Group 13 - Munch Wagon

Team Members: Erasto Astudillo, Felix Marecaux, Aidan Shotland, Carlos Anguiano
GitHub: https://github.com/canguiano13/cs250-designSpecificationAssignment

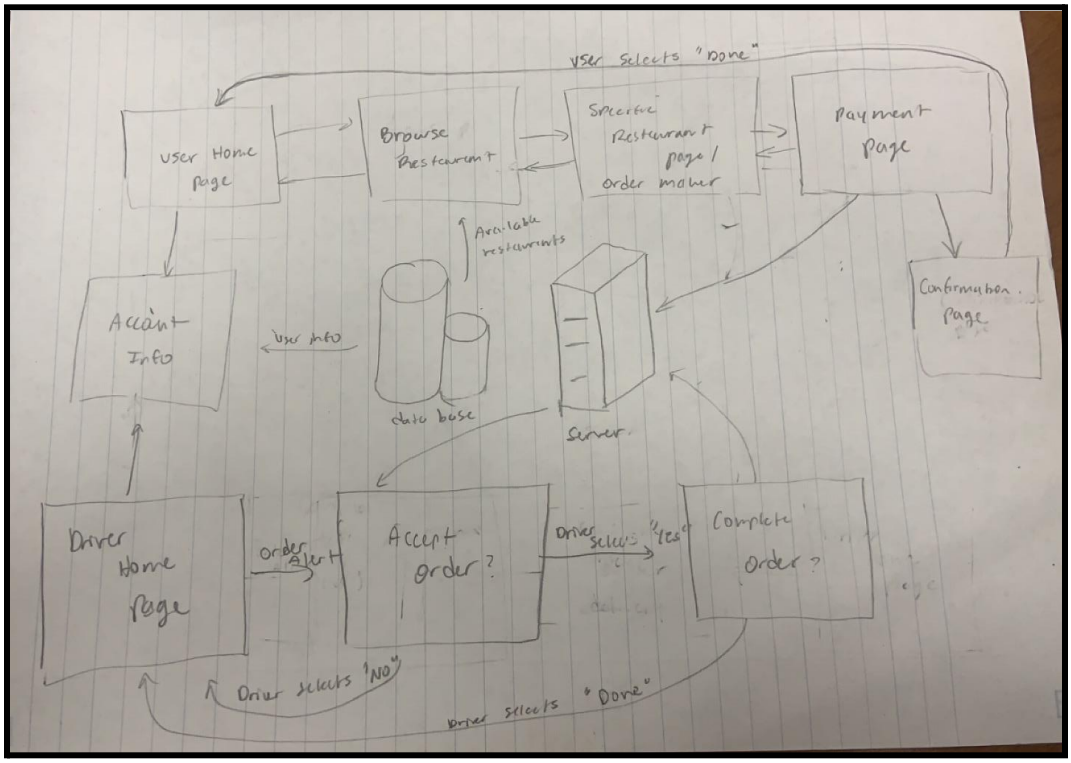
System Description:

-Brief Overview of System:

