PROJECT

Prepared by

Vishal S
Harish Vijay V
Sampreeth Nagabandi
Dheera Vikhyath Kothapalli

An Integrated Management System for Canteens, Departmental Stores, and Delivery

22AIE111 Object Oriented Programming in Java

Batch B Group 1

An Integrated Management System for Canteens, Departmental Stores, and Delivery

A Project Report

Submitted by

Vishal S - CB.EN.U4AIE22157

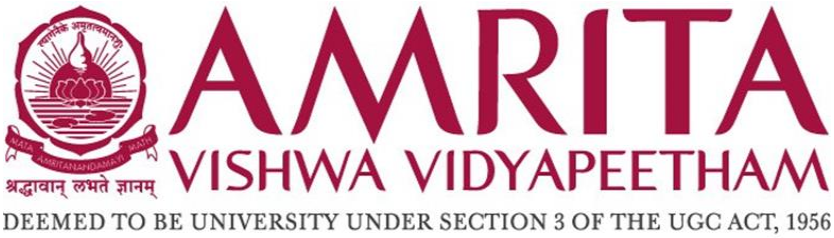
Harish Vijay V – CB.EN.U4AIE22124

Nagabandi Sampreeth - CB.EN.U4AIE22141

Dheera Vikhyath Kothapalli - CB.EN.U4AIE22115

As a part of the subject

Object Oriented Programming in Java



AMRITA SCHOOL OF ARTIFICIAL INTELLIGENCE

AMRITA VISHWA VIDYAPEETHAM

COIMBATORE – 641112 (INDIA)

JULY - 2023

AMRITA SCHOOL OF ARTIFICIAL INTELLIGENCE
AMRITA VISHWA VIDYAPEETHAM
COIMBATORE - 641 112

BONAFIDE CERTIFICATE

This is to certify that the thesis entitled “An Integrated Management System for Canteens, Departmental Stores, and Delivery” submitted by Vishal S (CB.EN.U4IE22157), Harish Vijay V (CB.EN.U4AIE22124), Sampreeth Nagabandi (CB.EN.U4AIE22141), and Dheera Vikhyath Kothapalli (CB.EN.22AIE22115) for the award of the Degree of Bachelor of Technology in the “CSE(AI) ” is a bonafide record of the work carried out by them under our guidance and supervision at Amrita School of Artificial Intelligence, Coimbatore.

Mr. Vipin Das

Project Guide

Dr. K.P. Soman

Professor and Head CEN

AMRITA SCHOOL OF ARTIFICIAL INTELLIGENCE
AMRITA VISHWA VIDYAPEETHAM
COIMBATORE - 641 112

DECLARATION

We, Vishal S (CB.EN.U4IE22157), Harish Vijay V (CB.EN.U4AIE22124), Sampreeth Nagabandi (CB.EN.U4AIE22141), and Dheera Vikhyath Kothapalli (CB.EN.22AIE22115), hereby declare that this thesis entitled "An Integrated Management System for Canteens, Departmental Stores, and Delivery" is the record of the original work done by us under the guidance of Mr. Vipin Das, Assistant Professor, Centre for Computational Engineering and Networking, Amrita School of Artificial Intelligence, Coimbatore. To the best of our knowledge this work has not formed the basis for the award of any degree/diploma/ associate ship/fellowship/or a similar award to any candidate in any University.

Place: Coimbatore

Date: 10-07-2023 **Signature of the Students**

Acknowledgement

We would like to express our special thanks of gratitude to our teacher Mr. Vipin Das Sir), who gave us the golden opportunity to do this wonderful project on the topic “An Integrated Management System for Canteens, Departmental Stores, and Delivery”, which also helped us in doing a lot of Research and we came to know about so many new things. We are thankful for the opportunity given.

We would also like to thank our group members, as without their cooperation, we would not have been able to complete the project within the prescribed time.

TABLE OF CONTENTS

Table of Figures

Abstract:

The Integrated Management System is an innovative project designed to revolutionize and automate administrative tasks in a canteen environment. It aims to provide a user-friendly graphical user interface (GUI) that simplifies operations and enhances the overall experience for both canteen employees and consumers.

The system encompasses various functionalities such as menu management, order processing, inventory tracking, and centralized database integration. With an intuitive GUI, canteen staff can effortlessly manage the menu by adding new items, modifying prices, and monitoring stock levels. Customers will enjoy a seamless experience as they browse the menu, place orders, and make payments.

Security is a crucial aspect of the system, and robust user authentication mechanisms will be implemented to safeguard sensitive information and restrict unauthorized access. This ensures that only authorized individuals can access and manage critical aspects of the system.

By implementing the Integrated Management System, the canteen aims to optimize its operations and enhance overall efficiency. Automating routine tasks and providing a streamlined user experience will simplify canteen management and contribute to a more organized operation. The system will reduce manual errors, improve order accuracy, and enable better inventory control.

The menu management feature allows canteen staff to easily update and modify menu items, including descriptions, prices, and availability. This functionality ensures that the menu is always up to date, providing accurate information to customers and minimizing confusion.

The order processing component streamlines the entire order lifecycle. Customers can browse the menu, select their desired items, and place orders through the user-friendly interface. Canteen staff can efficiently view and process these orders, ensuring timely preparation and delivery. The system can also handle customization requests and special instructions from customers, enhancing the flexibility and personalized experience.

Inventory tracking is an essential aspect of the system. By maintaining a centralized database, the system enables real-time monitoring of stock levels, ensuring that the canteen never runs out of popular items and minimizing waste. Canteen staff can receive automated notifications when stock levels are low, facilitating timely restocking and reducing inventory management complexities.

Furthermore, the system incorporates secure payment processing, enabling customers to make cashless transactions conveniently. By automating routine tasks and ensuring accurate inventory management, the system simplifies canteen management and contributes to a more organized and productive environment. In addition to the existing meal delivery service, the system will offer the convenience of ordering a wide range of grocery products, such as packaged food items, beverages, toiletries, stationery, and other daily essentials. This service will save individuals valuable time and eliminate the need to leave the college campus or rely on off-campus alternatives.

1 MODULE-1

1.1 PARTS INVOLVED AND OBJECTIVE

- Inventory
- Login Administration
- Graphical User Interface

Objective: Efficiently manage inventory and maintain a comprehensive database with a user-friendly Graphical User Interface.

1.2 DESCRIPTION OF THE MODULE

1.2.1 Admin Access

Admin module consists of three classes: "LoginFrame," "loginIDandPassword," and "Admin." These classes are part of a JavaFX application that manages the login functionality for an admin interface. Here's an overview of each class:

LoginFrame:

The LoginFrame class represents a data model for storing login credentials. It has properties for "username" and "password," implemented as StringProperties. It provides methods to get and set the username and password.

loginIDandPassword:

The loginIDandPassword class handles user login-related operations. It interacts with a text file called "LoginData.txt" located at the specified file path. Here are the methods and functionalities of this class:

- doesExist(String username): Checks if a user with the specified username exists in the "LoginData.txt" file.
- passwordMatch(String username, String password): Checks if the provided password matches the one stored for the given username in the "LoginData.txt" file.
- addUser(String username, String password): Adds a new user to the "LoginData.txt" file.
- updatePassword(String username, String newPassword): Updates the password for the user with the specified username in the "LoginData.txt" file.
- removeUser(String username): Removes the user with the specified username from the "LoginData.txt" file.
- updateUserID(String username, String newUserID): Updates the user ID for the user with the specified username in the "LoginData.txt" file.

Admin:

The Admin class serves as the controller for the admin interface. It handles user actions and manages the corresponding functionality. Here's an overview of its functionalities:

- addUser(ActionEvent event): Adds a user to the system by calling the necessary methods from loginIDandPassword.
- removeUser(ActionEvent event): Removes a user from the system by calling the necessary methods from loginIDandPassword.
- updatePassword(ActionEvent event): Updates the password for a user by calling the necessary methods from loginIDandPassword.
- updateUsername(ActionEvent event): Updates the username for a user by calling the necessary methods from loginIDandPassword.
- transactions(ActionEvent event): Navigates to the Transactions.fxml view.
- employee(ActionEvent event): Navigates to the Employee.fxml view.
- initialize(): Initializes the table columns for displaying login data and calls the populateTable() method.
- populateTable(): Reads the "LoginData.txt" file and populates the TableView with the login data.

