Unit I Lab Exercises III
MCA171 Python Programming
Department of Computer Science, Christ University Central Campus

TOJIN VARKEY SIMSON
2447253 MCA-B

Demonstrate Custom modules with functions

Your company uses **SmartScan Codes** to streamline user registration. You need to implement a system that reads user data from a SmartScan Code image and manages it using custom modules with lambda functions.

(a) Create a Python module named smartscan_registration_module.py that includes:

In-Memory Storage: Simulate a database using a list of dictionaries. Define lambda functions within the module for:

i. Creating a new user record.
ii. Inserting the user record into the list.
iii. Fetching all user records from the list.

SmartScan Code Scanning: Implement a function that reads and decodes the SmartScan Code. The SmartScan Code contains user information encoded as a comma-separated string in the format "name,email".

User Registration Function: Implement a function RegisterUserFromSmartScan that:

i. Uses the scanning function to extract user data.
ii. Uses the lambda functions to create and insert the user record into the in-memory list.
iii. Prints the list of all registered users after adding the new user.

(b) Place the above function in a separate module file and create another script to import this module and invoke the function within the script.

A):

```
# smartscan_registration_module.py
# Initialize a list to store user record
equipment_details = []


# Lambda function to create a new user record
```

```
create_equipment_record = lambda equipment_id, equipmentName, CompanyEmail: {"equipment_id": equipment_id, "equipmentName": equipmentName, "CompanyEmail": CompanyEmail}

# Lambda function to insert the user record into the list
insert_equipment_record = lambda record: equipment_details.append(record)

# Lambda function to fetch all user records from the list
fetch_all_equipment_records = lambda: equipment_details

def SmartScanCode(data):
    """Decode data from QR code."""
    # Assume data is a comma-separated string: "user_id,name,email"
    try:
        equipment_id, equipmentName, CompanyEmail = data.split(',')
        return equipment_id, equipmentName, CompanyEmail
    except ValueError:
        raise ValueError("Data entered is incorrect.")
```

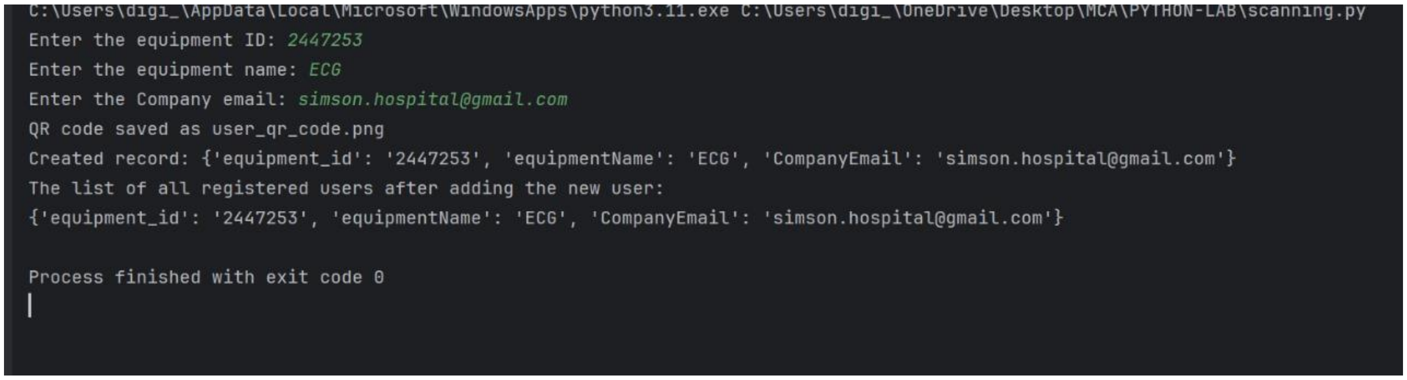
# User_register.py

```
from smartscan_registration_module import *

def RegisterUserFromSmartScan(data):
    """Register a new user from SmartScan Code data."""
    equipment_id, equipmentName, CompanyEmail = SmartScanCode(data)

    # Create a new equipment record
    new_equipment = create_equipment_record(equipment_id, equipmentName, CompanyEmail)
    print(f"Created record: {new_equipment}")

    # Insert user record into the list
    insert_equipment_record(new_equipment)

    # Print all registered users
    print("The list of all registered users after adding the new user:")
    for equipment in fetch_all_equipment_records():
        print(equipment)
```

# Main File
# DOMAIN:Medical Equipment Failure Prediction

```
import qrcode
from PIL import Image
import re
from User_register import RegisterUserFromSmartScan

def is_valid_equipment_id(equipment_id):
    """Validate equipment ID."""
    return len(equipment_id) > 0

def is_valid_equipmentName(equipmentName):
    """Validate equipment name to ensure it contains only letters and spaces."""
    return bool(re.match(r"^[A-Za-z\s]+$", equipmentName))

def is_valid_CompanyEmail(CompanyEmail):
    """Validate Company Email address format."""
    return bool(re.match(r"^[\w\.-]+@[\w\.-]+\.\w+$", CompanyEmail))

def generate_qr_code(data, filename='user_qr_code.png'):
    """Generate a QR code from the given data and save it to a file."""
    qr = qrcode.QRCode(
        version=1,
        error_correction=qrcode.constants.ERROR_CORRECT_L,
        box_size=10,
        border=4,
    )
    qr.add_data(data)
    qr.make(fit=True)
    img = qr.make_image(fill='black', back_color='white')
    img.save(filename)
    print(f"QR code saved as {filename}")

    # Display the QR code image
    img.show()

def main():
    # User details
    equipment_id = input("Enter the equipment ID: ")
    equipmentName = input("Enter the equipment name: ")
    CompanyEmail = input("Enter the Company email: ")

    # Validate user input
    if not is_valid_equipment_id(equipment_id):
        print("Invalid equipment ID. It should not be empty.")
        return

    if not is_valid_equipmentName(equipmentName):
        print("Invalid equipment name. It should contain only letters and spaces.")
        return

    if not is_valid_CompanyEmail(CompanyEmail):
        print("Invalid Company email format.")
        return

    # Combine details into a comma-separated string
    data = f"{equipment_id},{equipmentName},{CompanyEmail}"

    # Generate QR code
    generate_qr_code(data, 'user_qr_code.png')

    # Register user from QR code data
    RegisterUserFromSmartScan(data)

if __name__ == "__main__":
    main()
```

OUTPUT:



```
C:\Users\digi_\AppData\Local\Microsoft\WindowsApps\python3.11.exe C:\Users\digi_\OneDrive\Desktop\MCA\PYTHON-LAB\scanning.py
Enter the equipment ID: 2447253
Enter the equipment name: ECG
Enter the Company email: simson.hospital@gmail.com
QR code saved as user_qr_code.png
Created record: {'equipment_id': '2447253', 'equipmentName': 'ECG', 'CompanyEmail': 'simson.hospital@gmail.com'}
The list of all registered users after adding the new user:
{'equipment_id': '2447253', 'equipmentName': 'ECG', 'CompanyEmail': 'simson.hospital@gmail.com'}

Process finished with exit code 0
```