Proyecto Metro de Quito

Brandon Isaac Arellano Calderón[L00412284]

Universidad de las Fuerzas Armadas
biarellano@espe.edu.ec

Abstract.

En este ensayo, se presenta un programa informático para el metro de Quito, con el objetivo de mejorar la eficiencia y la experiencia de los pasajeros en el uso del servicio. El programa se basa en el lenguaje de programación Python para brindar una variedad de funciones, incluyendo el ingreso de saldo y conteo de viajes para los usuarios, y la gestión de trenes y chóferes para el personal administrativo.

1 Introducción

El objetivo general de este ensayo es proponer un programa informático para el metro de Quito que mejore la eficiencia y la experiencia de los pasajeros en el uso del servicio. Los objetivos específicos son: (1) presentar las funciones específicas del programa, tales como el ingreso de saldo, conteo de viajes, verificación de trenes, validación de códigos de entrada, verificación de entrada y salida de trenes, cronometración de viajes de los trenes y la asignación de chóferes a los trenes. (2) discutir cómo el programa ayudará a mejorar la gestión del servicio de metro y la experiencia del usuario, al utilizar tecnología de Python para analizar y proporcionar información precisa y actualizada sobre el servicio de metro y para identificar y resolver problemas en el servicio de manera rápida.

2 Desarrollo

El programa informático propuesto se basa en el lenguaje de programación Python y tiene 8 secciones diferentes. Para los usuarios, se incluye una sección para el ingreso de saldo y una sección para contar los viajes. En la sección de ingreso de saldo, los usuarios podrán cargar dinero en sus tarjetas de metro, lo que les permitirá pagar sus viajes de manera más rápida y cómoda. En la sección de conteo de viajes, los usuarios podrán verificar el número de viajes que han realizado y el saldo restante en sus tarjetas de metro.

Para el personal administrativo y los usuarios, se incluye una sección para el ingreso de usuarios y contraseñas. En esta sección, los usuarios podrán crear y gestionar sus cuentas de metro, mientras que el personal administrativo podrá

gestionar las cuentas de los usuarios y controlar el acceso a las diferentes funciones del programa.

Para la parte administrativa, el programa incluye funciones como la verificación de trenes, la validación de códigos de entrada, la verificación de entrada y salida de trenes, la cronometración de viajes de los trenes y la asignación de chóferes a los trenes. En la sección de verificación de trenes, el personal administrativo podrá verificar el vin de cada tren que se encuentra en vigor y uso del día. En la sección de validación de códigos de entrada, se podrá verificar que los usuarios estén ingresando códigos válidos al entrar al metro. En la sección de verificación de entrada y salida de trenes, se podrá controlar el número de pasajeros que entran y salen de cada tren. En la sección de cronometración de viajes de los trenes, se podrá medir el tiempo que tarda cada tren en completar un viaje, permitiendo un mejor control del servicio. En la sección de asignación de chóferes a los trenes, se podrá asignar chóferes a los trenes de manera automatizada o manual, para asegurar un mejor servicio.

El programa informático propuesto para el metro de Quito incluye una variedad de funciones, tanto para los usuarios como para el personal administrativo, que ayudarán a mejorar la eficiencia y la experiencia en el uso del servicio de metro. El uso de Python permitirá una mayor flexibilidad y escalabilidad

Compresión del problema El metro de Quito requiere de un sistema adecuado al proyecto en el que se trabajó. Para lograrlo, se necesita una gestión eficiente del personal administrativo, encargado de la administración de trenes, la cronometración del tiempo, la verificación de trenes disponibles, la validación de códigos de entrada para los usuarios con problemas, la asignación de conductores a los trenes. Además, para la parte de los usuarios, se debe gestionar el conteo de viajes y la función de carga de saldo para los viajes, para lo cual se requiere una función de registro y una función de inicio de sesión. Todo esto se manejará a través de una base de datos, en nuestro caso, el gestionamiento de archivos

Formular un modelo El metro de Quito es un sistema de transporte esencial para la ciudad, y su eficiencia y efectividad dependen de una buena gestión y planificación. Es por ello que es necesario desarrollar un modelo de resolución que permita optimizar la operación del metro y mejorar la experiencia de los usuarios.

- Recopilación y análisis de los datos necesarios para el proyecto, como la información sobre los trenes disponibles, los horarios de operación, la cantidad de usuarios y su frecuencia de viaje, entre otros.
- Diseño de un sistema de gestión de personal administrativo que permita la asignación de tareas, la cronometración del tiempo y la verificación de trenes disponibles.
- Implementación de un sistema de validación de códigos de entrada para los usuarios con problemas, y una función de registro y de inicio de sesión para los usuarios.