1.2.2 Inventory

This module consists of three classes: "InventoryActions," "InventoryItem," and "Inventory." These classes are part of a JavaFX application that manages an inventory system. Here's an overview of each class:

InventoryActions:

- The InventoryActions class handles various operations related to the inventory. It interacts with a text file called "Inventory.txt" located at the specified file path. Here are the functionalities provided by this class:
- addItem(String itemName, Integer price, int quantity): Adds a new item to the inventory by appending its details to the "Inventory.txt" file if the item does not already exist.
- removeItem(String itemName): Removes an item from the inventory by excluding its line from the "Inventory.txt" file.
- updateItemPrice(String itemName, Integer newPrice): Updates the price of an item in the inventory by modifying the corresponding line in the "Inventory.txt" file.
- update(String name, Integer price, Integer quantity): A helper method that calls the necessary methods (updateItemPrice and updateItemQuantity) to update an item's price and quantity in the inventory.
- updateItemQuantity(String itemName, Integer newQuantity): Updates the quantity of an item in the inventory by modifying the corresponding line in the "Inventory.txt" file.
- checkAvailability(String itemName, Integer quantity): Checks if a certain quantity of an item is available in the inventory.
- getItemPrice(String itemName): Retrieves the price of an item from the inventory.
- updateInventory(String itemName, Integer quantity): Updates the inventory by increasing or decreasing the quantity of an item.

InventoryItem:

- The InventoryItem class represents an item in the inventory. It has private fields for "item" (item name), "price," and "quantity." The class provides getters for these fields.

Inventory:

The Inventory class serves as the controller for the inventory interface. It handles user actions and manages the corresponding functionality. Here's an overview of its functionalities:

- add(ActionEvent event): Adds an item to the inventory by retrieving the entered item name, price, and quantity from the text fields and calling the addItem() method from InventoryActions. It then clears the text fields and updates the inventory table.
- remove(ActionEvent event): Removes an item from the inventory by retrieving the entered item name from the text field and calling the removeItem() method from InventoryActions. It then clears the text field and updates the inventory table.
- update(ActionEvent event): Updates an item's price, quantity, or both in the inventory by retrieving the entered item name, price, and quantity from the text fields and calling the necessary methods from InventoryActions. It then clears the text fields and updates the inventory table.
- initialize(): Initializes the table columns for displaying inventory data and calls the populateInventoryTable() method.
- populateInventoryTable(): Reads the "Inventory.txt" file and populates the TableView with the inventory data.

1.3 MOTIVATION FOR THE MODULE

The Admin Access Module serves a crucial purpose within the Application by providing authorized user access and facilitating user management. Its key motivations include:

- **Authorized User Access:** The module ensures that only authorized personnel can access the system, safeguarding against unauthorized manipulation. User authentication and authorization mechanisms prevent unauthorized access, enhancing overall security.
- **User Management:** The module enables administrators to add and remove users, ensuring control over system access. By modifying usernames and passwords, administrators can maintain user accountability and system integrity.
- **Sign-up Process:** The module does not include a sign-up button, allowing new users to be added only by authorized personnel. This approach safeguards against unauthorized account creation and ensures administrators oversee the onboarding process.
- **User Credential Modification:** Administrators can modify usernames and passwords, adapting to personnel changes and enforcing password security policies. This capability enhances system security and limits access to authorized users.

The Inventory Module within a Grocery and Canteen Management System is crucial for effective operations and financial management. Its significance can be highlighted under the following topics:

- **Stock Control and Management:** The inventory module enables accurate tracking and control of stock items, including their names, quantities, and prices. It provides real-time visibility into available stock levels, allowing businesses to manage inventory efficiently, avoid stockouts, and minimize wastage.
- **Order Fulfillment and Purchase Planning:** By maintaining an up-to-date inventory record, the module supports order fulfillment processes. It helps determine the availability of items, facilitates accurate order placement, and assists in planning

purchases based on demand and stock levels. This ensures timely delivery of products and reduces the risk of overstocking or understocking.

- **Financial Management and Cost Control:** The inventory module provides essential data for financial management and cost control. It enables accurate valuation of inventory, cost analysis, and profit calculations. By knowing the quantity and price of items in stock, businesses can make informed decisions regarding pricing strategies, discounts, and promotions, ultimately optimizing profitability.
- **Reporting and Analytics:** The module generates comprehensive reports and analytics related to inventory. This includes stock turnover rates, slow-moving or obsolete items, and sales trends. Such insights help businesses make data-driven decisions, identify opportunities for improvement, and enhance overall inventory management strategies.

1.4 Relevance for the module in the System

1.4.1 Admin Access

The Admin Access Module is crucial in the Integrated Management System as it provides security, user management, system administration, accountability, and system integrity. It ensures that only authorized personnel can access the system, preventing unauthorized access and manipulation. The module allows administrators to add, remove, and manage users, controlling their access and maintaining data integrity. It also enables system configuration, maintenance, and monitoring, ensuring smooth operation. By logging user activities and generating audit trails, the module promotes accountability and transparency. Overall, it plays a vital role in protecting the system, managing users, and maintaining the system's stability and integrity.

1.4.2 Inventory

The Inventory Module is essential in the Grocery and Canteen Management System as it enables efficient tracking, management, and control of stock. It helps maintain accurate records of product names, quantities, and prices, ensuring optimal inventory levels and preventing stockouts or overstocking. The module facilitates streamlined ordering, restocking, and inventory reconciliation processes, minimizing wastage and improving cost-effectiveness. It provides real-time insights into stock availability, enabling timely replenishment and preventing shortages. The Inventory Module plays a crucial role in ensuring smooth operations, minimizing errors, optimizing inventory utilization, and ultimately enhancing customer satisfaction by ensuring the availability of products and maintaining accurate pricing information.