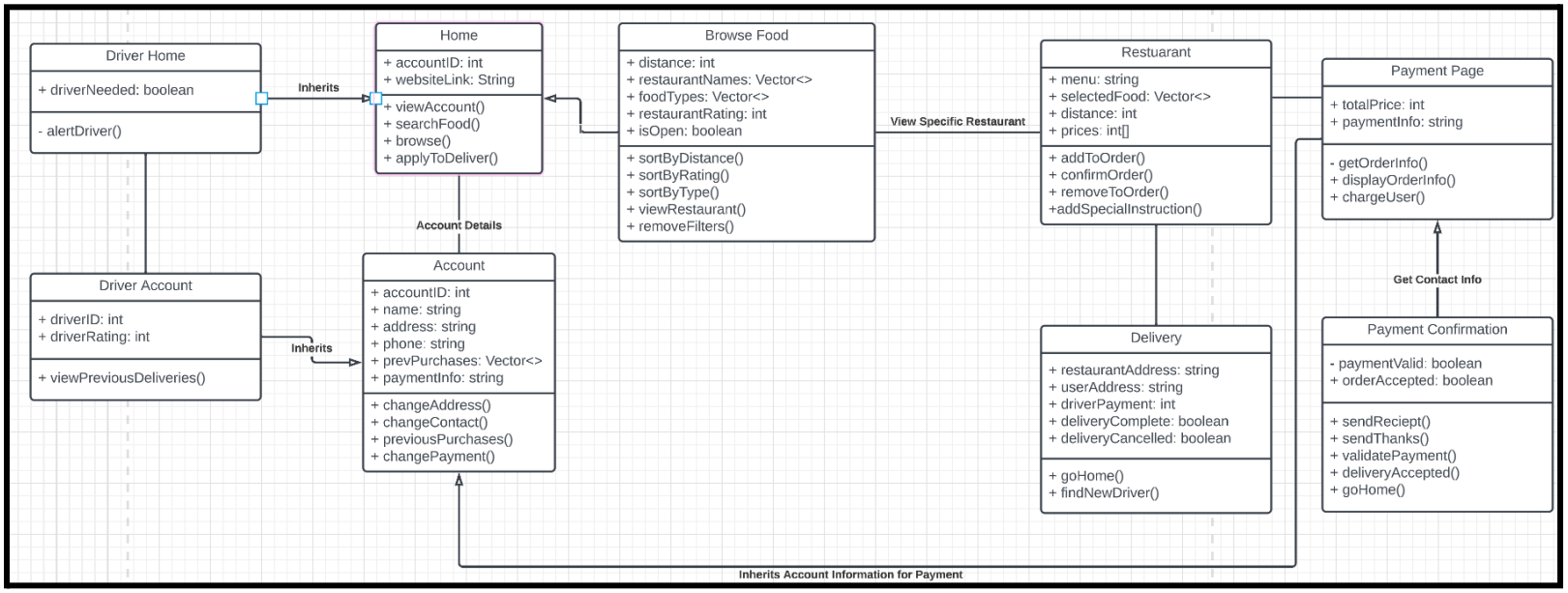
- The basic overview of the system is that of a delivery service, it begins with the drivers, the people who will deliver the food, drivers have a specialized driver account that gives them access to view past delivery details, and their driver rating. It also inherits the basic account information that a regular user would have to place orders. This is useful as the driver would not need to create another account to place orders allowing the drivers to order food as well. The driver would of course be able to view payment information regarding the order and select whether to accept or decline the order which gets sent back to our servers. For the user end, they begin at the Home Page, they can navigate to the account information that contains their address and other information which is all stored in our database. They can also navigate to browse the various restaurants. When a user does select a restaurant and choose their food items, they head to the payment page where they can input their payment information. After they submit their payment information, it gets sent to the bank's servers and gets validated or rejected, and sent back to us either to prompt for an order confirmation or to send the user back to the payment information page. If the user manages to get an order confirmation page, that as well prompts for the order to be sent to our server where we show when a driver is needed to deliver an order on the Driver Home Page. Overall the software allows access for the customer's need to place their orders at their desired restaurant that is listed, and receive details regarding their order. This then allows for the drivers to see when an order needs attention and helps to fulfill the customer's order, allowing the program to serve its purpose in a streamline manner.

Software Architecture Overview:

-Architectural Diagram of all Major Components:



-UML Class Diagram



-Description of Classes

- Home
  - The first page that the user sees when the user opens the software. From this class, the user can view their account info or can browse available restaurants that they may order from. The user can also use the search bar to search for specific restaurants.
- AccountInfo
  - The user may navigate to the AccountInfo page where they can view and edit any addresses, contact information, or payment information associated with their account. They can also view any previous purchases made under their accountID. Their previous purchases are imported from an internal server to a vector, and displayed on the page
- DriverAccountInfo
  - Inherited from the AccountInfo class, This class is used by any approved drivers to manage their information about their account. They also gain the ability to view their previous deliveries, and add bank information
- Browse
  - This class allows users to browse restaurants that are open and available for delivery. From this class, the user can sort the available restaurants by rating, type of food, or distance. The user can also use viewRestaurant() to access the Restaurant class which would allow them to place and manage orders

