Programming With Python

GROUP ASSIGNMENT

STUDENT NAME:

1. Lai Chun Hoong TP063514
2. Lee Zhixiang TP064098

LECTURER NAME: Armadeep

INTAKE CODE: UCDF2104ICT(SE)

Table of Contents

Introduction and assumptions

AEMS which also known as Asian Event Management Services is currently focusing on solving their main issue which is increase in demand for their services and came up with a solution which is to have a program which manages their business process online. With the help of program OEMS, also called the Online Event Management System, AEMS can solve their issue easily as OEMS is a system solely developed for users to manage events.

Before program OEMS is developed, it is assumed that the developers are not limited with the designs of the program, the way of coding and writing the pseudocode, and looks of flowchart. The main requirement is having all the functionalities mentioned to be fulfilled and ensures that the program runs smoothly as they are aiming to provide the users with the best experience when using their system. It is also assumed that the program will not be 100% bug-free however the developers will debug as much as they can to ensure the program can be run most of the time with the least error possible.

Design of the program

Pseudocode

#1 Function to start program

```
FUNCTION start()

    CALL FUNCTION clear()
    PRINT ('OEMS is loading...')
    OPEN cart.txt in WRITE mode with buffering set to 1 as cartfile
    CLOSE cartfile
    sleeps for 3 seconds
    PRINT ('Done!')
    sleep for 1 second
    CALL FUNCTION main_menu()
    RETURN

ENDFUNCTION
```

#2 Function to clear terminal screen

```
FUNCTION clear()

    Using os module:
        Clear terminal screen

ENDFUNCTION
```

#3 Function for main menu

```
FUNCTION main_menu()

    CALL FUNCTION clear()
    sleeps for 0.75 seconds

    TF = True

    IF session status == 'guest' THEN

        menu = "Welcome to OEMS, The Online Event Management System!
    What would you like to do?

    1. Log In
    2. Register An Account
```

```
3. Event Information
4. Exit

Choice: '"

    DOWHILE TF == True

        TRY
            PRINT (menu)
            Get answer
        CATCH
            PRINT('Invalid option, please try again.')
            sleeps for 0.75 seconds
            CALL FUNCTION clear()
            CONTINUE


        IF answer == 1 THEN
            CALL FUNCTION log_in()
            RETURN
        ELIF answer == 2 THEN
            CALL FUNCTION acc_register()
            RETURN
        ELIF answer == 3 THEN
            CALL FUNCTION list_event_menu()
            RETURN
        ELIF answer == 4 THEN
            CALL FUNCTION clear()
            CALL FUNCTION quit()
        ELSE
            PRINT('Invalid option, please try again.')
            sleeps for 0.75 seconds
            CALL FUNCTION clear()
            CONTINUE
        ENDIF

    ENDWHILE

ELIF session_status == 'admin' THEN

    PRINT(f'Welcome to the OEMS admin menu, {acc_name}')

    menu ='"What would you like to do?

1. Add New Event
2. Modify Event
```

```
3. Event Information
4. Customer Records
6. Exit

Choice: '"

   DOWHILE TF == True
      TRY
         PRINT(menu)
         Get answer
      CATCH
         PRINT('Invalid option, please try again.')
         sleeps for 0.75 seconds
         CALL FUNCTION clear()
         CONTINUE

      IF answer == 1 THEN
         CALL FUNCTION add_event()
         RETURN
      ELIF answer == 2 THEN
         CALL FUNCTION modify_event()
         RETURN
      ELIF answer == 3 THEN
         CALL FUNCTION list_event_menu()
         RETURN
      ELIF answer == 4 THEN
         CALL FUNCTION customer_record()
         RETURN
      ELIF answer == 5 THEN
         CALL FUNCTION log_out()
         RETURN
      ELIF answer == 6 THEN
         CALL FUNCTION clear()
         CALL FUNCTION quit()
      ELSE
         PRINT('Invalid option, please try again.')
         sleeps for 0.75 seconds
         CALL FUNCTION clear()
         CONTINUE
      ENDIF

   ENDWHILE

ELIF session status == 'registered' THEN

   PRINT (f'Welcome back {acc_name}!')
```

```
    menu ="'What would you like to do?

  1. View Event Information
  2. Add Events to cart
  3. View Cart
  4. Log Out
  5. Exit

  Attention: Please note that cart items will only remain for this session only. Cart items will be deleted on exit.

  Choice: "'

    DOWHILE TF == True
      TRY
        PRINT(menu)
        Get answer
      CATCH
        PRINT('Invalid option, please try again.')
        sleeps for 0.75 seconds
        CALL FUNCTION clear()
        CONTINUE

      IF answer == 1 THEN
        CALL FUNCTION list_event_menu()
        RETURN
      ELIF answer == 2 THEN
        CALL FUNCTION cart()
        RETURN
      ELIF answer == 3 THEN
        CALL FUNCTION view_cart()
        RETURN
      ELIF answer == 4 THEN
        CALL FUNCTION log_out()
        RETURN
      ELIF answer == 5 THEN
        CALL FUNCTION clear()
        CALL FUNCTION quit()
      ELSE
        PRINT('Invalid option, please try again.')
        sleeps for 0.75 seconds
        CALL FUNCTION clear()
        CONTINUE
      ENDIF

    ENDWHILE
```

```
   ENDIF

ENDFUNCTION
```

```
#4 Function to log in to program
FUNCTION log_in()
   CALL FUNCTION clear()
   sleeps for 0.75 seconds
   DECLARE session_status, acc_name as Global variable

   TF = True

   DOWHILE TF == True
      OPEN account_info.txt in READ MODE with buffering set to 1
      READ line of files as info_file

      PRINT('OEMS Login')

      PRINT('Please enter your username: ')
      Get acc_name

      IF acc_name == "" THEN
         PRINT('No username entered, please try again.')
         sleeps for 0.75 seconds
         CALL FUNCTION clear()
         CONTINUE
      ENDIF

      FOR line In info_file  STEP 1
         SPLIT line as acc_info with ',' as delimiter
         STRIP acc_info with index of 0 as acc_info_name
         STRIP acc_info with index of 1 as acc_info_password

         IF acc_name == acc_info_name THEN

            DOWHILE TF == True

               PRINT('Please enter your password')
               Get acc_password

               IF acc_password == "" THEN
                  PRINT('No password entered, please try again.')
                  sleeps for 0.75 seconds
                  CONTINUE
```

```
            ELIF acc_password == acc_info_password THEN
               CALL FUNCTION clear()
               sleeps for 0.75 seconds
               PRINT(f'Logged in successfully. Welcome back {acc_name}.')
               sleeps for 0.75 seconds

               DOWHILE TF == True

                  PRINT('Are you an admin? [y/n] ')
                  Get admin_confirmation

                  IF admin_confirmation == 'y' or admin_confirmation == 'Y' THEN
                     CALL FUNCTION clear()
                     sleeps for 0.75 seconds

                     DOWHILE TF == True
                        CALL FUNCTION clear()

                        TRY
                           PRINT('Please enter admin code (use this code for testing
purposes:000) \nCode: ')
                           Get admin_code
                        CATCH
                           PRINT('No admin code entered, please try again.')
                           sleeps for 0.75 seconds
                           CONTINUE

                        IF admin_code == 000 THEN
                           CALL FUNCTION clear()
                           sleeps for 0.75 seconds
                           PRINT('Thank you, redirecting you to the admin menu.....')
                           CLOSE info_file
                           session_status = 'admin'
                           TF = False
                           sleeps for 3 seconds
                           CALL FUNCTION main_menu()
                           RETURN

                        ELSE
                           CALL FUNCTION clear()
                           sleeps for 0.75 seconds
                           PRINT('Admin code does not exist, please try again.')
                           sleeps for 0.75 seconds
                           CONTINUE
```

```
                    ENDIF

                ENDWHILE

            ELIF admin_confirmation == 'n' or admin_confirmation == 'N' THEN

                CALL FUNCTION clear()
                sleeps for 0.75 seconds
                PRINT('Redirecting you to the menu...')
                info_file.close
                session_status = 'registered'
                TF = False
                CALL FUNCTION main_menu()
                RETURN

            ElSE
                PRINT("Answer not recognized, please try again.')
                CONTINUE

            ENDIF

        ELSE
            CALL FUNCTION clear()
            sleeps for 0.75 seconds
            PRINT('OEMS Login')
            PRINT('Your password is incorrect, please try again.')
            CONTINUE

        ENDIF

      ENDWHILE

    ELSE
      CONTINUE

    ENDIF

  ELSE
    CALL FUNCTION clear()
    sleeps for 0.75 seconds

    log_in_retry = '"The entered username does not exist.
What would you like to do?

1. Retry
```

```
    2. Register new account
    3. Main menu

    Choice: '"

        DOWHILE TF == True:
            TRY
                PRINT(log_in_retry)
                Get log_in_retry
            CATCH
                PRINT('Invalid option, please try again.')
                sleeps for 0.75 seconds
                CALL FUNCTION clear()
                CONTINUE

            IF log_in_retry == '1' THEN
                CALL FUNCTION log_in()
                RETURN

            ELIF log_in_retry == '2' THEN
                CLOSE info_file
                acc_name = None
                CALL FUNCTION acc_register()
                RETURN

            ELIF log_in_retry =='3' THEN
                CALL FUNCTION main_menu()
                RETURN

            ELSE
                PRINT('Invalid option, please try again.')
                sleeps for 0.75 seconds
                CONTINUE

            ENDIF

        ENDWHILE

    ENDFOR

    ENDWHILE

    RETURN

ENDFUNCTION
```

```
#5 Function for logging out
FUNCTION log_out()

    CALL FUNCTION clear()
    sleeps for 0.75 seconds
    DECLARE session_status and acc_name as global variable
    session_status = 'guest'
    acc_name = None
    PRINT('Logging out...')
    sleeps for 3 seconds
    CALL FUNCTION(main_menu()
    RETURN

ENDFUNCTION
```

```
#6 Function for registering an account
FUNCTION acc_register()

    TF = True
    CALL FUNCTION clear()
    sleeps for 0.75 seconds
    amount_spent = 0

    DOWHILE True

        status = True
        PRINT('OEMS Account Registration\n')
        PRINT('Please enter your username: ')
        Get acc_name

        IF acc_name == "" THEN
            PRINT('No username entered, please enter a username.')
            sleeps for 0.75 seconds
            CALL FUNCTION clear()
            CONTINUE
        ENDIF

        OPEN account_info.txt in READ mode
        READ line of file as fhandler
        FOR line In fhandler STEP 1
            IF line STARTS WITH acc_name THEN
                status = False
                break
```

```
    ENDIF
ENDFOR

IF status = False THEN
    CALL FUNCTION clear()
    PRINT("This username already exists, please use another username")
    sleeps for 3 seconds
    CALL FUNCTION clear()
    CONTINUE
ENDIF

DOWHILE TF == True

    PRINT('Please enter your credit/debit card number [**** **** **** ****]: ')
    Get payment_card

    IF payment_card == '' THEN
        PRINT('No credit/debit card entered, please enter your credit/debit card number.')
        sleeps for 0.75 seconds
        CONTINUE
    ENDIF
    BREAK

ENDWHILE

DOWHILE TF == True

    PRINT('Please enter your password: ')
    Get acc_password

    IF acc_password == '' THEN
        PRINT('No password entered, please enter your password.')
        sleeps for 0.75 seconds
        CONTINUE
    ENDIF
    BREAK

ENDWHILE

DOWHILE TF == True

    PRINT('Please confirm your password: ')
    Get confirmation

    IF confirmation == '' THEN
        PRINT('No password entered, please enter your password.')
```

```
            sleeps for 0.75 seconds
            CONTINUE
        ENDIF
        BREAK

    ENDWHILE


    IF acc_password != confirmation THEN
        CALL FUNCTION clear()
        sleeps for 0.75 seconds
        PRINT("Incorrect password please try again")
        CONTINUE

    ELIF acc_password = confirmation THEN

        CALL FUNCTION clear()
        sleeps for 0.75 seconds
        acc_info = [acc_name, acc_password]
        OPEN account_info.txt in APPEND mode with buffering set to 1
        APPEND acc_info into file as fhandler
        CLOSE fhandler
        PRINT('Your account has been successfully created.')
        sleeps for 3 seconds
        BREAK

    ENDIF

ENDWHILE

option = '"What would you like to do?
    1. Main Menu
    2. Log In
    3. Exit
Choice: "'

CALL FUNCTION clear()
sleeps for 0.75 seconds

DOWHILE True
    TRY
        PRINT(option)
        Get choice
    CATCH
        PRINT('Invalid option, please try again.')
        sleeps for 0.75 seconds
```

```
        CALL FUNCTION clear()
        CONTINUE

    IF choice == 1 THEN
        TF = False
        CALL FUNCTION main_menu()
        RETURN
    ELIF choice == 2 THEN
        CALL FUNCTION log_in()
        RETURN
    ELIF choice == 3 THEN
        CALL FUNCTION clear()
        CALL FUNCTION quit()
    ELSE
        PRINT("Invalid option please try again")
        continue
    ENDIF

  ENDWHILE

ENDFUNCTION
```

#7 Function for showing category

```
FUNCTION category()

  TF = True
  CALL FUNCTION clear()
  sleeps for 0.75 seconds

  events = '''Events Categories Available:

  1. Sports
  2. E-Sports
  3. Technology
  4. Art
  5. General Entertainment
  6. Back to Main Menu

  Choice: '''

  DOWHILE True
    TRY
      PRINT(events)
      Get answer
    CATCH
```

```
        PRINT('Invalid option, please try again.')
        sleeps 0.75 seconds
        CALL FUNCTION clear()
        CONTINUE

    IF answer <= 6 THEN

        IF answer == 6 THEN
            TF == False
            CALL FUNCTION main_menu()
            RETURN
        ELSE
            RETURN answer
        ENDIF

    ELSE
        CALL FUNCTION clear()
        sleeps for 0.75 seconds
        PRINT("Invalid option please try again.")
        CALL FUNCTION clear()
        sleeps for 0.75 seconds
        CONTINUE

    ENDIF

  ENDWHILE

ENDFUNCTION
```

#8 Function for adding event

```
FUNCTION add_event()

  TF = True
  choice = CALL FUNCTION category()

  IF choice == 1 THEN
    categoryid = 'Sports'
  ELIF choice == 2 THEN
    categoryid = 'E-Sports'
  ELIF choice == 3 THEN
    categoryid = 'Technology'
  ELIF choice == 4 THEN
    categoryid = 'Art'
  ELIF choice == 5 THEN
    categoryid = 'General Entertainment'
```

```
ENDIF

DOWHILE TF == True
   status = True

   PRINT('Please enter the event name: ')
   Get event_name
   IF event_name == "":
      PRINT('No event name entered, please enter an event name.')
      sleeps for 0.75 seconds
      CONTINUE
   ENDIF

   OPEN event.txt in READ mode
   READ line of file as fhandler
   FOR line In fhandler STEP 1

      SPLIT line as event_info using "," as delimiter
      STRIP event_info with an index of 2 as name_availability

      IF name_availability == event_name THEN
         status  = False
      ENDIF
   ENDFOR

   IF status == False:
      PRINT("Event exsits please try again")
      CONTINUE
   ENDIF

   DOWHILE TF == True

      TRY
         PRINT('How much is the event?(RM): ')
         Get event_price

         IF event_price == '' or event_price == 0 THEN
            PRINT('No price entered, please enter price.')
            sleeps for 0.75 seconds
            CONTINUE
         BREAK
      CATCH
         PRINT('No price entered, please enter a price.')
         sleeps for 0.75 seconds
         CONTINUE
```

```
        ENDWHILE

        listid=1
        OPEN event.txt in READ mode
        READ line of file as fhandler
        FOR line In fhandler STEP 1
            IF line ENDS WITH ("\n") THEN
                listid +=1
            ENDIF
        ENDFOR

        event_list = [listid, categoryid, event_name, event_price]
        OPEN event.txt in APPEND mode with buffering set to 1
        APPEND event_list into file as fhandler
        CLOSE fhandler

        clear terminal
        sleeps for 0.75 seconds
        PRINT("Your event has been added")
        sleeps for 3 seconds
        TF == False
        CALL FUNCTION main_menu()
        RETURN

    ENDWHILE

    RETURN

ENDFUNCTION
```

