L7 Informatics Internship Program Assignment

Project Title: IceCream Delight Manager

1.Overview

This project is an Ice Cream Shop API and a user interface built with Flask for the backend and Streamlit for the frontend. It allows users to search for flavors, add new flavors, add allergens to specific flavors, and manage a shopping cart.

The backend is implemented using Flask, SQLAlchemy for ORM, and SQLite as the database. The frontend is built using Streamlit, which interacts with the API to display and manage flavors and allergens.

2.Technologies Used

- **Backend**: Flask, SQLAlchemy, SQLite
- **Frontend**: Streamlit, Requests
- **Database**: SQLite
- **Containerization**: Docker
- **Testing**: Postman

3.Features

- **Search Flavors**: Allows users to search for ice cream flavors.
- **Add Flavor**: Adds new flavors to the database.
- **Add Allergen**: Adds allergens to specific flavors.
- **Shopping Cart**: Manages a shopping cart for users to add and view flavors

4.Project Structure

Copy code

```
├── app.py            Flask API for handling flavor and allergen logic
├── models.py         SQLAlchemy models for Flavor, Allergen, and Cart
├── ui.py             Streamlit UI for interacting with the backend
├── Dockerfile        Dockerfile for building the application container
└── requirements.txt
└── Readme.md Python dependencies
```

Directory structure:

Name	Status	Date modified	Type	Size
__pycache__	✓	09-12-2024 13:24	File folder	
app	✓	09-12-2024 13:24	File folder	
instance	✓	09-12-2024 15:23	File folder	
app	✓	09-12-2024 12:10	Python File	2 KB
Dockerfile	✓	09-12-2024 15:35	File	1 KB
models	✓	09-12-2024 10:47	Python File	1 KB
Readme	✓	09-12-2024 15:53	Markdown Source File	0 KB
requirements	✓	09-12-2024 15:57	Text Document	1 KB
ui	✓	09-12-2024 14:17	Python File	5 KB

5.Install Dependencies

Install the required Python dependencies using pip:

pip install -r requirements.txt
The requirements.txt file should include:

```
Flask==2.2.3
Flask-SQLAlchemy==2.5.1
requests==2.28.1
streamlit==1.10.0
altair>=4.2.2
```

6.Running the Application

Backend (Flask)

To run the Flask API, use the following command:
python app.py
This will start the Flask API at http://127.0.0.1:5000/.

Frontend (Streamlit)

To run the Streamlit UI, use the following command:

streamlit run ui.py

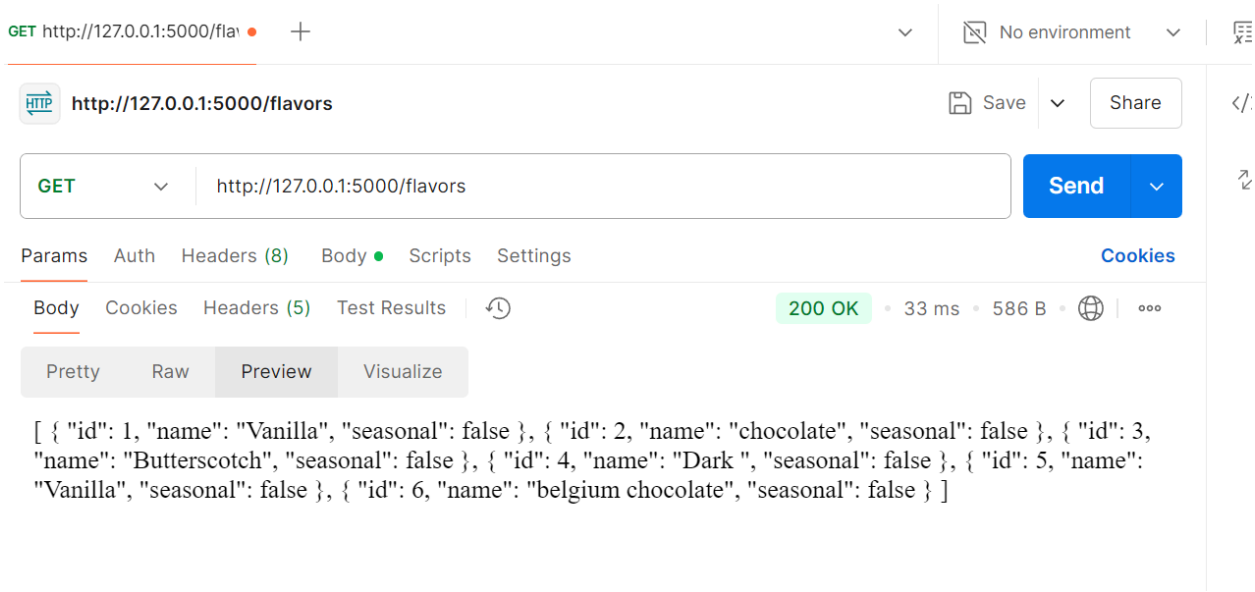
The frontend will be available at http://localhost:8501.

