NodeJS: Home Assignment #2

Requirement Details (Scenario):

You are building the API for a pizza-delivery company. Don't worry about a frontend, just build the API. Here's the spec from your project manager:

1. New users can be created, their information can be edited, and they can be deleted. We should store their name, email address, and street address.
2. Users can log in and log out by creating or destroying a token.
3. When a user is logged in, they should be able to GET all the possible menu items (these items can be hardcoded into the system).
4. A logged-in user should be able to fill a shopping cart with menu items
5. A logged-in user should be able to create an order. You should integrate with the Sandbox of Stripe.com to accept their payment. Note: Use the stripe sandbox for your testing. Follow this link and click on the "tokens" tab to see the fake tokens you can use server-side to confirm the integration is working: https://stripe.com/docs/testing#cards
6. When an order is placed, you should email the user a receipt. You should integrate with the sandbox of Mailgun.com for this. Note: Every Mailgun account comes with a sandbox email account domain (whatever@sandbox123.mailgun.org) that you can send from by default. So, there's no need to setup any DNS for your domain for this task https://documentation.mailgun.com/en/latest/faqs.html#how-do-i-pick-a-domain-name-for-my-mailgun-account

Important Note: If you use external libraries (NPM) to integrate with Stripe or Mailgun, you will not pass this assignment. You must write your API calls from scratch. Look up the "Curl" documentation for both APIs so you can figure out how to craft your API calls.

This is an open-ended assignment. You may take any direction you'd like to go with it, as long as your project includes the requirements. It can include anything else you wish as well.

And please: Don't forget to document how a client should interact with the API you create!

End Points:

Endpoints	Description	Request & Header

POST /users	User details such as name, email address, street address are posted to create a new user.	**Request Headers:**
		Accept: application/json
		Request Params:
		{
		"name": "Jayalakshmi",
		"email_address": "jaya@gmail.com",
		"street_address": "2,TN,India",
		"password": "123456",
		"confirm_password": "123456"
		}

GET /users	Lists stored details of a registered user. **NOTE:** To authorize user, need to send email_address in request parameter(GET) and token(a token generated when user login) in Authorization header.	**Request Headers:** Accept: application/json Authorization: **<TOKEN>** **Request Params:** email_address: jaya@gmail.com

PUT /users	User can update his/her details such as name, password, street address.
	NOTE: To authorize user, need to send email_address in request parameter(POST) and token(a token generated when user login) in Authorization header.
		Request Headers:
		Accept: application/json
		Authorization: **<TOKEN>**
		Request Params:
		{
		"name": "Jayalakshmi",
		"email_address": "jaya@gmail.com",
		"street_address": "Palani,TN",
		"password": "12345678"
		}

GET /users	User can delete their account. **NOTE:** To authorize user, need to send email_address in request parameter(GET) and token(a token generated when user login) in Authorization header.	**Request Headers:** Accept: application/json Authorization: **<TOKEN>** **Request Params:** email_address: jaya@gmail.com

POST /users/login	A registered user can login into our system by posting email address and password. Upon successful login, user will get Domino's Pizza menu items and an Authorization Token to access the features of our system.	**Request Headers:**
		Accept: application/json
		Request Params:
		{
		"email_address": "jaya@gmail.com",
		"password": "123456"
		}

GET /users/logout	A registered user logout from our system. **NOTE:** To authorize user, need to send email_address in request parameter(GET) and token(a token generated when user login) in Authorization header.	**Request Headers:** Accept: application/json Authorization: **<TOKEN>** **Request Params:** email_address: jaya@gmail.com

GET /tokens/refresh	If an authorized token expired, token will be refreshed by this API.	**Request Params:**
		{
		"email_address": "jaya@gmail.com",
		"token": "83W5FgeNtzdxQz5hw42x"
		}

GET /menus	Shows list of all available menus in Domino's Pizza. **NOTE:** To authorize user, need to send token(a token generated when user login) in Authorization header.	**Request Headers:** Accept: application/json Authorization: **<TOKEN>**

POST /cart	A registered user can create their own shopping cart by posting menu id (list on menu items after successful login) and quantity of item to be needed.
	NOTE: To authorize user, need to send token(a token generated when user login) in Authorization header.
		Request Headers:
		Accept: application/json
		Authorization: **<TOKEN>**
		Request Params:
		{
		"menu_id": "PIZZA04",
		"quantity": 2
		}

GET /cart	Shows list of items saved in shopping cart. **NOTE:** To authorize user, need to send token(a token generated when user login) in Authorization header.	**Request Headers:** Accept: application/json Authorization: **<TOKEN>**

PUT /cart	User can update the quantity of an item in his/her shopping cart.
	NOTE: To authorize user, need to send token(a token generated when user login) in Authorization header.
		Request Headers:
		Accept: application/json
		Authorization: **<TOKEN>**
		Request Params:
		{
		"menu_id": "PIZZA04",
		"quantity": 2
		}

DEL /cart	User removes an item from his/her shopping cart.
	NOTE: To authorize user, need to send token(a token generated when user login) in Authorization header.
		Request Headers:
		Accept: application/json
		Authorization: **<TOKEN>**
		Request Params:
		{
		"menu_id": "PIZZA04"
		}

POST /cart/empty	User clears / remove all items from his/her shopping cart. **NOTE:** To authorize user, need to send token(a token generated when user login) in Authorization header.	**Request Headers:** Accept: application/json Authorization: **<TOKEN>**
POST /orders	User place order for the items added in his/her shopping cart. Upon successful order, order receipt will be mailed to user's registered email address. **NOTE:** To authorize user, need to send token(a token generated when user login) in Authorization header.	**Request Headers:** Accept: application/json Authorization: **<TOKEN>**

GET **/orders**	User views all the orders he/she made with our system. **NOTE:** To authorize user, need to send token(a token generated when user login) in Authorization header.	**Request Headers:** Accept: application/json Authorization: **<TOKEN>**