#9 Function for modifying events

```
FUNCTION modify_event()

    CALL FUNCTION event_list()
    PRINT('\n')
    TF = True

    DOWHILE TF == True

        PRINT("Which event would you like to modify?[ID]: ")
        Get choice_id

        IF choice_id == "" THEN
            PRINT('No event ID entered, please enter an event ID.')
            sleeps for 0.75 seconds
```

```
        CONTINUE
    ELSE
        BREAK
    ENDIF

OPEN event.txt in READ mode with buffering set to 1
READ line in file as fhandler_read

FOR line IN fhandler_read STEP 1
    SPLIT line as event_info with ',' as delimiter
    STRIP event_info with index of 0 as event_id
    STRIP event_info with index of 1 as event_category
    STRIP event_info with index of 2 as event_name
    STRIP event_info with index of 3 as event_price

    CALL FUNCTION clear()
    sleeps for 0.75 seconds

    option = '''Options available:
1. Change event category
2. Change event name
3. Change event price
4. Delete event
5. Cancel

Choice: '''

    IF event_id == choice_id THEN

        DOWHILE TF == True
            TRY
                PRINT(option)
                Get option_input
            CATCH
                PRINT('Invalid option, please try again.')
                sleeps for 0.75 seconds
                CALL FUNCTION clear()
                CONTINUE

            IF option_input == 1 THEN
                choice = CALL FUNCTION category()

                IF choice == 1 THEN
                    new_category = 'Sports'
                    BREAK
                ELIF choice == 2 THEN
```

```
            new_category = 'E-Sports'
            BREAK
         ELIF choice == 3 THEN
            new_category = 'Technology'
            BREAK
         ELIF choice == 4 THEN
            new_category = 'Art'
            BREAK
         ELIF choice == 5 THEN
            new_category = 'General Entertainment'
            BREAK
         ENDIF

      ELIF option_input == 2 THEN
         DOWHILE TF == True
            PRINT('Please enter new event name: ')
            Get new_name
            IF new_name == "" THEN
               PRINT('No new event name entered, please enter a new event name.')
               sleeps for 0.75 seconds
               CONTINUE
            ELSE
               BREAK
            ENDIF

         ENDWHILE


      ELIF option_input == 3 THEN
         DOWHILE TF == True
            PRINT('Please enter new price[RM]: ')
            Get new_price
            IF new_price == '' or new_price == '0' THEN
               PRINT('No new price entered, please enter new price.')
               sleeps for 0.75 seconds
               CONTINUE
            ELSE
               BREAK
            ENDIF

         ENDWHILE

      ELIF option_input == 4 THEN
         BREAK

      ELIF option_input == 5 THEN
```

```
            CALL FUNCTION main_menu()
            RETURN

        ELSE
            PRINT('Invalid option, please try again.')
            sleeps for 0.75 seconds
            CALL FUNCTION clear()
            CONTINUE

        ENDIF

    ENDWHILE
    BREAK

  ENDIF

ENDFOR

CLOSE fhandler_read

WITH OPEN event.txt in READ mode as fhandler_read

  list_data_temp = []
  readlines in fhandler_read as list_data

  FOR line in list_data

    IF line STARTS WITH choice_id THEN

      IF option_input == 1 THEN
          line = replace event_category with new_category in line
      ELIF option_input == 2 THEN
          line = replace event_name with new_name in line
      ELIF option_input == 3 THEN
          line = replace event_price with new_price in line
      ELIF option_input == 4 THEN
          line = replace line with ""
      ENDIF

    ENDIF

    APPEND line into list_data_temp

  ENDFOR

WITH OPEN event.text in WRITE mode as fhandler_write
```

```
        FOR line in list_data_temp
            WRITE line in fhandler_write

        ENDFOR

    CALL FUNCTION clear()
    sleeps for 0.75 seconds
    PRINT("Modified complete, redirecting to main menu.....")
    sleeps for 3 seconds
    CALL FUNCTION main_menu()

    RETURN

ENDFUNCITON
```

```
#10 Function for listing events
choice = CALL FUNCTION category()
    CALL FUNCTION clear()
    sleeps for 0.75 seconds

    OPEN event.txt in READ mode
    READ line in file as event_file

    IF choice == 1 THEN
        categoryid = 'Sports'
    ELIF choice == 2 THEN
        categoryid = 'E-Sports'
    ELIF choice == 3 THEN
        categoryid = 'Technology'
    ELIF choice == 4 THEN
        categoryid = 'Art'
    ELIF choice == 5 THEN
        categoryid = 'General Entertainment'
    ENDIF

    PRINT('Category: ', categoryid, '\n')

    FOR line in event_file
        SPLIT line as event_info with ',' as delimiter
        STRIP event_info with index of 0 as event_info_id
        STRIP event_info with index of 1 as event_info_category
        STRIP event_info with index of 2 as event_info_name
        STRIP event_info with index of 3 as event_info_price
```

```
        IF categoryid == event_info_category THEN
            PRINT(f'ID:{event_info_id}          Event:{event_info_name}
Price:RM{event_info_price}')
        ENDIF
    ENDFOR

ENDFUNCTION
```

#11 Function for displaying the menu for the list event function

```
FUNCTION list_event_menu()

    CALL FUNCTION event_list()

    TF = True

    event_menu = "'What would you like to do?

    1.Back to category menu
    2.Back to main menu

    Choice: "'

    DOWHILE TF == True
        PRINT('\n')

        TRY
            PRINT(event_menu)
            Get choice
        CATCH
            PRINT('Invalid choice, please try again.')
            sleeps for 0.75 seconds
            CONTINUE

        IF choice == 1 THEN
            CALL FUNCTION list_event_menu()
            RETURN
        ELIF choice == 2 THEN
            TF = False
            CALL FUNCTION main_menu()
            RETURN
        ELSE
            PRINT('Invalid choice, please try again.')
            sleeps for 0.75 seconds
            CONTINUE
        ENDIF
```

```
    ENDWHILE

    RETURN

ENDFUNCTION
```

```
#12 Function for users to add events to cart
FUNCTION cart()
    CALL FUNCTION clear
    sleeps for 0.75 seconds
    CALL FUNCTION event_list()
    PRINT('\n')
    TF = True


    DOWHILE TF == True

        PRINT('\n')
        PRINT("Which event[ID] would you like to add to cart? (Type 'n' to cancel): ")
        Get event_choice

        IF event_choice == "" THEN
            PRINT('No event ID entered, please enter an event ID.')
            sleeps for 0.75 seconds
            CONTINUE
        ENDIF

        WITH OPEN event.txt in READ mode as event_file

            readlines in event_file as event_file_read

            FOR item in event_file_read
                SPLIT item as events with (',') as delimiter

                STRIP events with index of 0 as event_id
                STRIP events with index of 1 as event_category
                STRIP events with index of 2 as event_name
                STRIP events with index of 3 as event_price

                IF event_choice == event_id THEN
                    WITH OPEN cart.txt in APPEND mode with buffering set to 1 as cart_file
                        event_info = [event_id, event_category, event_name, event_price]
                        APPEND event_info into cart_file
                        CALL FUNCTION clear()
```

```
                sleeps for 0.75 seconds

                PRINT('Event successfully added to cart, would you like to add another event?
(y/n): ')
                Get add_another_event

                IF add_another_event == 'y' or add_another_event == 'Y' THEN
                   CALL FUNCTION event_list()
                   CONTINUE
                ELIF add_another_event == 'n' or add_another_event == 'N' THEN
                   CALL FUNCTION clear()
                   sleeps for 0.75 seconds
                   PRINT('Sending you to main menu...')
                   sleeps for 3 seconds
                   CALL FUNCTION main_menu()
                   RETURN
                ELSE
                   PRINT('Invalid option, please try again.')
                   sleeps for 0.75 seconds
                   CONTINUE
                ENDIF

            ELIF event_choice == 'n' or event_choice == 'N' THEN
               TF = False
               PRINT('Aborting...Sending you to main menu...')
               sleeps for 3 seconds
               CALL FUNCTION main_menu()
            ELSE
               PRINT('The event ID you entered does not exist, please try again.')
               sleeps for 0.75 seconds
               BREAK
            ENDIF

        ENDFOR

        CONTINUE

   ENDWHILE

   RETURN

ENDFUNCTION
```

```
#13 Function for view user's cart
FUNCTION view_cart()
    Declare acc_name as Global variable
    CALL FUNCTION clear()
    sleeps for 0.75 seconds
    TF = True

    TRY
        WITH OPEN cart.txt in READ mode as cart_file
            readlines in cart_file as cart_file_read
            total_price = 0

            PRINT(f"{acc_name}'s cart.\n")

            FOR item in cart_file_read
                SPLIT item as events with ',' as delimiter
                STRIP events with index of 0 as event_id
                STRIP events with index of 1 as event_category
                STRIP events with index of 2 as event_name
                STRIP events with index of 3 as event_price
                total_price = total_price + int(event_price)

                PRINT(f'ID:{event_id}        Category:{event_category}        Event:{event_name}
Price:RM{event_price}')
            ENDFOR

    CATCH
        CALL FUNCTION clear()
        sleeps for 0.75 seconds
        PRINT('No records in cart, redirecting to main menu...')
        sleeps for 3 seconds
        CALL FUNCTION main_menu()
        RETURN

    PRINT('\n')
    PRINT(f"Total Price: RM{total_price}")

    view_cart_menu = "'What would you like to do?

    1.Proceed to checkout
    2.Back to main menu

    Choice: "'

    DOWHILE TF = True
        TRY
```

```
        PRINT(view_cart_menu)
        Get answer
        BREAK
      CATCH
        PRINT('Invalid option, please try again.')
        sleeps for 0.75 seconds
        CONTINUE
  ENDWHILE

  IF answer == 1 THEN

    DOWHILE TF = True
      PRINT('Are you sure you would like to checkout all items? (y/n): ')
      Get confirmation
        IF confirmation == 'y' or confirmation == 'Y' THEN
          PRINT('Processing...')

          WITH OPEN account_info.txt in READ mode as fhandler_read

            account_data_temp = []
            readlines in fhandler_read as account_data
            FOR account in account_data
              SPLIT account as accounts with ',' as delimiter
              STRIP accounts with index of 0 as account_name
              STRIP accounts with index of 3 as amount_spent
              new_total_price = STR(total_price + int(amount_spent))

              IF account_name == acc_name THEN
                account = replace amount_spent with new_total_price in account
              ENDIF

              APPEND account into account_data_temp

            ENDFOR

          WITH OPEN account_info.txt in WRITE mode with buffering set to 1 as
fhandler_write
            FOR account in account_data_temp
              WRITE account in fhandler_write
            ENDFOR

          OPEN cart.txt in WRITE mode with buffering set to 1 as cart_file
          CLOSE cart_file

          CALL FUNCTION clear()
          sleeps for 0.75 seconds
```

```
            PRINT('Purchase complete, the receipt will be sent to you by the end of the
month, Thank you!')
            sleeps for 3 seconds
            CALL FUNCTION clear()
            PRINT('Redirecting you to the main menu...')
            sleeps for 3 seconds
            CALL FUNCTION main_menu()
            RETURN

         ELIF confirmation == 'n' or confirmation == 'N' THEN
            CALL FUNCTION view_cart()
            RETURN

         ELSE
            PRINT('Invalid option, please try again.')
            sleeps for 0.75 seconds
            CONTINUE

         ENDIF

      ENDWHILE

   ELIF answer == 2 THEN
      PRINT('Sending you to main menu...')
      sleeps for 3 seconds
      CALL FUNCTION main_menu()
      RETURN

   ENDIF

   RETURN

ENDFUNCTION
```

```
#14 Function for displaying all customer records
FUNCTION customer_records()

   CALL FUNCTION clear()
   sleeps for 0.75 seconds
   TF = True
   PRINT('Customer records\n')

   WITH OPEN account_info.txt in READ mode as fhandler
      readlines in fhandler as accounts
```

```
    FOR item in accounts
        SPLIT item as account with (',') as delimiter
        STRIP account with index of 0 as account_name
        STRIP account with index of 2 as amount_spent

        PRINT(f'Username:{account_name}          Ammount spent:{ammount_spent}')

        menu = '''What would you like to do?

        1.Back to main menu

        Choice: '''

        DOWHILE TF = True
            TRY
                PRINT(menu)
                Get option
            CATCH
                PRINT('Invalid option, please try again.')
                sleeps for 0.75 seconds
                CONTINUE

            IF option == 1 THEN
                CALL FUNCTION clear()
                sleeps for 0.75 seconds
                PRINT('Sending you back to the main menu...')
                sleeps for 3 seconds
                TF = False
                CALL FUNCTION main_menu()
                RETURN
            ELSE
                PRINT('\n')
                PRINT('Invalid option, please try again.')
                sleeps for 0.75 seconds
                CONTINUE
            ENDIF

        ENDWHILE

    ENDFOR

ENDFUNCTION
```

```
#15 Function to search and display specific customer record
FUNCTION specific_customer_record()
    TF = True


    DOWHILE TF = True
        PRINT("Please enter account's username: ")
        Get acc_name
        CALL FUNCTION clear
        sleeps for 0.75 seconds

        WITH OPEN account_info.txt as fhandler in READ mode
            readlines in fhandler as accounts
            FOR items in accounts
                SPLIT item as account with (',') as delimiter
                STRIP account with index of 0 as account_name
                STRIP account with index of 2 as amount_spent

                IF account_name == acc_name THEN
                    PRINT(f"{account_name}'s records\n")
                    PRINT(f'Username:{account_name}          Ammount spent:{ammount_spent}\n')
                    break
                ENDIF

            ELSE
                PRINT("Invalid username please try again")
                sleeps for 3 seconds
                CALL FUNCTION clear
                sleeps for 0.75 seconds
                CONTINUE

            ENDFOR
            BREAK
    ENDWHILE

        menu = '''What would you like to do?

    1.Back to main menu

    Choice: '''

    DOWHILE TF == True
        TRY
            PRINT(menu)
            Get option
        CATCH
```

```
        PRINT('Invalid option, please try again.')
        sleeps for 0.75 seconds
        CONTINUE

    IF option == 1 THEN
        CALL FUNCTION clear()
        sleeps for 0.75 seconds
        PRINT('Sending you back to the main menu...')
        sleeps for 3 seconds
        TF = False
        CALL FUNCTION main_menu()
        RETURN
    ELSE
        PRINT('\n')
        PRINT('Invalid option, please try again.')
        sleeps for 0.75 seconds
        CONTINUE
    ENDIF

  ENDWHILE

ENDFUNCTION
```