Testing the Application

Using Postman

GET /flavors: Fetch all ice cream flavors.

- Request: GET http://127.0.0.1:5000/flavors



- Response: A list of all flavors in JSON format.

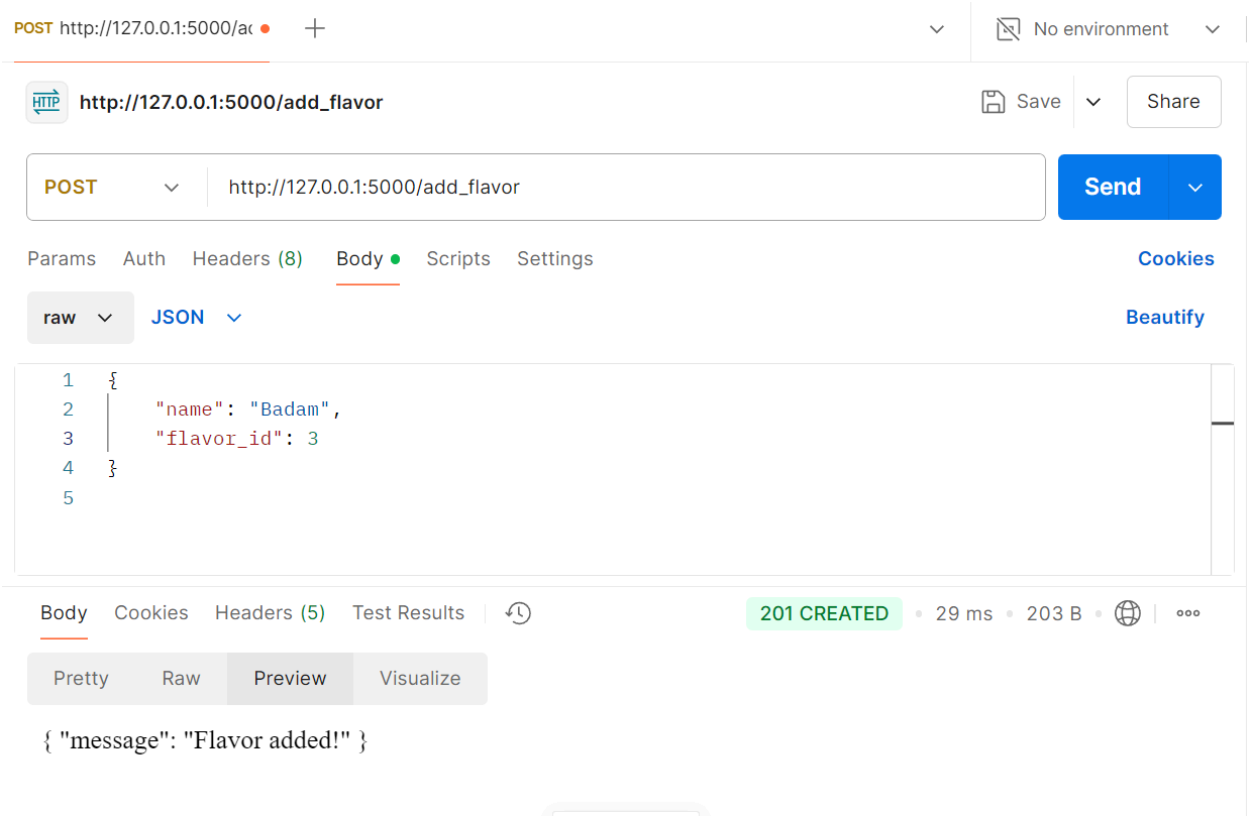
POST /add_flavor: Add a new flavor.

