CODE DOCUMENTATION

Name: Ragavarshini S

Reg no: 20MIS0192

Mail: ragavarshinis1025@gmail.com

```
import sqlite3

# Connect to the SQLite database
conn = sqlite3.connect('ice_cream_parlor.db')
cursor = conn.cursor()

# Create tables if they don't exist
cursor.execute('''
    CREATE TABLE IF NOT EXISTS flavors (
        id INTEGER PRIMARY KEY,
        name TEXT NOT NULL,
        description TEXT,
        seasonal BOOLEAN NOT NULL DEFAULT 0
    )
''')

cursor.execute('''
    CREATE TABLE IF NOT EXISTS ingredients (
        id INTEGER PRIMARY KEY,
        name TEXT NOT NULL
    )
''')

cursor.execute('''
    CREATE TABLE IF NOT EXISTS flavor_ingredients (
        flavor_id INTEGER,
        ingredient_id INTEGER,
        PRIMARY KEY (flavor_id, ingredient_id),
        FOREIGN KEY (flavor_id) REFERENCES flavors (id),
        FOREIGN KEY (ingredient_id) REFERENCES ingredients (id)
    )
''')

cursor.execute('''
    CREATE TABLE IF NOT EXISTS customer_suggestions (
        id INTEGER PRIMARY KEY,
        name TEXT NOT NULL,
        suggestion TEXT NOT NULL
    )
''')

cursor.execute('''
    CREATE TABLE IF NOT EXISTS allergens (
        id INTEGER PRIMARY KEY,
        name TEXT NOT NULL
    )
''')

cursor.execute('''
    CREATE TABLE IF NOT EXISTS flavor_allergens (
        flavor_id INTEGER,
        allergen_id INTEGER,
        PRIMARY KEY (flavor_id, allergen_id),
        FOREIGN KEY (flavor_id) REFERENCES flavors (id),
        FOREIGN KEY (allergen_id) REFERENCES allergens (id)
    )
''')

cursor.execute('''
    CREATE TABLE IF NOT EXISTS carts (
        id INTEGER PRIMARY KEY,
        customer_name TEXT NOT NULL
    )
''')

cursor.execute('''
    CREATE TABLE IF NOT EXISTS cart_items (
        cart_id INTEGER,
        flavor_id INTEGER,
        PRIMARY KEY (cart_id, flavor_id),
        FOREIGN KEY (cart_id) REFERENCES carts (id),
        FOREIGN KEY (flavor_id) REFERENCES flavors (id)
    )
''')

# Commit the changes
conn.commit()
```

```
# Function to add a new flavor
def add_flavor(name, description, seasonal):
    """
    Adds a new flavor to the database.

    Args:
        name (str): Name of the flavor.
        description (str): Description of the flavor.
        seasonal (bool): Indicates if the flavor is seasonal.

    Returns:
        int: ID of the newly added flavor.
    """
    cursor.execute('INSERT INTO flavors (name, description, seasonal) VALUES (?, ?, ?)', (name,
description, seasonal))
    conn.commit()
    return cursor.lastrowid

# Function to add a new ingredient
def add_ingredient(name):
    """
    Adds a new ingredient to the database.

    Args:
        name (str): Name of the ingredient.

    Returns:
        int: ID of the newly added ingredient.
    """
    cursor.execute('INSERT INTO ingredients (name) VALUES (?)', (name,))
    conn.commit()
    return cursor.lastrowid

# Function to associate an ingredient with a flavor
def add_flavor_ingredient(flavor_id, ingredient_id):
    """
    Links an ingredient to a flavor.

    Args:
        flavor_id (int): ID of the flavor.
        ingredient_id (int): ID of the ingredient.
    """
    cursor.execute('INSERT INTO flavor_ingredients (flavor_id, ingredient_id) VALUES (?, ?)',
(flavor_id, ingredient_id))
```

```
    conn.commit()

# Function to add a customer suggestion
def add_customer_suggestion(name, suggestion):
    """
    Adds a customer suggestion to the database.

    Args:
        name (str): Name of the customer.
        suggestion (str): Customer's suggestion.

    Returns:
        int: ID of the newly added suggestion.
    """
    cursor.execute('INSERT INTO customer_suggestions (name, suggestion) VALUES (?, ?)', (name,
suggestion))
    conn.commit()
    return cursor.lastrowid

# Function to add a new allergen
def add_allergen(name):
    """
    Adds a new allergen to the database.

    Args:
        name (str): Name of the allergen.

    Returns:
        int: ID of the newly added allergen.
    """
    cursor.execute('INSERT INTO allergens (name) VALUES (?)', (name,))
    conn.commit()
    return cursor.lastrowid

# Function to associate an allergen with a flavor
def add_flavor_allergen(flavor_id, allergen_id):
    """
    Links an allergen to a flavor.

    Args:
        flavor_id (int): ID of the flavor.
        allergen_id (int): ID of the allergen.
    """
    cursor.execute('INSERT INTO flavor_allergens (flavor_id, allergen_id) VALUES (?, ?)', (flavor_id,
allergen_id))
    conn.commit()

# Function to create a new shopping cart
def create_cart(customer_name):
    """
    Creates a new shopping cart for a customer.

    Args:
        customer_name (str): Name of the customer.

    Returns:
        int: ID of the newly created cart.
    """
    cursor.execute('INSERT INTO carts (customer_name) VALUES (?)', (customer_name,))
    conn.commit()
    return cursor.lastrowid

# Function to add a flavor to a cart
def add_to_cart(cart_id, flavor_id):
    """
    Adds a flavor to the customer's cart.

    Args:
        cart_id (int): ID of the cart.
        flavor_id (int): ID of the flavor.
    """
    cursor.execute('INSERT INTO cart_items (cart_id, flavor_id) VALUES (?, ?)', (cart_id, flavor_id))
    conn.commit()

# Function to search for flavors by name
def search_flavors(name):
    """
    Searches for flavors by name.

    Args:
        name (str): Name of the flavor to search for.

    Returns:
        list: List of matching flavors.
    """
    cursor.execute('SELECT * FROM flavors WHERE name LIKE ?', ('%' + name + '%',))
    return cursor.fetchall()
```

```
# Function to filter flavors by seasonality
def filter_flavors(seasonal):