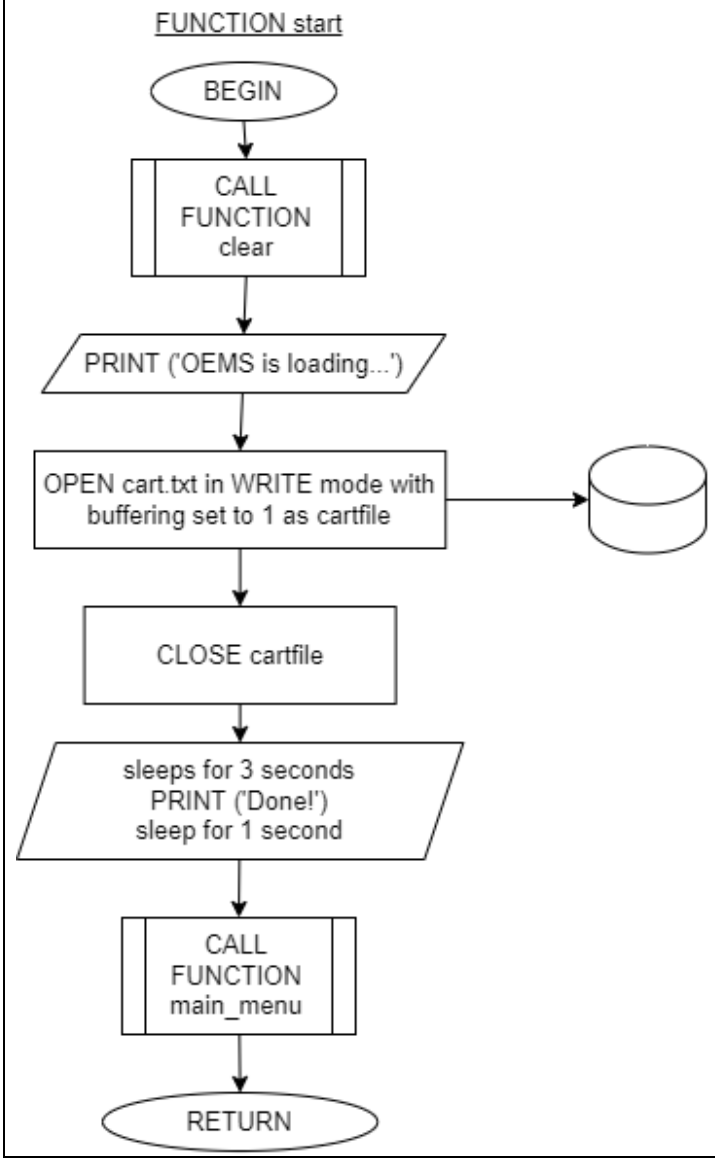
#16 Starting the Online Event Management System

```
Program OEMS
BEGIN

    DECLARE session_status, acc_name, menu, acc_name, acc_password,
admin_confirmation, payment_card, confirmation, acc_info, option, events, acc_info_name,
acc_info_password, event_info, event_id, event_category, event_name, even_price,
event_info, event_info_id, event_info_category, event_info_name, even_info_price,
name_availability, event_list, categoryid, choice_id, new_category, new_name, new_price,
list_data_temp, list_data, event_menu, event_choice, add aother_event, view_cart_menu,
account_data, account_data_temp, accounts, account_name, amount_spent, new_total_price as
an array of string
    DECLARE TF, status as a boolean
    DECLARE answer, admin_code, log_in_retry, amount_spent, choice, event_price, listid,
option_input, total_price as integer

    IMPORT os and time module

    session_status = 'guest'
    acc_name = None

    CALL FUNCTION start()

END
```

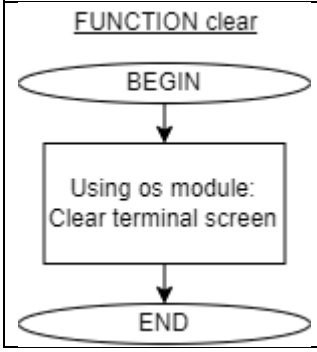
Flowchart

The full flowchart file can be found in the same zip file under the name “flowchart.drawio”, and can be viewed on draw.io.