- Request: POST http://127.0.0.1:5000/add_flavor

Json :

```
{
  "name": "Chocolate",
  "seasonal": false
}
```

- Response: { "message": "Flavor added!" }

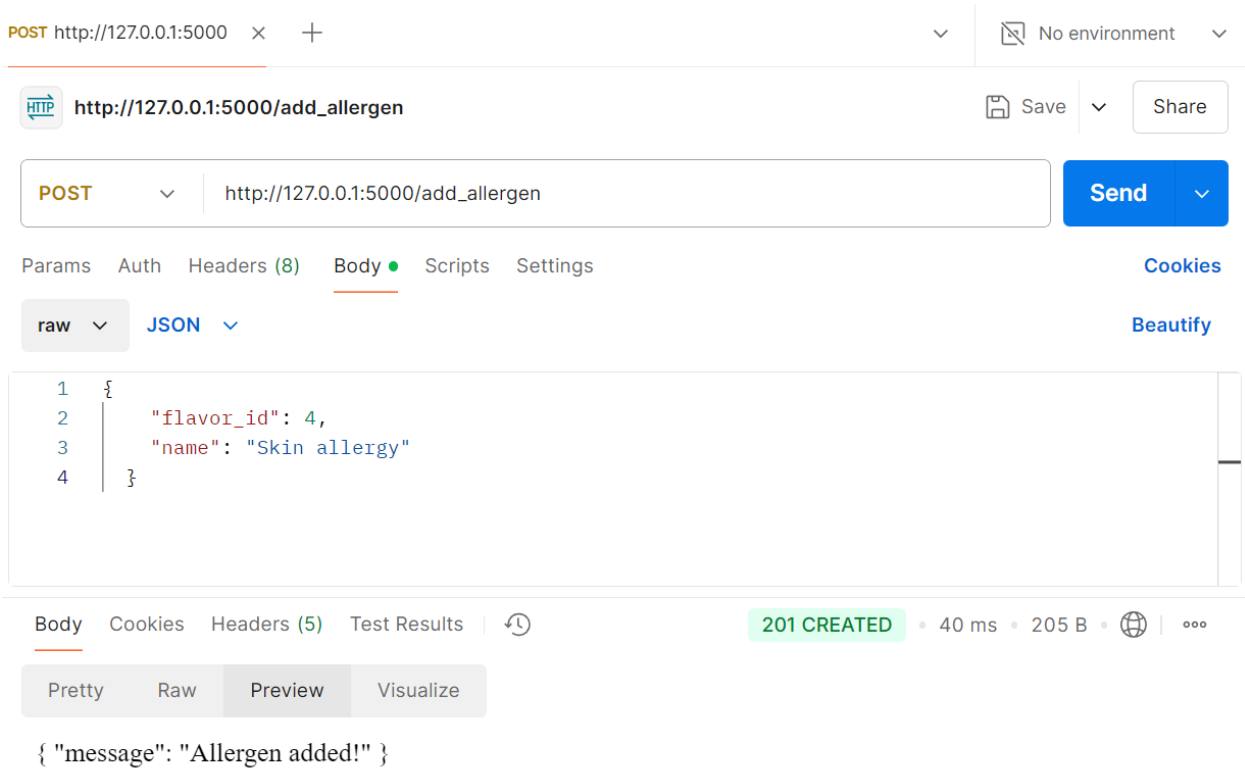


POST /add_allergen: Add an allergen to a specific flavor.