- Utilizar python para desarrollar el sistema, ya que es un lenguaje de programación con una gran cantidad de librerías y herramientas para el análisis de datos y la automatización de tareas.
- Prueba y evaluación del sistema, para asegurar su funcionamiento correcto y detectar posibles errores o problemas.
- Implementación del sistema en el metro de Quito y monitoreo continuo para asegurar su eficiencia y para identificar áreas de mejora.
- Ajustes y actualizaciones periódicas para mantener el sistema actualizado y adaptado a las necesidades cambiantes del metro.

Desarrollar un algoritmo Una vez que se ha comprendido el problema y se ha formulado un modelo de resolución, es necesario seguir una serie de instrucciones para poder codificar el programa. Para ayudar en este proceso se utiliza el pseudocódigo, el cual es una herramienta que permite detallar los pasos que el programa debe llevar a cabo cuando es codificado en un lenguaje de programación que la computadora pueda entender y ejecutar. El pseudocódigo es especialmente útil en la etapa temprana del desarrollo de un programa, ya que permite planificar el proceso de codificación de manera clara y ordenada, facilitando el trabajo del programador.

Como se muestra en la figura Figura 1 nos encontramos en el inicio del programa lo cual es un procedimiento el cual nos da un bienvenida para el metro de Quito.

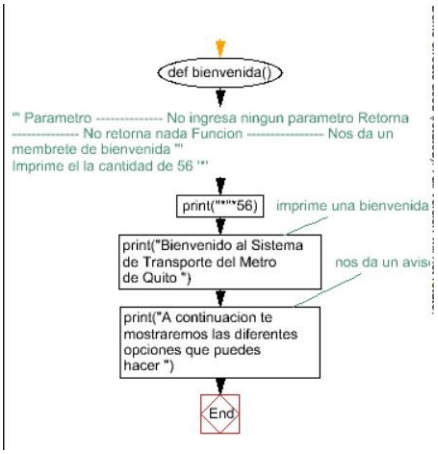

Fig. 1. Procedimiento de bienvenida

Como se puede ver en la figura sobre el procedimiento figura Figura 2 nos desplaza un menu el cual nos da las opciones para escoger y la opcion es guardada en una variable y esto nos retorna.

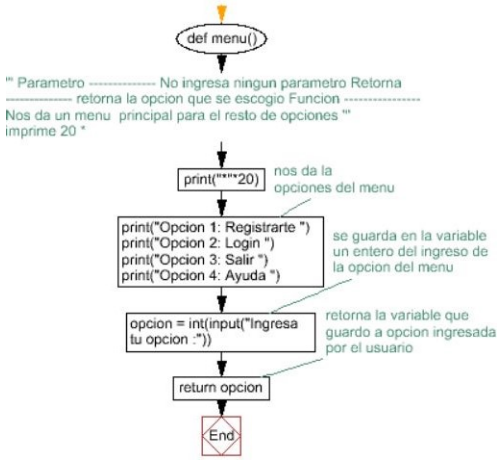

Fig. 2. Procedimiento de menu principal

Como procedimiento como se ve en la figura figura Figura 3 este procedimiento lo que hace es guardar el nombre y el password en un archivo externo para despues su manipulacion, registrando así a los usuarios.

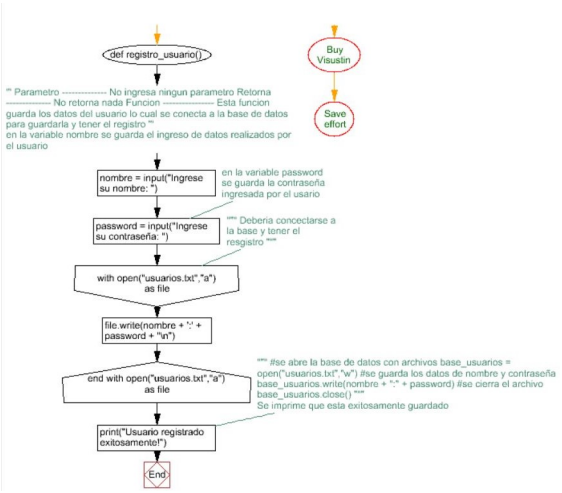

Fig. 3. Procedimiento del registro de usuario

En el procedimiento de login como se muestra a en la figura Figura 4 nos permite hacer una comparacion de los datos ingresados con el archivo pre exixtente con los usuarios registrados, validado esto nos permite tener la opcion de

ser usuario administrativo o usuario convencional, y con ello nos retorna esta opcion escogida que se usara oportunamente en un futuro.