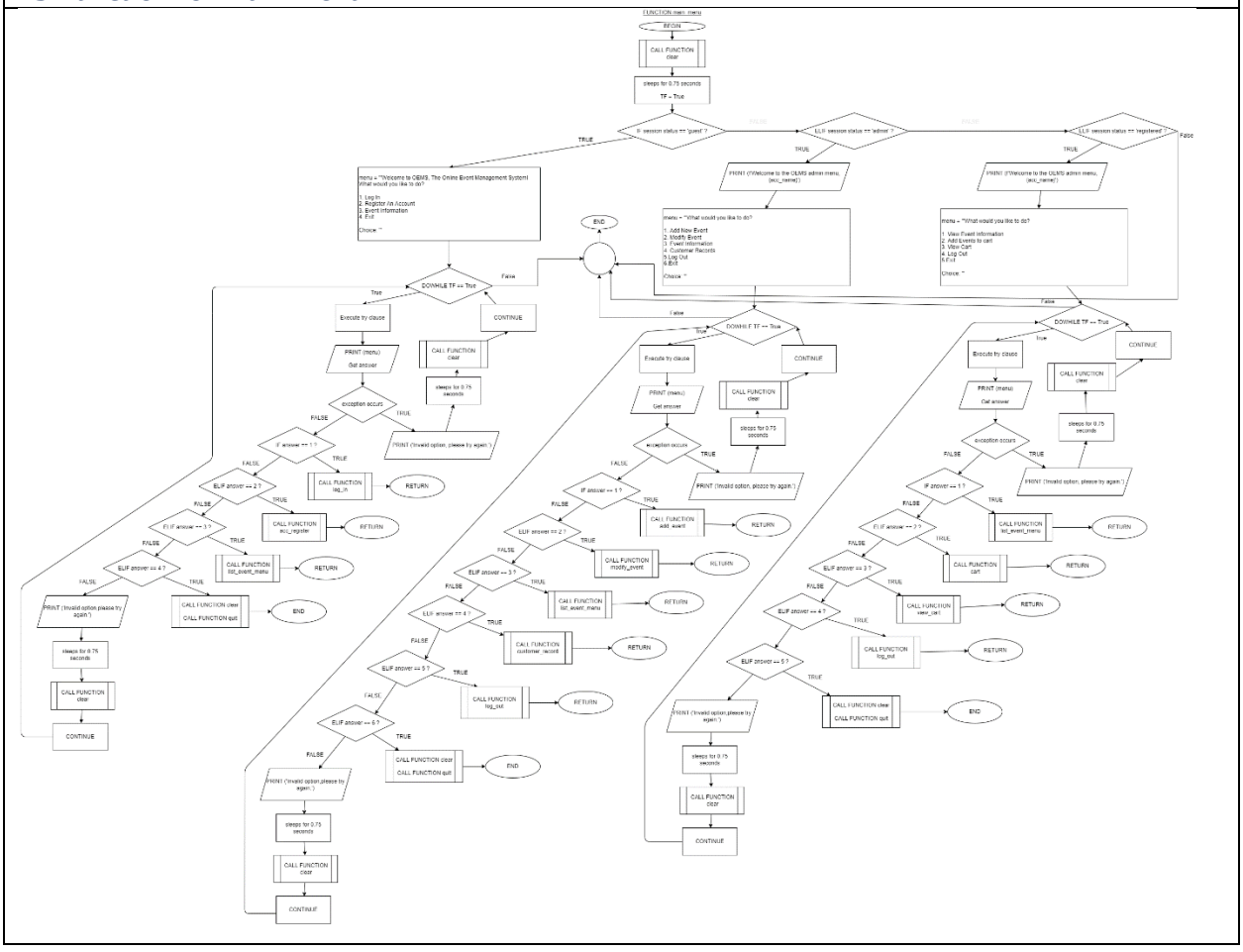
#1 Function to start program



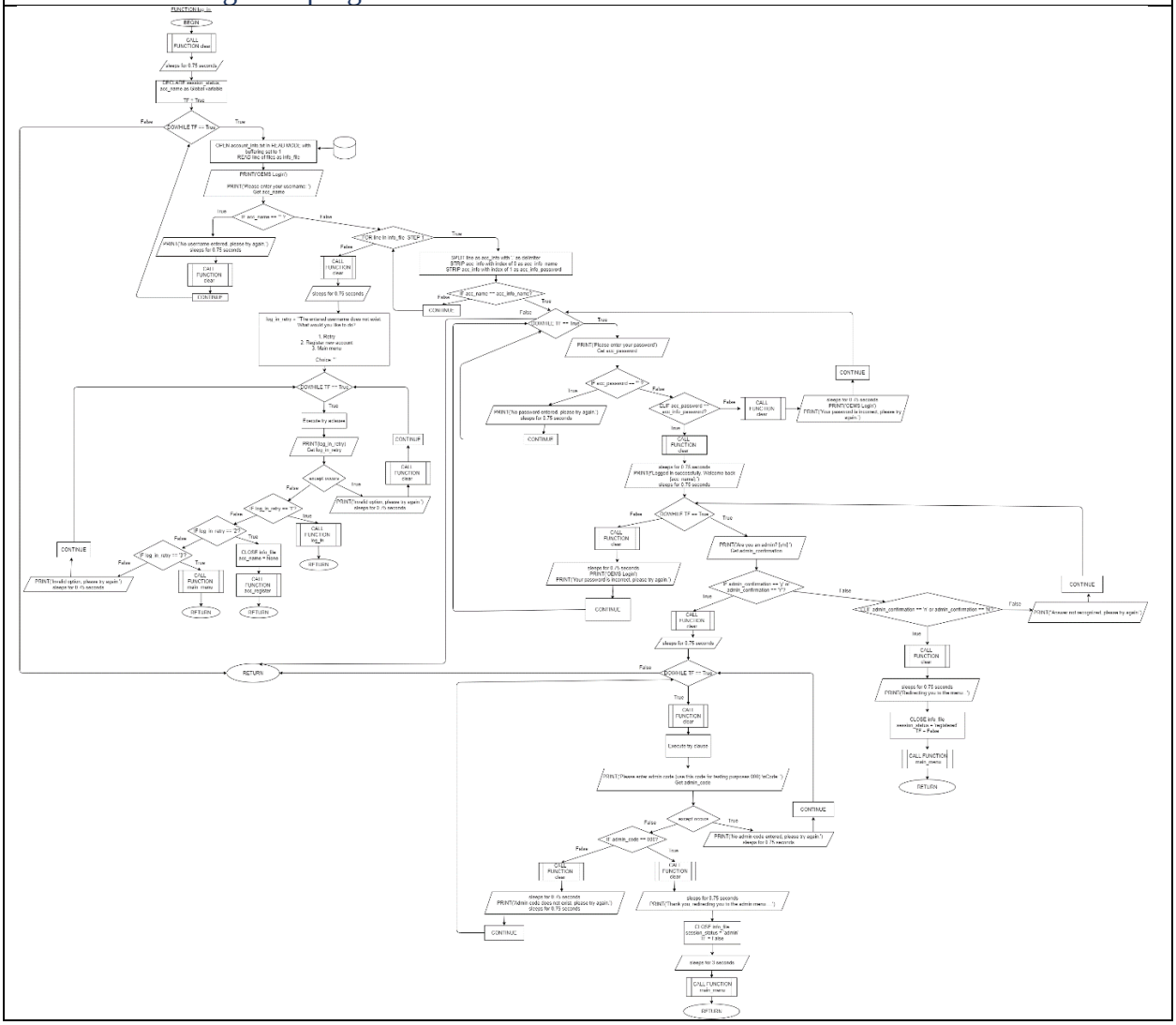
#2 Function to clear terminal screen



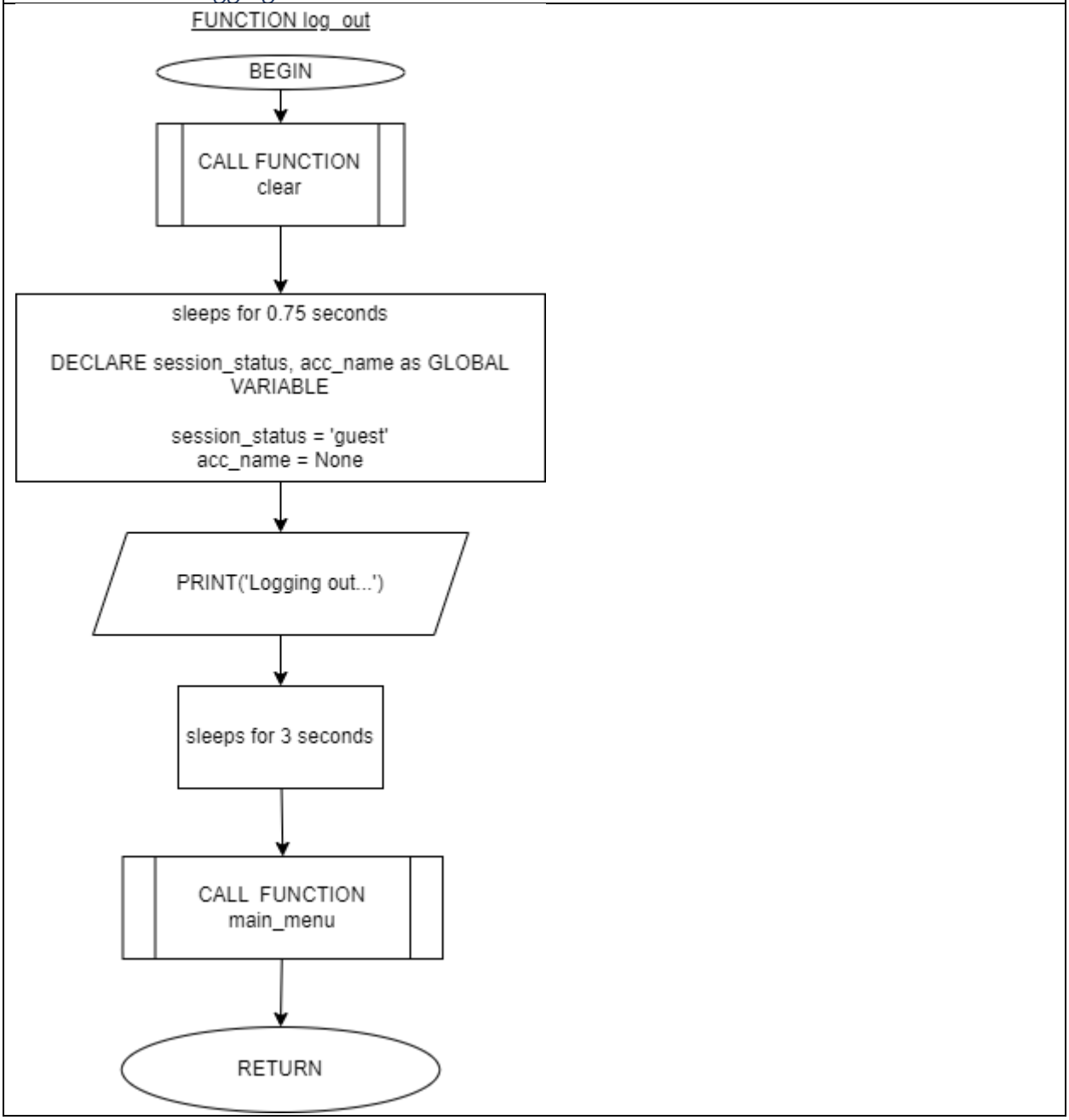
#3 Function for main menu



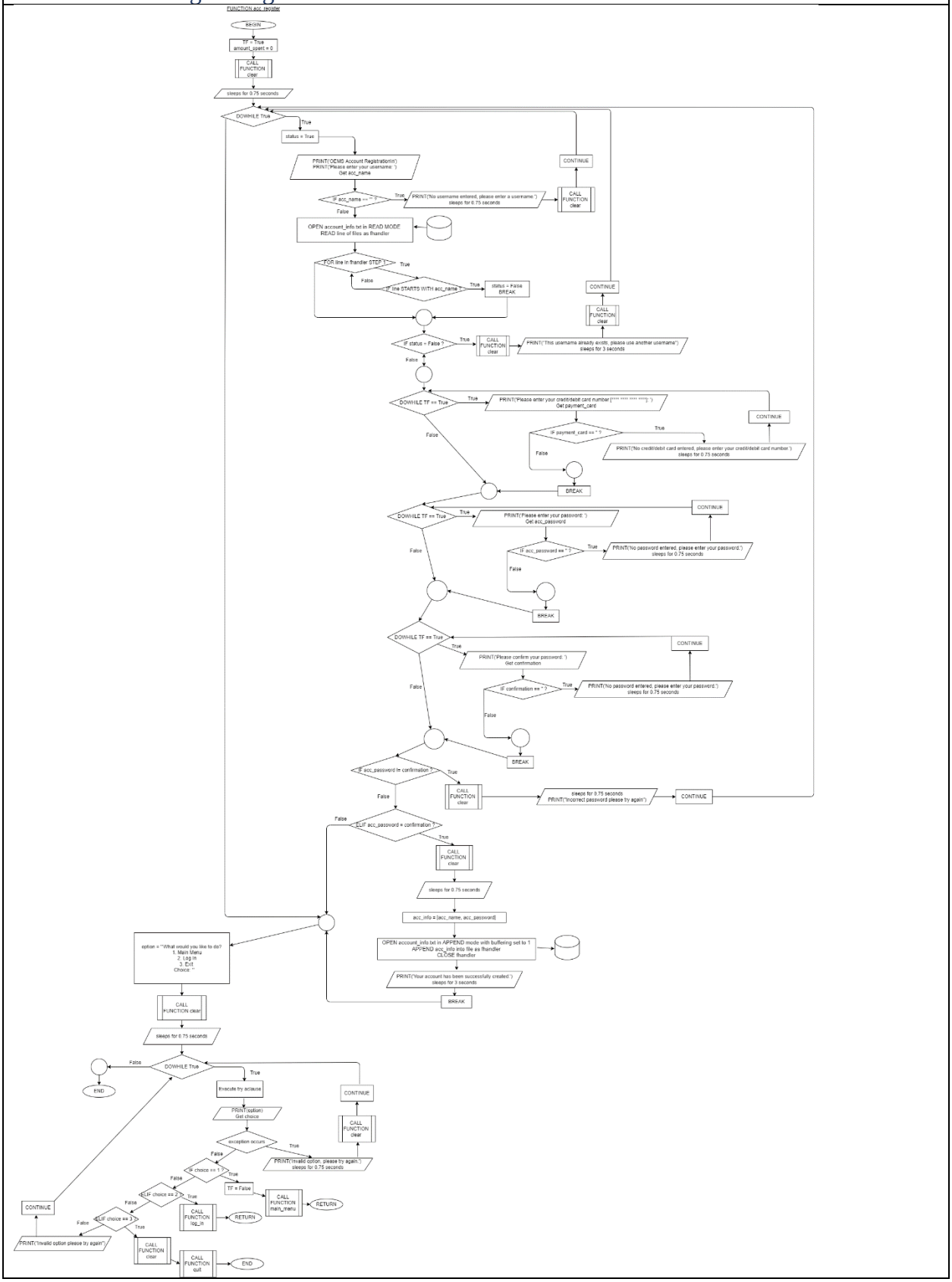
#4 Function to log in to program



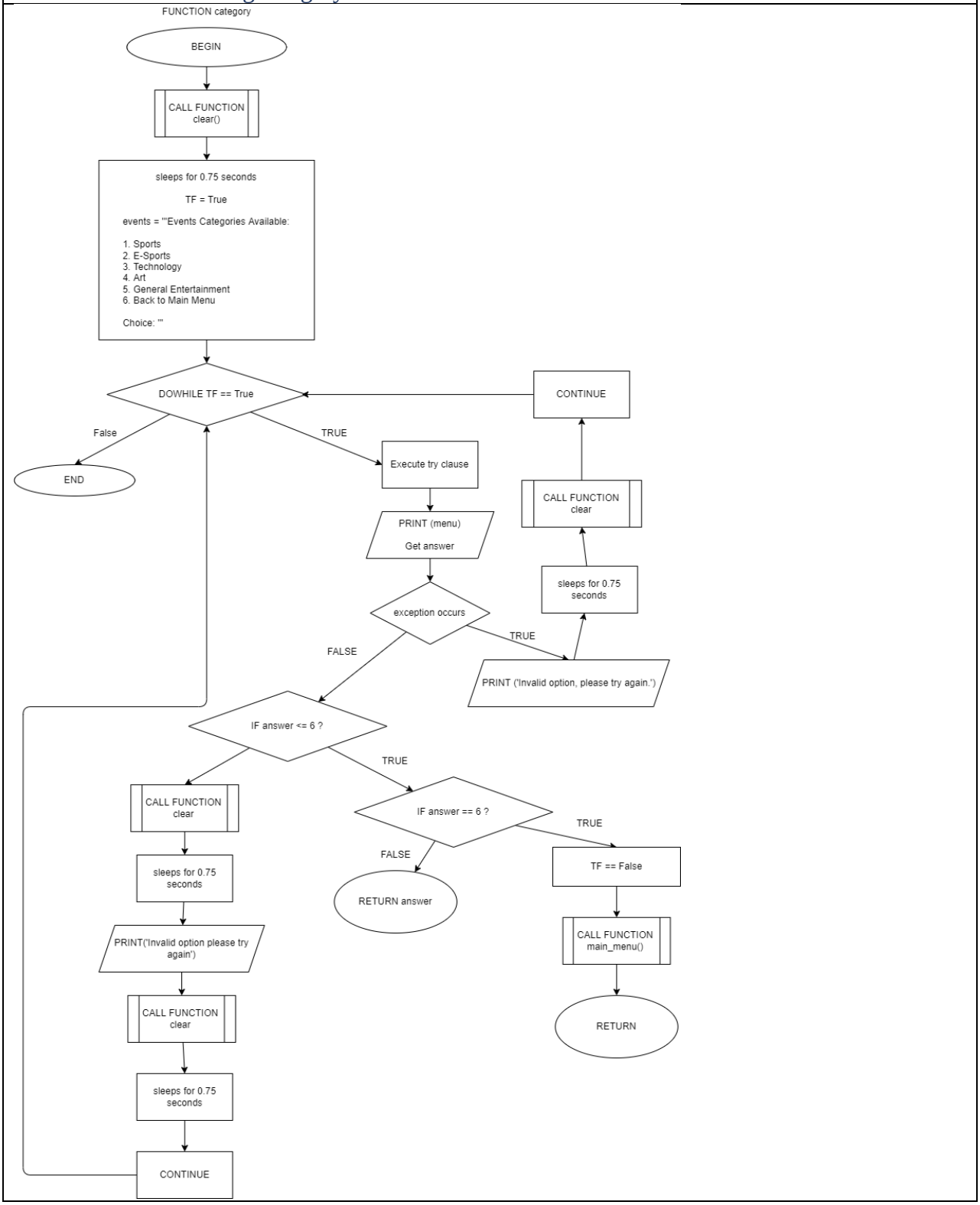
#5 Function for logging out



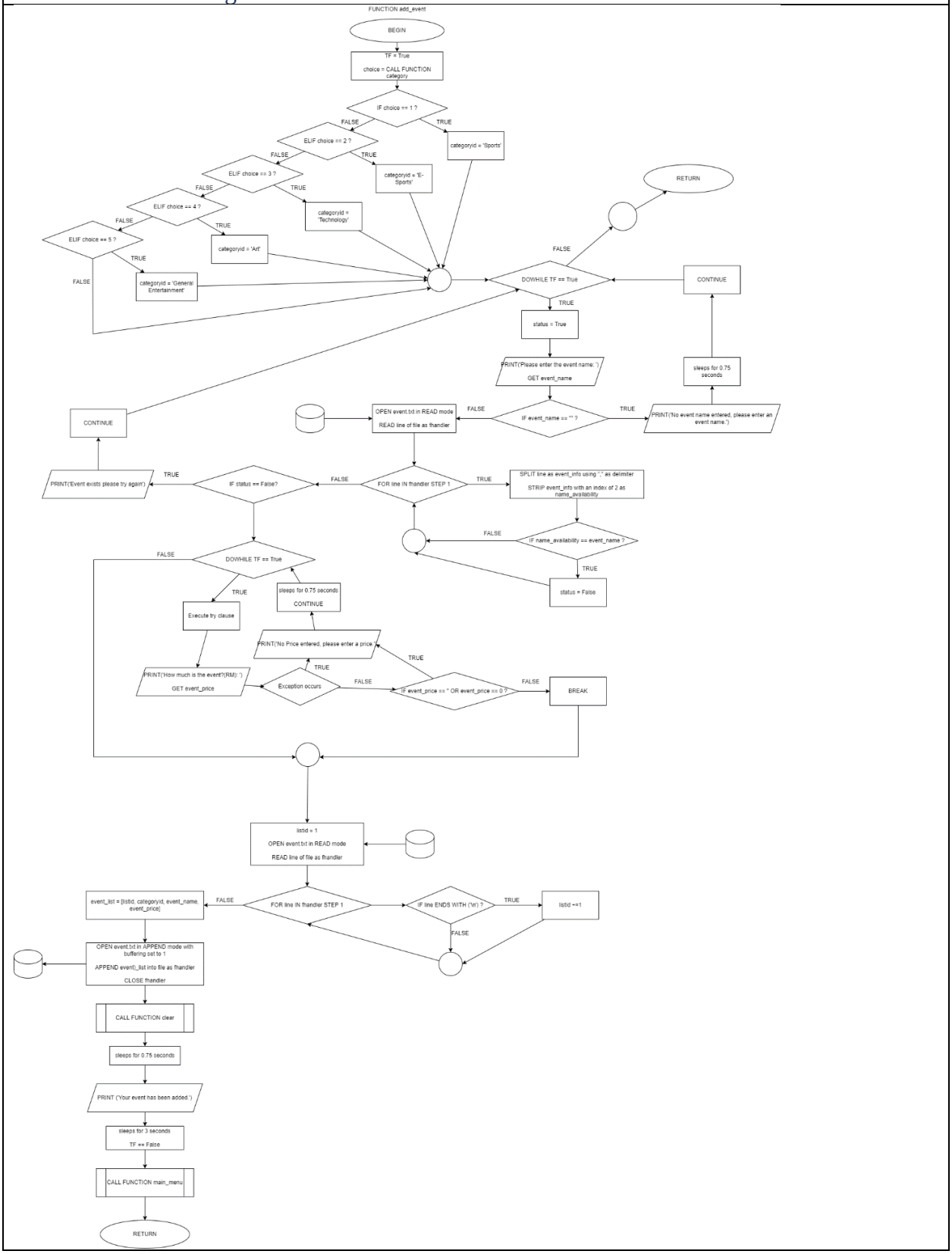
#6 Function for registering an account



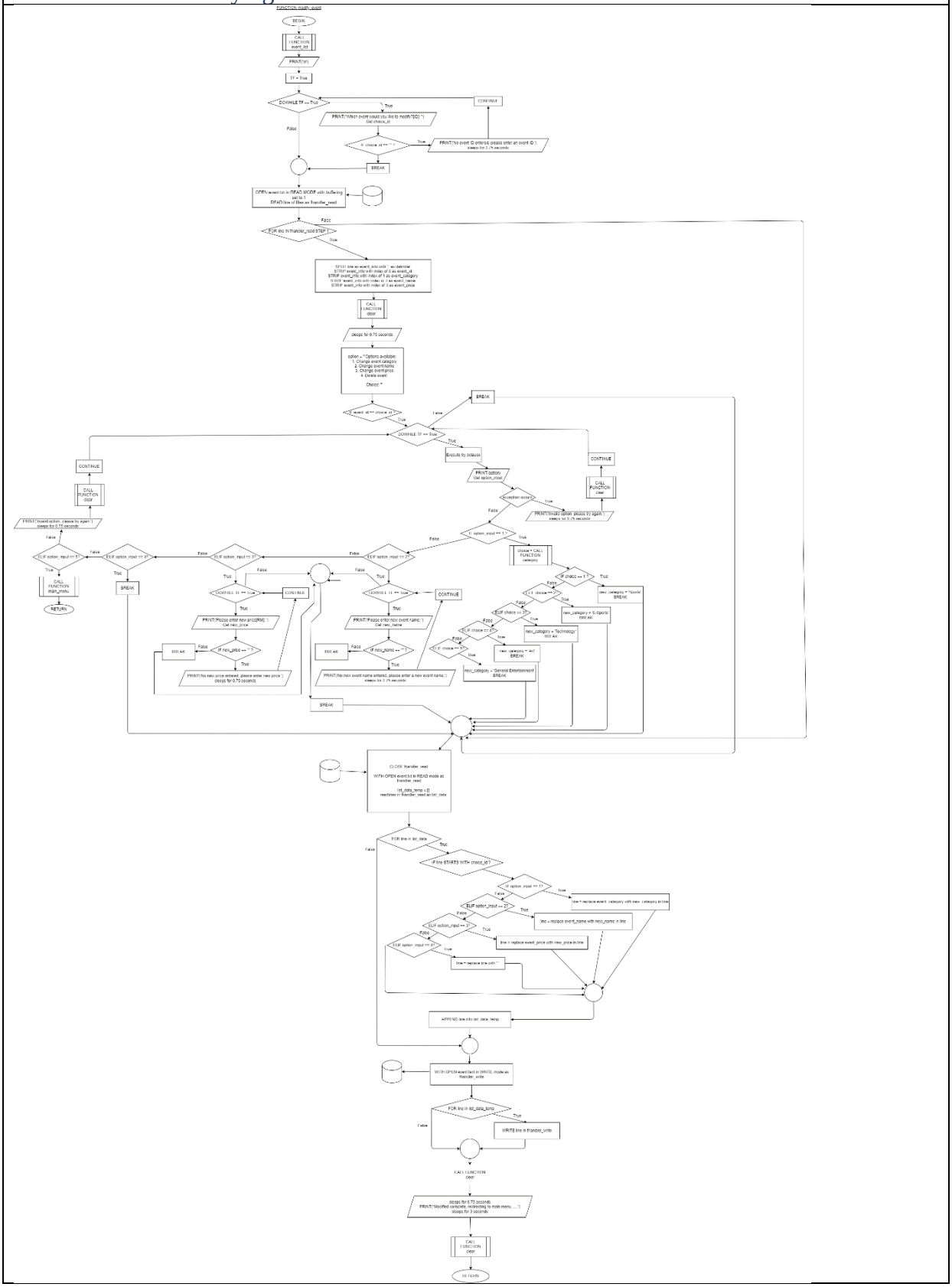
