Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

TEAM A: Villaincon Valley

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

AGILE STORIES

Story 1

- **As a store manager, I'd like to be able to process information for a customer's sales stored in separate files.**
    - **Description:** Other sales information should include purchase date, customer membership number, items purchased, and sales price quantity purchased.
    - **Story Point Est:** 3
    - **Assumptions:**
        - The documents are formatted to support inclusion
        - The text is formatted to pull specific data in order
        - The functionality should handle edge cases.
    - **Priority:** Sprint 2
    - **Assignee:** Johnny
    - **Tasks:**
        - Load from sales file into a database
        - Check if file is readable
        - Make sure that you add onto the database, as opposed to overwriting it
    - **Tests:**
        - Load from 3 different files
        - Check your database to see if all of the information is present
    - **Definition of Done:**
        - The user can load from the sales file

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

Story 2

- **As an Administrator, I'd like the ability to add and delete members**
    - **Description:**
        - When a member is deleted, their purchases should remain
        - When a member is added, there should be the option to add a new sale
    - **Story Point Est:** 5
    - **Assumptions:**
        - There is a dynamic data structure containing all Bulk Club members
        - There is a dynamic data structure containing all Bulk Club sales
        - There is a function to add new sales
    - **Priority:** Sprint 3
    - **Assignee:** Johnny
    - **Tasks:**
        - Create a function to delete a member
        - Create a function to add a member
        - Add gui support for the above functions
    - **Tests:**
        - Add a member and verify the updated data in the program is accurate
        - Add a member again, but add a sale with the member when prompted, verify the data in the program
        - Delete a member and verify the updated data in the program is accurate
    - **Definition of Done:**
        - When logged in as an admin, the user can add and delete members
        - The code fits the designated coding standards

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

Story 3

- **As a user, I would like to load in and access customer profiles to then determine display data about the customers.**
    - **Description:** The customer profile should contain the customer name, membership number, type of customer, and membership expiration date.
    - **Story Point Est:** 3
    - **Assumptions:**
        - The documents are formatted to support inclusion
        - The text is formatted to pull specific data in order
        - The functionality should handle edge cases
    - **Priority:** Sprint 2
    - **Assignee:** Johnny
    - **Tasks:**
        - Load from customer information file into a database
        - Check if file is readable
        - Make sure that you add onto the database, as opposed to overwriting it
    - **Tests:**
        - Load from 3 different files
        - Check your database to see if all of the information is present
    - **Definition of Done:**
        - The user can load from the customer information file

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

Story 4

- **As a user, I want my choices to be saved so that I can reduce time of searching and getting information on sales and customers**
    - **Description:** Changes must be saved between executions
    - **Story Point Est:** 2
    - **Assumptions:**
        - Features are already completed
        - Actions can be made in the program
        - The changes are not saved when the program is terminated
    - **Priority:** Sprint 3
    - **Assignee:** Not Assigned
    - **Tasks:**
        - Ensure that our data is in scope through the duration of the app
        - Make sure all of the files add onto our databases, instead of overwriting them
        - Ensure the program does not crash or terminate from an execution
    - **Tests:**
        - Load from multiple different files and ensure that the data is being added
        - Delete customers and sales in the middle of the database to ensure that it is adjusting accordingly
        - Terminate the program and restart it to ensure that it can obtain a fresh start
    - **Definition of Done:**
        - The program’s data has permanence for every action taken

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

Story 5

- **As a store manager, I would like to display the total purchases for any single member to ensure their purchases are correct.**
    - **Description:** Each purchase should include tax and they are searchable through either their membership number or name.
    - **Story Point Est:** 3
    - **Assumptions:**
        - There is data loaded from a file and stored into a database
        - There is a display to show all of purchases of a single customer
    - **Priority:** Sprint 2
    - **Assignee:** Sean
    - **Tasks:**
        - Display a series of purchases from one customer through UI
        - If the customer does not exist, display an error message
        - Give the user the option to use either a membership number or name, but not both
    - **Tests:**
        - .Find the purchases of a customer that exists
        - Try to find the purchases of a customer that doesn't exist
    - **Definition of Done:**
        - The user can navigate around the UI to display the total purchases for a single member

