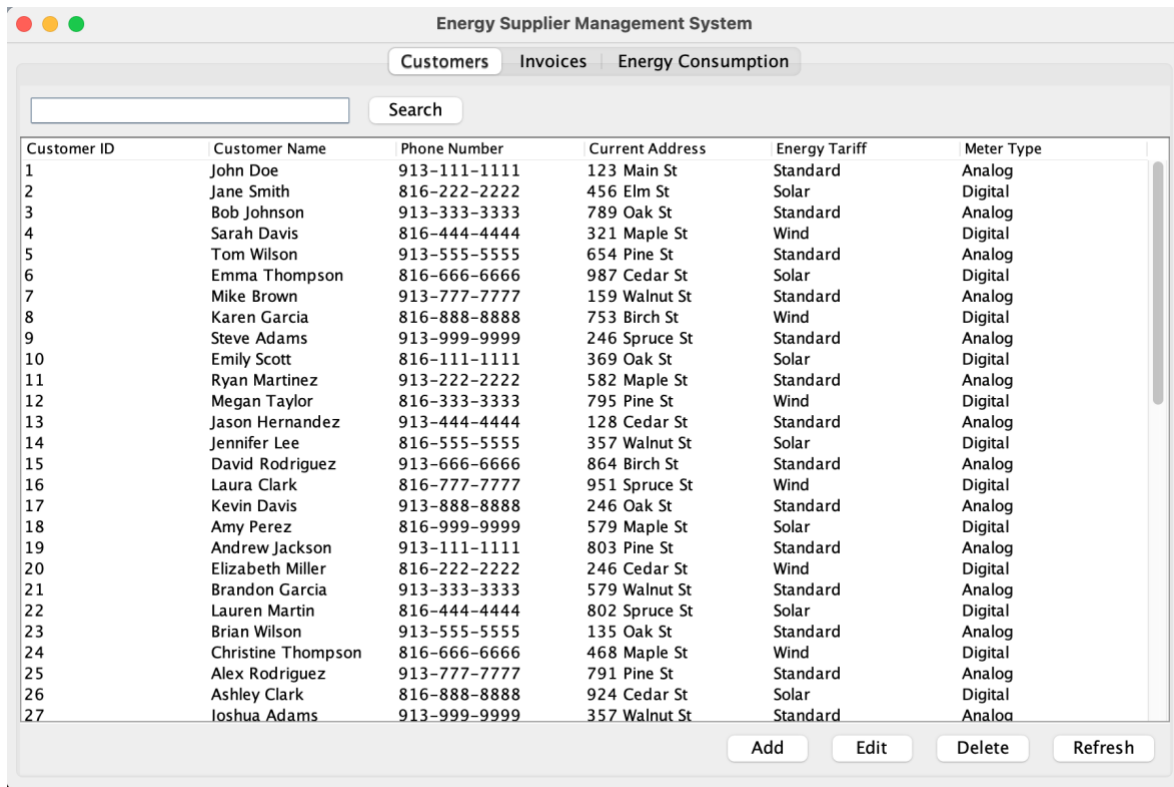
(Login and Login Validation)

(Customer Tab)

Energy Supplier Management System

Customers | Invoices | Energy Consumption

John [Search]

Customer ID	Customer Name	Phone Number	Current Address	Energy Tariff	Meter Type
1	John Doe	913-111-1111	123 Main St	Standard	Analog
3	Bob Johnson	913-333-3333	789 Oak St	Standard	Analog
35	David Johnson	305-000-0000	369 Cherry St	Solar	Digital
57	Isabella Johnson	202-000-0000	258 Cedar Ave	Solar	Digital

[Add] [Edit] [Delete] [Refresh]

(Search Customer)

Energy Supplier Management System

Customers | Invoices | Energy Consumption

[Search]