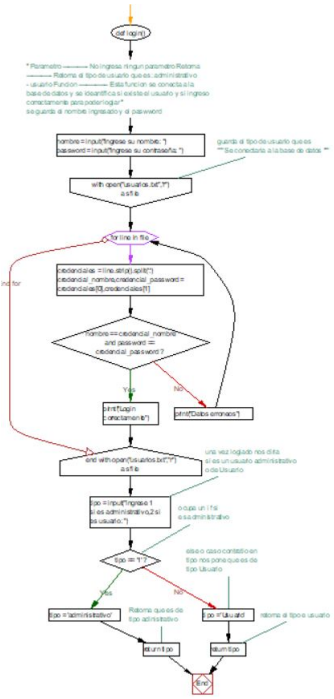

Fig. 4. Procedimiento de Login (iniciar sesión)

En el procedimiento como se muestra en la Figura 5 nos presenta un menu para los usuarios administrativos y se guarda su opcion para luego ser ocupada en algun otro procedimiento.

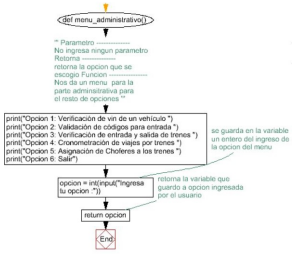

Fig. 5. Procedimiento de menu administrativo

En el procedimiento mostrado en la figura Figura 6 nos muestra un menu el cual es para los usuario convencionales y tiene menos opciones que el procedimiento e la figura Figura 5 es por el hecho de que los usarios tiene menos funciones y es de acuerdo a sus necesidades.

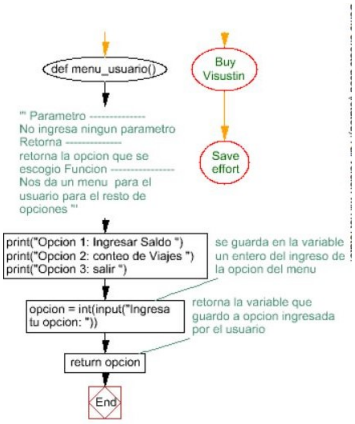

Fig. 6. Procedimiento del menu del usuario convencional

En el proceso de validacion de codigos como se muestra en la figura Figura 7 hace ua validacion de codigos para las personas que tengan problemas con dichos códigos y es administado por los ususarios administradores.

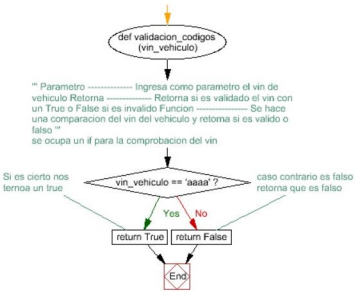

Fig. 7. Procedimiento de la validación de códigos

Los usuarios administradores tiene este procedimiento como se muestra en la figura Figura 8 para la validacion de los trenes que se encuntran disponibles.

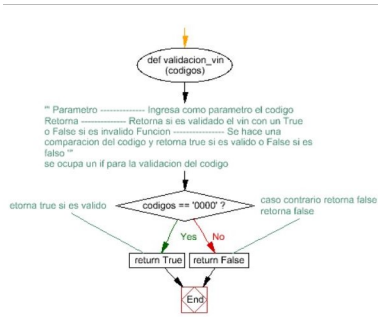

Fig. 8. Procedimiento de la validación de vin

Otra de las funciones con las que muenta el personal administrativo es la entrada y salida de trenes como se muetra a continuacion en la figura Figura 9 la cual llevan un registro de los trenes que entran y salen.

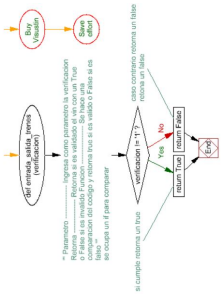

Fig. 9. Procedimiento de la entrada y salida de trenes

Un procedimiento con el que cuenta los usuarios convencionales es el de ecargar saldo como se muestra en la figura Figura 10 para poder acceder al servivio del uso del metro de Quito.

Fig. 10. Procedimiento de menu administrativo

Al igual que tendran el procedimiento de conteo de viajes como se muestra en la figura Figura 11 esto en referente a los usuaios convencionales.

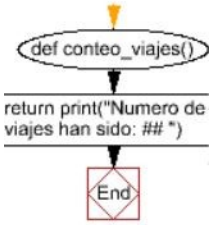

Fig. 11. Procedimiento del conteo de viajes

y por ultimo procedimiento en donde llaman al resto de prcedimientos y trabaja como un filtro y de acuerdo a las opciones que se escojan llaman al resto de procedimientos de acuerdo a lo que se esté necesitando, a continuacion como se muestra en la figura Figura 12 se puede observar como funciona dicho procedimiento.