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

Story 6

- **As an administrator, I'd like the ability to add and delete items in an inventory**
    - **Description:** have a separate tab to display the inventory, and have the ability to add, delete, and change items from the inventory
    - **Story Point Est:** 5
    - **Assumptions:**
        - There is a dynamic container containing the inventory(items bought) with their prices
    - **Priority:**
    - **Assignee:** Not Assigned
    - **Tasks:**
        - Create function to add a new item to the inventory
        - Create a function to delete an item from the inventory
        - Create a function to change information(i.e. price) about an item in the inventory
        - Add gui support for the above functions
    - **Tests:**
        - Add a new item and view the updated table
        - Delete an item and view the updated table
        - Make changes to an item and view the updated table
    - **Definition of Done:**
        - All features verifiably function based on the tests
        - The code fits the predesignated coding standards

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

Story 7

- **As a store manager, I would like to display how much an item has been sold so that I can update my inventory accordingly**
    - **Description:** The item must be searched by name and contains the total revenue without tax.
    - **Story Point Est:** 5
    - **Assumptions:**
        - There is an inventory that exists (either generated or read from a file)
        - There is UI that supports searching by name for an item
    - **Priority:** Sprint
    - **Assignee:** Max
    - **Tasks:**
        - Display in UI the quantity of an item sold and the total revenue
        - If the item cannot be found, display an error message
    - **Tests:**
        - Display an item that exists in the inventory
        - Try to find an item that doesn't exist in the inventory
    - **Definition of Done:**
        - An item can be found from the inventory, and if it isn't an error is displayed

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

Story 8

- **As a store manager, I'd like the ability to display the sales report by date**
    - **Description:**
        - Include items and quantities sold on that day
        - Includes the names of people that shopped on that day
        - Displays the total revenue including tax
        - Include number of unique executive members and regular members who shopped that day
    - **Story Point Est:** 8
    - **Assumptions:**
        - There is a populated container with all of the member information
        - There is a populated container with all of the sales information
    - **Priority:** Sprint 2
    - **Assignee:** Joshara
    - **Tasks:**
        - Create a function to clone sales data into a container separated by date
        - Create a function to calculate the total revenue before and after tax
        - Create a function that populates a container with unique members that shopped that day
        - Add gui support to display all of the above information
        - Only allow the user to display sales reports by dates that represent data files that have already been loaded into the program
    - **Tests:**
        - Make sure you can only display sales reports from dates that have sales reports loaded in
        - Sort by date and verify the displayed data
        - Unsort and resort to make sure nothing breaks
    - **Definition of Done:**
        - Data displays correctly based on the above tests
        - Code fits the predesignated coding standards

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

Story 9

- **As a store manager, I'd like to be able to retrieve the sales reports sorted by both date and membership type.**
    - **Description:** Display the sales reports sorted by date and allow the user the option to show either regular/executive members or to show reports from both types of member.
    - **Story Point Est:** 2
    - **Assumptions:**
        - Interface for reading/displaying data is in place
    - **Priority:** Sprint 2
    - **Assignee:** Not assigned
    - **Tasks:**
        - Create option to sort by date
        - Create option (combo box) to select which type of members are displayed
    - **Tests:**
        - Ensure correct data is being displayed
            - Check that only regular members are displayed when the appropriate option is selected
            - Check that only executive members are displayed when the appropriate option is selected
            - Check that both are displayed when option for both is selected
    - **Definition of Done:**
        - Doxygen style comments have been written
        - Data is displayed correctly and without omissions
        - Product owner signs off on code

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

Story 10

- **As a store manager, I'd like to be able to see all of the purchases made by each customer.**
    - **Description:** Display each purchase made sorted by customer membership number. Will also display a grand total at the bottom.
    - **Story Point Est:** 3
    - **Assumptions:**
        - Interface for reading purchase information is in place
    - **Priority:** Sprint 3
    - **Assignee:** Sean
    - **Tasks:**
        - Write functionality to sort by membership number
        - Write functionality to calculate grand total
    - **Tests:**
        - Ensure that elements are being sorted correctly by membership number
        - Ensure that the calculated total is correct
    - **Definition of Done:**
        - Doxygen comments have been written
        - Product owner has signed off on code
        - Data is displayed correctly and without omissions

