Service Layer Design

Kailee Stenseng

John E. Simon School of Business, Maryville University

SWDV 691: Software Development Capstone

Professor Joseph Gradecki

March 23, 2025

Service Layer Design

There are a variety of service endpoints that are required for the Niibish Aki web application MVP. The service layer facilitates interaction between the user interface and database by abstracting the underlying data operations. The backend of Niibish Aki will be implemented using Next.JS API routes, which handles HTTP requests within the /pages/api directory, and will use . Inside of this directory, there will be three subfolders for users, orders, and menu, and each API route corresponds to a file within their designated directories. To provide enhanced security, data integrity, and efficiency, middleware, likely NextAuth.js (I am still researching this), will be added to ensure users are authenticated for sensitive endpoints.

Specifications

The endpoints and routes that compose my service layer are listed below. I am hosting this application on Heroku, although all of the API routes are still in planning and has not been implemented yet. Only the home page exists at the moment, and there is no functionality in it, yet. I created a mocked Swagger page using https://editor.swagger.io/ and the exported YAML file is attached in the assignment submission.

User

Get User:

Method: GET

URL: https://niibish-aki-e5d55d879400.herokuapp.com/api/user/{userId}

Purpose: When the user goes to view their account settings (or address, but it's technically the same page), it will call this endpoint to ensure the data they see is the

most up to date. Once their data is processed and filled into the userContext, the data will be refreshed.

Example Request:

```
curl --request GET --url https://niibish-aki-e5d55d879400.herokuapp.com/api/user/660a2b3c4d5e6f7890123456
```

Responses:

Code	Description	Links
200	Successfully retrieved user info Media type: application/json Controls Accept header. Example Value | Schema	*No links*
400	Invalid userId specified Media type: application/json Examples: Response Example Value	*No links*
404	User not found Media type: application/json Examples: Response Example Value	*No links*


```
{
  "_id": "string",
  "firstName": "string",
  "lastName": "string",
  "userName": "string",
  "password": "string",
  "email": "user@example.com",
  "phone": "string",
  "address": {
    "line1": "string",
    "line2": "string",
    "city": "string",
    "state": "string",
    "zipcode": "string"
  }
}
```

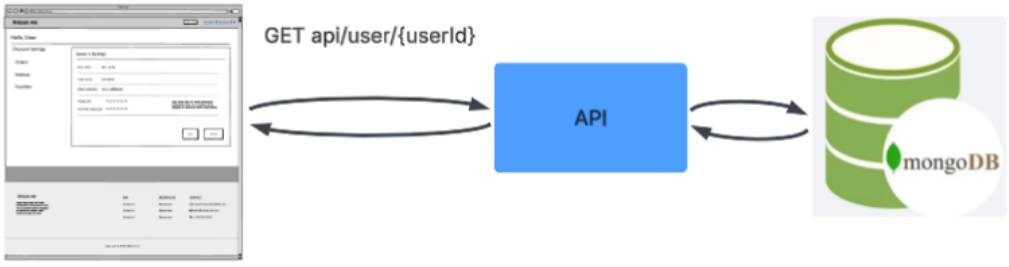


```
{
  "message": "Invalid user specified."
}
```



```
{
  "message": "User not found."
}
```


Diagram:



Update User:

Method: PUT

URL: https://niibish-aki-e5d55d879400.herokuapp.com/api/user/{userId}

Purpose: When the user goes to edit their account settings (or address, but it's technically the same page), it will call this endpoint once they click "Save". The new data will populate on the page once complete.

Example Request:

```
curl --request PUT --url https://niibish-aki-
e5d55d879400.herokuapp.com/api/user/660a2b3c4d5e6f7890123456 \
--header "Content-Type: application/json" \
--data '{
"firstName": "Jane",
"lastName": "Doe",
"userName": "janedoe",
"email": "janedoe@example.com",
"phone": "555-555-5556",
"address": {
"line1": "124 Main St",
"line2": "Apt 4C",
"city": "Springfield",
"state": "IL",
"zipcode": "62702"
}
}'
```

Responses:

Code	Description	Links
200	User successfully updated Media type: application/json Controls Accept header. Example Value | Schema	*No links*

```
{
  "_id": "string",
  "firstName": "string",
  "lastName": "string",
  "userName": "string",
  "password": "string",
  "email": "user@example.com",
  "phone": "string",
  "address": {
    "line1": "string",
    "line2": "string",
    "city": "string",
    "state": "string",
    "zipcode": "string"
  }
}
```

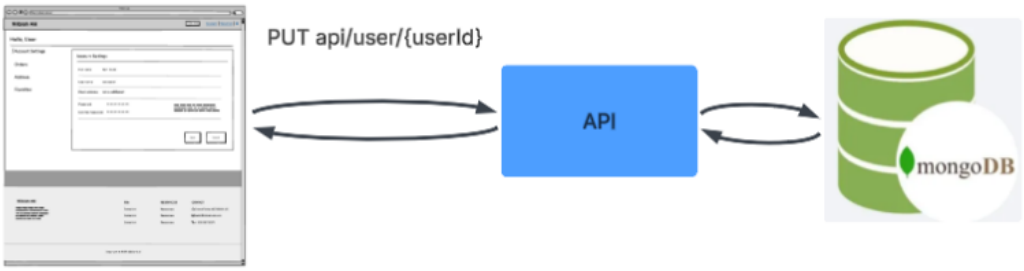
400	Invalid user data Media type: application/json Examples: Response Example Value	*No links*

```
{
  "message": "Invalid user data."
}
```

404	User not found Media type: application/json Examples: Response Example Value	*No links*

```
{
  "message": "User not found."
}
```

Diagram:



Delete User:

Method: DELETE

URL: https://niibish-aki-e5d55d879400.herokuapp.com/api/user/{userId}

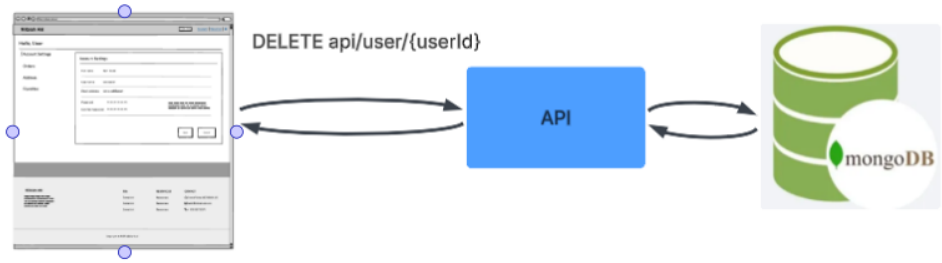
Purpose: When the user goes to delete their account settings, this will be called when they confirm they want to delete their account. Once complete, it will return them to the homepage. Note that this is a stretch goal and not part of MVP.

Example Request:

```
curl --request DELETE --url https://niibish-aki-e5d55d879400.herokuapp.com/api/user/660a2b3c4d5e6f7890123456
```

Responses:

Diagram:



Create User:

Method: POST

URL: https://niibish-aki-e5d55d879400.herokuapp.com/api/user/create

Purpose: When the user originally enters their information on a register or index page and click “Join Rewards”, their password will be hashed, and this will then create the

user in the database. Once they are created, the data will be processed and filled into the userContext, and they will be redirected to the account page.

Example Request:

```
curl --request POST --url https://niibish-aki-e5d55d879400.herokuapp.com/api/user/create \
--header "Content-Type: application/json" \
--data '{
"firstName": "John",
"lastName": "Smith",
"userName": "johnsmith",
"email": "johnsmith@example.com",
"phone": "555-555-5555",
"address": {
"line1": "567 Oak St",
"line2": null,
"city": "Peoria",
"state": "IL",
"zipcode": "61601"
}
}'
```

Responses:

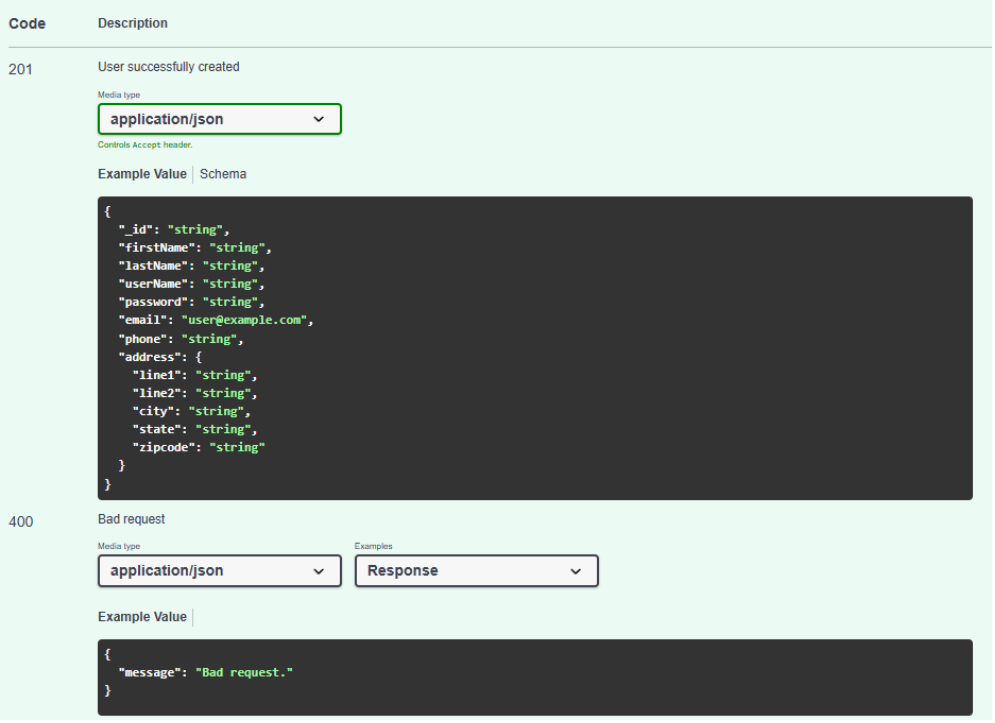
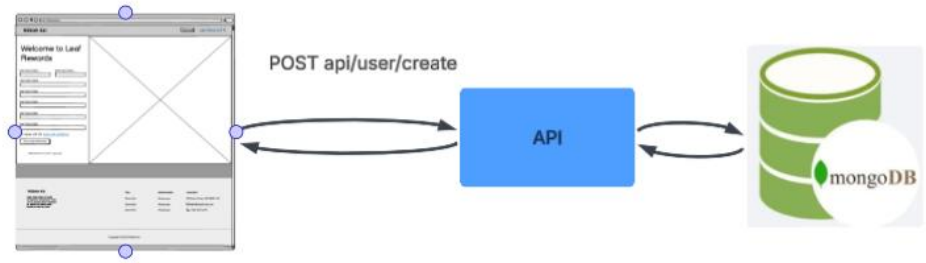


Diagram:



Login:

Method: POST

URL: https://niibish-aki-e5d55d879400.herokuapp.com/api/user/login

Purpose: When the user goes to sign into their account, this calls once they click on the "Login" button. Their password will be hashed. Once their data is processed and filled into the userContext, they will be redirected to their account page.