1.5 UML OF THE MODULE

Figure 1.1 Admin Access UML

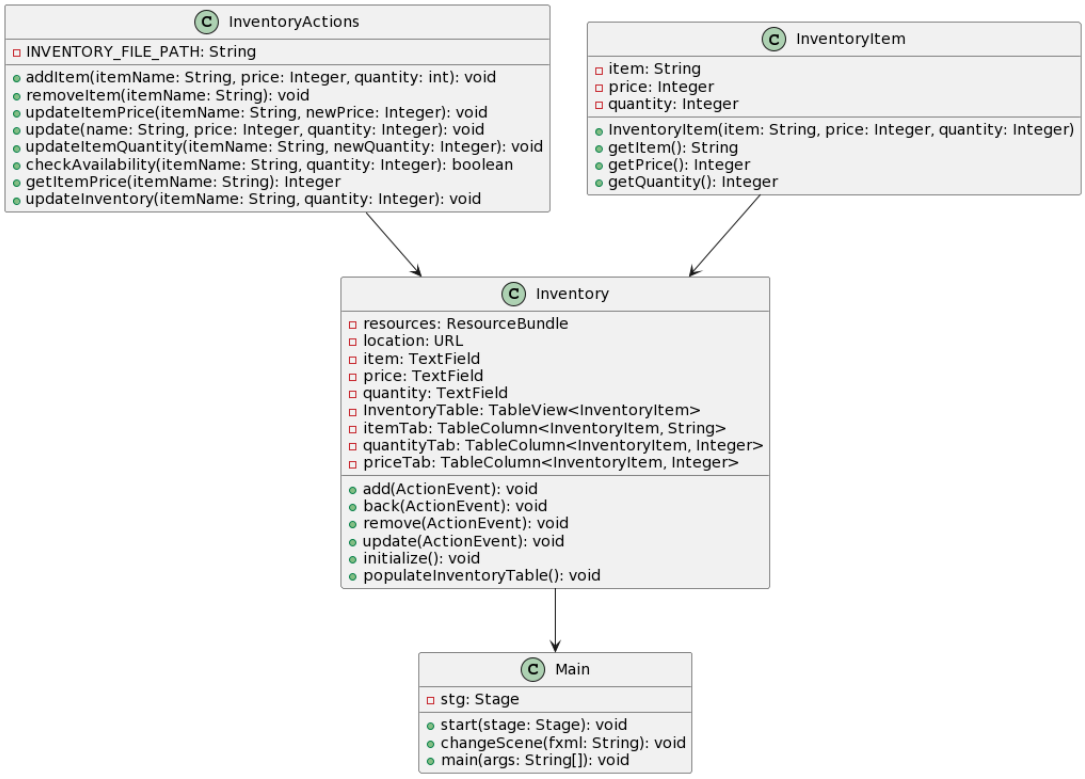


Figure 1.2 Inventory UML

2 MODULE – 2

2.1 DESCRIPTION OF THE MODULE

The given code defines the necessary imports and class declaration for the "Canteen" class. It includes private static final constants for file paths used to store canteen menu and cart data. The code also utilizes annotations (@FXML) to inject and link GUI components defined in an associated FXML file.

In terms of instance variables, the code declares TableView objects for displaying the cart and menu items, as well as various TextFields, CheckBoxes, and RadioButtons for user input.

The code creates ObservableList objects for storing grocery items (menu items) and cart items. These lists are used to populate the TableView objects with data.

The code contains event handling methods that handle various actions such as adding items to the cart, removing items from the cart, updating cart items, making payments, checking order status, and clearing the cart.

Additionally, there are helper methods included in the code. These methods handle saving cart data to a file, loading menu and cart data from files, calculating the total price of items in the cart, and populating the TableView objects with data.

An initialization method, annotated with @FXML initialize(), is present in the code. This method is called when the GUI is initialized and performs tasks such as setting up the TableView objects, loading menu and cart data, and calculating the total price.

Overall, the code provides functionality for managing a canteen application's menu, cart, payments, and order status using a graphical user interface implemented with JavaFX.

The `Cart` class manages shopping cart items by providing methods to load and save items from/to a file. It stores information such as the item name, quantity, and price. The `loadCartFromFile` method reads the cart items from a text file, while the `saveCartToFile` method writes the items back to the file. This class enables the management and persistence of shopping cart data.

2.2 MOTIVATION OF THE MODULE

Order Management:

- Streamlined Order Placement: The system allows customers to place orders easily and efficiently, reducing waiting times and enhancing the overall customer experience.
- Error Reduction: By automating the order process, human errors in taking orders and processing them manually are minimized, leading to greater accuracy and customer satisfaction.

- Order Tracking: Customers can check the status of their orders, ensuring transparency and keeping them informed about the progress.

Payment Processing:

- Multiple Payment Options: By offering various payment methods such as UPI, cash, and e-wallets, the system caters to a wider range of customer preferences, enhancing convenience and customer satisfaction.
- Faster Payments: Automating the payment process accelerates transaction times, reducing queues and enhancing operational efficiency.

2.3 Relevance of the Module in the System

It is responsible for managing the items that customers add to their cart during the ordering process.

- Order Management: The Cart module facilitates the management of orders placed by university members within the canteen system. It allows customers to add items to their cart, update quantities, remove items, and calculate the total cost of their order.
- Customer Experience: The Cart module enhances the overall customer experience by providing a convenient and user-friendly way to manage their food selections. It ensures that customers can review and modify their orders before proceeding to payment.
- Data Persistence: The Cart module handles the loading and saving of cart items to a file. It enables the system to remember the items in the cart even if the user logs out or closes the application. This persistence ensures that users can continue their order from where they left off.
- Integration with Payment Module: The Cart module interacts with the payment module to transfer the selected items and their total cost for payment processing. It provides the necessary information to generate the payment request and complete the transaction.
- Order Fulfilment: The cart items are crucial for the canteen staff to prepare and deliver the orders accurately. The cart information, such as the items, quantities, and prices, is used by the canteen management system to fulfil the customer's order effectively.
- Data Accuracy: By storing cart items separately from other components of the system, the Cart module ensures data accuracy and integrity. It prevents the loss or mix-up of items during the order management process.

In summary, the Cart module within the Canteen Management component of the university management system is essential for managing customer orders, providing a smooth ordering experience, facilitating payment processing, and ensuring accurate order fulfilment.