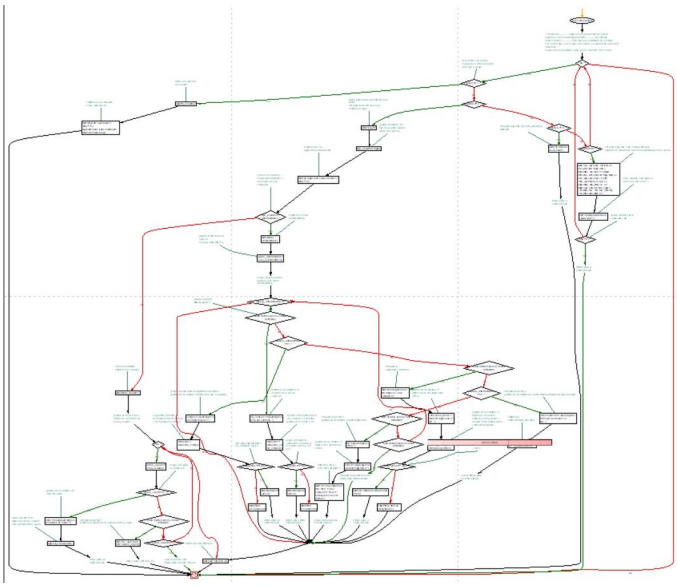

Fig. 12. Procedimiento de accion

Escribe el programa En la presente sección se deberá implementar el algoritmo a instrucciones las cuales serán comprendidas por la maquina, el lenguaje que se utilizo para la implementación del sistema del Metro de Quito en python.

Para la funcion de bienvenida como se muestra en el Listing 1.1 nos imprime en pantalla una caratula y nos da la bienvenida al programa, esta función es más para la parte estetica del programa para que tenga una mejor apariencia para los usuarios.

Listing 1.1. Función bienvenida

```
def bienvenida():
    '''
    Parametro
    ---------------
    No ingresa ningun parametro

    Retorna
    ---------------
    No retorna nada

    Funcion
    ---------------
    Nos da un membrete de bienvenida

    '''
    #Imprime el la cantidad de 56 '*'
    print("*"*56)
    #imprime una bienvenida
    print("Bienvenido al Sistema de Transporte del Metro
        de Quito ")
    #nos da un aviso
    print("A continuacion te mostraremos las diferentes
        opciones que puedes hacer ")
```

Por consiguiente la funcion menu se encuntra Listing 1.2, la funcion mensionada contiene un menu principal el cual será mostrado al usuario para que el escoga el item que necesita lo cual guada su opcion y lo retorna para que sea ocupado por alguna otra funcion.

Listing 1.2. Función menu

```
ef menu():
    '''
    Parametro
    ---------------
    No ingresa ningun parametro

    Retorna
    ---------------
```

```
    retorna la opcion que se escogio

    Funcion
    ---------------------
    Nos da un menu  principal para el resto de opciones

    '''
    #imprime 20 *
    print("*"*20)
    #nos da la opciones del menu
    print("Opcion 1: Registrarte ")
    print("Opcion 2: Login ")
    print("Opcion 3: Salir ")
    print("Opcion 4: Ayuda ")
    #se guarda en la variable un entero del ingreso de la
        opcion del menu
    opcion = int(input("Ingresa tu opcion :"))
    #retorna la variable que guardo a opcion ingresada
       por el usuario
    return opcion
```

La función Listing 1.3, esta función es donde se registra a los usuarios y los guarda en un archivo de texto como base de datos para ser ocupado posteriormente por otras funciones del prorama

Listing 1.3. Función Registro Usuario

```
def registro_usuario():
    '''
    Parametro
    ---------------------
    No ingresa ningun parametro

    Retorna
    ---------------------
    No retorna nada

    Funcion
    ---------------------
    Esta funcion guarda los datos del usuario lo cual se
       conecta a la base de datos para guardarla y tener
       el registro

    '''
    #en la variable nombre se guarda el ingreso de datos
       realizados por el usuario
    nombre = input("Ingrese su nombre: ")
```

```
    #en la variable password se guarda la contrase a
    ingresada por el usario
    password = input("Ingrese su contrase a: ")
    with open("usuarios.txt","a") as file:
        file.write(nombre + ':' + password + "\n")

    #se abre la base de datos con archivos
    base_usuarios = open("usuarios.txt","w")

    #se guarda los datos de nombre y contrase a
    base_usuarios.write(nombre + ":" + password)
    #se cierra el archivo
    base_usuarios.close()
    """
    #Se imprime que esta exitosamente guardado
    print("Usuario registrado exitosamente!")
```

En cambio en la función Listing 1.4, se valida los datos ingresados con la base de datos para poder iniciar sesion y entrar al resto de funciones.