Example Request:

```
curl --request POST --url https://niibish-aki-
e5d55d879400.herokuapp.com/api/user/login \
--header "Content-Type: application/json" \
--data '{
"userName": "johnsmith",
"password": "securepassword123"
}'
```

Responses:

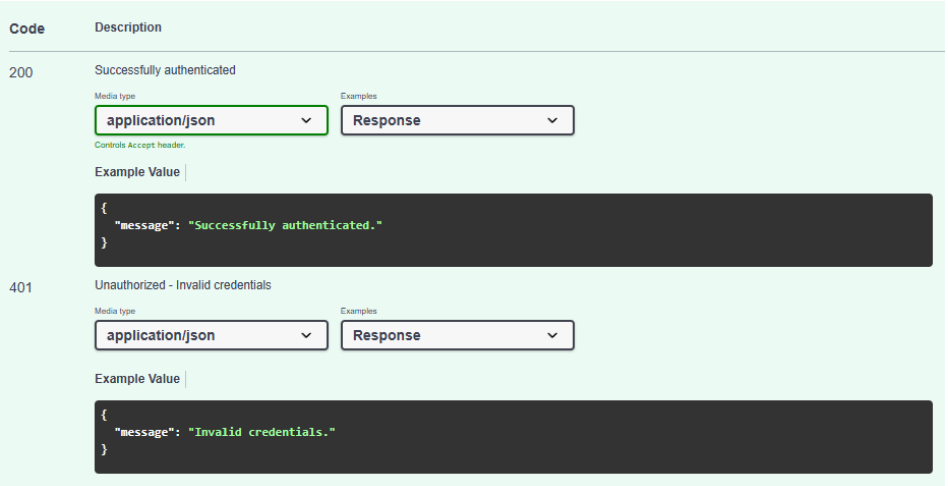
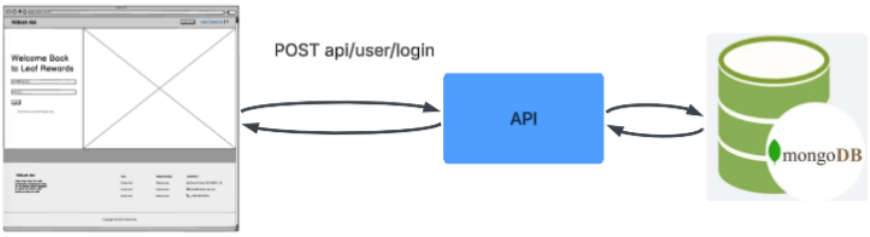


Diagram:



Get user favorites:

Method: GET

URL: https://niibish-aki-e5d55d879400.herokuapp.com/api/user/favorites/{userId}

Purpose: When the user goes to view their favorites on their account page, this will get called. Once the call is complete, it will populate their favorites page.

Example Request:

```
curl --request GET --url https://niibish-aki-
e5d55d879400.herokuapp.com/api/user/favorites/660a2b3c4d5e6f7890123456
\
--header "Content-Type: application/json" \
--data ' [
  {
    "_id": "string",
    "menuId": "string",
    "userId": "string",
    "name": "string",
    "addIns": [
      {
        "id": "string",
        "name": "string",
        "amount": 0,
        "price": 0
      }
    ],
    "price": 0
  }
]'
```

Responses:

Code | Description

200 — Successfully retrieved favorites

Media type: application/json

Controls Accept header.

Example Value | Schema

```
[
  {
    "_id": "string",
    "menuId": "string",
    "userId": "string",
    "name": "string",
    "addins": [
      {
        "id": "string",
        "name": "string",
        "amount": 0,
        "price": 0
      }
    ],
    "price": 0
  }
]
```

400 — Invalid userId specified

Media type: application/json | Examples: Response

Example Value

```
{
  "message": "Invalid userId specified."
}
```

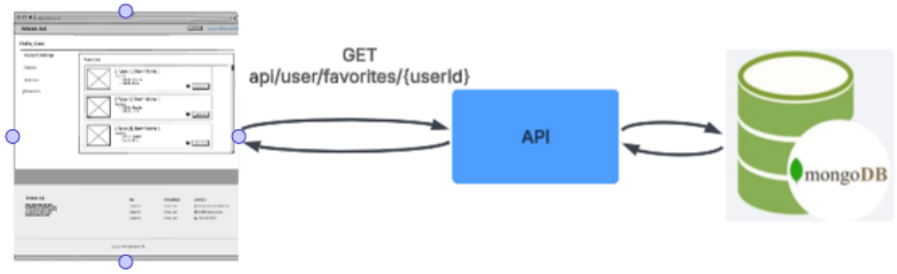
404 — User not found

Media type: application/json | Examples: Response

Example Value

```
{
  "message": "User not found."
}
```

Diagram:



Add user favorite:

Method: POST

URL: https://niibish-aki-e5d55d879400.herokuapp.com/api/user/favorites/{userId}

Purpose: When the user goes to view their orders on their account page and clicks on the heart icon on the item they wish to favorite, this will get called. Once the call is complete, it will stay on the orders page and they will see the update when they navigate to the favorites page.

Example Request:

```
curl --request POST --url https://niibish-aki-
e5d55d879400.herokuapp.com/api/user/favorites/660a2b3c4d5e6f7890123456
\
--header "Content-Type: application/json" \
--data ' {
  "_id": "string",
  "menuId": "string",
  "userId": "string",
  "name": "string",
  "addIns": [
    {
      "id": "string",
      "name": "string",
      "amount": 0,
      "price": 0
    }
  ],
  "price": 0
}'
```

Responses:

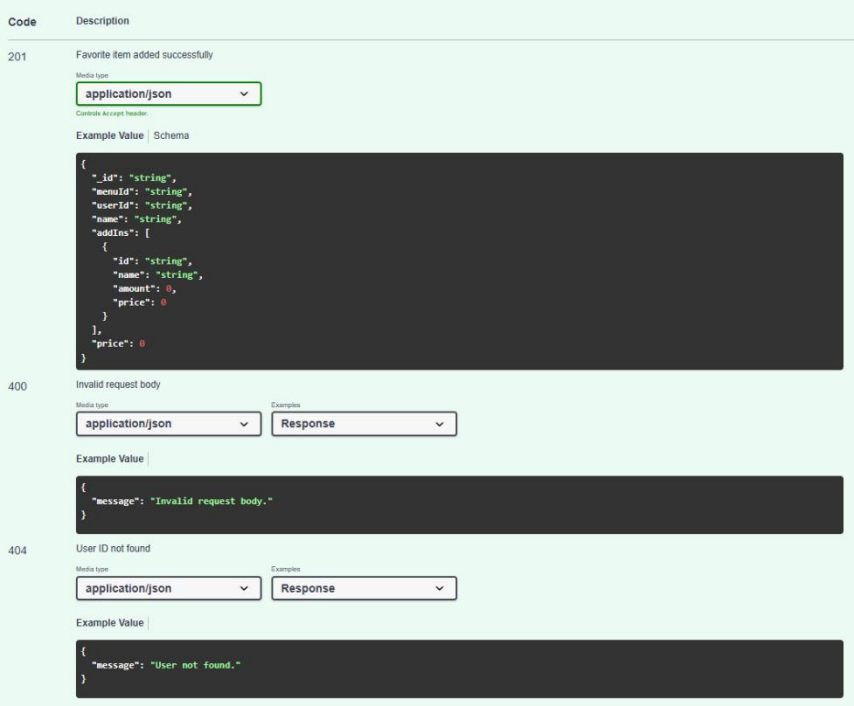
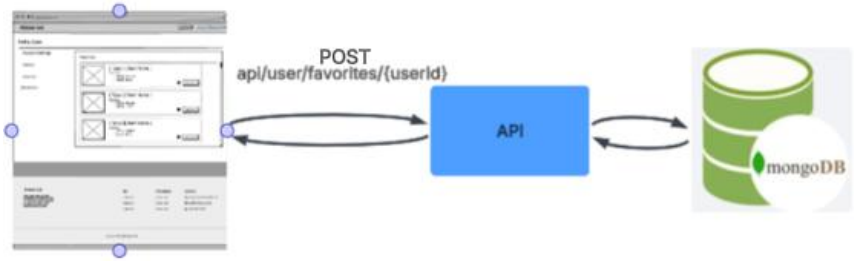


Diagram:



Remove user favorite:

Method: DELETE

URL: https://niibish-aki-e5d55d879400.herokuapp.com/api/user/favorites/{userId}

Purpose: When the user goes is viewing their favorites on the account page, this will get called when they click the filled in heart icon on a favorite. Once complete, the new list will be returned and processed on the favorites list.

Example Request:

curl --request DELETE --url https://niibish-aki-e5d55d879400.herokuapp.com/api/user/favorites/660a2b3c4d5e6f7890123456

Responses:

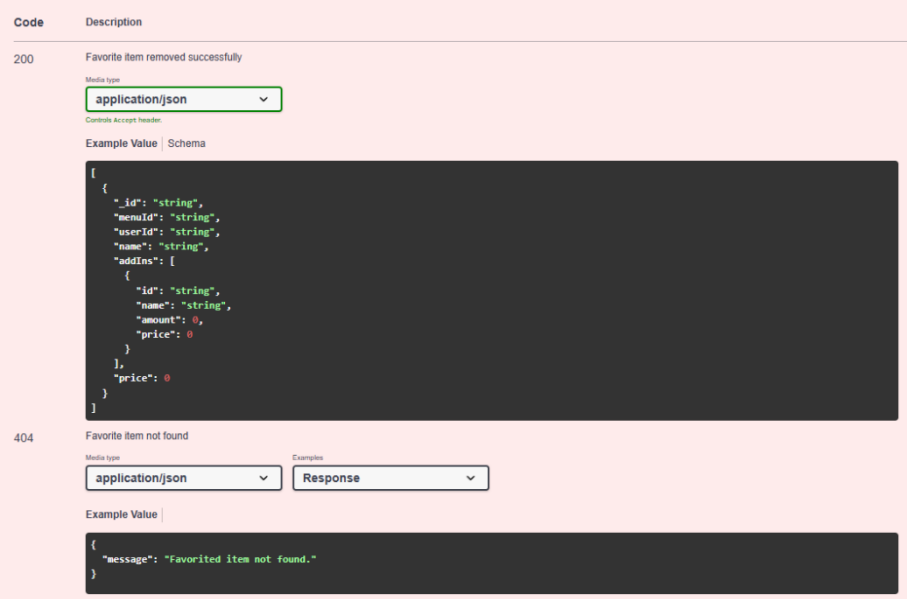
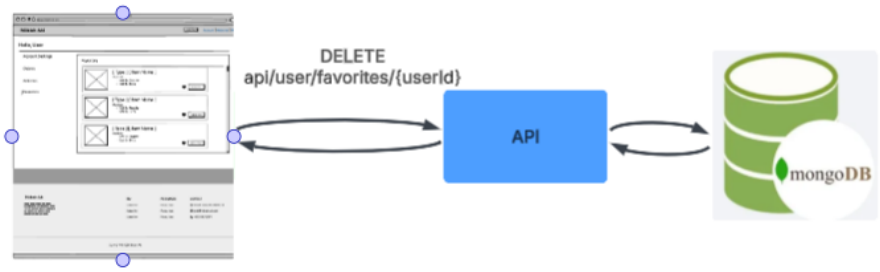


Diagram:



Order

Create order:

Method: POST

URL: https:// niibish-aki-e5d55d879400.herokuapp.com/api/order

Purpose: When the user clicks on Add To Order after customizing their drink (or reviewing the snack), this will be called if the user is signed in and there's no existing cart. Once complete, the user will be given a confirmation and they will return to the menu screen.