2.4 UML OF THE MODULE

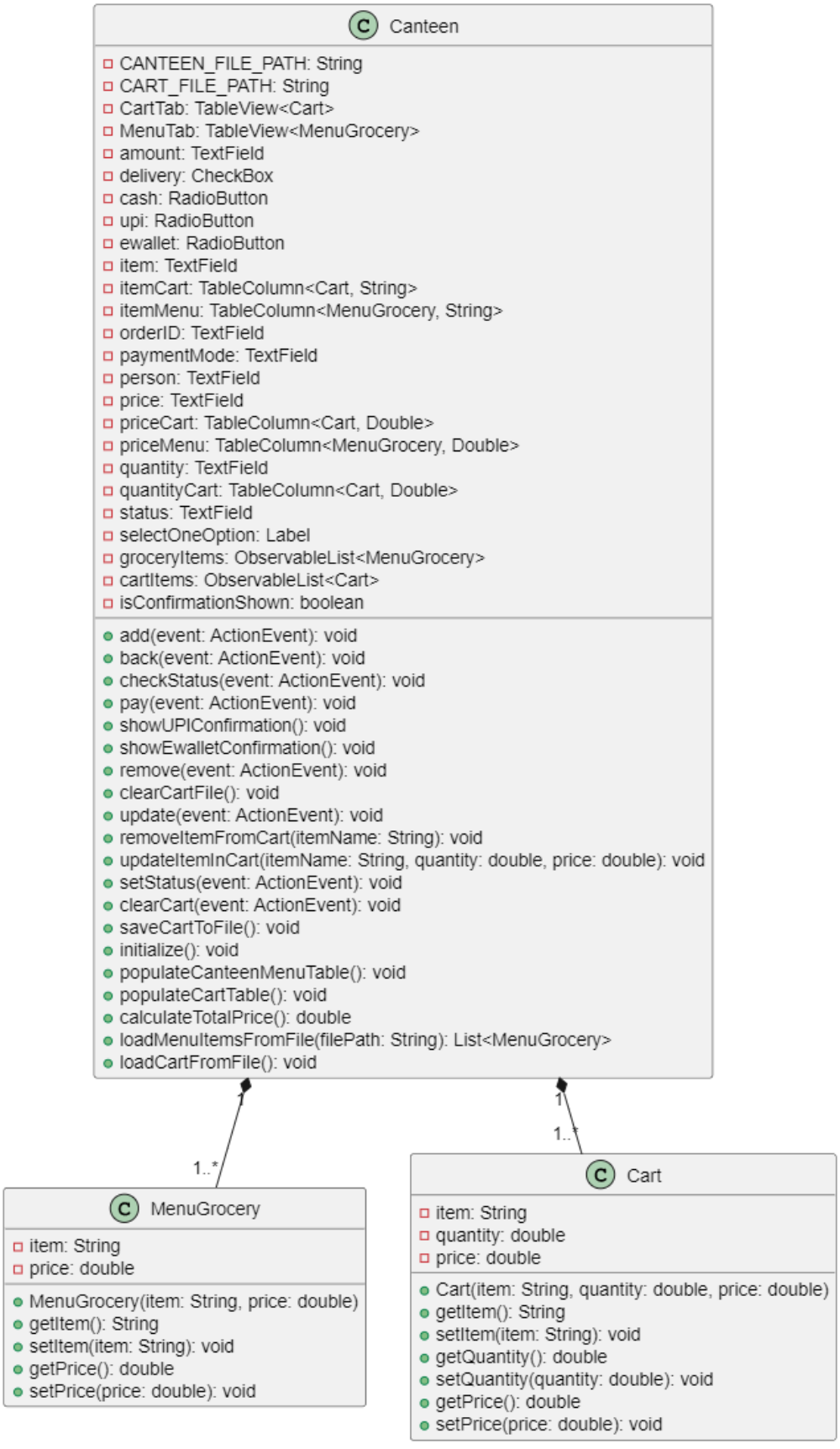


Figure 2.1 Canteen UML

Figure 2.2 Grocery UML

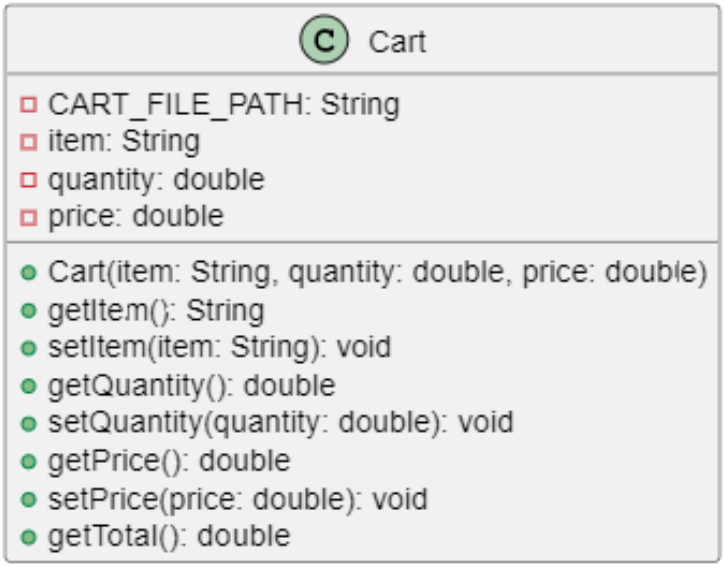


Figure 2.3 Cart UML

3 MODULE – 3

3.1 DESCRIPTION OF THE MODULE

The `Delivery` class is a controller for the "Delivery.fxml" view, handling actions related to delivery management. It assigns, checks, updates, and removes deliveries. It utilizes UI elements like checkboxes, text fields, and a table view. The `initialize` method sets up the UI, and `populateTable` retrieves delivery history from `DeliveryActions` to populate the table view.

The `DeliveryActions` class manages delivery operations, including assignment, status updates, deletion, and checking. It uses file I/O to store and retrieve delivery information.

The `DeliveryFrame` class represents a delivery with its associated information such as the order ID, delivery person, and status. It provides methods to access and modify these properties. The `toText` method converts the `DeliveryFrame` object into a string representation, where the properties are concatenated with commas. The `fromText` method takes a string and creates a `DeliveryFrame` object by parsing the string and extracting the order ID, delivery person, and status.

The `Menu` class is responsible for managing the menu items in the canteen and grocery menus. It provides methods for adding, removing, and updating menu items, as well as retrieving the price of an item. The class utilizes JavaFX for the UI components, such as checkboxes, text fields, and table views. It interacts with the `MenuItem` class to represent individual menu items and the `canteenMenuTable` and `groceryMenuTable` table views to display the menu items. The class reads menu items from text files, saves them to the files, and performs operations based on user actions.

The `MenuGrocery` class represents a grocery item in the menu. It has properties for the item name and price, along with getters and setters for accessing and modifying these properties.

The `MenuItem` class represents a menu item in the canteen menu. It has properties for the item name and price, along with getters and setters for accessing and modifying these properties. This class is used by the `Menu` class to represent and manipulate menu items in the canteen menu.

The `Slot` class manages slots for breakfast, lunch, and dinner, allowing users to add, remove, and clear items within each slot. It uses JavaFX for the user interface and maintains separate text files for each slot. The class provides methods to interact with the slot files, read their content, and update the UI accordingly. The `initialize` method sets up the UI, and a back button allows users to navigate to the menu screen.

3.2 MOTIVATION OF THE MODULE

The Delivery module in the Canteen Management System aims to provide convenience, save time, and increase accessibility for students and staff. By offering food delivery services, the module eliminates the need for individuals to physically visit the canteen, providing them with a more convenient way to order their meals. This is particularly beneficial for individuals with busy schedules or limited time between classes.

Moreover, the delivery system helps in saving time by allowing users to skip long queues at the canteen during peak hours. Instead, they can simply place their orders online or through a mobile app and have the food delivered to their desired location. This time-saving feature enhances efficiency and ensures that students can make the most of their break time.

Additionally, the module promotes increased accessibility. It caters to individuals who may face challenges in reaching the canteen due to physical disabilities or mobility constraints. By offering food delivery options, the system ensures that all students, regardless of their circumstances, can easily access and enjoy the canteen's offerings.

Real-time Updates: If the menu system is web-based or integrated with a mobile app, it can provide real-time updates on menu changes, availability of dishes, and promotional offers. Students can stay informed about the latest offerings and make informed decisions about their meals.

Improved food variety: With a menu system in place, canteen managers can easily introduce a wide range of food options. This enables students to have a greater variety of meals to choose from, catering to different preferences, dietary restrictions, and cultural backgrounds.

Efficient menu management: The module allows canteen managers or administrators to efficiently manage the menu items available during different meal slots. It provides options to add new items, remove existing items, and view all the items in each slot.

Customization and flexibility: By providing checkboxes for each meal slot (breakfast, lunch, dinner) and a text field for entering item names, the module enables customization and flexibility in managing the menu. Canteen managers can select the appropriate checkboxes and add or remove specific items based on the availability and preferences.

3.3 Relevance of the Module in the System

Efficient Delivery Management: The delivery codes enable the canteen management system to effectively manage and track the delivery process. By assigning deliveries to specific orders, updating the delivery status, and deleting deliveries when needed, the system ensures smooth coordination and efficient delivery operations.

Order Tracking and Transparency: The delivery codes provide visibility into the delivery status of each order. Customers can easily check the status of their orders, whether they have been delivered or not. This transparency enhances customer satisfaction and confidence in the canteen's delivery services.

Accurate Delivery Records: The codes ensure accurate and up-to-date delivery records within the system. By assigning deliveries, updating statuses, and deleting deliveries, the system maintains accurate information about each order's delivery history. This data can be used for reporting, analysis, and improving the overall delivery process.

MenuGrocery: This class represents a grocery item in the canteen's menu. It stores information about the item's name and price. It is relevant because it allows the canteen management system to include grocery items in the menu, expanding the range of available choices for customers.

MenuItem: This class represents a food item in the canteen's menu. It also stores information about the item's name and price. It is relevant because it enables the canteen management system to manage and display the food items available for ordering.

Menu: This class is responsible for handling the menu-related operations in the canteen management system. It provides functionalities to add, remove, update, and populate the canteen and grocery menus. It also interacts with files to load and save menu items. This class is relevant as it ensures that the menu information is properly managed and updated within the system, allowing for smooth ordering and management of food and grocery items in the canteen.

Adding Items: The code allows the addition of menu items to specific meal slots. When a user enters an item in the text field and clicks the "add" button, the code checks which meal slots (breakfast, lunch, or dinner) are selected through checkboxes and appends the item to the corresponding file (Breakfast_Slot.txt, Lunch_Slot.txt, or Dinner_Slot.txt). This functionality enables the canteen staff to update the menu items available in each meal slot.

Removing Items: Users can remove specific items from the meal slots by entering the item in the text field and clicking the "remove" button. The code reads the corresponding file for the selected meal slot, removes the matching item from the file, and updates the file accordingly. This feature allows for dynamic changes to the menu items in each meal slot.

Clearing Slots: The code provides an option to clear all items from a particular meal slot by selecting the respective checkbox and clicking the "clear" button. It clears the content of the corresponding file, effectively resetting the meal slot to an empty state. This functionality enables easy management of menu items in each meal slot.