Listing 1.4. Función Login

```
def login():
    '''
    Parametro
    ----------
    No ingresa ningun parametro

    Retorna
    ----------
    Retorna el tipo de usuario que es: administrativo -
        usuario

    Funcion
    ----------
    Esta funcion se conecta a la base de datos y se
        ideantifica si existe el usuario y si ingreso
        correctamente para poder logiar

    '''
    #se guarda el nombre ingresado y el paswword
    nombre = input("Ingrese su nombre: ")

    password = input("Ingrese su contrase a: ")
    #guarda el tipo de usuario que es
```

```
    """
    Se conectaria a la base de datos
    """
    with open("usuarios.txt","r") as file:
        for line in file:
            credenciales = line.strip().split(':')
            credencial_nombre,credencial_password =
                credenciales[0],credenciales[1]
            if nombre == credencial_nombre and password
               == credencial_password:
                print("Login correctamente")
                break
            else:
                print("Datos erroneos")
    #una vez logiado nos diria si es un usuario
        administrativo o de Usuario
    tipo = input("ingrese 1 si es administrativo,2 si es
        usuario: ")
    #ocupa un if si es administrativo
    if tipo == '1':
        tipo = 'administrativo'
        #Retorna que es de tipo adinistrativo
        return tipo
    #else o caso contratio en tipo nos pone que es de
        tipo Usuario
    else:
        tipo = 'Usuario'
        #retorna el tipo e usuario
        return tipo
```

Para la función Listing 1.5, nos muestra el menu si es un usuario administrativo con las diferentes funcoines de: Verificación de vin de un vehículo - Validación de códigos para entrada - Verificación de entrada y salida de trenes - Cronometración de viajes por trenes - Asignación de Choferes a los trenes - Salir ; de Acuerdo a la opción que se escoge cumplira una subsiguiente función o proceso.

Listing 1.5. Función Menu administrativo

```
def menu_administrativo():
    '''
    Parametro
    ---------
    No ingresa ningun parametro

    Retorna
```

```
    ----------
    retorna la opcion que se escogio

    Funcion
    ----------
    Nos da un menu  para la parte adminsitrativa para el
        resto de opciones

    '''

    print("Opcion 1: Verificaci n de vin de un veh culo
        ")

    print("Opcion 2: Validaci n de c digos para entrada
        ")

    print("Opcion 3: Verificaci n de entrada y salida de
        trenes ")

    print("Opcion 4: Cronometraci n de viajes por trenes
        ")

    print("Opcion 5: Asignaci n de Choferes a los trenes
        ")

    print("Opcion 6: Salir")

    #se guarda en la variable un entero del ingreso de la
         opcion del menu
    opcion = int(input("Ingresa tu opcion :"))
    #retorna la variable que guardo a opcion ingresada
        por el usuario
    return opcion
```

El uso del Listing 1.6 , en cambio para los usuarios que no son administrativos sino usuarios normales van a usar el sistema lo cual teiene muchas menos opciones las cuales son simples de acuerdo a su necesidad las cuales son: ingresar saldo - conteo de viajes - salir un sistema mas simplificado para cualquier tipo de usuario que utilice el servicio.

Listing 1.6. Función Menu Usuario

```
def menu_usuario():
    '''
    Parametro
    ----------
    No ingresa ningun parametro
```

```
    Retorna
    ----------
    retorna la opcion que se escogio

    Funcion
    ----------
    Nos da un menu  para el usuario para el resto de
        opciones

    '''
    print("Opcion 1: Ingresar Saldo ")
    print("Opcion 2: conteo de Viajes ")
    print("Opcion 3: salir ")
    #se guarda en la variable un entero del ingreso de la
         opcion del menu
    opcion = int(input("Ingresa tu opcion: "))
    #retorna la variable que guardo a opcion ingresada
        por el usuario
    return opcion
```

En esta función Listing 1.7 recibe los codigos con las personas que tienen problemas para ingresar al metro de Quito lo cual se verifica si son verdaderos o falsos dichos códigos.

Listing 1.7. Función validación códigos

```
def validacion_codigos(vin_vehiculo):
    '''
    Parametro
    ----------
    Ingresa como parametro el vin de vehiculo

    Retorna
    ----------
    Retorna si es validado el vin con un True o False si
        es invalido

    Funcion
    ----------
    Se hace una comparacion del vin del vehiculo y
        retorna si es valido o falso

    '''
    #se ocupa un if para la comprobacion del vin
    if vin_vehiculo == 'aaaa':
        #Si es cierto nos ternoa un true
```

```
        return True

    #caso contrario es falso
    else:
        #retorna que es falso
        return False
```

Al igual que en la función anterior esta Listing 1.8 valida los códigos de los trenes que se encuntrar aptos para su uso de esta forma validamos su codigo y poder tener un inventario.

Listing 1.8. Función validación vin

```
def validacion_vin(codigos):
    '''
    Parametro
    ----------------
    Ingresa como parametro el codigo

    Retorna
    ----------------
    Retorna si es validado el vin con un True o False si
        es invalido

    Funcion
    ----------------
    Se hace una comparacion del codigo y retorna true si
        es valido o False si es falso

    '''
    #se ocupa un if para la validacion del codigo
    if codigos == '0000':
        #etorna true si es valido

        return True
    #caso contrario retorna false
    else:
        #retorna false
        return False
```

En la función Listing 1.9 realiza una veificación de la entrada y salida de trenes para conocer si se encuntrar en uso o fuera de servicio.