#7 Function for showing category



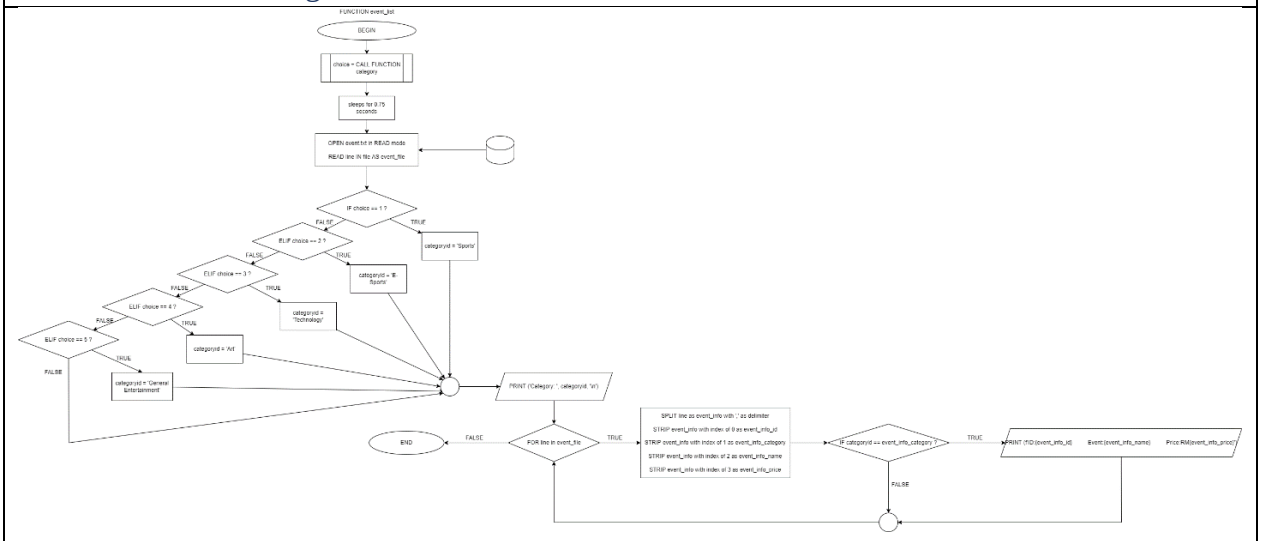
#8 Function for adding event



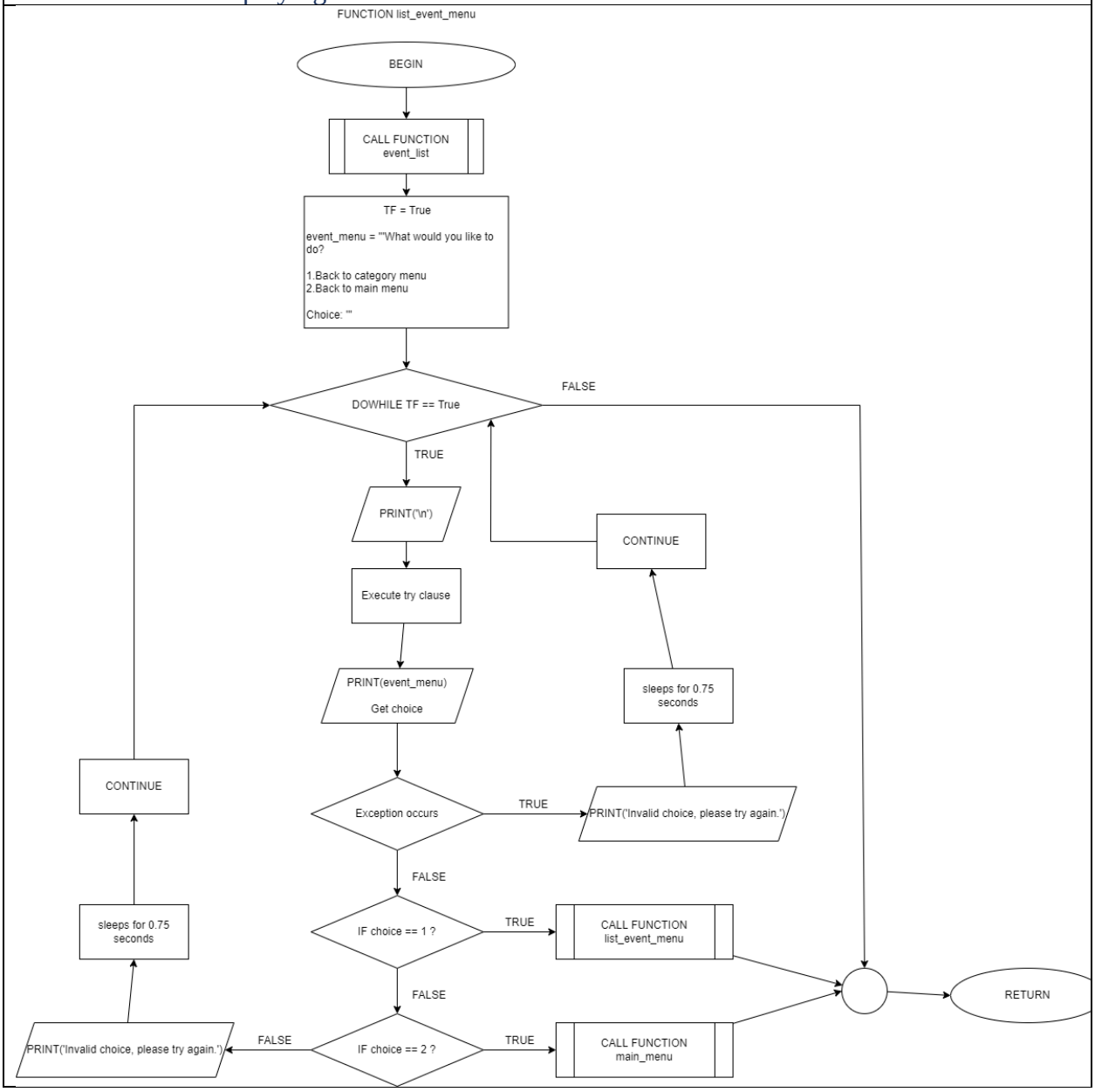
#9 Function for modifying events



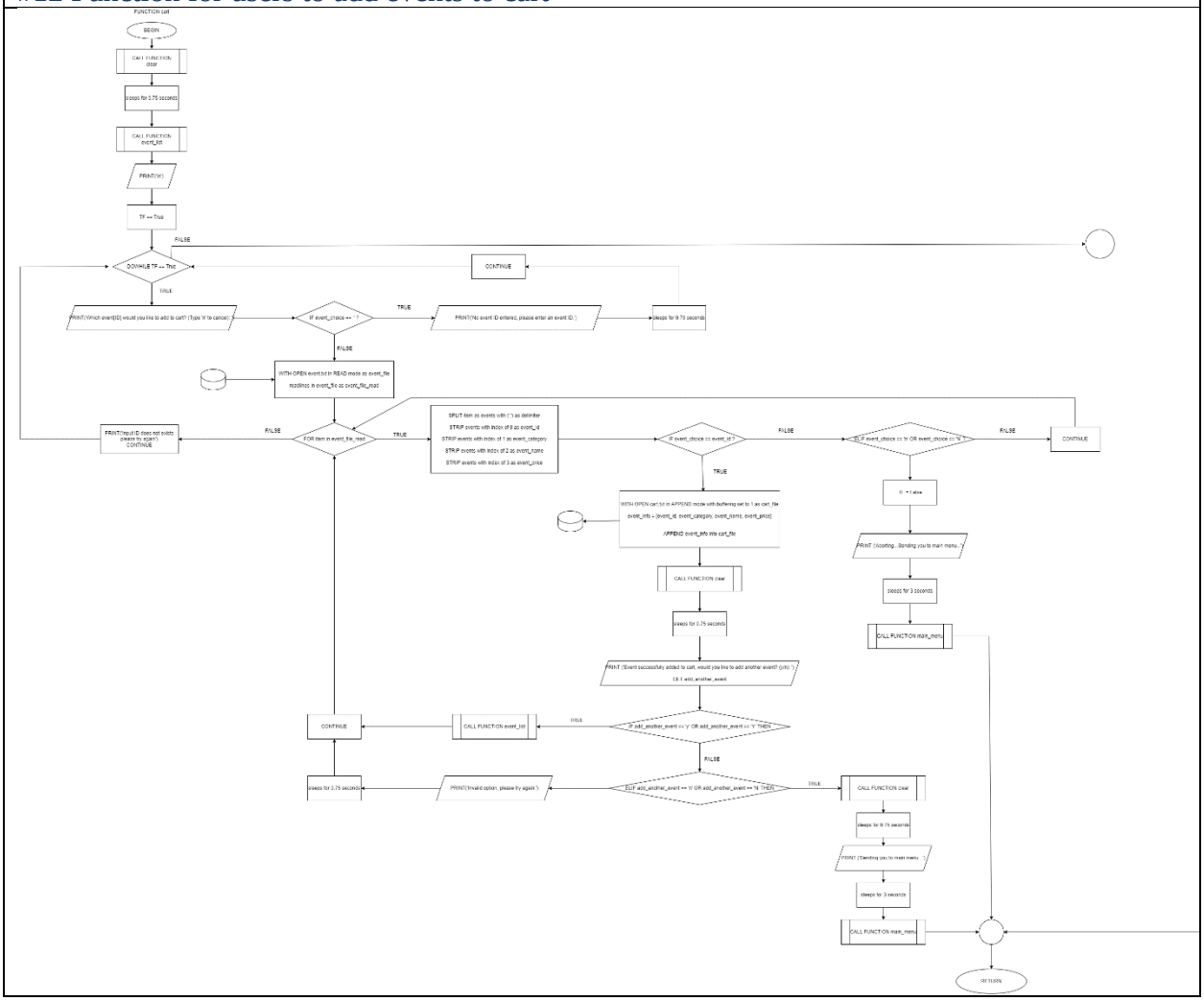
#10 Function for listing events



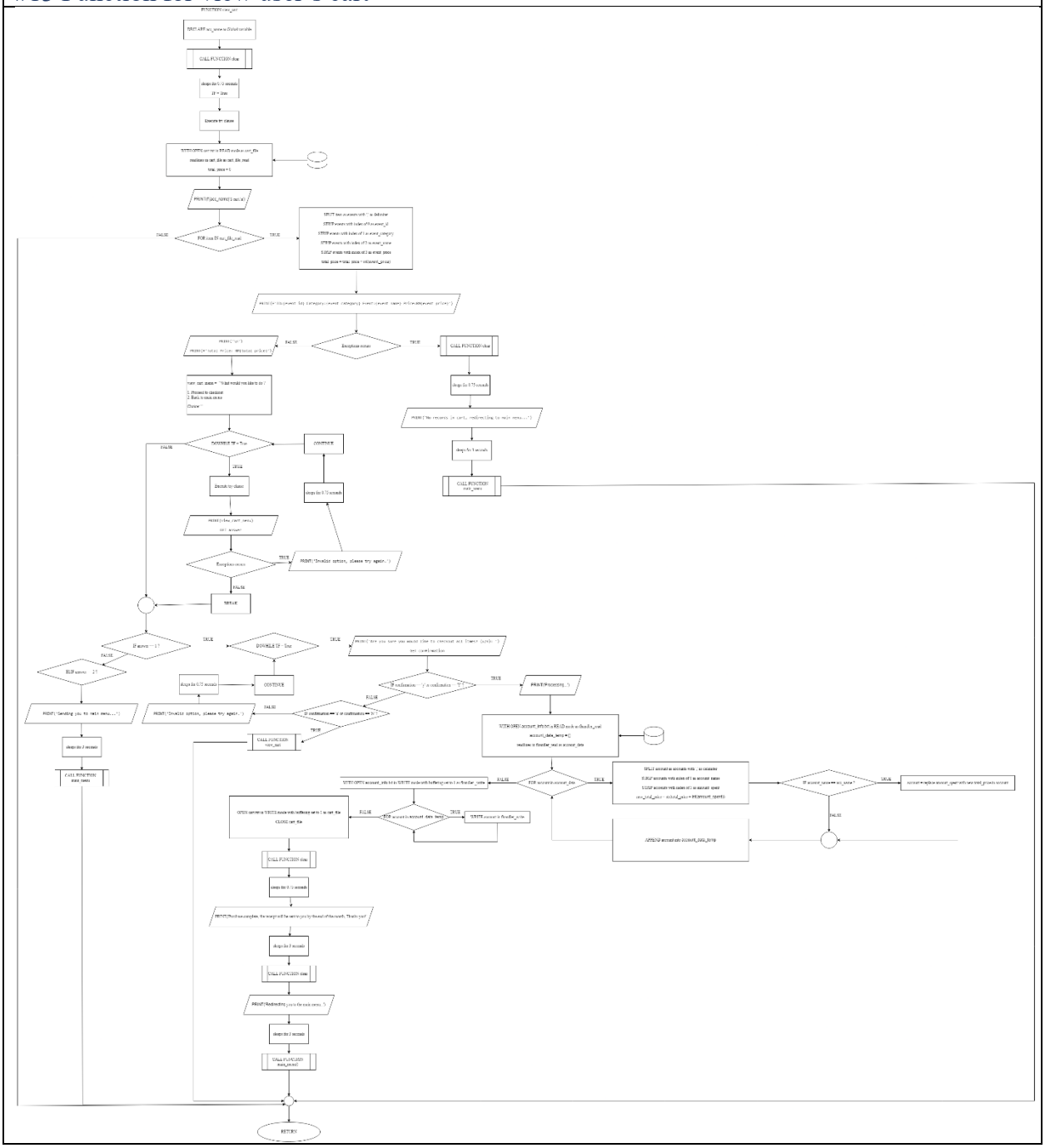
#11 Function for displaying the menu for the list event function



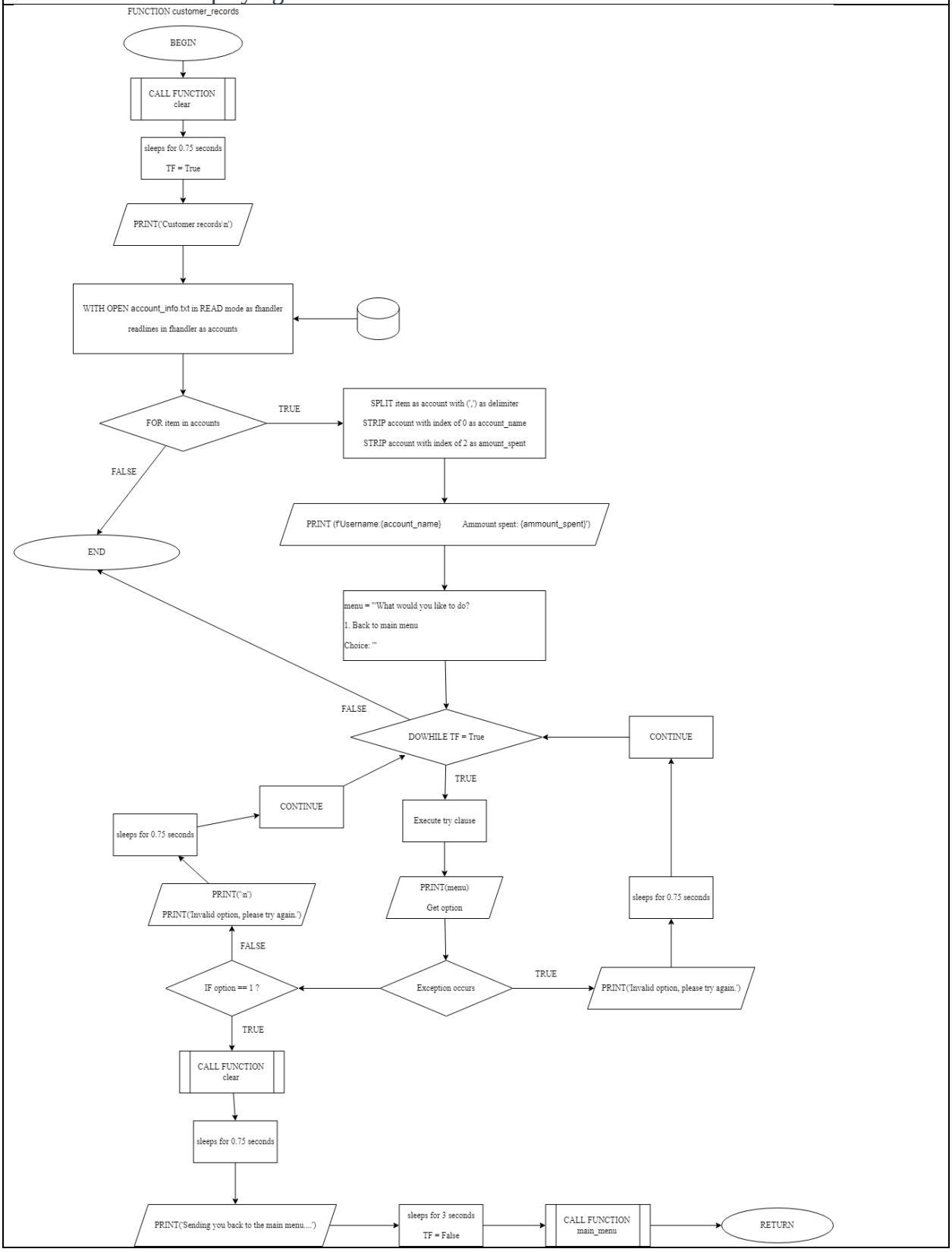
#12 Function for users to add events to cart