- Restaurant
  - This class sees users place and manage orders. Here, the user has already selected the specific restaurant that they wish to order from. The menu is displayed and they can add/remove items from their order. Once they have selected all of the items they wish to purchase, they can use confirmOrder() to navigate to the payment class. They may also add special instructions for their
- Payment
  - The payment class is a subclass of the userAccount class which allows users to pay for items in their order. Their order information is displayed for further verification that their order contains all of the correct items. This class inherits banking/payment information used to pay for the order
- Payment Confirmation Class
  - A subclass of the payment class, this class is used by the software to verify payments before sending an order alert to the primary server. Once the payment is verified, the user also receives a receipt and a thank you note.
- DriverHome
  - A homepage for verified delivery drivers. This class inherits the Home superclass. In addition to browsing restaurants and seeing account info, here drivers can get order alerts for outstanding deliveries that are requesting a driver.
- Delivery
  - The delivery class is used by drivers to execute user orders. From here, the driver will be able to confirm that they want to pick up a user's order and can complete an order to receive payment once they have delivered the user's order. Once they have completed an order, they can return to the home page.

-Description of Attributes:

- Home Class
  - accountID
    - Anyone using the software will have a distinct integer ID that is unique to them. This will be used for internal processes inside the software.
  - websiteLink
    - String of what the link is for users to apply as drivers of the application.
- BrowseFood Class
  - distance
    - Integer variable used to depict how far the restaurants are from the user's inputted address
  - restaurantNames
    - String list of all the different restaurant options that will be present on the browse page

- foodTypes
  - String list of the different categories of food options that will be available in the software.
- restaurantRating
  - An integer variable available for each restaurant that will be used as a rating.
- isOpen
  - A boolean variable that will be used to determine what restaurants are open based on the time at which the software is being used.

- Restaurant Class
  - menu
    - A string that will be used as a link to a list provided by the softwares internal servers.
  - selectedFood
    - A vector list that will be used to store all the selected food items as the user selects what they want.
  - distance
    - An integer value that will be used to distinguish how far the restaurant is from the user's inputted address.
  - prices
    - An integer array that will store all the prices of the menu items in the order at which they appear.
- PaymentPage Class
  - totalPrice
    - An integer value that will summarize the total cost of all the selected food items.
  - paymentInfo
    - A string that will include the card information, such as the 16 digits, expiration date, and security code.
- DriverAccount Class
  - driverID
    - An integer value that will be used to distinguish the different drivers for internal processes made by the software.
  - driverRating
    - An integer value that serves to provide trustworthy and good drivers by giving them a rating based on performance.
- Account Class
  - accountID
    - Anyone using the software will have a distinct integer ID that is unique to them. This will be used for internal processes inside the software.

- name
  - A string value that will be used to hold the users name
- address
  - A string that will hold the user's address
- phone
  - A string used to hold the user's phone number
- prevPurchases
  - A vector list used to hold the user's previous made orders/purchases
- paymentInfo
  - A string used to hold the user's payment information for check out.

❖ DriverHome Class
- driverNeeded
  - A boolean that will be used to represent when a driver is needed to perform a delivery

❖ Delivery Class
- restaurantAddress
  - A string which holds the address to which the delivery driver will need to go in order to pick up the food.
- userAddress
  - A string that holds the address to which the delivery driver will drop off the food after pick up
- driverPayment
  - How much the driver will make from the delivery, held in an integer variable.
- deliveryComplete
  - A boolean to signify that a delivery has been completed
- deliveryCancelled
  - A boolean used to signify that a delivery has been canceled by the driver.

❖ PaymentConfirmation Class
- paymentValid
  - A boolean to show whether the payment is accepted or not
- orderAccepted
  - A boolean that shows when a driver has accepted the order

-Description of Operations:

❖ Home Class
- viewAccount()
  - Sends user to account information page

- searchFood()
  - Allows user to search for restaurants in their surrounding area; returns a list of restaurant names
- browse()
  - Allows user to view browse page
- applyToDeliver()
  - Returns website link for user to apply to be a delivery driver

❖ Account Class
- changeAddress()
  - Allows user to change the address orders are delivered to and surrounding restaurants are selected from
- changeContact()
  - Allows user to modify the phone number and email that is tied to their account
- previousPurchases()
  - Allows user to view a list of their previous food delivery purchases
- changePayment()
  - Allows user to modify the payment information tied to their account

❖ BrowseFood Class
- sortByDistance()
  - Allows users to sort surrounding restaurants by the distance the restaurant is from the address tied to their account from least to most (max 15 miles) distant.
- sortByRating()
  - Allows user to sort surrounding restaurants by the average rating (/5) of the restaurant
- sortByType()
  - Allows user to filter surrounding restaurants to only display restaurants that sell food of a given type
- removeFilters()
  - Allows user to remove any filters that affect surrounding restaurants being displayed
- viewRestaurant()
  - Allows user to navigate to a specific restaurant's page to view their menu and order food, among other things.