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

Story 11

- **Display all members who expire in a given month**
    - **Description:** The user should be able to select a month and view all the users which expire in that month, along with the cost for each user to renew
    - **Story Point Est:** 2
    - **Assumptions:**
        - There should be a “Member” class with an expiration date data member
        - There should be a data member representing renewal cost
    - **Priority:** Sprint 2
    - **Assignee:** Johnny
    - **Tasks:**
        - Create function to filter Members by expiration month
        - Create a function to sort filtered Members by expiration day
        - Create UI functionality for the above functions
    - **Tests:**
        - Select every month available in various orders
        - Swap between this view and various other views(viewing full sales, viewing monthly sales, etc.)
    - **Definition of Done:**
        - The feature correctly displays the data and passes the above tests
        - The code conforms the predesignated coding standards

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

Story 12

- **As an administrator, I would like an application that requires my login information in order to prevent anyone else from accessing certain features such as adding or deleting members of the Bulk Club.**
    - **Description:** The act of an administrator logging in should enable additional features from the general application view and the store manager view.
    - **Story Point Est:** 1
    - **Assumptions:**
        - The user has already logged into the store manager view of the application.
        - All sale data and customer data is accessible to the user and available to edit.
    - **Priority:** Sprint 2
    - **Assignee:** Sean
    - **Tasks:**
        - Create a login window with an encrypted password that allows the user to open a unique view of the application.
    - **Tests:**
        - The user cannot access administrator functionality without first logging in to the administrator view.
        - If the username or password entered by the user is incorrect, the user cannot access administrator features of the application.
        - Upon successful login to the administrator view, the user gains access to previously hidden features of the application.
    - **Definition of Done:**
        - The user can choose to log in as an administrator to the application.

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

Story 13

- **As an administrator of the Bulk Club, I would like to determine if any Regular members of the Bulk Club should be promoted to Executive membership or if any Executive members should be demoted to Regular membership in order to determine the efficiency of the membership system.**
    - **Description:** The application should calculate the rebate of every Regular customer, and display members that met or exceeded the cost of promoting their membership to Executive. The application should also calculate the rebate of every Executive member, and display members that did not meet the cost of promoting their membership to Executive.
    - **Story Point Est:** 2
    - **Assumptions:**
        - Every customer has an associated list of sales records
        - The annual dues for a Regular membership to the Bulk Club is known
        - The annual dues for an Executive membership to the Bulk Club is known
        - The rebate percentage granted to Executive members is known
    - **Priority:** Sprint 3
    - **Assignee:** Not assigned
    - **Tasks:**
        - Sum the rebate of every customer
        - Calculate and compare the annual rebate of every customer to the difference between annual Regular and Executive membership dues
    - **Tests:**
        - An executive member with less than $55 in rebate shows in the Demote list
        - A regular member with more than $55 in their calculated rebate shows in the Promote list
    - **Definition of Done:**
        - A button, when pressed, activates a filter through the list of Regular customers, displaying those that should’ve promoted their membership to Executive

- A button, when pressed, activates a filter through the list of Executive customers, displaying those that should've demoted their membership to Regular

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

Story 14

- **As a store manager of the Bulk Club, I want to be able to view the rebate of every Executive member.**
    - **Description:** The user should be able to view the rebate amount of every member of the Bulk Club, sorted by membership number or based on total purchases before tax.
    - **Story Point Est:** 2
    - **Assumptions:**
        - Every member is distinguished by Executive or Regular membership type.
        - Every member has an associated membership number and associated purchases in a given year.
    - **Priority:** Sprint 3
    - **Assignee:** Not assigned
    - **Tasks:**
        - Display every Executive member in order of increasing or decreasing amount spent at the Bulk Club before tax
        - Display every Executive member in order of increasing or decreasing rebate amount earned
    - **Tests:**
        - No Regular members are displayed while this function is being used.
    - **Definition of Done:**
        - A button, when pressed, filters out the Regular members from a default view and sorts based on either rebate amount earned or total purchase amount in a given time period.

