

Beautiful Babe Boutique

Web APPlication

SERVICE LAYER

DESIGN

CrystaLyn Wegner

Introduction

For my backend Service Layer, I will be implementing a REST API using the Heroku Connect API. This be used for the interaction between my database in MongoDB and the User Interface. The languages and or frameworks I will use will be React, Redux, Node.js, Express, and Stripe for this e-commerce application.

To describe how my application will work within the Service Layer I will explain the different routes that the application can complete.

1. User Register/Create Login Route
2. Search Route
3. Cart Checkout Route

All the routes listed will be requested to interact with the database in

MongoDB

User Create Route

User creates a login/Register a Profile

Method: POST

URL: https://beaugiebabe.herokuapp.com/api/register/:id

Purpose: when a user creates (POST) a login/password and information profile. This example input will be called. Once they hit the ‘’Register” button they will be directed back to the previous page the user was on and it will display in the right corner the profile that is logged in.

Example Request:

**curl -X 'POST' \**

**'https://virtserver.swaggerhub.com/beaugiebabe/beaugiebabe/1.0.0/loginandprofile' \**

**-H 'accept: \*/\*' \**

**-H 'Content-Type: application/json' \**

**-d '{**

**"userid": string",**

**"name": "string",**

**"email": "string",**

**"password": {**

**"phone": "integer",**

**"home address": "string"**

**}**

**}'**

Success Response:

Text

Description automatically generated

Graphical user interface, application

Description automatically generatedThis is an example; it will read as User created.

Error Response:

No error response to display but it will tell the user what they are missing if the User did not complete the profile and their information. The site will not allow them to save the user login and will display a message in the fields that are missing information.

Graphical user interface, application

Description automatically generated

Icon

Description automatically generated

Search Route

*User Searches for a clothing(i.e.Dress) item*

Method: GET

URL:https://beaugiebabe.herokuapp.com/api/dress/search:id

Purpose: When a user searches for a particular clothing item i.e. a dress. The request will pass through the Heroku API and MongoDB database to retrieve all search results of dresses. It will display on the search results page. The User will be able to search by type of item, color and size. This is a sample URL, it will be generated in Heroku.

Example Request:

**curl -X 'GET' \**

**'https://virtserver.swaggerhub.com/beaugiebabe/beaugiebabe/1.0.0/dress?searchString=dress&skip=0&' \**

**-H 'accept: application/json'**

Success Response:

Text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Error Response:

Graphical user interface, application

Description automatically generated

Graphical user interface, application, Teams

Description automatically generated

Shopping Cart Checkout Route

Shopping Cart Route

Method: POST

URL: https://beaugiebabe.herokuapp.com/api/cart:id

Purpose: The User will be directed to this URL to checkout with the items they added to the shopping cart. After the user hit the cart icon, the request will pass through the Heroku and then to MongoDB where it is storing the shopping cart information and items. It will allow an order to be created with a order number and direct the user to enter payment information. This will be done using stripe.

Example Request:

**curl -X 'POST' \**

**'https://besugiebabe.swagger.io/v2/checkout/request \**

**-H 'accept: application/json' \**

**-H 'Content-Type: application/json' \**

**-d '{**

**"order number": 0,**

**"Total Items": 0,**

**"Order Total": 0,**

**"Request Date": "2022-11-06T21:18:11.558Z",**

**"status": "Request Submitted",**

**"complete": true**

**}'**

Success Response:

Graphical user interface

Description automatically generated with medium confidence

Error Response:

A picture containing shape

Description automatically generated