Example Request:

```
curl --request POST --url https://niibish-aki-
e5d55d879400.herokuapp.com/api/order \
--header "Content-Type: application/json" \
--data '{
  "_id": "string",
  "userId": "string",
  "items": [
    {
      "type": "tea",
      "itemId": "string",
      "addIns": [
        {
          "id": "string",
          "name": "string",
          "amount": 0,
          "price": 0
        }
```

```
      ],
      "quantity": 0,
      "price": 0
    }
  ],
  "bill": {
    "subtotal": 0,
    "tax": 0,
    "tip": 0,
    "total": 0
  },
  "status": "cart",
  "createdAt": "2025-03-24T05:28:55.597Z"
}'
```

Responses:

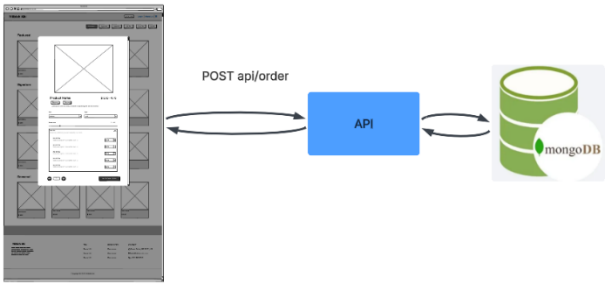
Code	Description
201	Order created successfully Media type: application/json Controls Accept header. Example Value | Schema


```
{
  "_id": "string",
  "userId": "string",
  "items": [
    {
      "type": "tea",
      "itemId": "string",
      "addIns": [
        {
          "id": "string",
          "name": "string",
          "amount": 0,
          "price": 0
        }
      ],
      "quantity": 0,
      "price": 0
    }
  ],
  "bill": {
    "subtotal": 0,
    "tax": 0,
    "tip": 0,
```


Code	Description
400	Invalid request body Media type: application/json Examples: Response Example Value


```
{
  "message": "Invalid request body."
}
```


Diagram:



Get order:

Method: GET

URL: https://niibish-aki-e5d55d879400.herokuapp.com/api/order/{orderId}

Purpose: When the user goes to checkout, this will be called to ensure their cart is up to date once starting the checkout process. After the data comes back and is processed, the user will see their checkout page.

Example Request:

```
curl --request GET --url https://niibish-aki-e5d55d879400.herokuapp.com/api/order/660a2b3c4d5e6f7890123456
```

Responses:

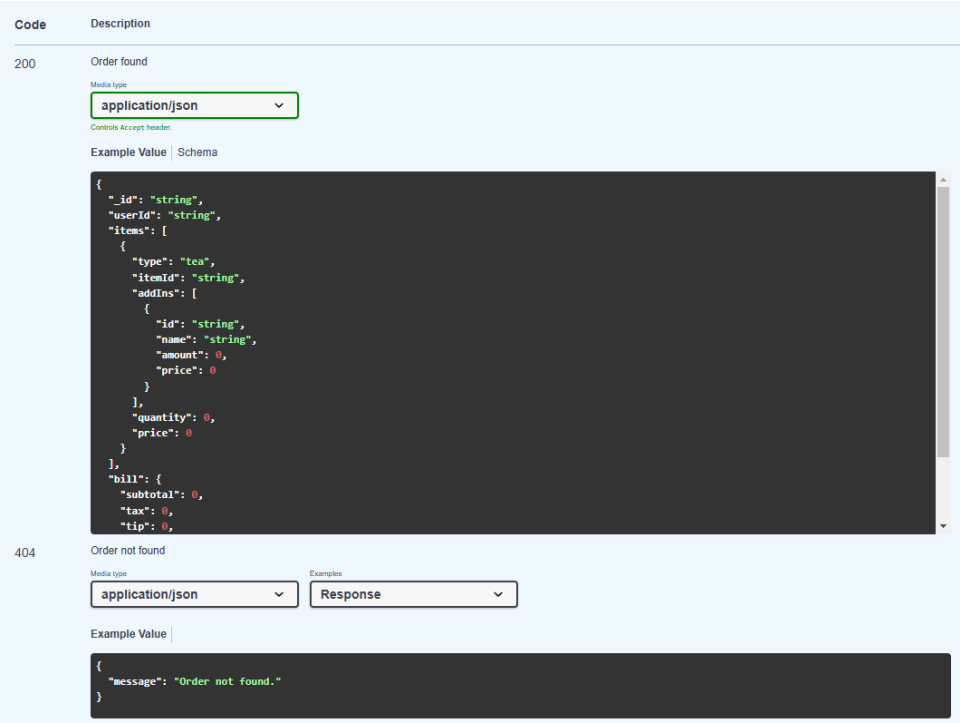
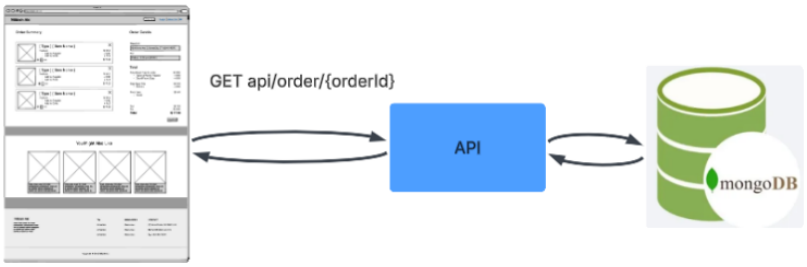


Diagram:



Update order:

Method: PUT

URL: https://niibish-aki-e5d55d879400.herokuapp.com/api/order/{orderId}

Purpose: When the user clicks on “Complete Order”, this will be called to update the order as a “complete” status, and once the data is returned and processed, it will give them a recap on the order confirmation page.