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

Story 15

- **As a store manager I'd like to be able to view how much of each item has been sold.**
    - **Description:** Display the quantity of each item sold sorted by item name that includes a grand total of all items sold (including tax) at the bottom.
    - **Story Point Est:** 5
    - **Assumptions:**
        - Interface for retrieving/viewing information is in place
    - **Priority:**
    - **Assignee:** Max
    - **Tasks:**
        - Create inventory UI
        - Retrieve information for sales
        - Sort sales by membership number
        - Calculate total (including tax)
    - **Tests:**
        - Ensure that all of the information is being retrieved
        - Ensure that the sales are being appropriately sorted
        - Ensure that the total has been calculated correctly
    - **Definition of Done:**
        - Doxygen comments have been written
        - Product owner has signed off on code
        - Data is correctly read into and manipulated by code

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

TEAM RULES

- Everyone does equal work.
- Let someone know if you are behind on work.
- Team Check-in @ 7:30 pm (M & W) on Discord.
- Double checking of work (instead of paired programming) at any time through Discord.
- No fisticuffs/fighting.

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

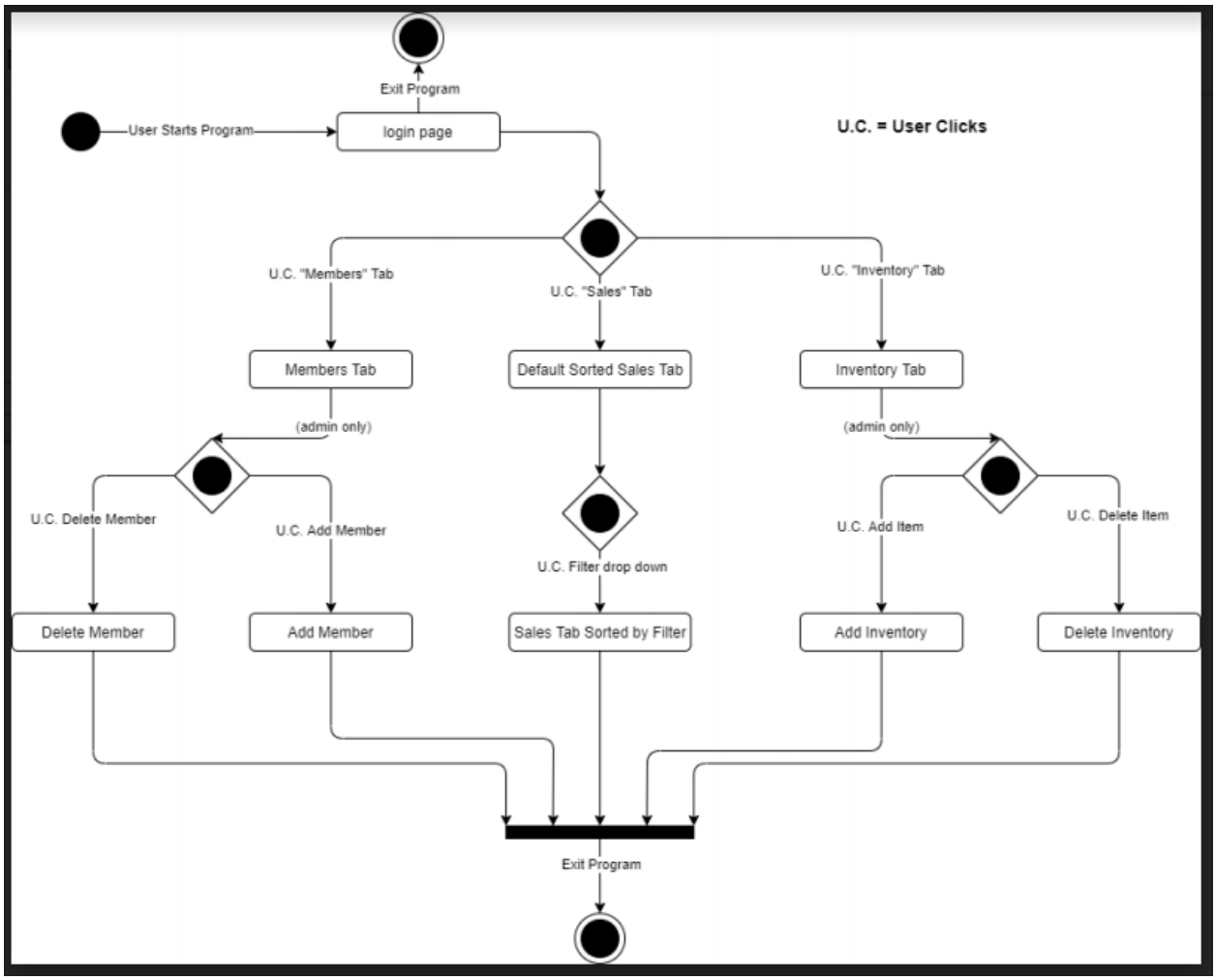
CODING STANDARD

- Comment your code clearly and concisely using DOXYGEN syntax. (ex: ////)
- Functions should be as simple and/or straightforward as possible.
- Document each file (at the top) what this file does and what functions it supports.
- camelCase your variables.
- Make sure your variable/function names are descriptive and self-commenting.
- Line up curly braces on the same line.
- Limit the number of global variables.
- Put function definitions and function signatures in different files.
    - Function signatures in a header file
    - Function definitions in a cpp file
- Use all caps for creating global variables.
- Work within your own personal branch, which is mirrored from the testing branch
    - Main branch → Testing branch → Individual branches (per person).
- MUST double check work before committing code to main.