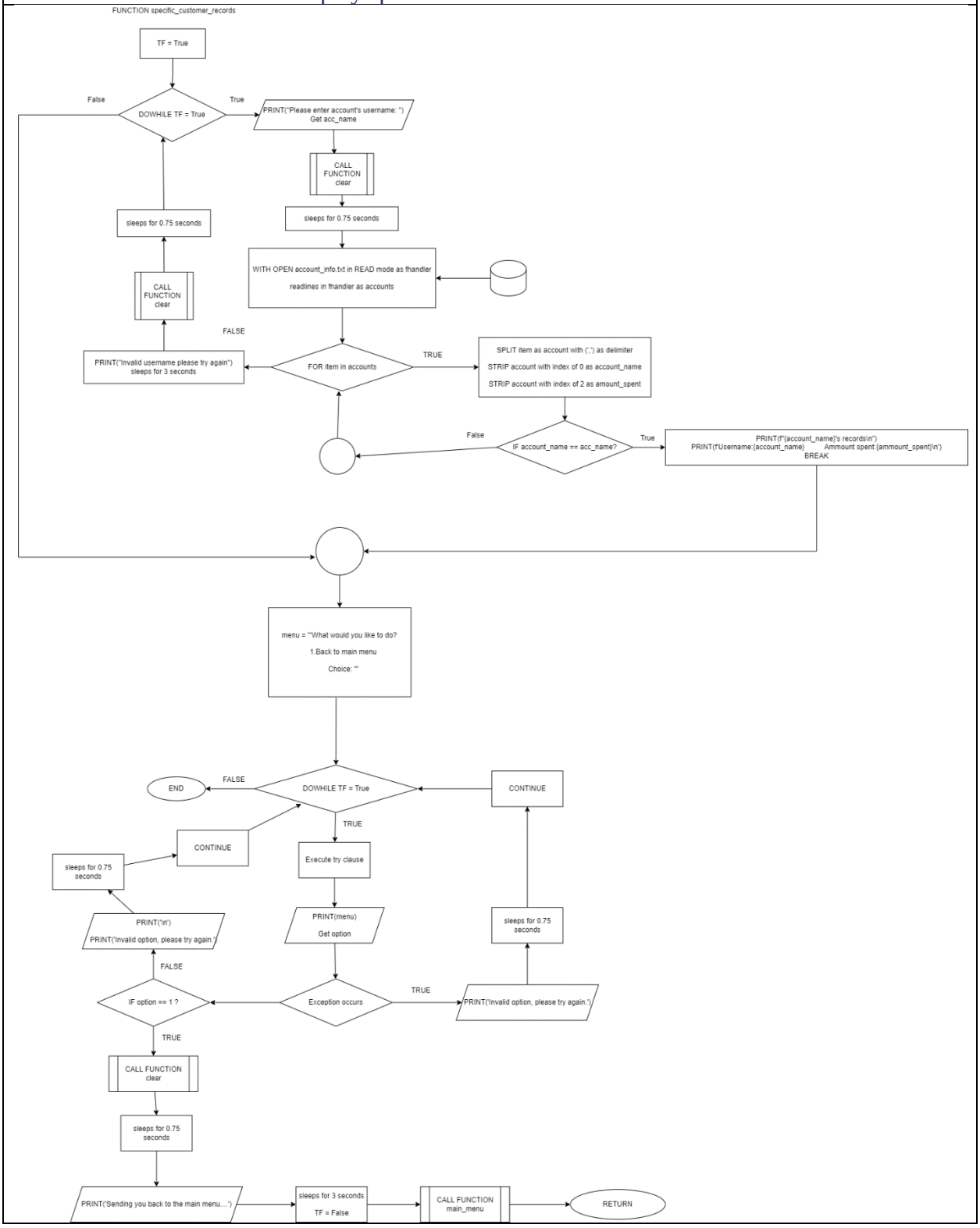
#13 Function for view user's cart



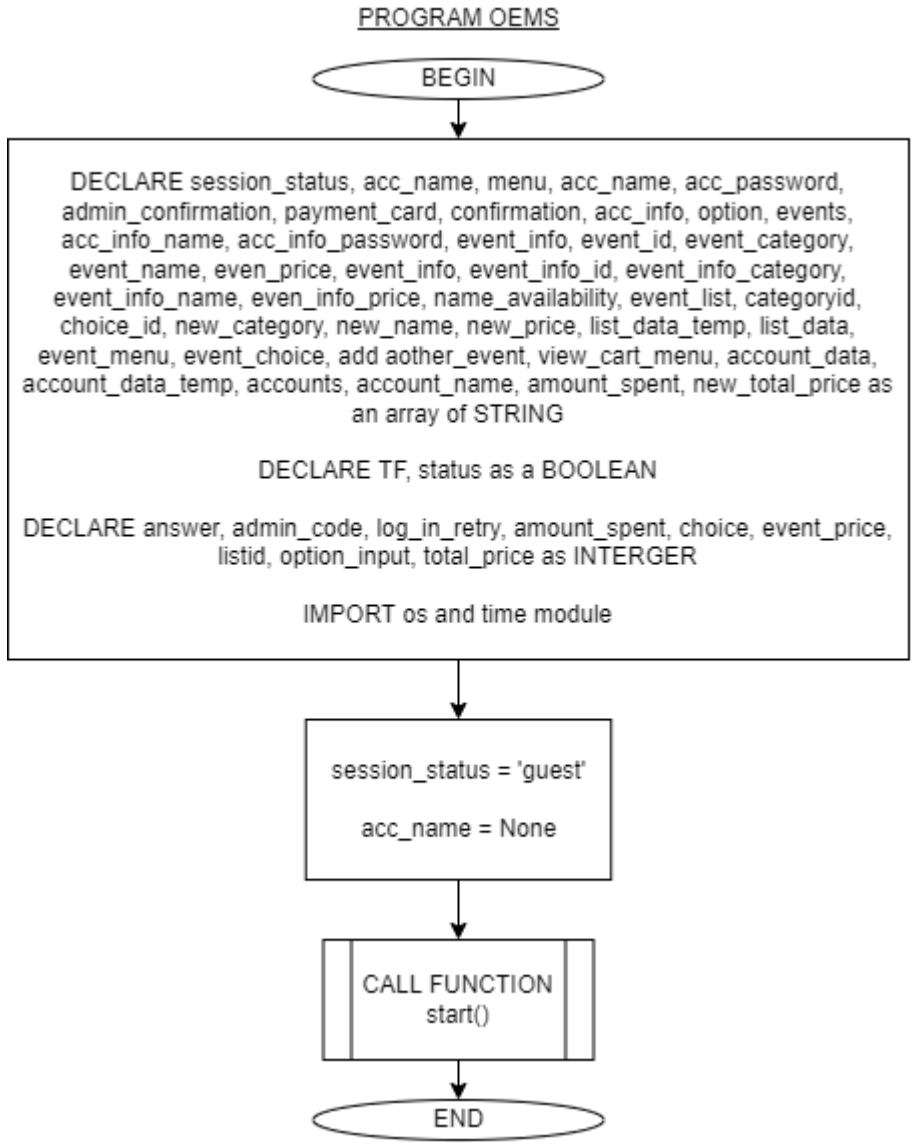
#14 Function for displaying all customer records



#15 Function to search and display specific customer record



#16 Starting the Online Event Management System



Source code & additional features

Start the program

```
import time
import os

session_status = ‘guest’
acc_name = None


***
PLACEHOLDER FOR FUNCTIONS AS THERE ARE ALOT
***


start()
```

Explanation: At the start of the program, the time and os module are imported to give the program more functionality. The time module is only used for its sleep function to give the program some delay so that it does not overwhelm the user with information on the terminal. The os module is only used for its system function with the ‘clear’ value for the use of clearing the terminal so that the program is presented in a more appealing manner. Then, variables, session_status and acc_name are created and given the default values of ‘guest’ and ‘None’, as these variables are important for multiple functions in the program. Then after listing out all the functions, the start function is called to officially start the program and can be used by the user.

Functions

#1 start() function

```
def start():
    clear()
    print('OEMS is loading...')
    cartfile = open('cart.txt', 'w', 1)  #used to clear the cart, as the cart works on a per session basis
    cartfile.close()
    time.sleep(3)
    print('Done!')
    time.sleep(1)
    main_menu()
    return
```

Explanation: This function is used to initiate the program. Some of the things this function does is to clear the cart file from the previous session due to the cart's per session design where the cart is cleared every new session. Besides that, the function prints outputs as a form of interaction with the user. And at the end the function moves on by calling the main menu function.

#2 clear() function

```
def clear():
    if os.name == 'nt':
        os.system('cls')
    else:
        os.system('clear')
```

Explanation: This function is used to clear the terminal so that the program won't look too cluttered and messy, it is mainly for aesthetic purposes.

#3 main_menu() function

```
def main_menu():
    clear()
    time.sleep(0.75)

    TF = True

    #checks if session status is guest
    if session_status == 'guest':

        menu ='''Welcome to OEMS, The Online Event Management System!
What would you like to do?

1. Log In
2. Register An Account
3. View Event Information
4. Exit

Choice: '''

        #execute user's choice
        while TF == True:
            try:
                answer = int(input(menu))
            except:
                print('Invalid option, please try again.')
                time.sleep(0.75)
                clear()
                continue

            if answer == 1:
                log_in()
                return
            elif answer == 2:
                acc_register()
                return
            elif answer == 3:
                list_event_menu()
                return
            elif answer == 4:
                clear()
                quit()
            else:
                print('Invalid option, please try again.')
                time.sleep(0.75)
```

```
            clear()
            continue



    #checks if session status to "admin"
    elif session_status == 'admin':

        print(f'Welcome to the OEMS admin menu, {acc_name}')

        menu ='''What would you like to do?

1. Add New Event
2. Modify Event
3. View Event Information
4. Customer Records
5. Logout
6. Exit

Choice: '''

        #execute user's choice
        while TF == True:
            try:
                answer = int(input(menu))
            except:
                print('Invalid option, please try again.')
                time.sleep(0.75)
                clear()
                continue
            if answer == 1:
                add_event()
                return
            elif answer == 2:
                modify_event()
                return
            elif answer == 3:
                list_event_menu()
                return
            elif answer == 4:
                customer_records()
                return
            elif answer == 5:
                log_out()
                return
            elif answer == 6:
```

```
            clear()
            quit()
        else:
            print('Invalid option, please try again.')
            time.sleep(0.75)
            clear()
            continue



    #checks if session status to "registered"
    elif session_status == 'registered':

        print(f'Welcome back {acc_name}!')

        menu ='''What would you like to do?

1. View Event Information
2. Add Events to cart
3. View Cart
4. Log Out
5. Exit

Attention: Please note that cart items will only remain for this session only. Cart items will be deleted on exit.

Choice: '''

        #executes user's choice
        while TF == True:
            try:
                answer = int(input(menu))
            except:
                print('Invalid option, please try again.')
                time.sleep(0.75)
                clear()
                continue
            if answer == 1:
                list_event_menu()
                return
            elif answer == 2:
                cart()
                return
            elif answer == 3:
                view_cart()
                return
```

```
        elif answer == 4:
            log_out()
            return
        elif answer == 5:
            clear()
            quit()
        else:
            print('Invalid option, please try again.')
            time.sleep(0.75)
            clear()
            continue
```

Explanation: This function is made to help show users the main menu of the program for the ease of access to other functions in the program. The function will check the session_status variable and display the main menu according to the session_status, as there are different menus from different account types, which are guest, admin and registered. Once checked, the program will then display the main menu of the account types with different options that lead to the execution of different functions.

#4 log_in() function

```
def log_in():
    clear()
    time.sleep(0.75)
    global session_status
    global acc_name
    TF = True

    #asks user to input a username
    while TF == True:
        info_file = open('account_info.txt', 'r', 1)
        print('OEMS Login')
        acc_name = input('Please enter your username: ')
        if acc_name == "":
            print('No username entered, please try again.')
            time.sleep(0.75)
            clear()
            continue

        for line in info_file:
            acc_info = line.split(',')
            acc_info_name = acc_info[0].strip()
```

```
        acc_info_password = acc_info[1].strip()

        if acc_name == acc_info_name:

            #asks user to input password
            while TF == True:

                acc_password = input('Please enter your password: ')

                if acc_password == "":
                    print('No password entered, please try again.')
                    time.sleep(0.75)
                    continue

                elif acc_password == acc_info_password:
                    clear()
                    time.sleep(0.75)
                    print(f'Logged in successfully. Welcome back {acc_name}.')
                    time.sleep(0.75)

                    while TF == True:

                        #user is required to answer y/n, to confirm if they are/aren't an admin
                        admin_confirmation = input('Are you an admin? (y/n) ')

                        if admin_confirmation == 'y' or admin_confirmation == 'Y':
                            clear()
                            time.sleep(0.75)

                            while TF == True:
                                clear()
                                try:
                                    #if user is an admin, an admin code is required, here "000" is used for
an example
                                    admin_code = int(input('Please enter admin code (use this code for
testing purposes:000) \nCode: '))
                                except:
                                    print('No admin code entered, please try again.')
                                    time.sleep(0.75)
                                    continue

                                #checks the admin code
                                if admin_code == 000:
                                    clear()
                                    time.sleep(0.75)
                                    print('Thank you, redirecting you to the admin menu...')
```

```
                        info_file.close
                        session_status = 'admin'
                        TF = False
                        time.sleep(3)
                        main_menu()
                        return

                    else:
                        clear()
                        time.sleep(0.75)
                        print('Admin code does not exist, please try again.')
                        time.sleep(0.75)
                        continue

                elif admin_confirmation == 'n' or admin_confirmation == 'N':
                    clear()
                    time.sleep(0.75)
                    print('Redirecting you to the menu...')
                    info_file.close
                    session_status = 'registered'
                    TF = False
                    main_menu()
                    return

                else:
                    print('Answer not recognized, please try again.')
                    continue

            else:
                clear()
                time.sleep(0.75)
                print('OEMS Login')
                print('Your password is incorrect, please try again.')
                continue
        else:
            continue
    else:
        clear()
        time.sleep(0.75)

        log_in_retry = ""The entered username does not exist.
What would you like to do?

1. Retry
2. Register new account
3. Main menu
```

```
Choice: '"
        #executes user's choice
        while TF == True:
            try:
                log_in_retry = int(input(log_in_retry))
            except:
                print('Invalid option, please try again.')
                clear()
                time.sleep(0.75)
                continue

            if log_in_retry == 1:
                log_in()
                return

            elif log_in_retry == 2:
                info_file.close
                acc_name = None
                acc_register()
                return

            elif log_in_retry == 3:
                main_menu()
                return

            else:
                print('Invalid option, please try again.')
                time.sleep(0.75)
                continue



    return
```

Explanation: This function helps the user to log in to the program, into either a normal registered account, or an admin account. To log in, the user will be asked to enter their username and password. Once logged in, the user will be asked if they are an admin or not. If yes, they will be asked to enter an admin code and once done, they will be sent to the admin main menu. If no, they will be sent to the main menu of a normal registered account.

#5 log_out() function

```
def log_out():
    clear()
    time.sleep(0.75)
    global session_status
    global acc_name
    session_status = 'guest'
    acc_name = None
    print('Logging out...')
    time.sleep(3)
    main_menu()
    return
```

Explanation: This function helps with logging out the user from a logged in session back into a guest session. It mainly just sets the session_status variable as well as the acc_name variable back to default and then call the main_menu function to go back to the main menu. It also prints outputs just for some interactivity.