Example Request:

```
curl --request PUT --url https://niibish-aki-
e5d55d879400.herokuapp.com/api/order/660a2b3c4d5e6f7890123456\
--header "Content-Type: application/json" \
--data ' {
  "_id": "string",
  "userId": "string",
  "items": [
    {
      "type": "tea",
      "itemId": "string",
      "addIns": [
        {
          "id": "string",
          "name": "string",
          "amount": 0,
          "price": 0
        }
      ],
      "quantity": 0,
      "price": 0
    }
  ],
```

```
  "bill": {
    "subtotal": 0,
    "tax": 0,
    "tip": 0,
    "total": 0
  },
  "status": "cart",
  "createdAt": "2025-03-24T05:28:55.597Z"
}'
```

Responses:

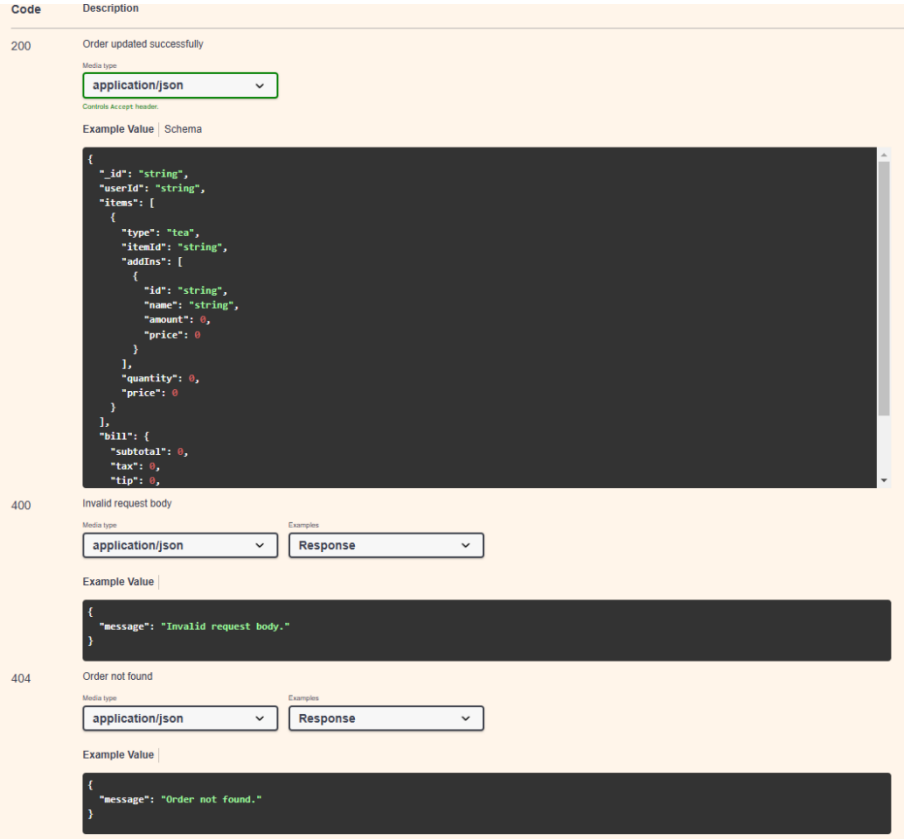
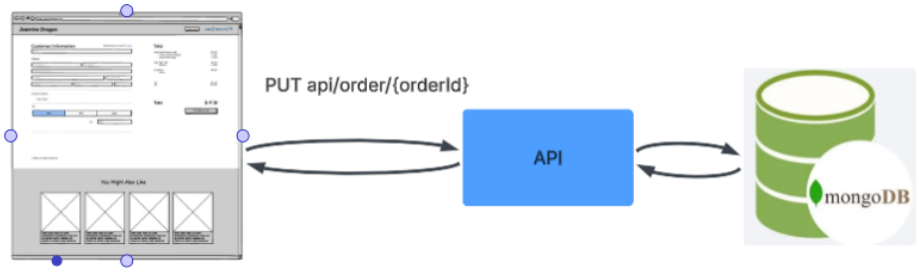


Diagram:



Remove order:

Method: DELETE

URL: https://niibish-aki-e5d55d879400.herokuapp.com/api/order/{orderId}

Purpose: When the user removes the last item in their cart, this will be called to remove their cart from the database. Upon deletion, the cart will give a message that it is empty and to add something, which they can click on to redirect them to the menu.

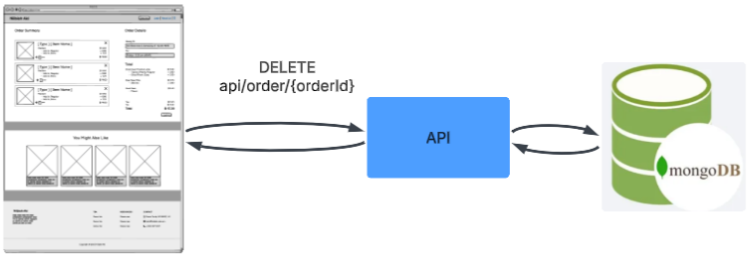
Example Request:

```
curl --request DELETE --url https://niibish-aki-e5d55d879400.herokuapp.com/api/order/660a2b3c4d5e6f7890123456
```

Responses:

Code	Description
200	Order successfully deleted Media type: application/json Examples: Response Controls Accept header. Example Value `{ "message": "Order successfully deleted." }`
404	Order not found Media type: application/json Examples: Response Example Value `{ "message": "Order not found." }`

Diagram:



Get user's orders:

Method: GET

URL: https://niibish-aki-e5d55d879400.herokuapp.com/api/ /order/user/{userId}

Purpose: When the user goes to their account page and selects Orders, this will be called to fetch their orders. Once the data is returned and processed, the user will be able to see all of their past orders.