Listing 1.9. Función entrada y salida de trenes

```
def entrada_salida_trenes(verificacion):
    '''
    Parametro
```

```
    ----------
    Ingresa como parametro la verificacion

    Retorna
    ----------
    Retorna si es validado el vin con un True o False si
        es invalido

    Funcion
    ----------
    Se hace una comparacion del codigo y retorna true si
        es valido o False si es falso

    '''
    #se ocupa un if para comparar
    if verificacion != '1':
        #si cumple retorna un true
        return True
    #caso contrario retorna un false
    else:
        #retona un false
        return False
```

Esta función Listing 1.10 se encuntra para los usuarios los cuales requieran realizar recargas de salgo para el uso del servivio del metro de Quito.

Listing 1.10. Función saldo

```
def ingresar_saldo(saldo):
    '''
    Parametro
    ----------
    Ingresa como parametro el saldo

    Retorna
    ----------
    El saldo a adido

    Funcion
    ----------
    Esta funcion se conectara a la base de datos para
        actualizar el saldo de la persona

    '''
    #imprime el saldo que se ha ingresado
    print("Se ha ingresado: ",saldo)
```

Si los usuarios desean tener un conteo de los viajes que realian entonces la función Listing 1.11 les permite observar la cantidad de viajes que han realizado.

Listing 1.11. Función conteo viajes

```
def conteo_viajes():

    return print("Numero de viajes han sido: ## ")
```

Por finalizar y una de las funciones mas fuertes ya que llama incluso a lresto de funciones es la función Listing 1.13 en esta función es en donde se filtran los datos y de acuerdo a esto va llamando a una función u otra de acuerdo al requerimiento.