Displaying Slots: The code retrieves the content from the three files (Breakfast_Slot.txt, Lunch_Slot.txt, and Dinner_Slot.txt) and displays them in the respective text areas (breakfastSlot, lunchSlot, and dinnerSlot). This functionality allows users to view the current menu items available in each meal slot.

3.4 UML OF THE MODULE

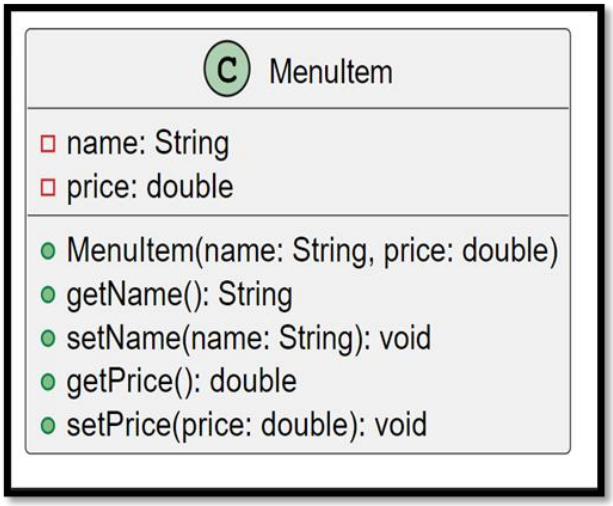


Figure 3.1 MenuItem UML

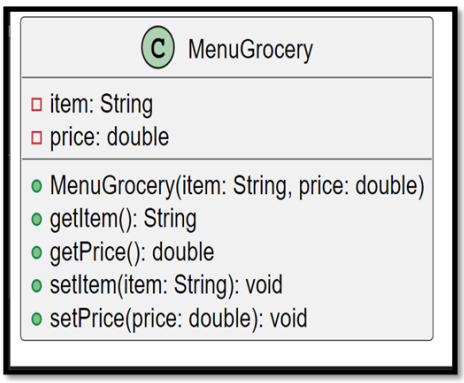


Figure 3.2 MenuGrocery UML

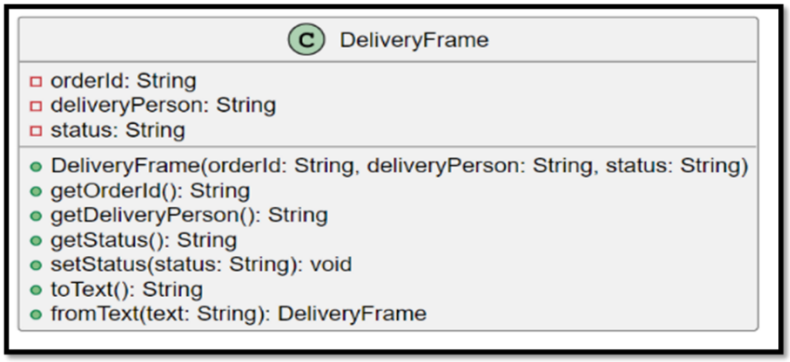


Figure 3.3 DeliveryFrame UML

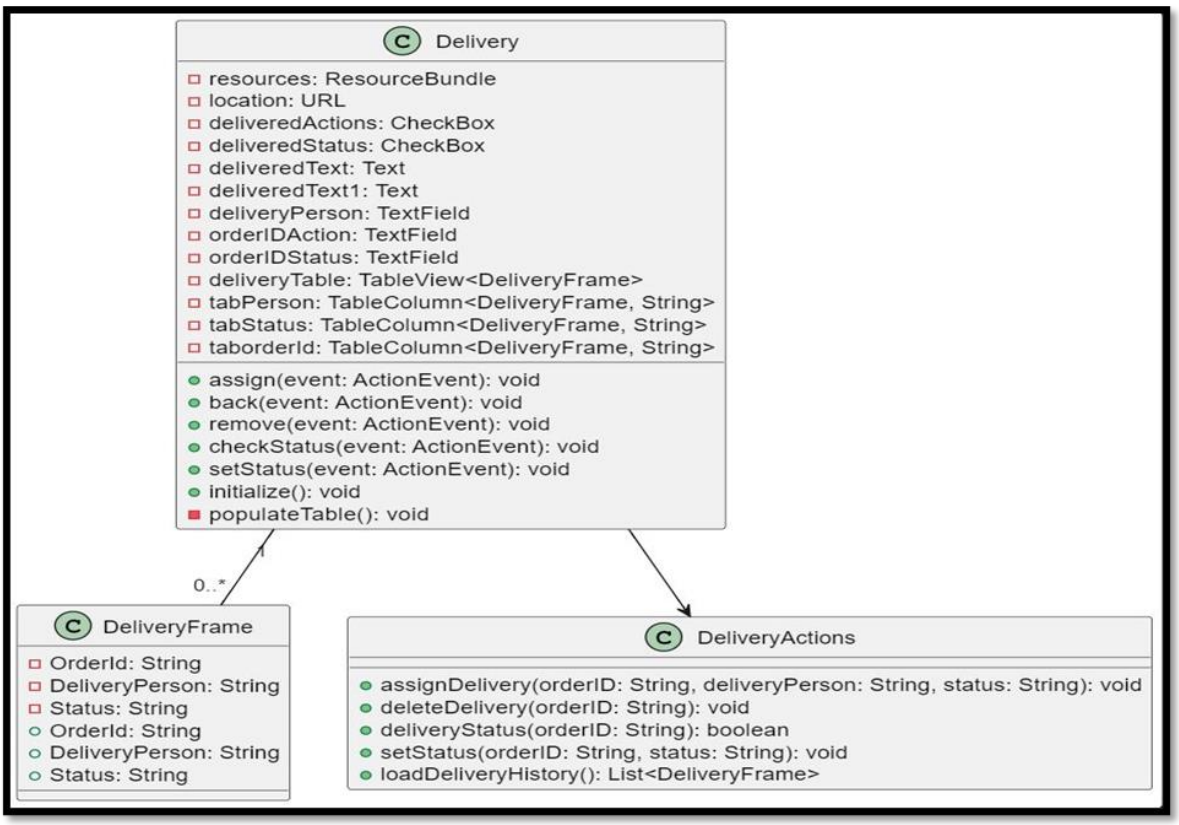


Figure 3.4 Delivery UML

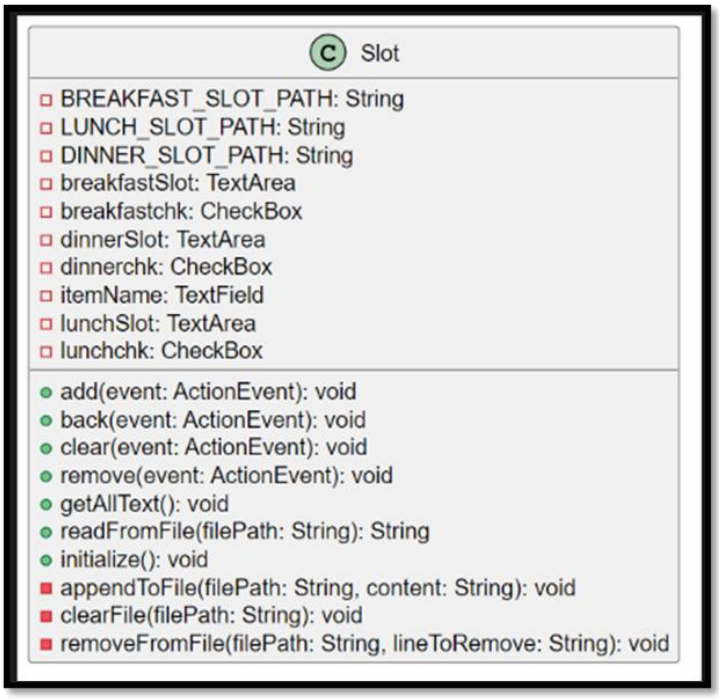


Figure 3.5 Slot UML

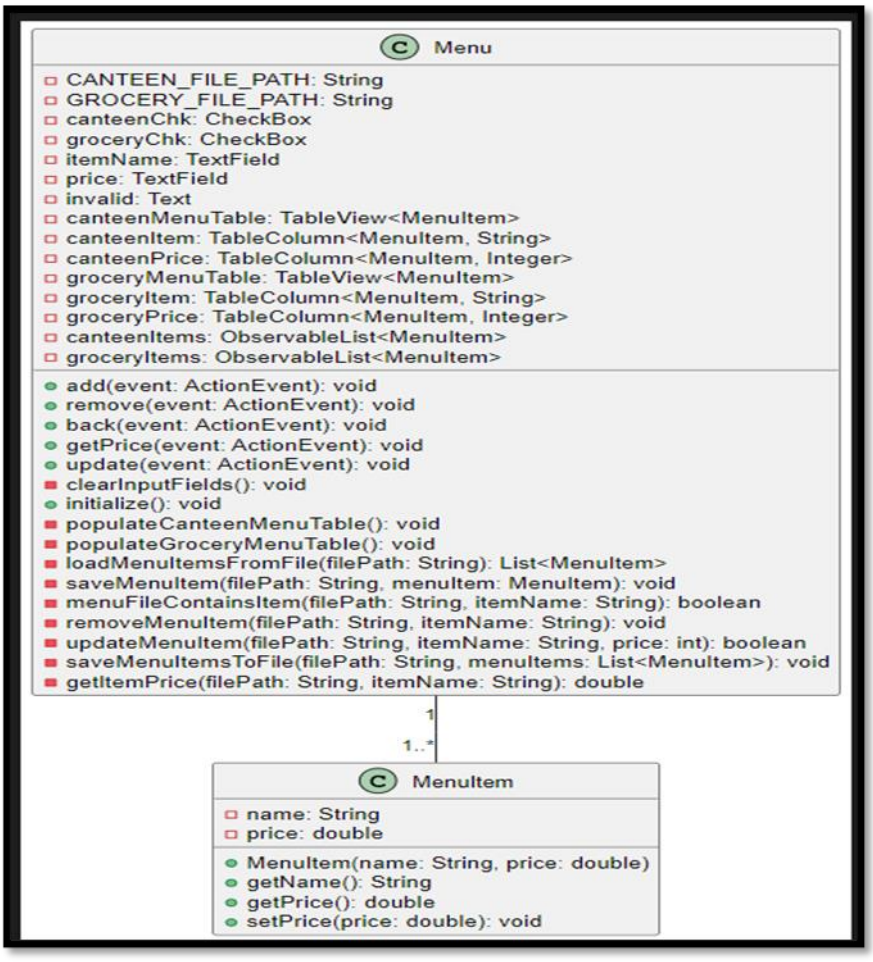


Figure 3.6 Menu UML

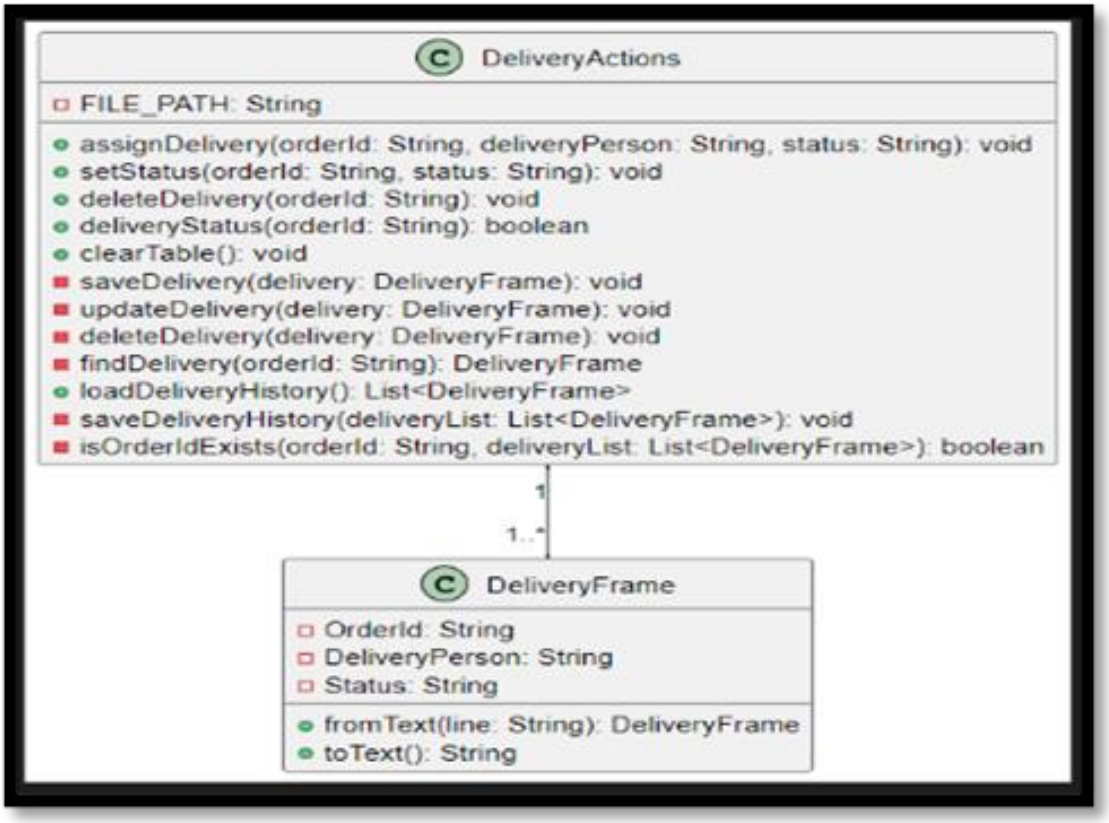


Figure 3.7 Delivery Actions UML

4 MODULE - 4

4.1 DESCRIPTION OF THE MODULE

Transaction System:

The Transaction class represents a transaction object. It has private instance variables for orderID, amount, and paid, along with corresponding getter and setter methods. The class provides two constructors, one that takes the orderID, amount, and paid as parameters, and another constructor that takes TextField objects for orderID and amount as well as a Boolean value for paid. This class is responsible for storing and retrieving transaction-related information.

The TransactionHandler class provides methods for handling transactions, including loading transactions from a file, saving transactions to a file, adding a new transaction, removing a transaction, and updating an existing transaction. It contains a constant TRANSACTIONS_FILE_PATH that holds the file path where the transaction data is stored.

The Transactions class is a controller class associated with a graphical user interface (GUI) defined in an FXML file.