Customer ID	Customer Name	Phone Number	Current Address	Energy Tariff	Meter Type
1	John Doe	913-111-1111	123 Main St	Standard	Analog
2	Jane Smith				Digital
3	Bob Johnson				Analog
4	Sarah Davis				Digital
5	Tom Wilson				Analog
6	Emma Thomp				Digital
7	Mike Brown				Analog
8	Karen Garcia				Digital
9	Steve Adams				Analog
10	Emily Scott				Digital
11	Ryan Martinez				Analog
12	Megan Taylor				Digital
13	Jason Hernan				Analog
14	Jennifer Lee				Digital
15	David Rodrigu				Analog
16	Laura Clark				Digital
17	Kevin Davis				Analog
18	Amy Perez				Digital
19	Andrew Jacks				Analog
20	Elizabeth Mille				Digital
21	Brandon Garcia	913-333-3333	579 Walnut St	Standard	Analog
22	Lauren Martin	816-444-4444	802 Spruce St	Solar	Digital
23	Brian Wilson	913-555-5555	135 Oak St	Standard	Analog
24	Christine Thompson	816-666-6666	468 Maple St	Wind	Digital
25	Alex Rodriguez	913-777-7777	791 Pine St	Standard	Analog
26	Ashley Clark	816-888-8888	924 Cedar St	Solar	Digital
27	Joshua Adams	913-999-9999	357 Walnut St	Standard	Analog

Add Customer

Name: TestUser

Phone Number: 913-000-0000

Current Address: 123 Main st.

Energy Tariff: Solar

Meter Type: Digital

[Save]

[Add] [Edit] [Delete] [Refresh]

(Add customer)

60	William Anderson	617-000-0000	789 Oak Ave	Standard	Analog
61	Stephanie Lee	816-333-3333	146 Oak St	Solar	Digital
62	TestUser	913-000-0000	123 Main st.	Solar	Digital

(Customer added to table)

Energy Supplier Management System

Customers | Invoices | Energy Consumption

[ ] Search

Customer ID	Customer Name	Phone Number	Current Address	Energy Tariff	Meter Type
36	Olivia Wilson	617-000-0000	789 Oak Ave	Standard	Analog
37	Sofia Rodriguez	415-000-0000	456 Elm St	Solar	Digital
38	Mia Anderson	214-000-0000	123 Main St	Standard	Analog
39	Ethan Martin	347-000-0000	246 Maple Rd	Solar	Digital
40	Avery Campbell	408-000-0000	369 Pine St	Standard	Analog
41	James Parker				Digital
42	Liam Bailey				Analog
43	Madison Smith				Digital
44	Ava Brown				Analog
45	Elijah Davis				Digital
46	Emma Wilson				Analog
47	Noah Miller				Digital
48	Abigail King				Analog
49	Evelyn Wright				Digital
50	Mason Green				Analog
51	Sophia Taylor				Digital
52	Benjamin Adam				Analog
53	Michael Scott				Digital
54	Chloe Allen				Analog
55	Evelyn Davis				Digital
56	Elizabeth Garcia				Analog
57	Isabella Johnson	202-000-0000	258 Cedar Ave	Solar	Digital
58	Alexander Taylor	631-000-0000	147 Walnut Ln	Standard	Analog
59	Ella Miller	305-000-0000	369 Cherry St	Solar	Digital
60	William Anderson	617-000-0000	789 Oak Ave	Standard	Analog
61	Stephanie Lee	816-333-3333	146 Oak St	Solar	Digital
62	TestUser	913-000-0000	123 Main st.	Wind	Analog

Confirm Delete

Are you sure you want to delete this row?

Customer ID: 51
Customer Name: Sophia Taylor
Phone Number: 305-000-0000
Current Address: 369 Cherry St
Energy Tariff: Solar
Meter Type: Digital

No | Yes

Add | Edit | Delete | Refresh

(Delete Customer)

50	Mason Green	631-000-0000	147 Walnut Ln	Standard	Analog
52	Benjamin Adams	617-000-0000	789 Oak Ave	Standard	Analog
53	Michael Scott	213-000-0000	456 Elm St	Solar	Digital

(Customer Deleted from Table)

Edit Customer

Name: TestUser

Phone Number: 913-000-0000

Current Address: 123 Main st.

Energy Tariff: Wind

Meter Type: Analog

Save

(Edit Customer)

61	Stephanie Lee	816-333-3333	146 Oak St	Solar	Digital
62	TestUser	913-000-0000	123 Main st.	Wind	Analog

(Customer Edited)