- Request: POST http://127.0.0.1:5000/add_allergen

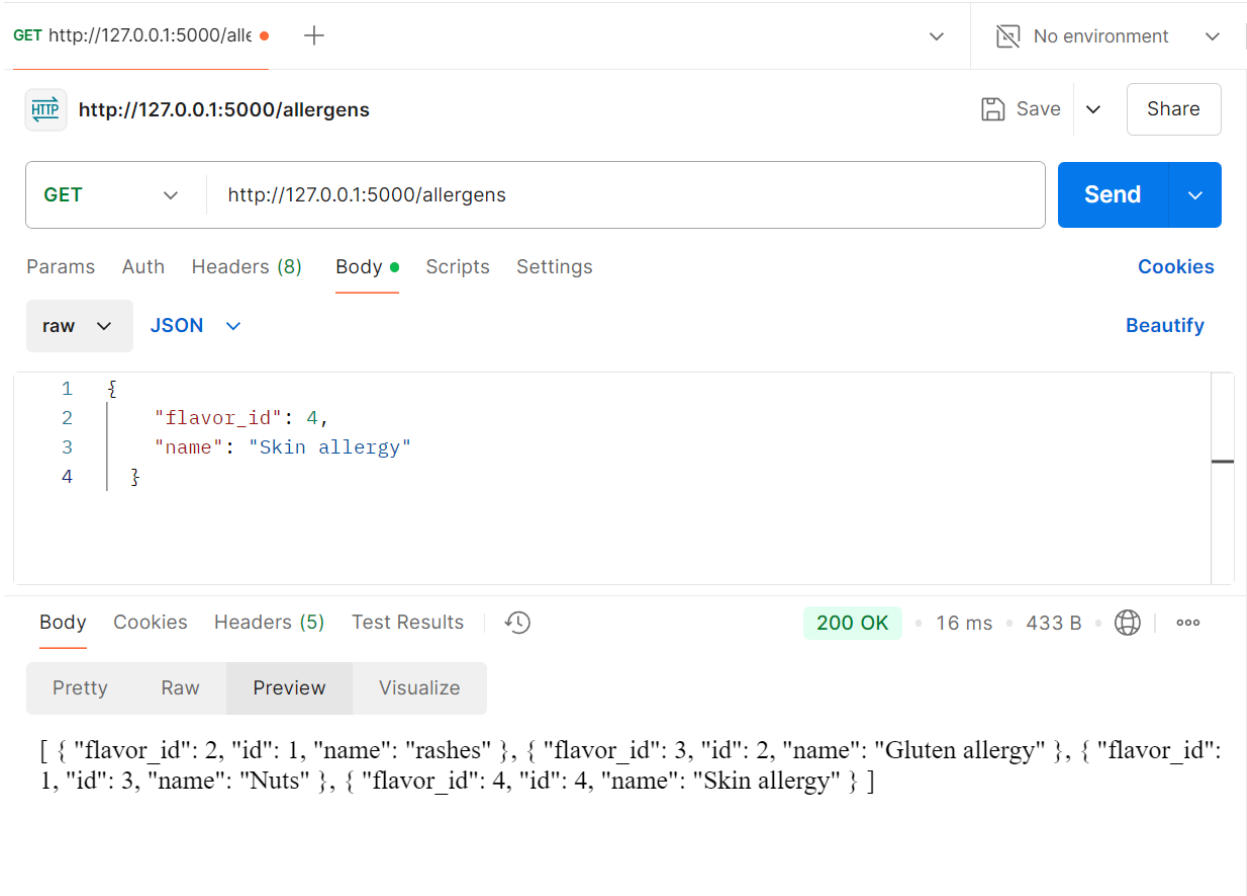
Json:
{
 "name": "Peanut",
 "flavor_id": 1
}

- Response: { "message": "Allergen added!" }



GET /allergens: Fetch allergens or allergens for a specific flavor.

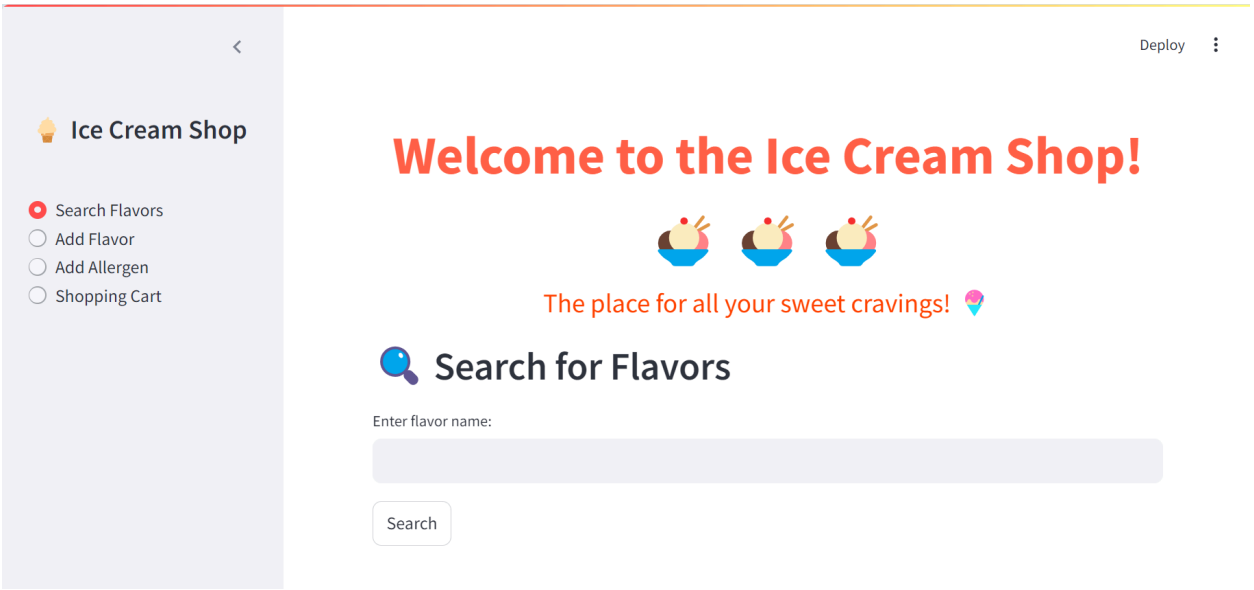
- Request: GET http://127.0.0.1:5000/allergens?flavor_id=1
- Response: A list of allergens for the specified flavor.

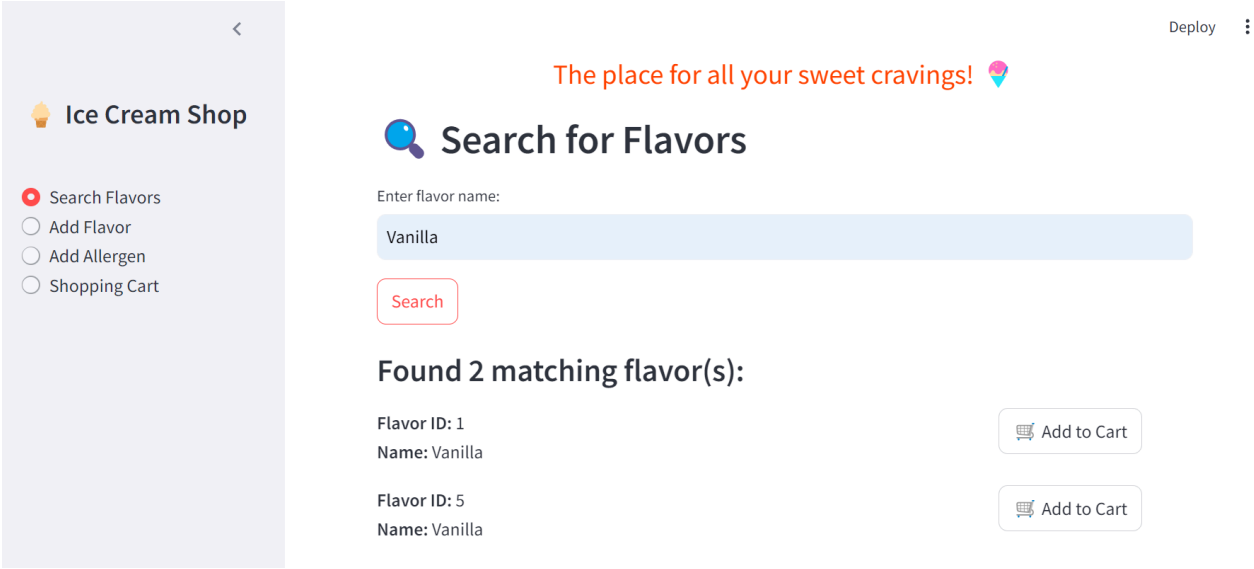


7.Using Streamlit UI

- **Search Flavors**: Enter a flavor name to search for and display results.
- **Add Flavor**: Add new flavors to the system.
- **Add Allergen**: Add allergens to specific flavors.
- **Shopping Cart**: View and manage the shopping cart by adding flavors.

User Interface Screenshots





Deploy

Ice Cream Shop

- Search Flavors
- Add Flavor
- Add Allergen
- Shopping Cart

The place for all your sweet cravings!

Add a New Flavor

New Flavor Name:

Blackcurrent

Add Flavor

Flavor added!

Deploy

Ice Cream Shop

- Search Flavors
- Add Flavor
- Add Allergen
- Shopping Cart

The place for all your sweet cravings!

Add Allergen

Allergen Name:

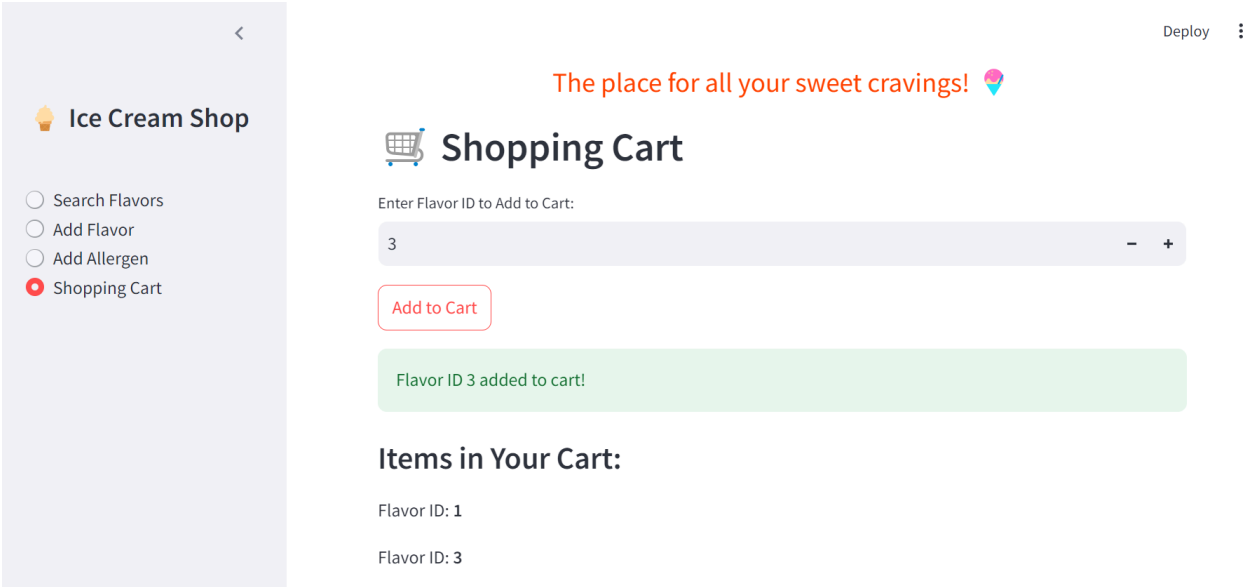
Chocolate allergy

Flavor ID:

2

Add Allergen

Allergen added!



8.Docker Setup

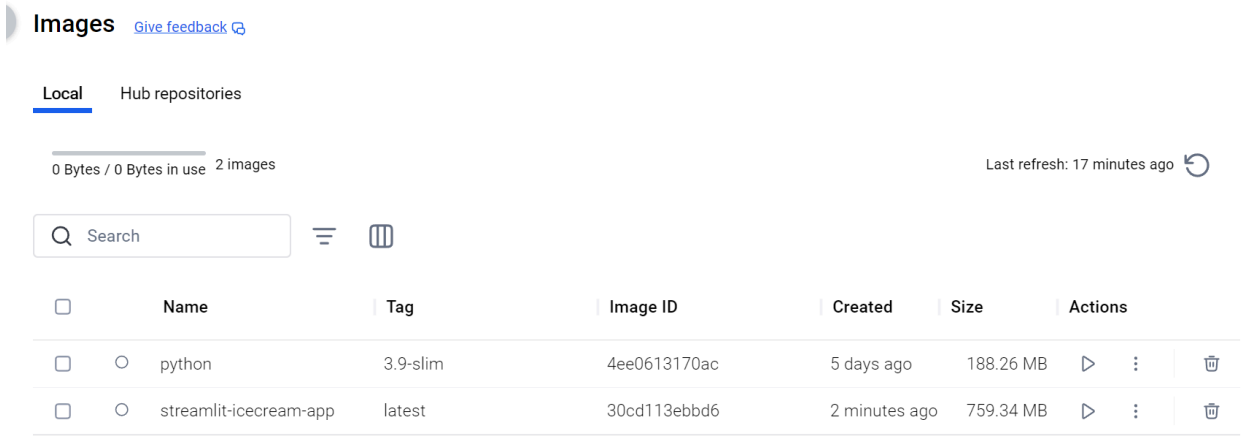
1. Dockerfile : The Dockerfile is used to create a Docker image for the application.

```
# Use an official Python runtime as the base image
FROM python:3.9-slim

# Set the working directory in the container
WORKDIR /app

# Copy the requirements file to the container
COPY requirements.txt requirements.txt

# Install dependencies
RUN pip install --no-cache-dir -r requirements.txt

# Copy the application files into the container
COPY . .

# Expose the port Streamlit will run on
EXPOSE 8501

# Run the Streamlit app
CMD ["streamlit", "run", "ui.py", "--server.port=8501",
"--server.address=0.0.0.0"]
```

2. Building and Running the Docker Container

Build the Docker image:

docker build -t streamlit-icecream-app .



Run the container:

docker run -p 8501:8501 streamlit-icecream-app

This will expose the Flask API on port 5000 and the Streamlit UI on port 8501. You can now access both through your browser.

<u>SQL Query or ORM Abstraction Implementation</u>

The backend uses SQLAlchemy ORM to interact with the SQLite database. Below are the queries handled through SQLAlchemy:

- **Fetch Flavors**:
  Flavor.query.all()
- **Add a Flavor**:
  new_flavor = Flavor(name="Chocolate", seasonal=False)
  db.session.add(new_flavor)
  db.session.commit()
- **Fetch Allergens for a Flavor**:
  Allergen.query.filter_by(flavor_id=flavor_id).all()
- **Add an Allergen**:
  new_allergen = Allergen(name="Peanut", flavor_id=flavor_id)
  db.session.add(new_allergen)
  db.session.commit()

8.Code Documentation

app.py

- **home()**: A welcome route for the application.
- **get_flavors()**: Fetches all ice cream flavors.
- **add_flavor()**: Adds a new flavor to the database.
- **add_allergen()**: Adds an allergen to a flavor.
- **get_allergens()**: Fetches allergens, optionally filtered by flavor_id.
- **search():**Search icecream name by using longersubsequence

models.py

- **Flavor Model**: Represents an ice cream flavor with name, seasonal, and a relationship with allergens.
- **Allergen Model**: Represents an allergen with name and a foreign key to the Flavor model.
- **Cart Model**: (Optional) Represents the shopping cart functionality, linking flavors to user carts.

ui.py

- **Fetch Flavors**: Fetches flavors from the Flask API.
- **Add Flavor**: Sends a POST request to add a new flavor.
- **Add Allergen**: Sends a POST request to add an allergen.
- **Shopping Cart**: Manages a session-based shopping cart using Streamlit.

9.Conclusion

This application allows users to explore and manage ice cream flavors and allergens through a Flask API and interact with the API via a user-friendly Streamlit interface. The project also provides Docker containerization for easy deployment.