Employee system:

The Employee class is a controller class associated with an FXML file. It contains various @FXML-annotated components such as TableView, TableColumn, and TextField for displaying and manipulating employee data. The class includes event handling methods for adding, removing, and updating employees based on user input.

The EmployeeActions class provides methods for performing actions on employee data, including adding an employee, removing an employee, and updating employee information such as name and salary.

The EmployeeFrame class represents an employee object and serves as a data model for employee-related information. It contains private instance variables for id, name, and salary, along with corresponding getter and setter methods

4.2 MOTIVATION OF THE MODULE

- **Accurate Profit Calculation:** By maintaining a transaction history, the system can accurately calculate profits. It tracks all sales, purchases, and expenses, providing a comprehensive overview of the financial activities. This eliminates the chances of losing bills or missing transactions, ensuring accurate profit calculations.
- **Elimination of Discrepancies**: With a transaction history, the system can identify and resolve any discrepancies or inconsistencies in financial records. It becomes easier to track down any missing or mismatched transactions, allowing for better financial control and reducing the risk of financial losses.
- **Efficient Record Keeping:** The module enables centralized and efficient record keeping of all transactions. It provides a single platform to store and manage transaction data, eliminating the need for manual or paper-based record-keeping systems. This not

only saves time but also ensures data integrity and easy access to historical transaction information.