Energy Supplier Management System

Customers   **Invoices**   Energy Consumption

[   ]   Search   Add   Edit   Delete   Refresh

Invoice ID	Customer ID	Invoice Date	Meter Fee	Paid?:
1	1	2023-04-25	35.0	YES
2	2	2023-04-28	20.0	NO
3	3	2023-04-19	35.0	YES
4	4	2023-04-22	20.0	YES
6	6	2023-04-23	20.0	NO
7	7	2023-04-27	35.0	YES
8	8	2023-04-21	20.0	NO
9	9	2023-04-24	35.0	NO
10	10	2023-04-20	20.0	YES
11	11	2023-04-25	35.0	YES
12	12	2023-04-28	20.0	NO
13	13	2023-04-19	35.0	YES
14	14	2023-04-22	20.0	YES
15	15	2023-04-26	35.0	NO
16	16	2023-04-23	20.0	NO
17	17	2023-04-27	35.0	YES
18	18	2023-04-21	20.0	NO
19	19	2023-04-24	35.0	NO
20	20	2023-04-20	20.0	YES
21	21	2023-04-25	35.0	YES
22	22	2023-04-28	20.0	NO
23	23	2023-04-19	35.0	YES
24	24	2023-04-22	20.0	YES
25	25	2023-04-26	35.0	NO
26	26	2023-04-23	20.0	NO
27	27	2023-04-27	35.0	YES
28	28	2023-04-21	20.0	NO
31	45	2023-04-20	20.0	NO

Generate Invoice

(Invoice Tab)

Energy Supplier Management System

Customers   **Invoices**   Energy Consumption

[NO   ]   Search   Add   Edit   Delete   Refresh

Invoice ID	Customer ID	Invoice Date	Meter Fee	Paid?:
2	2	2023-04-28	20.0	NO
6	6	2023-04-23	20.0	NO
8	8	2023-04-21	20.0	NO
9	9	2023-04-24	35.0	NO
12	12	2023-04-28	20.0	NO
15	15	2023-04-26	35.0	NO
16	16	2023-04-23	20.0	NO
18	18	2023-04-21	20.0	NO
19	19	2023-04-24	35.0	NO
22	22	2023-04-28	20.0	NO
25	25	2023-04-26	35.0	NO
26	26	2023-04-23	20.0	NO
28	28	2023-04-21	20.0	NO
31	45	2023-04-20	20.0	NO
32	40	2023-04-20	35.0	NO

Generate Invoice

(Invoice Search)

Energy Supplier Management System

Customers | **Invoices** | Energy Consumption

Search | Add | Edit | Delete | Refresh

Invoice ID	Customer ID	Invoice Date	Meter Fee	Paid?:
2	2	2023-04-28	20.0	NO
3	3	2023-04-19	35.0	YES
4	4	2023-04-22	20.0	YES
6	6	2023-04-23	20.0	NO
7	7			YES
8	8			NO
9	9			NO
10	10			YES
11	11			YES
12	12			NO
13	13			YES
14	14			YES
15	15			NO
16	16			NO
17	17			YES
18	18			NO
19	19			NO
20	20			YES
21	21			YES
22	22			NO
23	23			YES
24	24			YES
25	25			NO
26	26	2023-04-23	20.0	NO
27	27	2023-04-27	35.0	YES
28	28	2023-04-21	20.0	NO
31	45	2023-04-20	20.0	NO

Add Invoice

Customer ID: 40

Invoice Date (YYYY-MM-DD): 023-04-20

Meter Fee: 35.0

Paid(YES/NO): NO

Save | Cancel

Generate Invoice

(Add Invoice)

32	40	2023-04-20	35.0	NO

(Invoice Added)

Energy Supplier Management System

Customers | **Invoices** | Energy Consumption

Search | Add | Edit | Delete | Refresh

Invoice ID	Customer ID	Invoice Date	Meter Fee	Paid?:
2	2	2023-04-28	20.0	NO
3	3	2023-04-19	35.0	YES
4	4	2023-04-22	20.0	YES
6	6	2023-04-23	20.0	NO
7	7			YES
8	8			NO
9	9			NO
10	10			YES
11	11			YES
12	12			NO
13	13			YES
14	14			YES
15	15			NO
16	16			NO
17	17			YES
18	18			NO
19	19			NO
20	20			YES
21	21			YES
22	22			NO
23	23			YES
24	24			YES
25	25			NO
26	26	2023-04-23	20.0	NO
27	27	2023-04-27	35.0	YES
28	28	2023-04-21	20.0	NO
31	45	2023-04-20	20.0	NO
32	40	2023-04-20	35.0	NO

Edit Invoice Info

Customer ID: 40

Invoice Date (YYYY-MM-DD): 2020-04-20

Meter Fee: 20.0

Paid (Yes/No?): Yes

Save | Cancel

Generate Invoice

(Edit Invoice)

32	40	2020-04-20	20.0	Yes

(Invoice Edited)