Listing 1.12. Función Accion

```
def accion(opcion):
    '''
    Parametro
    ----------
    Ingresa como parametro la opcion elejida en el menu
        principal

    Retorna
    ----------
    No rertorna nada

    Funcion
    ----------
    Esta funcion va filtrando de acuerdo a la opcion que
        se escogio en el menu y va llamando al resto de
        funciones
    '''

    #ocupa un ciclo repetitivo para que no culmine en lo
        menus
    while True:
        #de acuerdo a la opcion escogida se toma la
            opcion en la que cumple
        if opcion == 1:
            #llama a la funcion de usuario
            registro_usuario()
            #imprime que el registro se ha culminado
            print("Registro culminado")
            print("*"*15)
            input("presione para continuar")
            return accion(menu())
            #break para hacer un cierre forzado
            #break
```

```
#elif para la opcion que sea 2
elif opcion == 2:
    #imprime el login
    print("Login")
    #guarda el retorno  del tipo de usuario
        cuando llama a la funcion
    tipo_usuario = login()
    #imprime que ha ngresado exitosamente
    print("Ha ingresado exitosamente")
    print("*"*15)
    #se hace un if para el usuarioadministrativo
        y si cumple hace la sentencia
    if tipo_usuario == 'administrativo':
        #imprime el menu administrativo
        print("Menu administrativo")
        #guarda el retorno de la funcion
            menu_administrativo
        opcion_administativo =
            menu_administrativo()
        #ocupa un while para la opciones del menu
             administrativo
        while opcion_administativo<5:
            #ocupa un if para filtrar la opcion
            if opcion_administativo == 1:
                #guarda lo que ingresa el usuario
                     como vin de vehiculo
                vin_vehiculo = input("Ingrese el
                    vin del vehiculo: ")
                #manda como parametro el
                    vin_vehiculo y el retorno
                    guarda si es valido o no
                validacion = validacion_vin(
                    vin_vehiculo)
                #ocupa un if para la validacion
                    con el tru si es valido y
                    false si no lo es
                if validacion == True:
                    print("Validacion Exitosa")
                    #break para cerrar
                        forzadamente
                    break
                else:
                    print("Error de validacion")
                    break
```

```
#ocupa un elfin para la siguientes
    opciones
elif opcion_administativo == 2:
    #guarda en la variable el input
        de lo que se ingreso
    codigos = input("Ingrese el
        codigo de pase: ")
    #se guarda el retorno de la
        validacion de la funcion y se
        manda los codigos
    validacion = validacion_codigos(
        codigos)
    #un if para comprobar si es
        verdadero o falso
    if validacion == False:
        print("Validacion Exitosa")
        #break para un cierre forzado
        break
    else:
        print("Error de validacion")
        break
#elif para la sigueintes opciones
elif opcion_administativo == 3:

    print("Si ha ingresado el tren
        ingrese 1 caso contrario 0")
    #guarda en la variable el ingreso
         quesi ha ingresado el tren
    verificacion_trenes = input("Ha
        ingresado el tren : ")
    #guarda en la variable el retorno
         de la funcion
        entrada_salida_trenes y manda
        como parametro la
        verificacion_trenes
    verificacion =
        entrada_salida_trenes(
        verificacion_trenes)
    #ocupa un if para la validacion
        si es verdadero o falso
    if verificacion == True:
        print("Ha entrado
            exitosamente el tren")
        #break para el cierre forzado
        break
```

```
                else:
                    print("Error el tren no ha
                        ingresado")
                    break
            #elif para la opcion 4
            elif opcion_administativo == 4:
                #guarda en la variable la
                    conometracion ingresada por el
                     usario
                cronometracion = input("Ingrese
                    el timepo que realizo el tren:
                    ")
                #imprime la cronometracion del
                    tren
                print("La cronometracion fue :",
                    cronometracion)
                #usa un break para el cierre
                    forzado
                break
            #elif para la opcion 5
            elif opcion_administativo ==5:
                #guarda en la variable el dato
                    ingresado
                tren = input("Ingrese el tren: ")
                #guarda en la variable el ingreso
                     del nombre del chofer
                chofer = input("Ingrese al chofer
                    asignado: ")
                #imprime el tren y el choefer
                    asignado
                print ("se ha asignado al tren ",
                    tren,"con el conductor ",
                    chofer,"encargado con esta
                    unidad")
                #ocupa un break para el cierre
                    forzado
                break
            #elif para la opcion de salir
            elif opcion_administativo == 6:
                #break para cierre forzado
                break
    #en caso contratio entonces es usuario
    else:
            print("Menu Usuario")
```

```
                #guarda en la opcion el retorno de la
                    funcion de menu_usuario
                while True:
                    opcion_usuario = menu_usuario()
                    #ocupa un if para la validacion
                    if opcion_usuario == 1:
                        #guarda en la variable un
                            float del saldo
                        saldo = float(input("Ingresa
                            la cantidad de saldo: "))
                        #llama a la funcion
                            ingresar_saldo y manda
                            como parametro el saldo
                        ingresar_saldo(saldo)
                        #break para un cierre forzado
                        return
                    #elif para la opcion 2
                    elif opcion_usuario == 2:
                        #imprime el retorno de la
                            funcion conteo_viajes
                        print("Tus viajes han sido",
                            conteo_viajes())
                        #break para cerre forazdo
                        return
                        #para la opcion salir
                    elif opcion_usuario ==3:
                        #break para el cierre forzado
                        return
            #return al menu principal
            return accion(menu())


        #Elif para la opcion del menu principal
        elif opcion == 3:

            #print par
            print("Se termino de ejecutar")
            #break para el cierre forzado
            break
        #elif para la opcion 4 del menu principal
        elif opcion== 4:
            #imprime las intrucciones del menu principal
                como ayuda
            print("Las opciones indicadas nos permiten
                realizar las diferentes acciones",
```

```
        "\npara OPCION 1: Nos permite registrarnos en
            la aplicacion del metro",
        "\nen la OPCION 2: Una vez registrado
            deberiamos de ingresar con nuestro usuario
            y contrase a ",
        "\nen la OPCION 3: Cierra el programa  ")
    #en la variable salir ingresa si desea salir
        con un 1
    salir = int(input("presione 1 para salir: "))
    #if para validar con la vaariable salir
    if salir == 1:
        #brreak para el cierre forzado
        break
```

Y por finaliar en donde es la cabeza del proyecto el main Listing **??**, aquí se encuentra llamada a las funciones para el funcionamiento del código.

Listing 1.13. Main

```
if __name__ == "__main__":
    #llama a la fncion bienvenida
    bienvenida()
    #llama a la funcion accion y como parametro la
        funcion menu
    accion(menu())
```

Pruebe el programa

Evalué la solución Después de evaluar el modelo de resolución propuesto, se puede concluir que es una estrategia viable para optimizar la operación del metro de Quito y mejorar la experiencia de los usuarios. El diseño del sistema de gestión de personal administrativo, la implementación de un sistema de validación de códigos de entrada y una función de registro y inicio de sesión para los usuarios son medidas necesarias para garantizar una operación eficiente del metro. Utilizar Python para desarrollar el sistema es una buena opción debido a las numerosas librerías y herramientas disponibles para el análisis de datos y la automatización de tareas. La realización de pruebas y evaluaciones para asegurar el correcto funcionamiento del sistema, así como el monitoreo continuo y los ajustes periódicos son medidas necesarias para garantizar su eficiencia y adaptabilidad a las necesidades cambiantes del metro de Quito. Por lo que puedo afirmar que este modelo de resolución es una estrategia sólida y viable para mejorar la operación del metro de Quito.

A continuacion se mostrara las siguien figuras las cuales mostraran las pruebas del programa.

En la siguiente figura Figura 13 se puede la corrida del programa cuando se hace un registro de un usuario.