#6 acc_register() function

```
def acc_register():

    TF = True
    clear()
    time.sleep(0.75)
    amount_spent = 0

    while TF == True:

        status = True
        print('OEMS Account Registration\n')
        acc_name = input('Please enter your username: ')
            #ask for username
        if acc_name == "":
            print('No username entered, please enter a username.')
            time.sleep(0.75)
            clear()
            continue

        fhandler = open ('account_info.txt','r')
        for line in fhandler:
            if line.startswith(acc_name):
```

```
            status = False
            break

    if status == False:
        clear()
        print("This username already exists, please use another username.")
        time.sleep(3)
        clear()
        continue
        #checking name availability

    while TF == True:
        #asks user for credit/debit card info for payment purposes
        payment_card = input('Please enter your credit/debit card number [**** **** ****
****]: ')

        if payment_card == '':
            print('No credit/debit card entered, please enter your credit/debit card number.')
            time.sleep(0.75)
            continue
        break

    while TF == True:
        #asks user to enter password
        acc_password = input('Please enter your password: ')

        if acc_password == "":
            print('No password entered, please enter your password.')
            time.sleep(0.75)
            continue
        break

    while TF == True:
        #asks user to confirm their password
        confirmation = input('Please confirm your password: ')

        if confirmation == "":
            print('No password entered, please enter your password.')
            time.sleep(0.75)
            continue
        break



    if acc_password != confirmation:
        clear()
```

```
            time.sleep(0.75)
            print("Incorrect password, please try again.")
            continue
            #password confirmation

        elif acc_password == confirmation:
            clear()
            time.sleep(0.75)
            acc_info = [acc_name, acc_password, payment_card, amount_spent]
            file = open('account_info.txt', 'a',1)
            file.write (str(acc_info).strip('[]').replace("'", '') + '\n')
            file.close
            print('Your account has been successfully created.')
            time.sleep(3)
            break
            #account registered

    option = '''What would you like to do?

1. Main Menu
2. Log In
3. Exit

Choice: '''
    #options for user
    clear()
    time.sleep(0.75)
    #executes user's choice
    while TF == True:
        try:
            choice = int(input(option))
        except:
            print('Invalid option, please try again.')
            time.sleep(0.75)
            clear()
            continue

        if choice == 1:
            TF = False
            main_menu()
            return
        elif choice == 2:
            TF = False
            log_in()
            return
        elif choice == 3:
```

```
            clear()
            quit()
        else:
            print("Invalid option, please try again.")
            continue
```

Explanation: This function helps users register an account for the program so they can get the benefits that come with a registered account. To register the account, the function will first ask the user for a username, and if the username already exists, it will ask the user to try again. Then the user would have to enter their credit or debit card number for any purchasing purposes. After that, the user will be asked for their password and confirm it to finish the registration, once done, their account will be added to the accounts information file and a menu will ask the user for their next action.

#7 category() function

```
def category():
    TF = True
    clear()
    time.sleep(0.75)
    events = '''Events Categories Available:

1. Sports
2. E-Sports
3. Technology
4. Art
5. General Entertainment
6. Back to Main Menu

Choice: '''

    #executes user's choice
    while TF == True:
        try:
            answer = int(input(events))
        except:
            print('Invalid option, please try again.')
            time.sleep(0.75)
            clear()
            continue

        if answer <= 6:
```

```
            if answer == 6:
                TF = False
                main_menu()
                return
            else:
                return answer

        else:
            clear()
            time.sleep(0.75)
            print("Invalid option, please try again.")
            time.sleep(0.75)
            clear()
            continue
```

Explanation: This function helps with displaying all the existing categories and allows the user to choose their desired category. It also has a “Back to Main Menu” option just in case the user changes their mind.

#8 add_event() function

```
def add_event():
    TF = True
    choice = category()

    if choice == 1:
        categoryid = 'Sports'
    elif choice == 2:
        categoryid = 'E-Sports'
    elif choice == 3:
        categoryid = 'Technology'
    elif choice == 4:
        categoryid = 'Art'
    elif choice == 5:
        categoryid = 'General Entertainment'
    #setting category based on user's input

    while TF == True:
        status = True

        #asks for the name of event
        event_name = input('Please enter the event name: ')
        if event_name == "":
            print('No event name entered, please enter an event name.')
```

```
        time.sleep(0.75)
        continue

    fhandler = open('event.txt','r')
    for line in fhandler:
            event_info = line.split(',')
            name_availability = event_info[2].strip()

            if name_availability == event_name:
                status = False

    #checks for the availability of event name
    if status == False:
        print("Event exists please try again")
        continue

    #asks user price of event
    while TF == True:
        try:
            event_price = int(input('How much is the event?(RM): '))
            if event_price == '' or event_price == 0:
                print('No price entered, please enter price.')
                time.sleep(0.75)
                continue
            break
        except:
            print('No price entered, please enter a price.')
            time.sleep(0.75)
            continue
    #asking for name and price


    #adds the information of event to file
    listid=1
    fhandler = open ('event.txt','r')
    for line in fhandler:
        if line.endswith("\n"):
            listid +=1

    event_list = [listid, categoryid, event_name, event_price]
    fhandler = open ('event.txt','a',1)
    fhandler.write (str(event_list).strip('[]').replace("'", '') + '\n')
    fhandler.close
    #appending what was input into text file
    clear()
    time.sleep(0.75)
```

```
        print('Your event has been added.')
        time.sleep(3)
        TF == False
        main_menu() #once done, sends user back to main menu
        return
    return
```

Explanation: This function helps admins to add new events to the list of events. Firstly, it will call the category function to display all categories and lets the user choose which category in which the new event will be added to. Then the function will ask the user for the event name and check the availability of the name. If there are no problems, the user will then be asked to enter the price of the event. Once that's done, the event will be added to the events file and returns the user back to the main menu.

#9 modify_event() function

```
def modify_event():
    event_list()
    print('\n')
    TF = True

    while TF == True:

        #asks user the ID of event which they would likke to modify
        choice_id = input("Which event would you like to modify?[ID]: ")

        if choice_id == "":
            print('No event ID entered, please enter an event ID.')
            time.sleep(0.75)
            continue

        else:
            break

    #reads information of selected event
    fhandler_read = open('event.txt','r', 1)

    for line in fhandler_read:
        event_info = line.split(',')
        event_id = event_info[0].strip()
        event_category = event_info[1].strip()
        event_name = event_info[2].strip()
```

```
        event_price =  event_info[3].strip()

        clear()
        time.sleep(0.75)
        option = '''Options available:

1. Change event category
2. Change event name
3. Change event price
4. Delete event
5. Cancel

Choice: '''
        if event_id == choice_id:

            while TF == True:
                try:
                    option_input = int(input(option))

                except:
                    print('Invalid option, please try again.')
                    time.sleep(0.75)
                    clear()
                    continue

                #category of event is to be modified, lists all categories for user to select
                if option_input == 1:
                    choice = category()

                    if choice == 1:
                        new_category = 'Sports'
                        break
                    elif choice == 2:
                        new_category = 'E-Sports'
                        break
                    elif choice == 3:
                        new_category = 'Technology'
                        break
                    elif choice == 4:
                        new_category = 'Art'
                        break
                    elif choice == 5:
                        new_category = 'General Entertainment'
                        break

                #name of event is to be modified, asks user to input new name for the event
```

```
            elif option_input == 2:
                while TF == True:
                    new_name = input('Please enter new event name: ')
                    if new_name == "":
                        print('No new event name entered, please enter a new event name.')
                        time.sleep(0.75)
                        continue
                    else:
                        break

            #price of event is to be modified, asks user to input new price for the event
            elif option_input == 3:
                while TF == True:
                    new_price = (input('Please enter new price[RM]: '))
                    if new_price == '' or new_price == '0':
                        print('No new price entered, please enter new price.')
                        time.sleep(0.75)
                        continue
                    else:
                        break

            elif option_input == 4:
                break

            #cancel modification and return to main menu
            elif option_input == 5:
                main_menu()
                return

            else:
                print('Invalid option, please try again.')
                time.sleep(0.75)
                clear()
                continue
        break

fhandler_read.close()

#assigns new changes to file
with open('event.txt') as fhandler_read:

    list_data_temp = []
    list_data = fhandler_read.readlines()
    for line in list_data:

        if line.startswith(choice_id):
```

```
            if option_input == 1:
                line = line.replace(event_category, new_category)
            elif option_input == 2:
                line = line.replace(event_name, new_name)
            elif option_input == 3:
                line = line.replace(event_price, new_price)
            elif option_input == 4:
                line = ""
        list_data_temp.append(line)

    with open('event.txt', 'w') as fhandler_write:
        for line in list_data_temp:
            fhandler_write.write(line)

    clear()
    time.sleep(0.75)
    print("Modified complete, redirecting to main menu.....")
    time.sleep(3)
    main_menu() #sends users back to main menu when done
    return
```

Explanation: This function helps admins to modify or delete the events that were previously added. The function will ask the user to choose which event they would like to modify. Once chosen, the function will then ask the user what they would like to do with the event, which includes changing the event category, the event name, the event price, as well as an option to delete the event entirely. Once chosen, the function will then proceed with the user's choice and if needed, ask the user to enter new information. Once done, the function will write the new information into the events file and send the user back to the main menu.

#10 event_list() function

```
def event_list():
    choice = category()
    clear()
    time.sleep(0.75)

    event_file = open('event.txt', 'r')

    #checks which category has been selected by user
    if choice == 1:
        categoryid = 'Sports'
    elif choice == 2:
        categoryid = 'E-Sports'
    elif choice == 3:
        categoryid = 'Technology'
    elif choice == 4:
        categoryid = 'Art'
    elif choice == 5:
        categoryid = 'General Entertainment'



    print('Category:',categoryid,'\n')

    for line in event_file:
        event_info = line.split(',')
        event_info_id = event_info[0].strip()
        event_info_category = event_info[1].strip()
        event_info_name = event_info[2].strip()
        event_info_price = event_info[3].strip()

        #prints out information of events under selected category
        if categoryid == event_info_category:
            print(f'ID:{event_info_id}          Event:{event_info_name}
Price:RM{event_info_price}')
```

Explanation: This function helps with listing all events that are under a specific category chosen by the user. The category function is first called to display all categories for the user to choose, then the function will read the events file and only print the events with the user's chosen category, which prints the event's id, name and price.

#11 list_event_menu() function

```
def list_event_menu():
    event_list()


    TF = True

    event_menu = "'What would you like to do?

1.Back to category menu
2.Back to main menu

Choice: "'

    #executes user's choice
    while TF == True:
        print('\n')
        try:
            choice = int(input(event_menu))
        except:
            print('Invalid choice, please try again.')
            time.sleep(0.75)
            continue

        if choice == 1:
            list_event_menu()
            return
        elif choice == 2:
            TF = False
            main_menu()
            return
        else:
            print('Invalid choice, please try again.')
            time.sleep(0.75)
            continue


    return
```

Explanation: This function shows a menu for the user's next action after the event_list function.

#12 cart() function

```
def cart():
    clear()
    time.sleep(0.75)
    event_list()
    print('\n')
    TF = True

    #asks which event user would like to add to their cart
    while TF == True:

        event_choice = input("Which event[ID] would you like to add to cart? (Type 'n' to
cancel): ")
        if event_choice == "":
            print('No event ID entered, please enter an event ID.')
            time.sleep(0.75)
            continue

        with open('event.txt', 'r') as event_file:

            event_file_read = event_file.readlines()

            for item in event_file_read:
                events = item.split(',')

                event_id = events[0].strip()
                event_category = events[1].strip()
                event_name = events[2].strip()
                event_price = events[3].strip()

                #writes information into cart file
                if event_choice == event_id:
                    with open('cart.txt', 'a', 1) as cart_file:
                        event_info = [event_id, event_category, event_name, event_price]
                        cart_file.write (str(event_info).strip('[]').replace("'", '') + '\n')
                        clear()
                        time.sleep(0.75)

                        #asks if user would like to add other events to cart
                        add_another_event = input('Event successfully added to cart, would you like to
add another event? (y/n): ')


                        if add_another_event == 'y' or add_another_event == 'Y':
                            event_list()
```

```
                    continue
                elif add_another_event == 'n' or add_another_event == 'N':
                    clear()
                    time.sleep(0.75)
                    print('Sending you to main menu...')
                    time.sleep(3)
                    main_menu()
                    return
                else:
                    print('Invalid option, please try again.')
                    time.sleep(0.75)
                    continue

            elif event_choice == 'n' or event_choice == 'N':
                TF = False
                print('Aborting...Sending you to main menu...')
                time.sleep(3)
                main_menu()
                return

            else:
                print('The event ID you entered does not exist, please try again.')
                time.sleep(0.75)
                break
        continue

    return
```

Explanation: This function helps user's add events they want into their cart. Firstly, the event_list function will be called, which reads the events file, to show the events to the user, and once chosen, the function will write the chosen event's information into the cart file, the user also has an option to cancel adding item to cart if they change their mind. The function then will ask the user if they would like to add another event. If yes, the user will be sent back to the start and repeat the cart function. If no, the user will be sent back to the main menu.

#13 view_cart() function

```
def view_cart():
    global acc_name
    clear()
    time.sleep(0.75)
    TF = True
```

```
    #reads and shows the cart information
    try:
        with open('cart.txt') as cart_file:
            cart_file_read = cart_file.readlines()
            total_price = 0

            print(f"{acc_name}'s cart.\n")


            for item in cart_file_read:
                events = item.split(',')
                event_id = events[0].strip()
                event_category = events[1].strip()
                event_name = events[2].strip()
                event_price = events[3].strip()
                total_price = total_price + int(event_price)

                print(f'ID:{event_id}          Category:{event_category}          Event:{event_name}
Price:RM{event_price}')

    except:
        clear()
        time.sleep(0.75)
        print('No records in cart, redirecting to main menu...')
        time.sleep(3)
        main_menu()
        return



    print('\n')
    print(f"Total Price: RM{total_price}")
    print('\n')

    view_cart_menu = "'What would you like to do?

1.Proceed to checkout
2.Back to main menu

Choice: "'

    #executes user's choice
    while TF == True:
        try:
            answer = int(input(view_cart_menu))
```

```
            break
        except:
            print('Invalid option, please try again.')
            time.sleep(0.75)
            continue

    if answer == 1:

        while TF == True:

            #confirms with user if they are certain to checkout
            confirmation = input('Are you sure you would like to checkout all items? (y/n): ')


            if confirmation == 'y' or confirmation == 'Y':
                print('Processing...')


                #reads and calculates required information
                with open('account_info.txt') as fhandler_read:

                    account_data_temp = []
                    account_data = fhandler_read.readlines()
                    for account in account_data:
                        accounts = account.split(',')
                        account_name = accounts[0].strip()
                        amount_spent = accounts[3].strip()
                        new_total_price = str(total_price + int(amount_spent))

                        if account_name == acc_name:
                            account = account.replace(amount_spent, new_total_price)

                        account_data_temp.append(account)


                #writes new information into user's account
                with open('account_info.txt', 'w', 1) as fhandler_write:
                    for account in account_data_temp:
                        fhandler_write.write(account)

                #clears the cart file after checkout
                cartfile = open('cart.txt', 'w', 1)
                cartfile.close()

                clear()
                time.sleep(0.75)
```

```
            print('Purchase complete, the receipt will be sent to you by the end of the month, Thank you!')
            time.sleep(3)
            clear()
            print('Redirecting you to the main menu...')
            time.sleep(3)
            main_menu() #sends user back to main menu when done
            return

        elif confirmation == 'n' or confirmation == 'N':
            view_cart()
            return

        else:
            print('Invalid option, please try again.')
            time.sleep(0.75)
            continue



    elif answer == 2:
        print('Sending you to main menu...')
        time.sleep(3)
        main_menu()
        return
    return
```

Explanation: This function allows the user to view the items they have added to their cart by reading the cart file. It also gives them the option to checkout their cart. If the user chooses to checkout, a confirmation will confirm the user's choice, and after that, the items in the cart will be purchased using the user's credit/debit card information they provided during registration, and the total amount will be updated to the user's account details in the account information file. Once the checkout process is complete, the cart file will be cleared and return the user back to the main menu.

#14 customer_records() function

```
def customer_records():
    clear()
    time.sleep(0.75)
    TF = True
    print('Customer records\n')
```

```
    #reads the account information file
    with open('account_info.txt') as fhandler:
        accounts = fhandler.readlines()
        for item in accounts:
            account = item.split(',')
            account_name = account[0].strip()
            ammount_spent = account[2].strip()

            #prints information of all registered accounts
            print(f'Username:{account_name}          Ammount spent:{ammount_spent}')

    menu = "'What would you like to do?

1.Back to main menu

Choice: "'

    #executes user's choice
    while TF == True:
        try:
            option = int(input(menu))
        except:
            print('Invalid option, please try again.')
            time.sleep(0.75)
            continue

        if option == 1:
            clear()
            time.sleep(0.75)
            print('Sending you back to the main menu...')
            time.sleep(3)
            TF = False
            main_menu()
            return

        else:
            print('\n')
            print('Invalid option, please try again.')
            time.sleep(0.75)
            continue
```

Explanation: This function allows admins to view information about all registered accounts, such as their username and the amount spent on the account.

#15 specific_customer_records() function

```
def specific_customer_records():
    TF = True

    #ask for account name
    while TF == True:
        acc_name = input("Please enter account's username: ")
        clear()
        time.sleep (0.75)

        with open ('account_info.txt') as fhandler:
            accounts = fhandler.readlines()
            for item in accounts:
                account = item.split(',')
                account_name = account[0].strip()
                ammount_spent = account[3].strip()
                if account_name == acc_name:
                    print(f"{account_name}'s records\n")
                    print(f'Username:{account_name}          Ammount spent:{ammount_spent}\n')
                    break
            else:
                print("Invalid username please try again")
                time.sleep(3)
                clear()
                time.sleep(0.75)
                continue
            break
    #read every lines to find corresponding name then print line

    menu = "'What would you like to do?

1.Back to main menu

Choice: "'

    #executes user's choice
    while TF == True:
        try:
            option = int(input(menu))
        except:
            print('Invalid option, please try again.')
            time.sleep(0.75)
            continue

        if option == 1:
            clear()
```

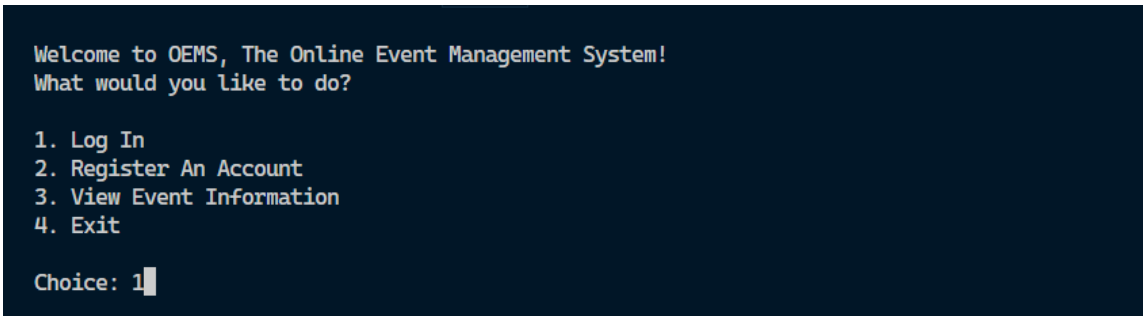
```
            time.sleep(0.75)
            print('Sending you back to the main menu...')
            time.sleep(3)
            TF = False
            main_menu()
            return

        else:
            print('\n')
            print('Invalid option, please try again.')
            time.sleep(0.75)
            continue
```

Explanation: This function allows the admin to search for specific customer record by entering their username.

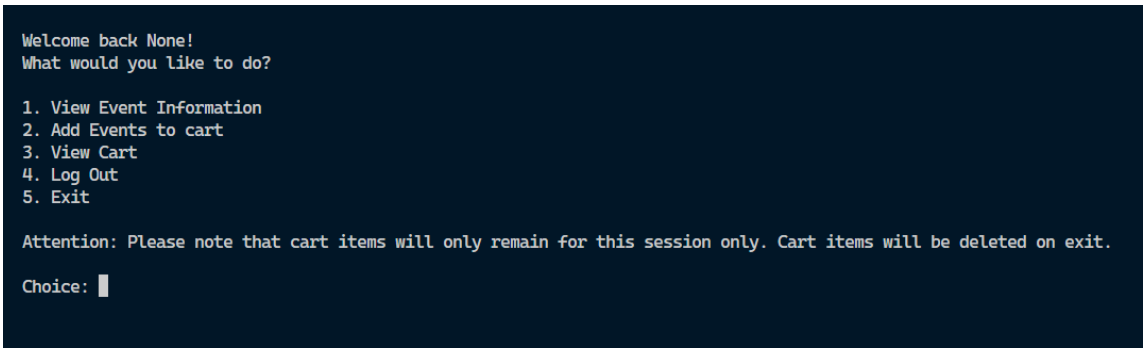
Sample Input/Output in OEMS

#1 Interface for main menu

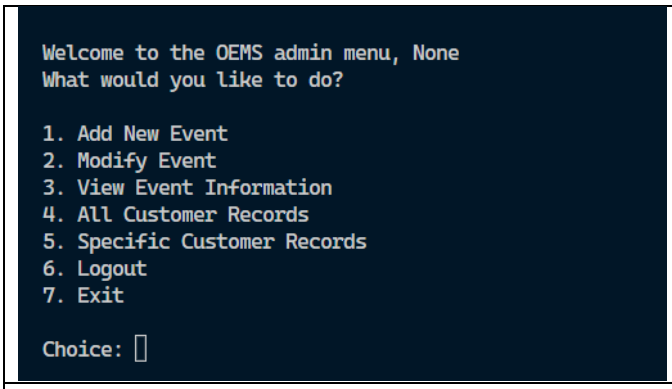
Guest menu:



Registered menu:

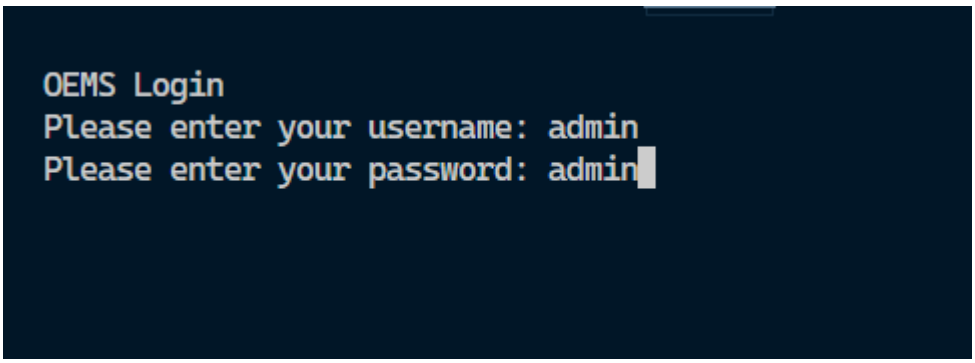


Admin menu:



Explanation: The main menu interface allows the user to choose what they would like to do; the example above are the main menus for the guest, registered and admin sessions. It first greets the user and asks them what they would like to do, by listing out the options, then asks them to input their choice.

#2 Interface for login



Explanation: Login interface allows the user to input their username and password. They will be logged in once the correct details are entered.

#3 Interface when adding an event

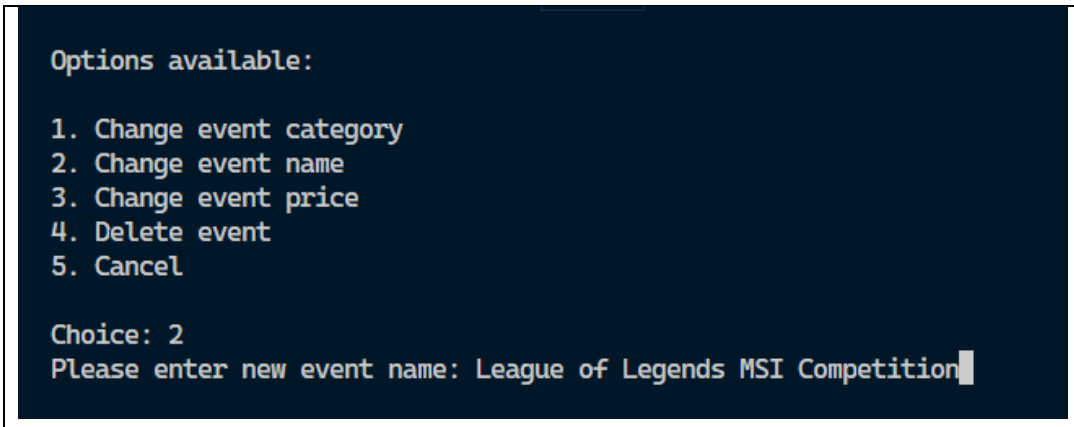
```
Events Categories Available:

1. Sports
2. E-Sports
3. Technology
4. Art
5. General Entertainment
6. Back to Main Menu

Choice: 2
Please enter the event name: League of Legends
How much is the event?(RM): 200
```

Explanation: When adding a new event, admin is allowed to choose the category of the event they want to add then the system will ask for the event name and price. Once all the details are entered, the event information will be saved in a text file.

#4 Interface when modifying event

```
Events Categories Available:

1. Sports
2. E-Sports
3. Technology
4. Art
5. General Entertainment
6. Back to Main Menu

Choice: 1
```

```
Category: E-Sports

ID:2          Event:Apex Legends Semi-finals          Price:RM120
ID:7          Event:Minecraft Building Competition          Price:RM150
ID:11          Event:League of Legends          Price:RM200


Which event would you like to modify?[ID]: 11
```



Explanation: When modifying an event, the admin will be asked to choose the category the event is under, then the list of events will be shown for the admin to choose. Once chosen, the admin will be asked to choose a modify option and enter new information. Once done, the new information will be written to the text file.

#5 Interface when choosing event category

Explanation: The category interface shows all existing categories for the user to choose.

#6 Interface for listing all events in a category

```
Category: Sports

ID:1          Event:Badminton          Price:RM100
ID:6          Event:Basketball          Price:RM110


What would you like to do?

1.Back to category menu
2.Back to main menu

Choice: 1
```

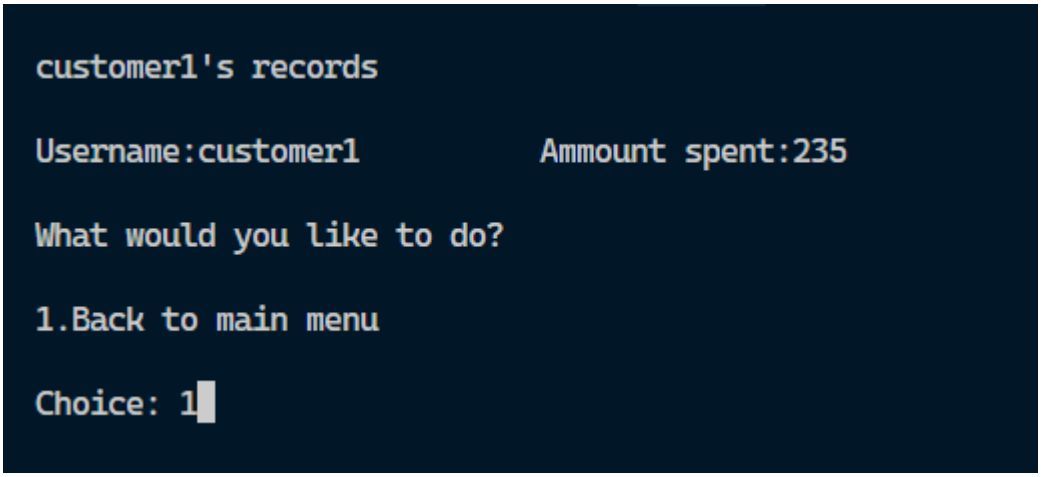
Explanation: This interface lists all events in a specific category and asks what the user would like to do next.

#7 Interface for showing all customer records

```
Customer records

Username:admin          Ammount spent:0
Username:customer1          Ammount spent:235
Username:customer2          Ammount spent:0


What would you like to do?

1.Back to main menu

Choice: 1
```

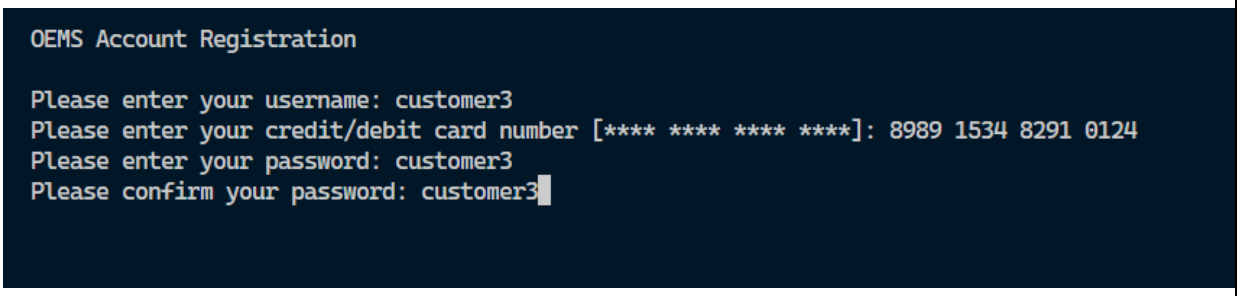
Explanation: This interface lists out all registered accounts in the system, then asks what the user would like to do.

#8 Interface that shows specific customer's records



Explanation: This interface lists information about a specific customer after inputting the customer's username, the asks what the user would like to do.

#9 Interface for account registration



Explanation: The account registration guides users through the registration process, by first asking for their username, then credit/debit card number for payment purposes, then lastly their password and confirm it.

#10 Interface for adding events to cart

```
Category: Sports

ID:1          Event:Badminton          Price:RM100
ID:6          Event:Basketball          Price:RM110


Which event[ID] would you like to add to cart? (Type 'n' to cancel): 1
```

Explanation: This interface first lists out all events under the chosen category, then asks which event the user would like to add to cart by entering the event's ID. They also have the option to cancel by inputting 'n'.

#11 Interface for viewing cart and payment

```
None's cart.

ID:6          Category:Sports          Event:Basketball          Price:RM110
ID:1          Category:Sports          Event:Badminton          Price:RM100
ID:2          Category:E-Sports          Event:Apex Legends Semi-finals          Price:RM120


Total Price: RM330


What would you like to do?

1.Proceed to checkout
2.Back to main menu

Choice: 1
Are you sure you would like to checkout all items? (y/n): y
```

```
Purchase complete, the receipt will be sent to you by the end of the month, Thank you!
```

Explanation: This interface first shows all items that are in the user's cart, and it shows the total price of all the events. Then the user would be asked what they would like to do, which include the options to checkout or go back to main menu. If they choose to checkout, a confirmation will be needed to confirm the purchase. Once confirmed, the credit/debit card information

provided during registration will be used for payment and an output will notify the user about the receipt.

Conclusion

The OEMS program will help AEMS in managing their online business as this program consists of all necessary functions and additional features requested by the event agency which allow the users to manage events freely and easily. Even though the program is working fine most of the time, there may still be some bugs here and there which are yet to be debugged as the program is not 100% perfect, however users are most likely going to have a good experience in using the program. Overall, the project to create this program has been a very educational experience and will be very helpful in the future.