    """
    Filters flavors based on whether they are seasonal.

    Args:
        seasonal (bool): Indicates if the flavors to filter should be seasonal.

    Returns:
        list: List of matching flavors.
    """
    cursor.execute('SELECT * FROM flavors WHERE seasonal = ?', (seasonal,))
    return cursor.fetchall()

# Function to get items in a cart
def get_cart_items(cart_id):
    """
    Retrieves the items in a customer's cart.

    Args:
        cart_id (int): ID of the cart.

    Returns:
        list: List of items in the cart.
    """
    cursor.execute('SELECT f.name, f.description FROM cart_items ci JOIN flavors f ON ci.flavor_id
= f.id WHERE ci.cart_id = ?', (cart_id,))
    return cursor.fetchall()

# Main function to run the application
def main():
    """
    Main function to run the interactive command-line interface for the ice cream parlor
application.
    """
    while True:
        print('Ice Cream Parlor Cafe Application')
        print('-------------------------------')
        print('1. Add flavor')
        print('2. Add ingredient')
        print('3. Add customer suggestion')
        print('4. Add allergen')
        print('5. Create cart')
        print('6. Add to cart')
        print('7. Search flavors')
        print('8. Filter flavors')
        print('9. View cart')
        print('10. Exit')
        choice = input('Choose an option: ')

        if choice == '1':
            name = input('Enter flavor name: ')
            description = input('Enter flavor description: ')
            seasonal = input('Is this a seasonal flavor? (y/n): ')
            add_flavor(name, description, seasonal.lower() == 'y')
            print('Flavor added successfully!')

        elif choice == '2':
            name = input('Enter ingredient name: ')
            add_ingredient(name)
            print('Ingredient added successfully!')

        elif choice == '3':
            name = input('Enter customer name: ')
            suggestion = input('Enter customer suggestion: ')
            add_customer_suggestion(name, suggestion)
            print('Customer suggestion added successfully!')

        elif choice == '4':
            name = input('Enter allergen name: ')
            add_allergen(name)
            print('Allergen added successfully!')

        elif choice == '5':
            customer_name = input('Enter customer name: ')
            cart_id = create_cart(customer_name)
            print(f'Cart created successfully with ID: {cart_id}')

        elif choice == '6':
            cart_id = int(input('Enter cart ID: '))
            flavor_id = int(input('Enter flavor ID: '))
            add_to_cart(cart_id, flavor_id)
            print('Item added to cart successfully!')

        elif choice == '7':
            name = input('Enter flavor name to search: ')
            results = search_flavors(name)
```

```
            if results:
                print('Search results:')
                for row in results:
                    print(f'ID: {row[0]}, Name: {row[1]}, Description: {row[2]}, Seasonal: {row[3]}')
            else:
                print('No results found.')

        elif choice == '8':
            seasonal = input('Filter by seasonal flavors? (y/n): ')
            results = filter_flavors(seasonal.lower() == 'y')
            if results:
                print('Filter results:')
                for row in results:
                    print(f'ID: {row[0]}, Name: {row[1]}, Description: {row[2]}, Seasonal: {row[3]}')
            else:
                print('No results found.')

        elif choice == '9':
            cart_id = int(input('Enter cart ID: '))
            results = get_cart_items(cart_id)
            if results:
                print('Cart items:')
                for row in results:
                    print(f'Name: {row[0]}, Description: {row[1]}')
            else:
                print('No items in cart.')

        elif choice == '10':
            break

        else:
            print('Invalid option. Please try again.')

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

WORKING:

 Database Connection and Table Creation:

- The application connects to an SQLite database named ice_cream_parlor.db.
- It creates the necessary tables (flavors, ingredients, flavor_ingredients, customer_suggestions, allergens, flavor_allergens, carts, cart_items) if they do not already exist.

 Function Definitions:

- **add_flavor**: Adds a new flavor to the flavors table.
- **add_ingredient**: Adds a new ingredient to the ingredients table.
- **add_flavor_ingredient**: Links an ingredient to a flavor in the flavor_ingredients table.
- **add_customer_suggestion**: Adds a customer suggestion to the customer_suggestions table.
- **add_allergen**: Adds a new allergen to the allergens table.
- **add_flavor_allergen**: Links an allergen to a flavor in the flavor_allergens table.
- **create_cart**: Creates a new shopping cart for a customer in the carts table.
- **add_to_cart**: Adds a flavor to a customer's cart in the cart_items table.
- **search_flavors**: Searches for flavors by name in the flavors table.
- **filter_flavors**: Filters flavors based on whether they are seasonal.
- **get_cart_items**: Retrieves the items in a customer's cart.

 Main Function:

- Provides an interactive command-line interface for the user.
- The user can choose options to add flavors, ingredients, customer suggestions, allergens, create carts, add items to carts, search and filter flavors, and view cart items.
- The loop continues to prompt the user until they choose to exit by selecting option 10.