Fig. 13. Registro de usuarios

En la siguiente figurra Figura 14 se puede ver la ejecución del inicio de sesión y el tipo de usuario que puede ser, administrativo o usuario. En la siguiente figura

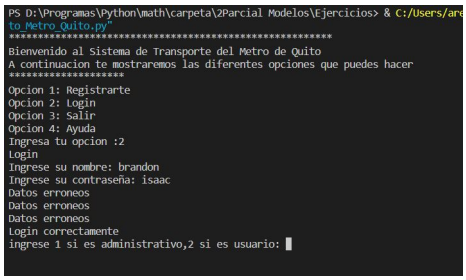

Fig. 14. Inicio de sesión

Figura 15 se puede ver el menu de usuario administrador y las opciones que tiene. En la siguiente figura Figura 16se puede ver la opcion de validacion de

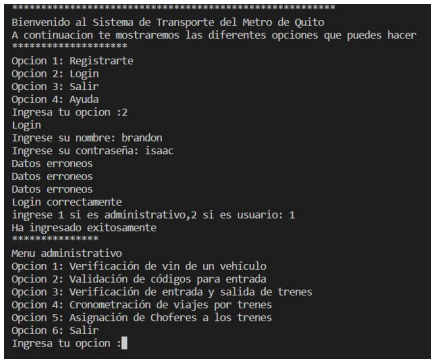

Fig. 15. Menú personal administrativo

códigos y como funciona En la siguiente figura Figura 17 se puede ver la opcion

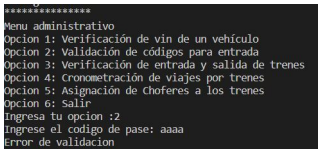

Fig. 16. Validacion de Codigos

de validacion del vin de los trenes y como se ejecuta En la siguiente figura Figura

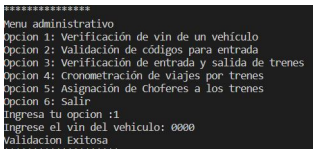

Fig. 17. Validacion de Vin de trenes

18 se puede ver la opcion cronometración y como se mostro En la siguiente figura

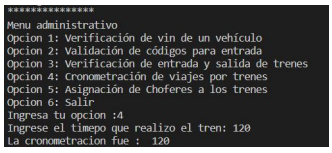

Fig. 18. Cronometracion de trenes

Figura 19 se puede ver la opcion de ingreso de los trenes.

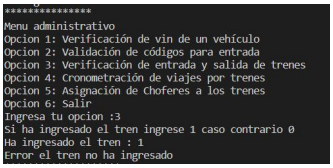

Fig. 19. Ingreso y salida de trenes

3 Discusión

El programa informático propuesto para el metro de Quito tiene como objetivo mejorar la eficiencia y la experiencia de los pasajeros en el uso del servicio. Al utilizar el lenguaje de programación Python, el programa cuenta con una gran escalabilidad, lo que permite agregar nuevas funciones y adaptarse a las necesidades cambiantes del servicio de metro. Esto también permite una mayor flexibilidad al momento de tomar decisiones y resolver problemas en el servicio de metro.

Además, el programa permite una mayor precisión en la gestión del servicio de metro, al proporcionar información precisa y actualizada sobre el servicio, incluyendo información sobre, los horarios de llegada y salida de los trenes. Esto ayudará a los pasajeros a planificar sus viajes de manera más eficiente, lo que a su vez reducirá los retrasos y la congestión en el servicio.

En cuanto a la experiencia del usuario, el programa les permite contar sus viajes y tener una mejor información sobre el servicio de metro. Además, les permite cargar saldo de manera rápida y sencilla, mejorando la eficiencia en la gestión de las tarjetas de metro.

Para más detalles sobre el proyecto se puede visitar el siguiente link: `https://github.com/BrandonArellanoC/Proyecto_Metro_de_Quito`, donde se encuentra subido a Github con el código fuente y documentación para su consulta y estudio. En general, el programa informático propuesto para el metro de Quito tiene el potencial de mejorar significativamente la eficiencia y la experiencia de los pasajeros en el uso del servicio de metro.

4 Conclusiones

El programa informático propuesto para el metro de Quito tiene como objetivo mejorar la eficiencia y la experiencia del usuario en el uso del servicio. Se ha presentado las funciones específicas del programa y se ha discutido cómo estas ayudarán a mejorar la gestión del servicio de metro y la experiencia del usuario. El programa cuenta con una gran escalabilidad y flexibilidad, lo que permite adaptarse a las necesidades cambiantes del servicio de metro y agregar nuevas funciones. El programa tiene el potencial de mejorar significativamente la eficiencia y la experiencia en el uso del servicio de metro en Quito. Es importante que se implemente un programa de esta naturaleza ya que mejorará la calidad de vida de las personas que usan el metro diariamente, ayudando a que sean más seguros, eficientes y cómodos