- **Employee Transaction Management:** The module includes functionality to manage employee transactions, such as salary payments. It allows for easy tracking of employee expenses and ensures transparency in financial operations. By maintaining employee transaction records, the system provides a comprehensive overview of all financial aspects related to the workforce.

4.3 Relevance of the Module in the System

The Transaction class represents individual transactions made within the canteen, such as orders and payments. It stores essential information like the order ID, amount, and payment status. This class is relevant as it provides the data structure to handle and manage transaction details, ensuring accurate recording and tracking of financial activities within the canteen management system.

The TransactionHandler class contains methods to load, save, add, remove, and update transactions. These operations are crucial for maintaining and managing the transaction history. This class ensures the relevance and integrity of transaction data by providing functionalities to retrieve, modify, and store transaction information efficiently. It enables effective financial control and accurate profit calculation within the canteen management system.

The Transactions class acts as a controller for the transaction-related GUI components defined in the associated FXML file. It handles events such as adding, removing, and updating transactions based on user input. This class ensures the user interface's relevance by providing functionalities to interact with the transaction data. It also populates the GUI with transaction information and allows for easy navigation and management of transactions within the canteen management system.

The Employee class represents individual employees working within the canteen. It stores relevant information like employee ID, name, and salary. This class is relevant as it provides the data structure to manage and track employee details within the canteen management system. It ensures accurate recording of employee information, enabling effective employee

The EmployeeActions class contains methods to add, remove, and update employee details within the canteen management system. These operations are crucial for maintaining employee records, managing salaries, and updating employee information. This class ensures the relevance of employee data by providing functionalities to handle employee-related operations efficiently, contributing to streamlined employee management.

The EmployeeFrame class serves as a data model for storing employee information. It includes attributes like ID, name, and salary. This class is relevant as it provides a structured representation of employee data, facilitating data integrity and accurate reporting within the canteen management system. It helps in populating tables, generating reports, and providing insights into employee-related financial aspects.