❖ Restaurant Class
- addToOrder()
  - Allows user to add a food item from the restaurant's menu to their order
- removeFromOrder()
  - Allows user to remove an item in the list of their current order.

- addSpecialInstruction()
  - Allows user to add a message to send to the restaurant via external messaging system
- confirmOrder()
  - Allows user to finalize their order; sends user to payment page

- PaymentPage Class
  - displayOrderInfo()
    - Allows user to view items currently in their order; can be used as verification before paying for order
  - chargeUser()
    - Once the user has confirmed their order, the software will acquire the correct amount of money from the user and alert available drivers nearby that there is an outstanding delivery request
  - getOrderInfo()
    - Used internally by the system once the user has confirmed their payment information to send the items in the user's order to the restaurant via external messaging system
- PaymentConfirmation Class
  - sendReceipt()
    - Sends user receipt of purchase via phone and/or email
  - sendThanks()
    - Displays a thank you image for the user
  - validatePayment()
    - Validates user's payment transfer is successful
  - deliveryAccepted()
    - Notifies user via phone and/or email that a delivery driver has accepted their outstanding order request
  - goHome()
    - Allows user to navigate to home class
- Delivery
  - goHome()
    - Allows delivery drivers to view the Driver Home Page
  - findNewDriver()
    - Allows driver to restart alert and cancel their current delivery.
- Driver Home Class
  - alertDriver()
    - Alerts the driver that there is an outstanding delivery request that they may accept or decline. Once a driver accepts, the alert is removed from other drivers' home page.
- Driver Account Class

- ➢ viewPreviousDeliveries()
  - ■ Displays a list of a driver's previously completed deliveries.

Development Plan and Timeline:

-Partitioning of Tasks

Erasto Astudillo - Requirements Gathering - **deadline 3/1/23**

- Establish needs of the customer

Felix Marecaux - Design

- Determine structure of software **-deadline 3/1/23**
- Create useable UML diagram for software

Carlos Anguiano - Implementation **- deadline 4/1/24**

- Build software
- Code and test

Aidan Shotland - Verification / Maintenance **- deadline 4/20/24**

- Test if system works
- Are stakeholders happy?
- Fix bugs

-Team Member Responsibilities

Regardless of individual responsibility, entire team will work and help each other

Erasto Astudillo will find out what the stakeholders and customers need. He will find out how well the system serves its purpose. He will ask the stakeholders specific questions regarding budget, manpower, deadlines, functionalities, etc. To find out what the customer needs he will research similar platforms with similar functionalities that have had success in the past. In addition, he will collect data by conducting surveys on all social media platforms along with asking relatives and peers their opinions. The collection of data will be completed by **March 1st 2023.**

Felix Marecuix is in charge of design. He needs to specify the structure of the software along with designing the architecture. He will make a shared document with his team members to help brainstorm ideas on the design. This document will include a few elements, a brief overview, a UML and architecture diagram, description of classes and attributes and operations, and finally this timeline of responsibilities. The overview is essentially a summary or introduction to the functionalities of the software. The architecture and UML diagram will provide team members with a visual representation of the design. The descriptions will breakdown every step of the implementation of code. This responsibility will have a deadline of **March 1st, 2023**.

Carlos Anguiano will be in charge of the actual implementation of the code. He is in charge of the team at this stage but everyone will be responsible for contributing something to implementation. Each member of the group will be assigned a different job by Carlos and he will assist everyone in need. Depending on individual strengths, the jobs will be divided into: classes, attributes, and operations. This division of jobs will allow the job to be done more efficiently and effectively with constant communication among team members. The actual implementation of the code's timeline is TBD. The hard deadline for the stakeholders is **April 1st 2024.**

Aidan Shotland will be responsible for the failures. He will be constantly testing and debugging the program to ensure that it is exactly what the stakeholders and customers require. The main goals of testing will be to detect, locate, fix, failures, faults and errors. Any problems such as bad customer reviews, unhappy stakeholders, and angry management will all go through him. He will be assigned this position for the foreseeable future however, the service will need to be available for consumers by **April 20th, 2024.**