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

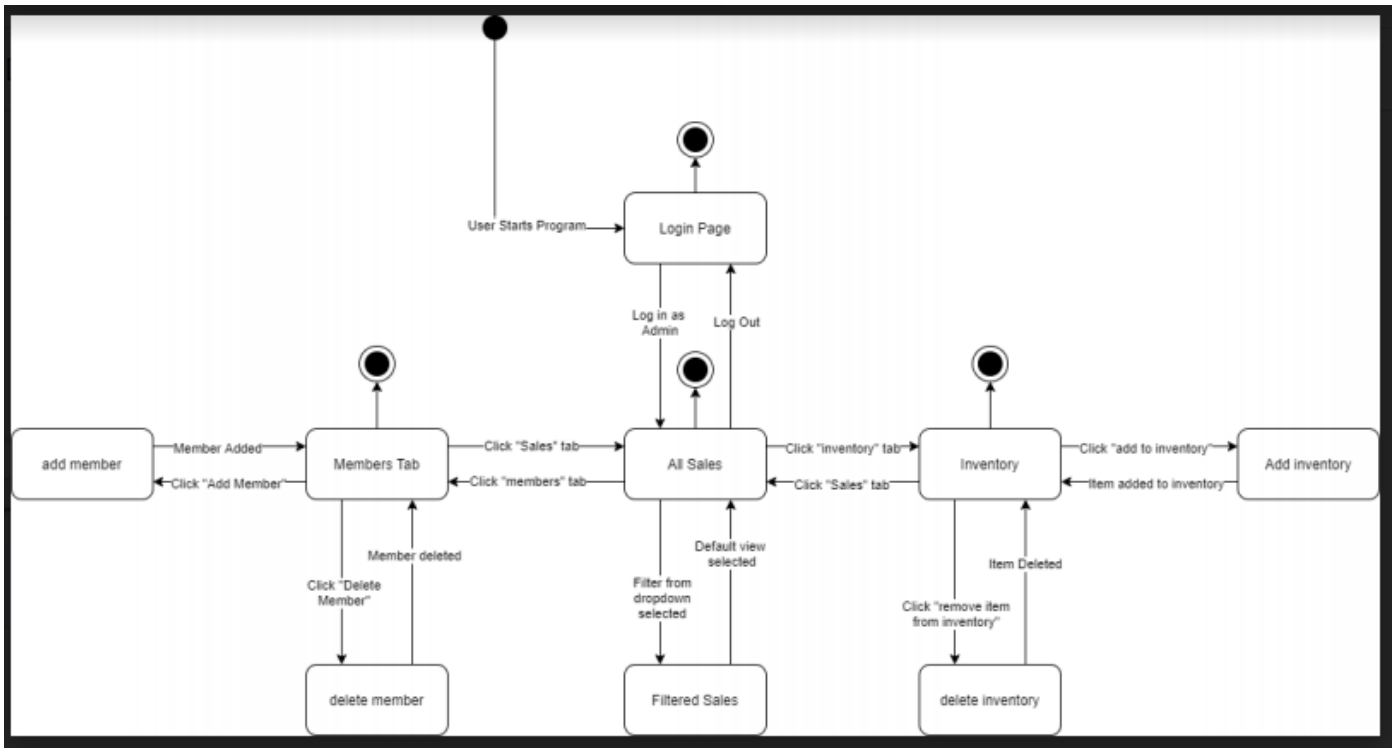
UML DIAGRAMS

Activity Diagram

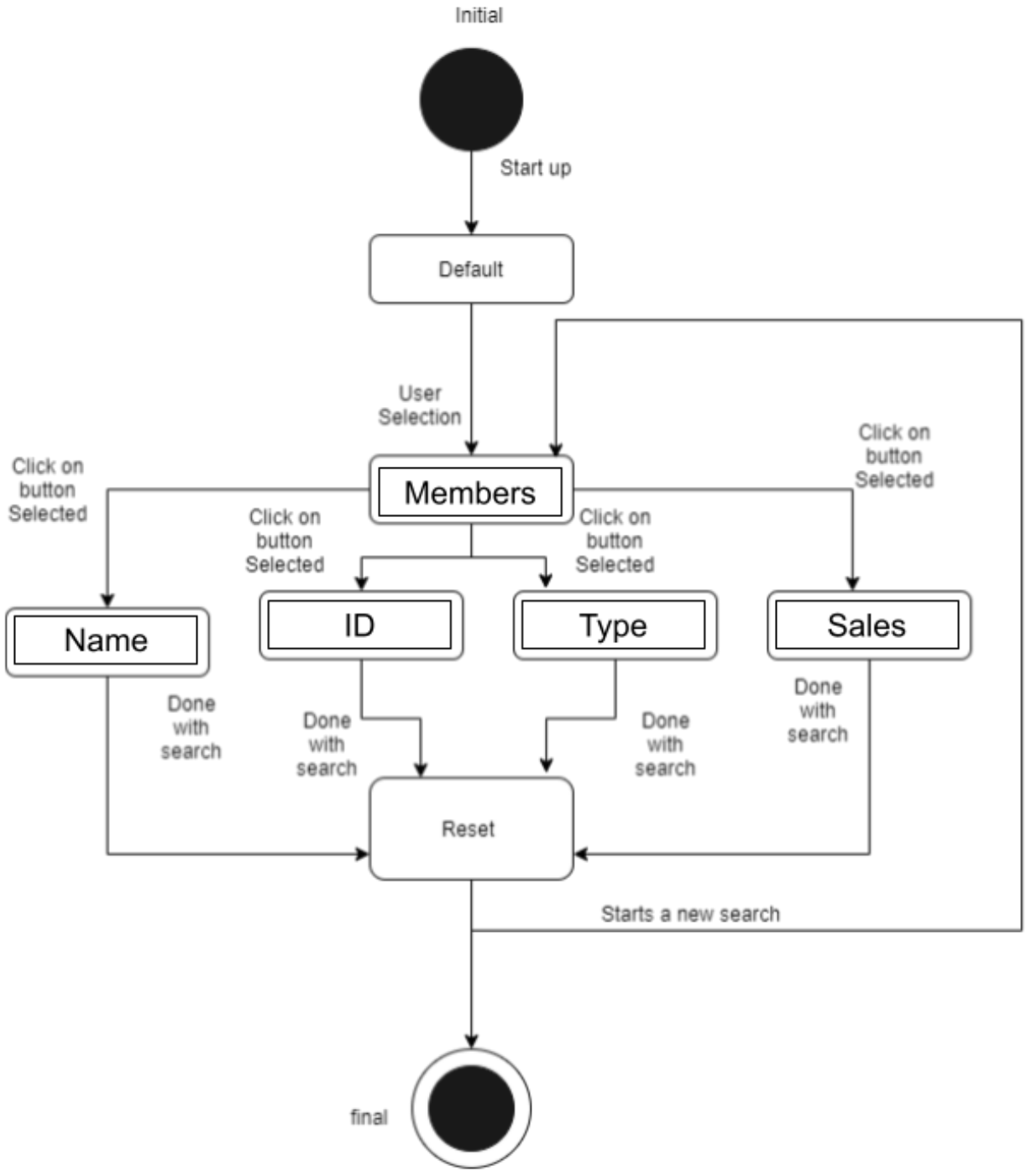


Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

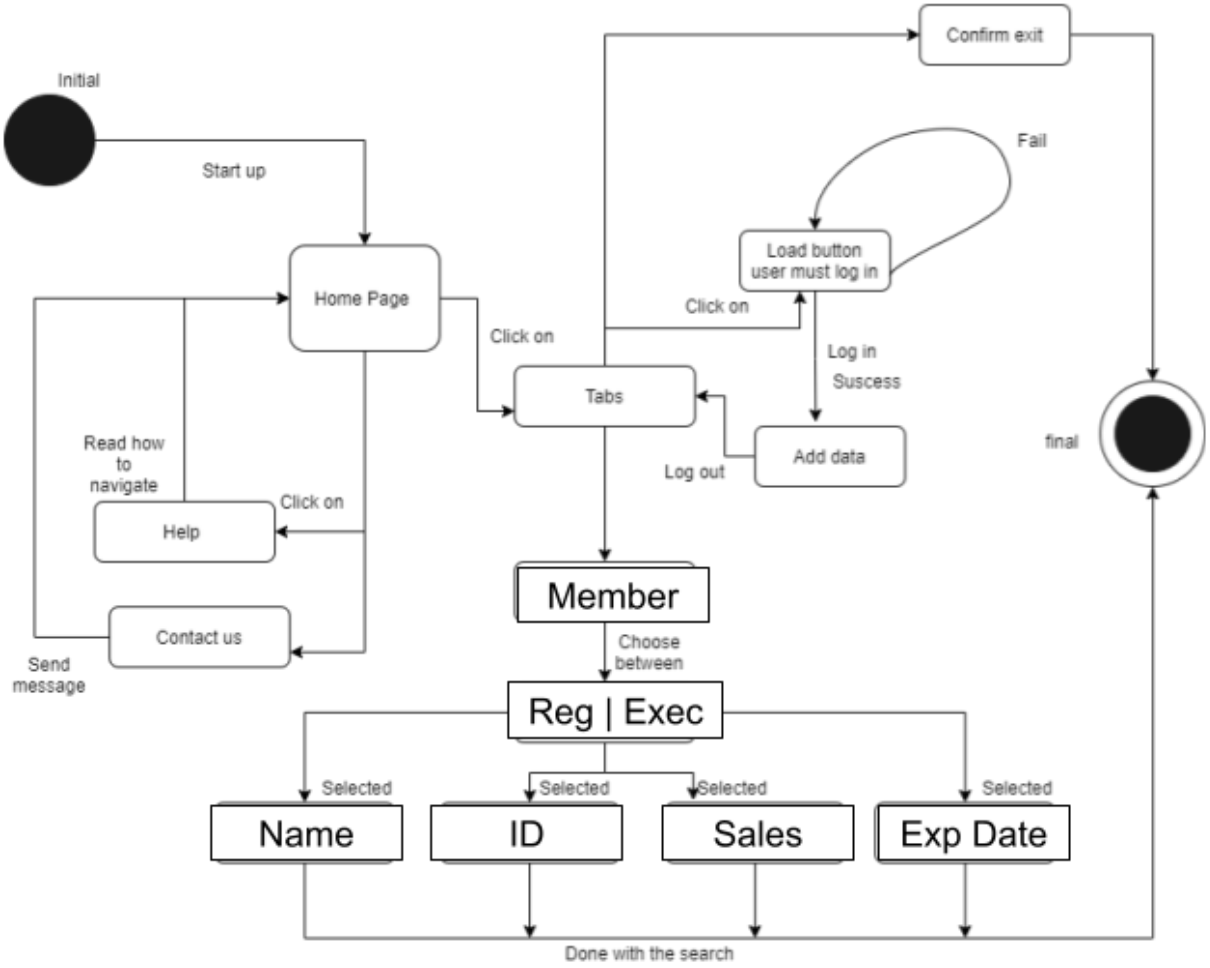
State Diagrams



Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker



Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker



Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

USE CASES

Use Case Name: Administrator Adds a Member

Actors

- Valid Administrator: Has a pre-existing username and password
- Invalid Administrator: Does not have a pre-existing username and password
- Bulk Club Application: A way for the user to interact with the sales and membership database

Trigger

- The user indicates that they want to login into the administrator view of the application by choosing this option in the menu bar, clicking the login option

Preconditions

- The user already has access to a valid password
- The user already has access to the complete data for a member they wish to add to the list of members displayed in the application

Post-Conditions

- The new member and all associated information with that member, including membership ID number, data has been stored in the database appropriately