Example Request:

```
curl --request GET --url https://niibish-aki-e5d55d879400.herokuapp.com/api/order/user/660a2b3c4d5e6f7890123456
```

Responses:

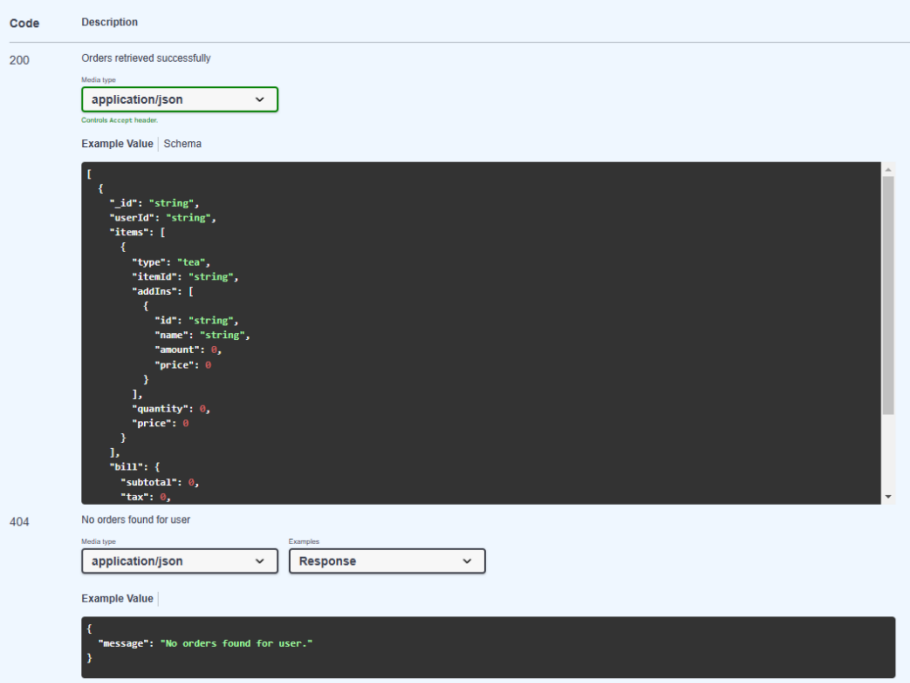
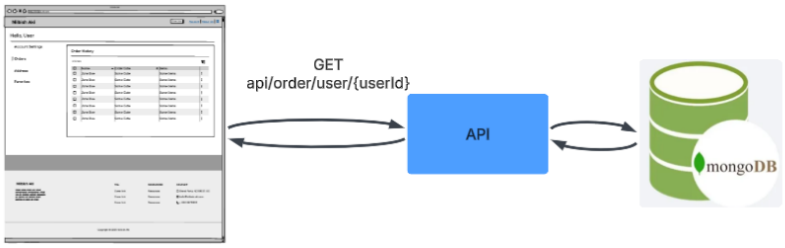


Diagram:



Menu

Get menu:

Method: GET

URL: https://niibish-aki-e5d55d879400.herokuapp.com/api/menu/

Purpose: When the user goes to the menu page, this will be called and once the data is returned and processed, the user will be able to view each item on their own product card on the page.

Example Request:

```
curl --request GET --url https://niibish-aki-e5d55d879400.herokuapp.com/api/menu
```

Responses:

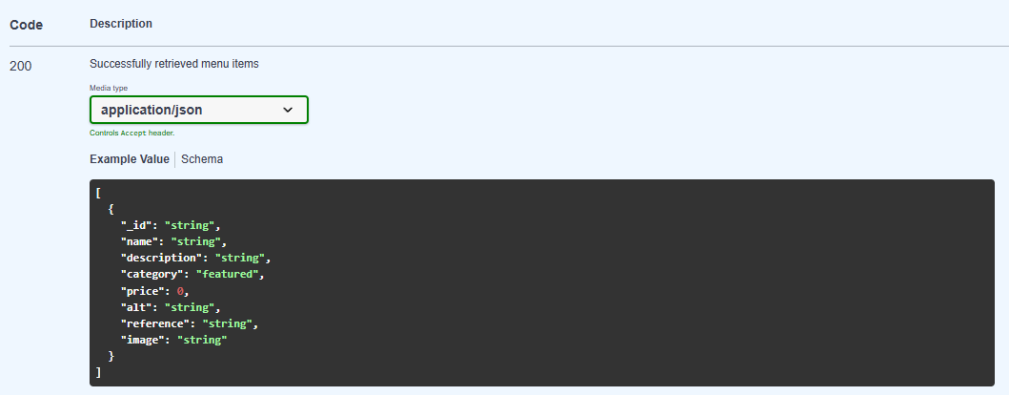
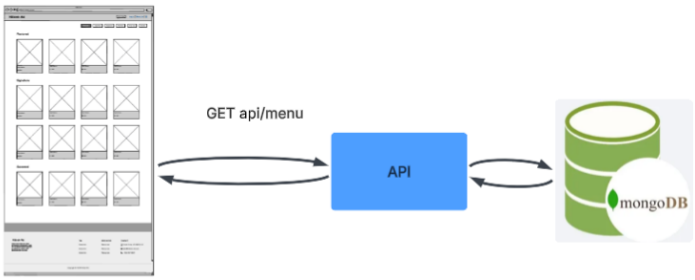


Diagram:



Get menu item:

Method: GET

URL: https://niibish-aki-e5d55d879400.herokuapp.com/api/menu/{itemId}

Purpose: When the user is in the checkout page and they want to modify an item, they will click on the “Modify” button and this will be called. Once the data is returned and processed, the edit modal will appear (the same as the one they see on the menu page).