4.4 UML OF THE MODULE

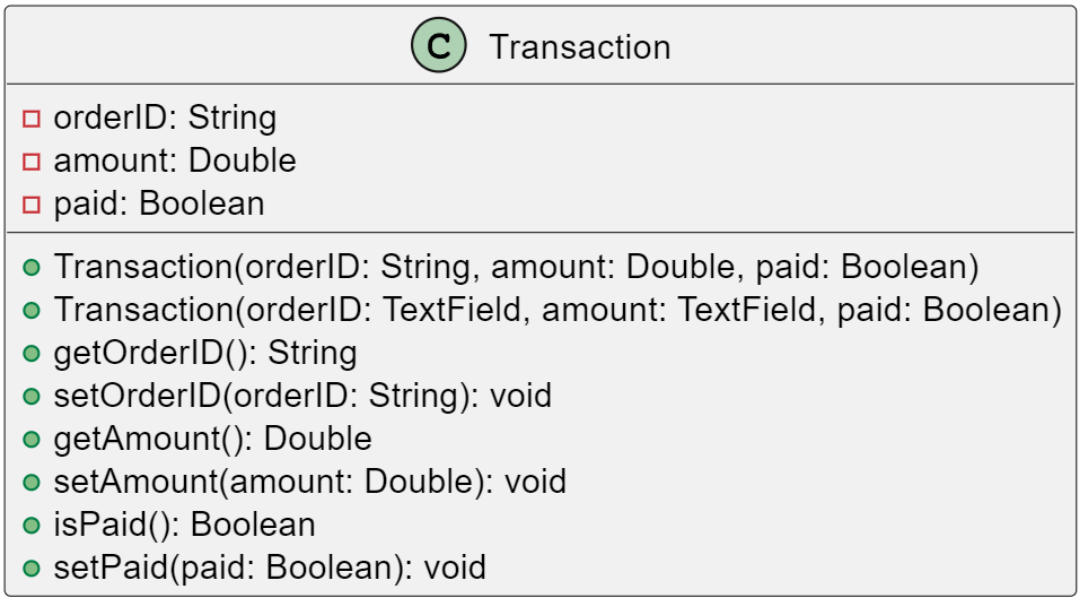


Figure 4.1 Transaction UML

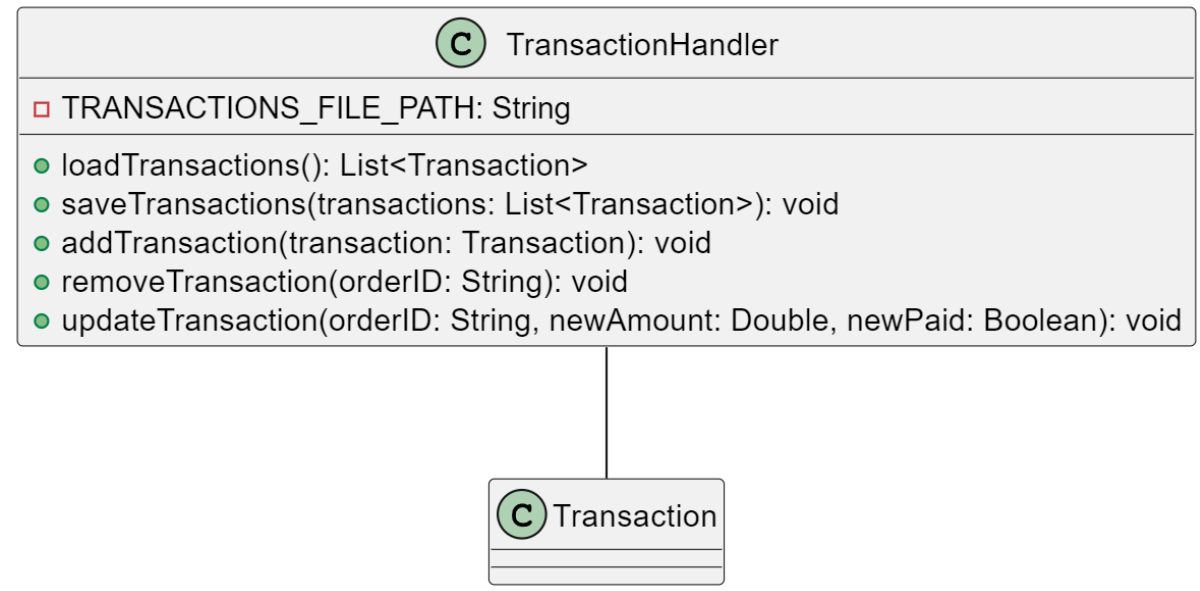


Figure 4.2 Transaction Handler UML

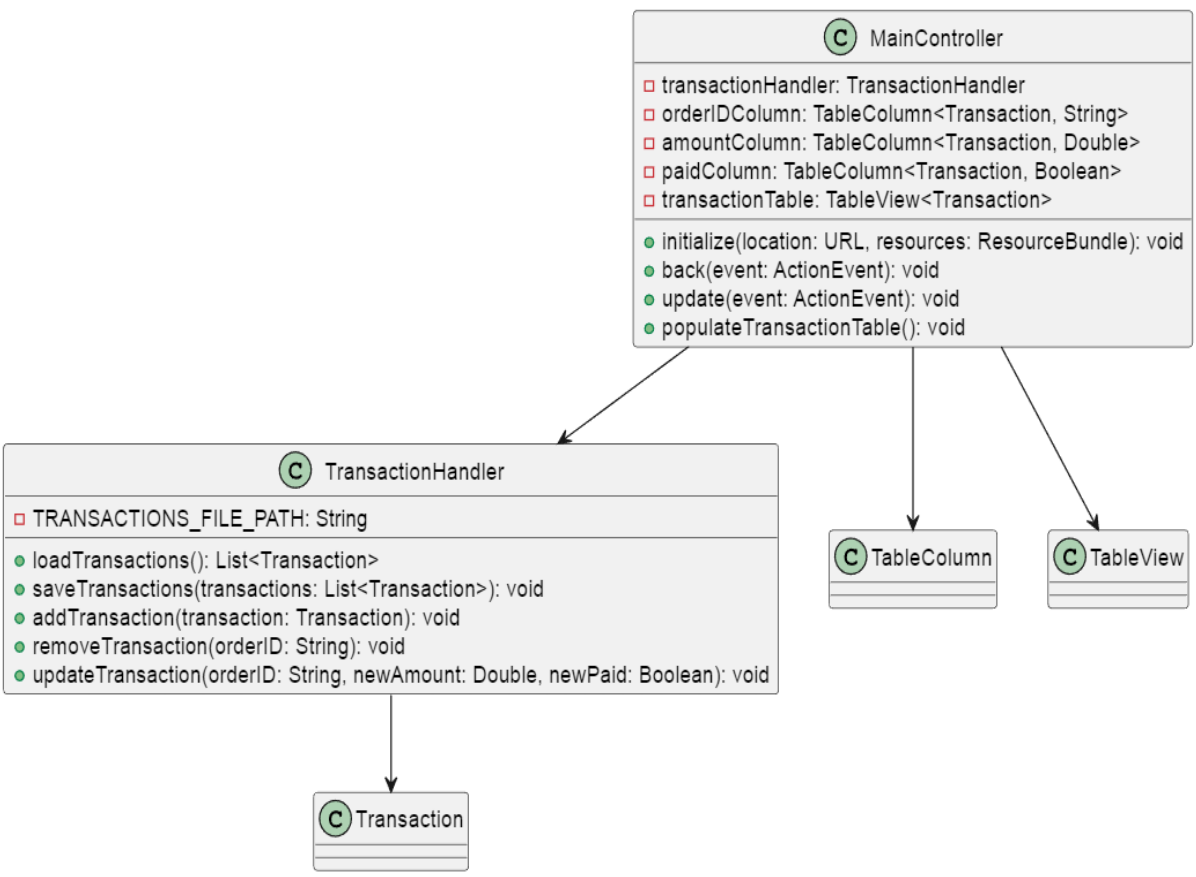


Figure 4.3 MainController UML

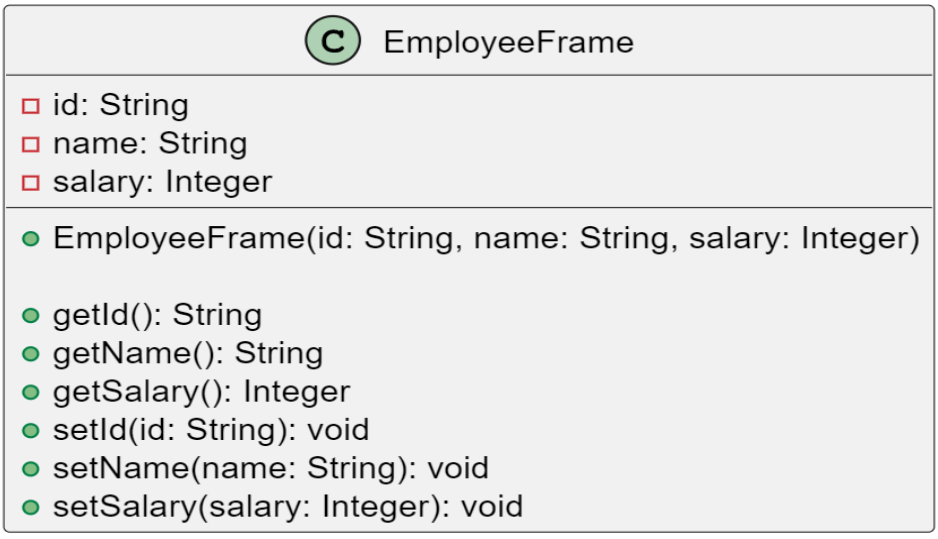


Figure 4.4 Employee Frame UML

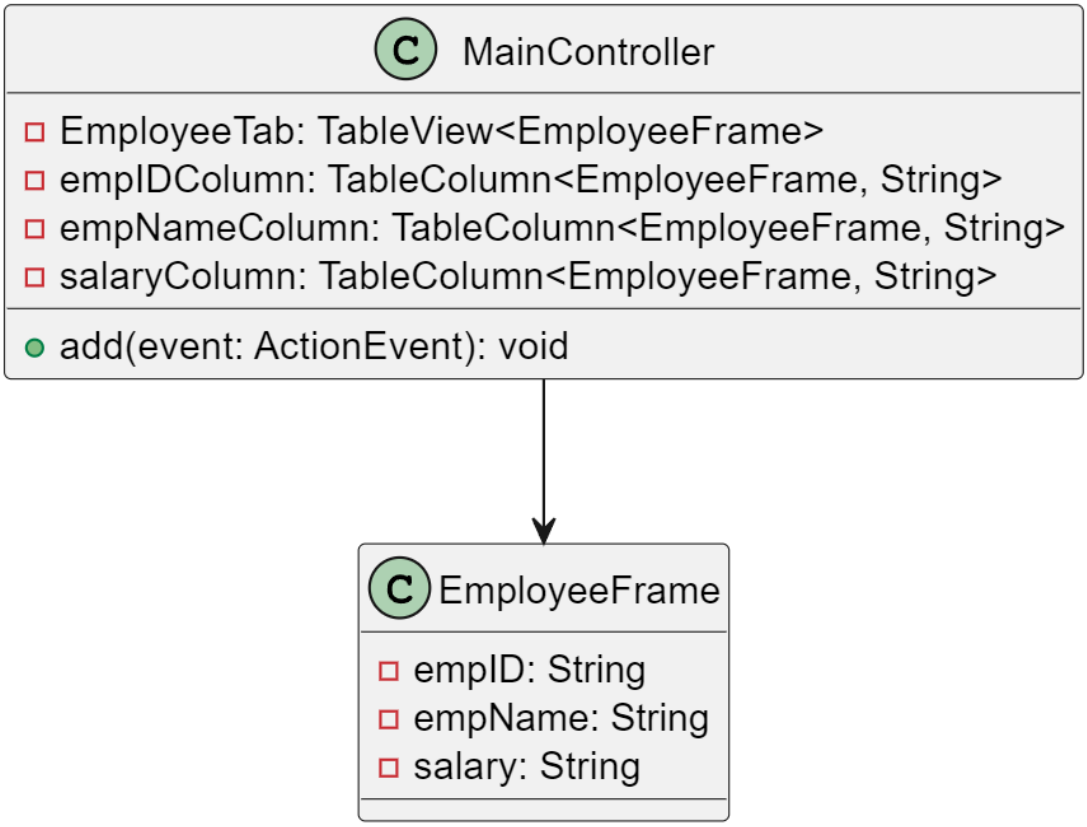


Figure 4.5 MainController UML

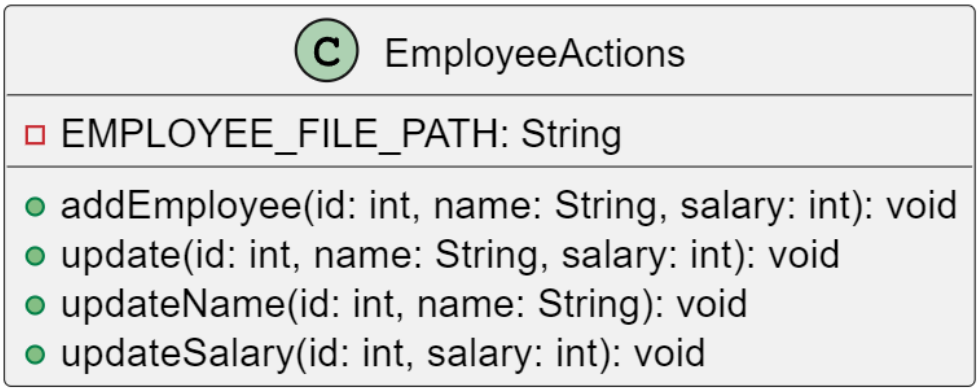


Figure 4.6 Employee Actions UML