Customers   Invoices   Energy Consumption

| | Search | | Add | Edit | Delete | Refresh |

Invoice ID	Customer ID	Invoice Date	Meter Fee	Paid?:
3	3	2023-04-19	35.0	YES
4	4	2023-04-22	20.0	YES
6	6	2023-04-23	20.0	NO
7	7	2023-04-27	35.0	YES
8	8	2023-04-21	20.0	NO
9	9			NO
10	10			YES
11	11			YES
12	12			NO
13	13			YES
14	14			YES
15	15			NO
16	16			NO
17	17			YES
18	18			NO
19	19			NO
20	20			YES
21	21			YES
22	22			NO
23	23			YES
24	24	2023-04-22	20.0	YES
25	25	2023-04-26	35.0	NO
26	26	2023-04-23	20.0	NO
27	27	2023-04-27	35.0	YES
28	28	2023-04-21	20.0	NO
31	45	2023-04-20	20.0	NO
32	40	2020-04-20	20.0	Yes

Confirm Delete

Are you sure you want to delete this row?

Invoice ID: 20
Customer ID: 20
Invoice Date: 2023-04-20
Total Amount: 20.0
Paid (Yes/No?): YES

| No | Yes |

Generate Invoice

(Delete Invoice)

19	19	2023-04-24	35.0	NO
21	21	2023-04-25	35.0	YES

(Invoice Deleted)

Customers   Invoices   Energy Consumption

Invoice

Invoice ID
1
2
3
4
6
7
8
9
10
11
12
13
14
15
16
17
18
19
21
22
23
24
25
26
27
28
31

Customer Name: Jane Smith   Customer Address: 456 Elm St
KC Energy Company   Invoice ID#: 2
Invoice Date: 2023-04-28

Invoice for customerID: 2
Customer Name: Jane Smith
Address: 456 Elm St

KC Energy Company
Invoice Date: 2023-04-28

Breakdown of charges for this month:

You are using Solar which has a energy tarrif rate of 0.1 KW/h
as well as having a Digital meter which has a fee of $20.0 and all of
these are factors which make up your bill.

This means that your total is calculated as follows...

Subtotal: (872.0 KWh x 0.1) + $20.0
Total: $107.2 x Sales tax of 6.5%

Please pay the balance of $114.17 within the next 30 days.

Total Charges: $114.17

Make PDF

Generate Invoice

(Generate Invoice for Customer)

Energy Supplier Management System

Customers | Invoices | **Energy Consumption**

Search

ID	Customer ID	Days Since Reading	Reading Date	KWh (Meter Reading)	KWh Energy Tariff Rate
1	1	23.5	2023–04–20	935.0	0.17
2	2	28.2	2023–03–12	872.0	0.1
3	3	25.8	2023–02–25	938.0	0.17
4	4	21.1	2023–01–17	749.0	0.15
6	6	22.7	2023–03–28	1200.0	0.1
8	8	29.6	2023–01–22	918.0	0.15
9	9	26.8	2023–04–13	825.0	0.17
10	10	20.5	2023–03–04	863.0	0.1
11	11	23.5	2023–04–20	735.0	0.17
12	12	28.2	2023–03–12	872.0	0.1
13	13	25.8	2023–02–25	938.0	0.17
14	14	21.1	2023–01–17	749.0	0.15
15	15	27.9	2023–04–03	1000.0	0.17
16	16	22.7	2023–03–28	1300.0	0.1
17	17	24.3	2023–02–10	700.0	0.17
18	18	29.6	2023–01–22	918.0	0.15
19	19	26.8	2023–04–13	825.0	0.17
21	21	23.5	2023–04–20	1235.0	0.17
22	22	28.2	2023–03–12	1872.0	0.1
23	23	25.8	2023–02–25	938.0	0.17
24	24	21.1	2023–01–17	1549.0	0.15
25	25	27.9	2023–04–03	1000.0	0.17
26	26	22.7	2023–03–28	1300.0	0.1
27	27	24.3	2023–02–10	700.0	0.17
28	28	29.6	2023–01–22	1918.0	0.15

Add | Edit | Delete | Refresh

(Energy Consumption Tab)

Energy Supplier Management System

Customers | Invoices | **Energy Consumption**

0.15 | Search

ID	Customer ID	Days Since Reading	Reading Date	KWh (Meter Reading)	KWh Energy Tariff Rate
4	4	21.1	2023–01–17	749.0	0.15
8	8	29.6	2023–01–22	918.0	0.15
14	14	21.1	2023–01–17	749.0	0.15
18	18	29.6	2023–01–22	918.0	0.15
24	24	21.1	2023–01–17	1549.0	0.15
28	28	29.6	2023–01–22	1918.0	0.15