Flow

1. The user indicates they want to enter a new member into the sales database by toggling the “Admin” button found in the application tab
2. The application presents a login page, prompting the user to enter a valid password
3. The user will enter their password
4. The interactive pamphlet will check if the entered data matches the preexisting data, which will trigger the interactive pamphlet to either
    a. Allow the (now verified) administrator access to the Administrative functionality and display a message box of the successful login or
    b. Deny the user access to the Administrative functionality and display a message box of the unsuccessful login, returning the user to step 1
5. The valid administrator will then have the added capability from the normal view of the application to add or remove sales or members
    a. Enter valid information for either a sale or a member, after which the pamphlet will indicate the operation was a success or
    b. Enter invalid information for either a sale or a member, after which the pamphlet will indicate the operation was unsuccessful and return the user to beginning of step 5
6. The administrator will exit the application, and the data will exist in the database according to how they altered it

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

Use Case: Store ManagerAccesses Sales Records

Actors

- Store Manager: A registered store manager seeking to access/display various elements of the sales record(s)
- Program Administrator: An admin looking to update the inventory/member list
- Bulk Club Application: an interface for the user to interact with the membership/sales databases

Trigger

- The user logs in as either a store manager or an administrator and accesses the tab(s) relating to sales reports

Precondition

- User provided valid login credentials

Post Condition

- All information is correctly displayed based on the user's choice(s)

Flow

1. The user has the choice to display the sales reports by
    a. Date
    b. Date & Membership Type
    c. Total purchases by member
    d. Quantity of each item sold
2. The user indicates that they want to view membership information such as
    a. rebate for all executive members
    b. all memberships that will expire in a given month
    c. Quantity of item sold from inventory
    d. Total purchases for a given member
3. The program displays the information that the user selected in an organized, orderly manner

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

Use Case: Administrator Determines if Member Should have Demoted or Promoted Membership

Actors

- Valid Administrator: An administrator for the Bulk Club, having successfully logged in to the administrative view of the application, and looking to determine the status of every membership
- Bulk Club Application: Provides a clean, orderly interface for the database containing the sales information and member information

Trigger

- The administrator indicates that they wish to determine which Regular members of the Bulk Club should promote their membership to Executive based on their hypothetical rebate amount, and conversely determine which Executive members of the Bulk Club should demote their membership based on their rebate amount.

Preconditions

- The database has a calculated field for the rebate amount corresponding to a percentage of the member's total purchases.

Post-conditions

- The user can view one of two lists
    - A list of Executive members who should demote their membership
    - A list of Regular members who should promote their membership

Flow

1. The administrator indicates that they wish to determine the Regular and Executive members' status of membership by clicking on the corresponding tab of the main view
2. The user will have the option to either
    a. Show a list of the Regular members of the Bulk Club that have a rebate amount greater than the amount it costs to upgrade their membership status
    b. Show a list of the Executive members of the Bulk Club that have a rebate amount lower than the amount is costs to upgrade their membership status
3. The administrator will have the option to return to the main view of the application by clicking the appropriate tab of the menu

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

SPRINT RETROSPECTIVES

Sprint 1

TO DO (✓/X)**:**

[✓] Activity diagram
[✓] State diagram
[✓] Use cases

[✓] - AGILE STORIES
[✓] - UML DIAGRAMS
[✓] - RETROSPECTIVE

RETROSPECTIVE:

Sprint 1 went smoothly, with solid guidance from our project manager Joshara. He formatted our goals nicely, and we evenly delegated work accordingly. We divided work on Agile stories evenly, and worked on the UML diagram collaboratively. By group consensus, there is seemingly nothing we can or would like to change, as this is already going many times better than our last project's first sprint.

Joshara Edwards
Sean Hostetter
Max Kwatcher
Johnny Wannamaker

Sprint 2

TO DO (✓/X)**:**

[✓] - Load from SQL
[✓] - Login (Manager/Admin)
[✓] - Sorting and Filtering

RETROSPECTIVE:

Sprint 2 was an interesting milestone as it started to solidify more of the project and our design choices. Not only did we have to make decisions about the UI, but also our code design and efficiency. A lot was discussed concerning efficiency, scalability, and navigating around data, but ultimately it was most practical to start coding based on some assumptions to start. As a team we have been able to pick up tasks and get them done exceptionally well.