CRTICAL THINKING- APPLIED PROJECT

NAME: JENNIFER AROKIADOSS

PROJECT: DESIGN THE AUTOMATED TRAIN ALGORITHM FUNCTIONALITY

The purpose of this applied project is to develop the algorithms (artificial intelligence) to operate the automatic train that can effectively detect and scans the specific features of the passenger. It ensures it does what it is designed to do safely and successfully. It also focus on the overall algorithm including some of the features such as face characteristics like eye, hair etc, the trip distances, and use modules where possible.

FUNCTIONALITY: AUTOMATIC FEE DEDUCTION

```
PassengerAccount
   Passenger_info()
   Passenger_scan(eye_color,face_length, DB)
   Passenger_trip(TripSelection,passengerFee_account, fee_message)

END

Passengerinfo()
      PassengerName = GetPassengerName("Type your Name:  ")
      return(PassengerName)
END

Passenger_scan(face_color,face_length, DB)
  //Use  the camera to take a photo:
    Turn on camera
     passenger_photo = take photo with the camera
  //Scan the following to check the person is rightone
     face_color = face_color_in_database
     face_length = face_length_in_database
  //Open the passenger database
```

```
    DB = OPEN www.metro.com.au/passengers.csv
        DOWHILE(READ PassangerDatabase from DB) is successfully verified
            IF  face_color && face_length = DB THEN
                PRINT "Account successfully verified"
            ELSE PRINT "Repeat passenger_photo"
             ENDIF
        ENDDO
END
```

```
Passenger_trip(TripSelection, passengerFee_account, fee_message)
  select trip = GetName("Please select a Trip from the following : ")
  Trip_Fee = x
  passengerFee_account = fee_message

        IF Trip < 10 km:
           DISPLAY("x is 3$")
           Detect x from  passengerFee_account
        ENDIF

        IF Trip >= 10 km, but < 80 km:
           PRINT("x is 10$")
           Detect x from  passengerFee_account
         ENDIF

        IF Trip >= than 80 km :
           PRINT("x is 15$")
           Detect x from  passengerFee_account
        ELSEIF Max trip is 130km
        ENDIF

        //Printing fee message if needed
        IF passenger = fee_message THEN
            Print passengerFee_account
         ELSEIF "Exit"
        ENDIF
END
```

- **<u>DEFINING DIAGRAM FOR TASK 1</u>**

INPUT	PROCESSING	OUTPUT
Name	• Prompt for and Get Passenger Name.	Fee detection
Face Color & Face Length	• Scan the features with photo. • Read through database	
Trip selection	• Calculate distance • Display fee accordingly • Detect fee from Account • Print fee message if needed	

❖ FUNCTIONALITY: ENTRY & TRAVEL & EXIT

```
Passenger_travel()
   Passenger_entry()
   Train_speed(speed, location , distance )
   Train_door_opening(speed, reached_destination)
   Train_door_closing( Time, reached_destination)
   Passenger_scan(face_color,face_length, DB)
   Process_Trip(total_trip, distance_of_trip, passenger_DB)
END

Passenger_entry()
    train_station = PRINT("Type your route option: ")
     Read route
     Route 1 = Distance >= 10km && < 80km
     Route 2 = Distance >= 80km
     Route 3 = Distance > 80km && <= 130km
          IF Route1:THEN
             DISPLAY("Platform No: 1")
          ENDIF

          IF Route2:THEN
             DISPLAY("Platform No: 2")
          ENDIF

         IF Route 3:THEN
            DISPLAY("Platform No: 3")
          ENDIF
   END

//speed of the train
```

```
Train_speed(speed, location , distance )
  N= arrived destination
//setting max distance using GPS as 20km to reach destination
    CASE N
        1: speed = distance & location = 0km DISPLAY 'Start'
        2: speed = distance & location >= 50km DISPLAY 'Fast'
        3: speed = distance & location <= 10km DISPLAY ‘Slow'
        4: speed = distance & location == N DISPLAY ‘Stop'
    ENDCASE
END



Passenger_Board(Counting_sensor,Passenger_intial_count)
    DO Passenger_intial_count  1 to 1000 USING Counting_sensor
       PRINT Passenger_intial_count
    ENDDO
END


Passenger_Leaving(Counting_sensor,Passenger_exit_count)
    DO Passenger_exit_count 1000 to 0 USING Counting_sensor
       PRINT Passenger_exit_count
    ENDDO
END

Train_door_opening(speed, reached_destination)
    T = 60 secs
       WHILE  speed = 0km && reached_destination DO
          Open_door = T
       ENDWHILE
END

Train_door_closing( Time, reached_destination)
    D ="door is closing"
     T = 60 secs
     speed = 0
        WHILE reached_destination + open_door = T DO
            Close_door = D THEN
            speed = speed + 1
        ENDWHILE
END

Passenger_scan(face_color,face_length, DB)
  //Use  the camera to take a photo:
```

```
    Turn on camera
    passenger_photo = take photo with the camera
 //Scan the following to check the person is right one
    face_color = face_color_in_database
    face_length = face_length_in_database
  //Open the passenger database
     DB = OPEN www.metro.com.au/passengers.csv
        DOWHILE(READ PassengerDatabase from DB) is successfully verified
             IF  face_color && face_length = DB
                 PRINT "Account successfully verified"
             ELSE PRINT "Repeat passenger_photo"
             ENDIF
       ENDDO
END

Process_Trip(total_trip, distance_of_trip, passanger_DB)
  SET total_trip to Zero
  GET distance_of_trip
     IF Passenger_Fee_detection = distance_of_trip THEN
        GET Verificatied = “Succesful”
         add total_trip to passanger_DB
     ELSE calculate_newfee with distance_of_trip
          PRINT  New_PassengerFee
          GET Verificatied = “Succesful”
          add total_trip to passenger_DB
     ENDIF
END
```