Write a python program that helps a buyer to choose his desired car based on available cars of different models and parameters from a car shop. Consider the following while developing the python code:

1. **Car**: This class will represent a car with its attributes (price, fuel efficiency, brand, etc.).
2. **CarShop**: This class will represent the car shop that contains available cars.
3. **CarBuyer**: This class will assist the buyer in choosing a car based on their preferences.

**Python Code for Car Recommendation System:**

import pandas as pd

# Class representing a single car

class Car:

def \_\_init\_\_(self, model, price, fuel\_efficiency, brand, horsepower, safety\_rating, color):

self.model = model

self.price = price

self.fuel\_efficiency = fuel\_efficiency

self.brand = brand

self.horsepower = horsepower

self.safety\_rating = safety\_rating

self.color = color

def \_\_repr\_\_(self):

return f"{self.model} ({self.brand}): ${self.price}, {self.fuel\_efficiency} MPG, Safety: {self.safety\_rating} Stars"

# Class representing the car shop with available cars

class CarShop:

def \_\_init\_\_(self):

self.cars = []

def add\_car(self, car):

self.cars.append(car)

def get\_available\_cars(self):

return self.cars

# Class representing a car buyer, who can choose a car based on preferences

class CarBuyer:

def \_\_init\_\_(self, budget, min\_fuel\_efficiency, brand\_preference, min\_safety\_rating):

self.budget = budget

self.min\_fuel\_efficiency = min\_fuel\_efficiency

self.brand\_preference = brand\_preference

self.min\_safety\_rating = min\_safety\_rating

def filter\_cars(self, cars):

# Filter cars based on the buyer's preferences

matching\_cars = [

car for car in cars

if car.price <= self.budget

and car.fuel\_efficiency >= self.min\_fuel\_efficiency

and (self.brand\_preference.lower() in car.brand.lower() or self.brand\_preference == "")

and car.safety\_rating >= self.min\_safety\_rating ]

return matching\_cars

def recommend\_car(self, car\_shop):

available\_cars = car\_shop.get\_available\_cars()

filtered\_cars = self.filter\_cars(available\_cars)

if not filtered\_cars:

print("No cars match your criteria. Please adjust your preferences.")

else:

print(f"\nHere are the cars that match your preferences (Price <= ${self.budget}, Min Fuel Efficiency >= {self.min\_fuel\_efficiency} MPG, Brand: {self.brand\_preference}, Min Safety Rating >= {self.min\_safety\_rating}):")

for car in filtered\_cars:

print(car)

# Example usage:

# 1. Create a CarShop and add some cars

car\_shop = CarShop()

# Adding some car models to the car shop

car\_shop.add\_car(Car('Honda Civic', 22000, 30, 'Honda', 158, 4.5, 'Red'))

car\_shop.add\_car(Car('Toyota Corolla', 21000, 32, 'Toyota', 139, 4.7, 'Blue'))

car\_shop.add\_car(Car('BMW 3 Series', 35000, 25, 'BMW', 255, 4.2, 'Black'))

car\_shop.add\_car(Car('Ford Focus', 20000, 28, 'Ford', 160, 4.3, 'White'))

car\_shop.add\_car(Car('Chevrolet Malibu', 23000, 26, 'Chevrolet', 160, 4.4, 'Silver'))

# 2. Ask for buyer preferences and create a CarBuyer object

print("Welcome to the car shop!\n")

# Get buyer preferences (for simplicity, input is taken as plain text)

try:

budget = float(input("Enter your maximum budget (in USD): $"))

min\_fuel\_efficiency = float(input("Enter the minimum fuel efficiency (in MPG): "))

brand\_preference = input("Enter your preferred brand (leave blank for any): ")

min\_safety\_rating = float(input("Enter the minimum safety rating (out of 5): "))

buyer = CarBuyer(budget, min\_fuel\_efficiency, brand\_preference, min\_safety\_rating)

# 3. Get recommendations based on buyer preferences

buyer.recommend\_car(car\_shop)

except ValueError:

print("Invalid input! Please enter valid numbers for budget, fuel efficiency, and safety rating.")