Add | Edit | Delete | Refresh

(Energy Consumption Search)

Add Customer Meter Info

Customer ID: 37

Days Since Reading: 29

Reading Date (YYYY-MM-DD): 2023-05-20

KWh (Meter Reading): 979

KWh Energy Tariff Rate: 0.1

Save

(Add customer consumption info)

Energy Supplier Management System

Customers | Invoices | Energy Consumption

Search

ID	Customer ID	Days Since Reading	Reading Date (YYYY-MM...	KWh (Meter Reading)	KWh Energy Tariff Rate
1	1	23.5	2023-04-20	935.0	0.17
2	2	28.2	2023-03-12	872.0	0.1
3	3	25.8	2023-02-25	938.0	0.17
4	4	21.1	2023-01-17	749.0	749.0
6	6	22.7	2023-03-28	1200.0	0.1
8	8				0.15
9	9				0.17
10	10				0.1
11	11				735.0
12	12				0.1
13	13				0.17
14	14				0.15
15	15				0.17
16	16				0.1
17	17				0.1
18	18				0.13
19	19				0.17
21	21				0.17
22	22				0.1
23	23				0.17
24	24				0.15
25	25	27.9	2023-04-03	1000.0	0.17
26	26	22.7	2023-03-28	1300.0	0.1
27	27	24.3	2023-02-10	700.0	0.17
28	28	29.6	2023-01-22	1918.0	0.15
31	37	30.0	2023-05-20	989.0	0.15

Confirm Delete

Are you sure you want to delete this row?

Customer Consumption ID: 25
Customer ID: 25
Days Used: 27.9
Reading Date: 2023-04-03
Total Watt Usage: 1000.0
Meter Fee: 0.17

No   Yes

Add   Edit   Delete   Refresh

(Delete customer consumption info)

Energy Supplier Management System

Customers | Invoices | Energy Consumption

Search

ID	Customer ID	Days Since Reading	Reading Date	KWh (Meter Reading)	KWh Energy Tariff Rate
1	1	23.5	2023-04-20	935.0	0.17
2	2	28.2	2023-03-12	872.0	0.1
3	3				0.17
4	4				0.15
6	6				0.1
8	8				0.15
9	9				0.17
10	10				0.1
11	11				0.17
12	12				0.1
13	13				0.17
14	14				0.15
15	15				0.17
16	16				0.1
17	17				0.17
18	18				0.15
19	19				0.17
21	21				0.17
22	22				0.1
23	23				0.17
24	24				0.15
25	25	27.9	2023-04-03	1000.0	0.17
26	26	22.7	2023-03-28	1300.0	0.1
27	27	24.3	2023-02-10	700.0	0.17
28	28	29.6	2023-01-22	1918.0	0.15
31	37	29.0	2023-05-20	979.0	0.1

Edit Customer's Meter Info

Customer ID: 37

Days Since Reading: 30.0

Reading Date (YYYY-MM-DD): 2023-05-20

KWh (Meter Reading): 989.0

KWh Energy Tariff Rate: 0.15

Save   Cancel

Add   Edit   Delete   Refresh

(Edit customer consumption info)

References

AndrewMcAndrewMc 1111 silver badge22 bronze badges, & AlexAlex 4. (1962, November 1). *How to connect java classes to Gui?* Stack Overflow. Retrieved May 5, 2023, from https://stackoverflow.com/questions/35529282/how-to-connect-java-classes-to-gui

Design patterns - MVC pattern. Tutorials Point. (n.d.). Retrieved May 5, 2023, from https://www.tutorialspoint.com/design_pattern/mvc_pattern.htm

Fadatare, R. (2020, September 14). *Java Swing Application With Database Connection*. Java Guides. Retrieved May 5, 2023, from https://www.javaguides.net/2019/07/java-swing-application-with-database-connection.html

GeeksforGeeks. (2022, January 13). *Java database connectivity with mysql*. GeeksforGeeks. Retrieved May 5, 2023, from https://www.geeksforgeeks.org/java-database-connectivity-with-mysql/

GeeksforGeeks. (2023, March 31). *MVC design pattern*. GeeksforGeeks. Retrieved May 5, 2023, from https://www.geeksforgeeks.org/mvc-design-pattern/

Hartman, J. (2023, March 11). *Java Swing Tutorial: How to create a GUI application in Java*. Guru99. Retrieved May 5, 2023, from https://www.guru99.com/java-swing-gui.html