How I approached in solving the problem:

To make it easier, I divided the app into customer and admin and created backend followed by frontend for each of them.

1. Firstly, I went through the requirement specifications document, listed what functionality is required, what is to be stored and how. Assumed how a basic front-end app might look like.
2. Then I decided to create a basic backend schema for the data to be stored in backend like user data, food items, orders, etc.
3. Created the basic backend schema for user and customer registration, and login. At first decided to go with different backends for restaurant admin and customer authentication, but I thought of a real apps and continued with the same email role-based approach for both restaurant and customer.
4. Redesigned the authentication system, as one system for both admin and customer as user. Each user may have multiple roles like customer and admin. But one user can be admin for one restaurant only.
5. Tested the admin and customer registration, login authentication based on role for its authenticity using postman. Changed the logic and schema as required after testing.
6. Created the basic frontend functionality for the admin and customer registration in React using tailwinds.
7. After creating the frontend, tested the app once again and continued with the restaurant login backend part.
8. Started with the creation of backend for restaurant, creating restaurant itself after registering the user as admin, food items creation, deletion, updating. Tested it and designed the frontend for the restaurant creation, food items CRUD.
9. Created the front end for the customer showing only the food items, then I moved to the customer backend for add the items to the cart, orders.
10. After testing, created the fronted for the customer for the cart, checkout, orders. After the orders, created frontend and backend for restaurant admin for received orders to accept orders, change the stock and availability.
11. After the creation of the app, tested both admin and customer apps.
12. Later I found that any user who is logged in can access the fronted routes. With empty data, then there was a need that admin or customer frontend routes to need protection.
13. Created a routes protection for both admin and customer. Also created a backend admin access protection for restaurant admin only in addition to the authentication middleware. By this only admin can access the backend and fronted routes if logged in as admin only.
14. Finally added toast notification messages to the app.
15. Deployed the backend on Render and frontend on Netlify.