Example Request:

```
curl --request GET --url https://niibish-aki-e5d55d879400.herokuapp.com/api/menu/660a2b3c4d5e6f7890123456
```

Responses:

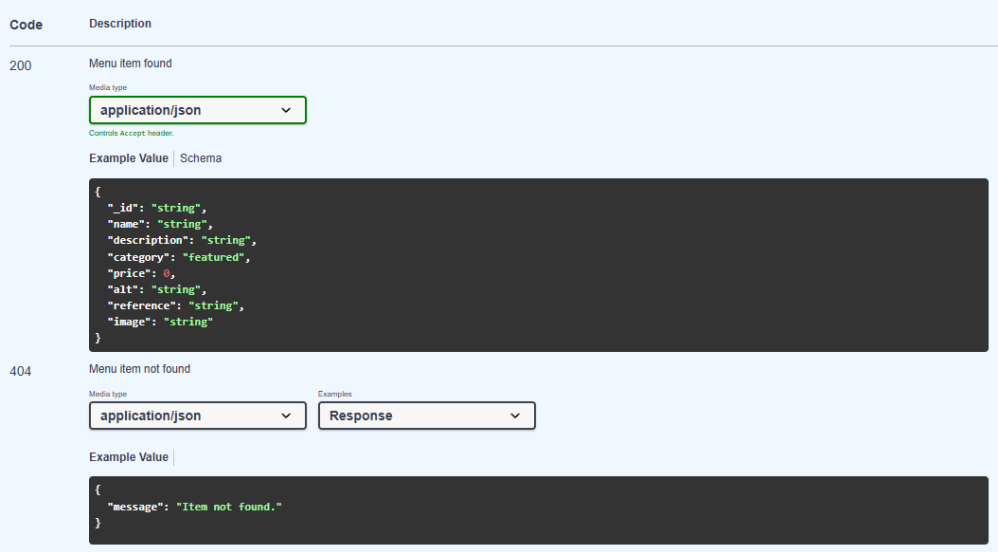
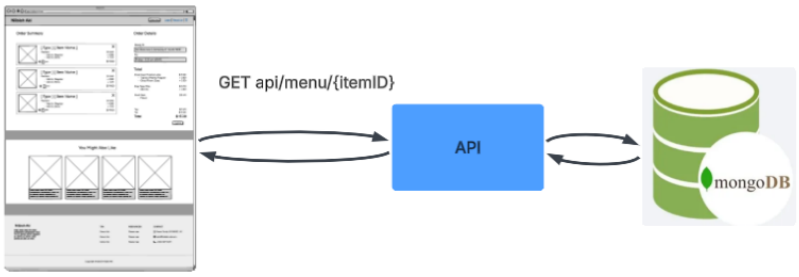


Diagram:



Update item:

Method: PUT

URL: https://niibish-aki-e5d55d879400.herokuapp.com/api/menu/{itemId}

Purpose: This is a stretch goal and not part of MVP. Once an admin goes their page (which will be a user page with extra tab options), they will be able to click on “Menu” and see the menu, and once they click on “Edit”, they can edit the fields on an item and once updated, this is called and the data is processed, with the new list being loaded and displayed.

Example Request:

```
curl --request PUT --url https://niibish-aki-e5d55d879400.herokuapp.com/api/menu/660a2b3c4d5e6f7890123456
```

Responses:

Code | **Description**

200 — Item updated successfully

Media type: application/json

Controls Accept header.

Example Value | Schema

```
{
  "_id": "string",
  "name": "string",
  "description": "string",
  "category": "featured",
  "price": 0,
  "alt": "string",
  "reference": "string",
  "image": "string"
}
```

400 — Invalid request body

Media type: application/json | Examples: Response

Example Value

```
{
  "message": "Invalid request body."
}
```

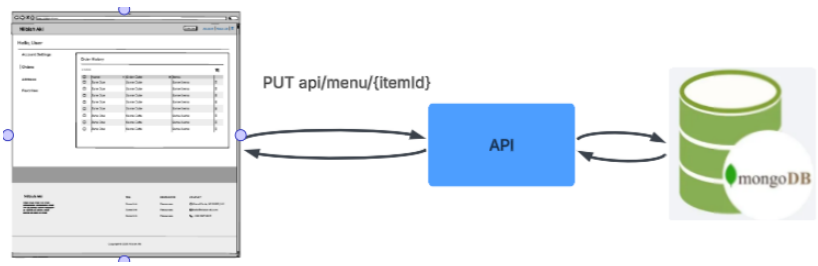
404 — Menu item not found

Media type: application/json | Examples: Response

Example Value

```
{
  "message": "Item not found."
}
```

Diagram:



Get featured items:

Get add-ins:

Method: GET

URL: https://niibish-aki-e5d55d879400.herokuapp.com/api/menu/addIns

Purpose: When the user goes to the menu page and clicks on a product card, this will be called and once the data is returned and processed, the user will be able to customize their beverage.

Example Request:

curl --request GET --url https://niibish-aki-e5d55d879400.herokuapp.com/api/menu/addIns
Responses:

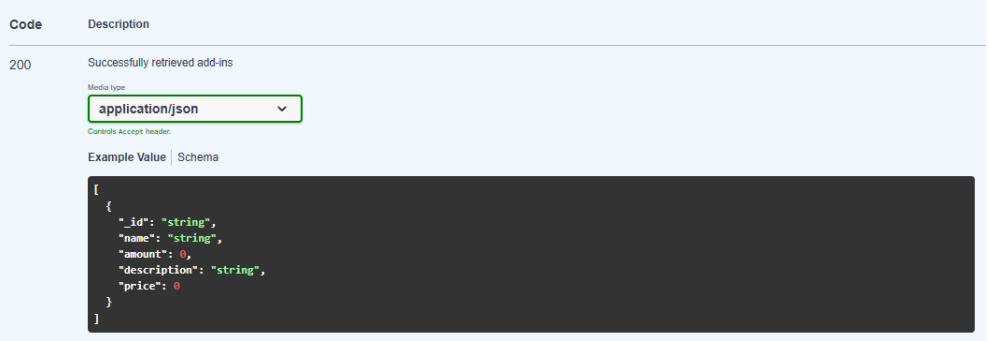


Diagram: