Assignment Description:

Part 1 – Base Code: CRUD Operations:

Ice Cream Menu structure

The server will use file-based data storage to store the menu. The menu items will be stored as a JSON list. Each ice cream specialty on the menu will be represented as a JSON object. An ice-cream specialty has the following attributes:

• code: A string of digits that uniquely identifies the item.

• name: a string that represents the name of the ice cream specialty.

• ingredients: A list of strings representing the ingredients of the ice cream specialty.

• price: A number representing the price of the ice cream scoop.

• availability: A Boolean indicating whether the item is available or has been sold out.

Server Endpoints

The following endpoints are defined for the server:

1. /welcome: o Description: Displays a welcome message to the client. This endpoint will be the default route for the server and will implement only the GET operation. o Details: The server will retrieve a welcome text message from a text file and send it to the client using streams and pipes, after appropriately setting response headers. The server needs to handle all errors on this path.

2. /menu - /menu/{code}: o Description: Implements the CRUD operations on the ice cream menu. Operations will be sent to the server using POSTMAN. The server code needs to handle all errors on this path for all CRUD operations.

o CRUD Operations:

1. Create:

Endpoint: /menu or /menu/ Details: When the server starts for the first time, it should not assume that the menu file exists. The file should be created with the first POST request. When the server receives a POST request, it parses the query string of the request to get the details of the ice cream specialty and adds it to the menu file. The server needs to ensure that each item added to the menu is unique and should not allow duplication. After the item is added, the server needs to reply with status code 200 and the string ‘OK’ to the client. If the item already exists in the menu, the server needs to reply with status code 400 and the string ‘BAD REQUEST’ to the client.

1. Read:

Endpoint: /menu or /menu/ Details: When the server receives a GET request, it will read the menu file and send it to the client using streams and pipes after appropriately setting response headers. The server needs to reply with status code 200 and a JSON list representing the menu. If the menu file does not exist, the server needs to reply with the status code 404 and the string ‘Not Found’ to the client.

1. Update:

Endpoint: /menu/{code} or /menu/{code}/

Description: Updates one of the ice cream items identified by {code}, where {code} is the ice cream code that uniquely identifies the ice cream. Details: For an ice cream specialty, three attributes can be updated: price, ingredients, and availability, or any combination thereof. It can be assumed that only these three attributes will be sent in the client request. For each update operation, the menu will be retrieved from the JSON file, the specified attribute of the ice cream specialty will be updated, and then the menu will be saved back to the file. After the item is updated, the server needs to reply with status code 200 and the string ‘OK’ to the client. If the menu file does not exist, or the item to be updated does not exist in the menu, the server needs to reply with the status code 404 and the string ‘Not Found’ to the client.

1. Delete:

Endpoint: /menu/{code} or /menu/{code}/

Description: Deletes one of the ice cream items identified by {code}, where {code} is the ice cream code that uniquely identifies the ice cream. Details: The manager at Chilly Delights might decide to stop serving one of the ice cream specialties on the menu. Accordingly, the server needs to allow deletions from the menu. For deletions, the ice cream specialty to be deleted is identified by its code. The server will retrieve the menu, delete the specified item, and then store the menu back to the file. After the item is deleted, the server needs to reply with status code 200 and the string ‘OK’ to the client. If the menu file does not exist, or the item to be deleted does not exist in the menu, the server needs to reply with the status code 404 and the string ‘Not Found’ to the client.

Setup and Implementation Hints for Part 1:

1. Use the code developed in Lab 8 as a reference.

2. The welcome file is stored in the server working path.

3. Provide two helper functions for retrieving and storing the menu. Use these functions for POST, DELETE and PUT operations.

4. Use streams and pipes for GET method.

5. Minimize the nesting of callbacks

6. Provide code to catch any invalid path under ‘/menu’ and respond with ‘400 BAD REQUEST’

7. Develop the server code incrementally and always check that the response is received by POSTMAN.

Test Cases for Part 1:

Following are values for the ‘key:value’ pairs used for inserting ice cream items in the menu for the POST method.

code: 10233

name: Honey Dew Sorbet

ingredient: [‘Honey Dew’]

code: 10234

name: Banana Pudding

code: 10345

name: Strawberry Lemonade Sorbet

price: 17

avail: true

ingredient: [‘Banana’, ‘Vanilla’, ‘Wafer Cookies’]

price: 15

avail: true

ingredient: [‘Lemonade’, ‘Organic Strawberries’]

price: 12

avail: true

code: 10232

name: Coconut Almond Brownie

ingredient: [‘Coconut’, ‘Almond’, ‘Brownie’]

price: 17

avail: true

code: 10202

name: Black Raspberry Chip

ingredient: [‘Black Raspberry’, ‘Chocolate’]

price: 13

avail: true

code: 10237

name: Summer Peach Melba

ingredient: [‘Peach’, ‘Raspberry’]

price: 18

avail: true

The snapshots provided below are a sample of the expected responses received on POSTMAN. For the following test cases, it is assumed that the server is running on port 3030.

/welcome Endpoint: Note that with GET method, regardless of the pathname provided after the ‘/’ the server defaults to the welcome message.

Part 2 – Skill Demonstration: Adding a New Route

The manager at Chilly Delights would like to add the functionality of displaying an ice cream item by its code. Accordingly, you are required to add a new route to provide this functionality. The new route description and details are given below.

/menu/{code}:

o Description: Displays one of the ice cream items identified by {code}, where {code} is

the ice cream code that uniquely identifies the ice cream.

o Details: For this endpoint, the server will retrieve the menu from the file, locate the ice

cream item by its code, and respond to the client with status code 200 and a JSON object

representing the ice cream item. If the menu file does not exist, or the item to be

displayed does not exist in the menu, the server needs to reply with the status code 404

and the string ‘Not Found’ to the client.

Test Cases for Part 2: /menu/{code} Endpoint: The server accepts ‘/menu/{code}’ or

‘/menu/{code}/’ as the pathname for the GET method in addition to the PUT an

DELETE methods. - Displaying an ice cream item by code