Project Design Phase
Solution Architecture

Date	13 April 2025
Team ID	SWTID1743680479
Project Name	SB Foods - Food Ordering App
Maximum Marks	4 marks

Solution Architecture:

1) **Client:**
User: Uses a mobile or web interface to interact with the food delivery platform and can browse restaurants, place orders, track deliveries, and manage account info.

2) **Browser:**
Acts as the interface through which the user accesses the web application and it also sends HTTP requests to the backend server when users perform actions like ordering food or viewing menus

3) **WebServer:**

- Node.js: The JavaScript runtime environment used for building the server-side of the application.
- Express.js: A web application framework for Node.js, used to handle HTTP requests and route them to the appropriate handlers.
-

4) **AppServer**:
React.js: Provides the user interface of the food delivery app and handles dynamic rendering of menus, order forms, and delivery tracking. it interacts with backend APIs via HTTP calls.

5) **Database**:
MongoDB: The NoSQL database used to store user information, food details, order details and other relevant data for the web app.

6) **ExternalServices**:

- SB Food Data API: Provides food information, menu updates, and pricing.
- Track Service: Integrates with food processing services to monitor order delivery in real-time.

Interaction Flow:

1) **Client Interaction:**
The user launches the food delivery app via a browser, and they navigate through menus, select items, and place an order.

2) **Request Flow:**
The browser sends an **HTTP request** to the **Node.js/Express.js WebServer** to fetch required data or process an action (e.g., order placement). The WebServer serves the React frontend and handles backend API calls.

3) **User Actions:**
Actions include browsing menus, creating accounts, logging in, adding items to the cart, checking out, and tracking order status.

4) **Data Operations:**
On order placement, the WebServer interacts with **MongoDB** to create or update records. The database handles operations like reading menu items, saving order history, and updating delivery status.

5) **External Services Interaction:**
WebServer fetches real-time restaurant/menu data via SB Food Data API. Sends/receives delivery updates using the Track Service to monitor and display order status. External service data is also saved in the database for analytics and historical records.

6) **Response Flow:**
API responses are returned to the React frontend. UI components update dynamically (e.g., order confirmation). The final output is displayed to the client through the browser.

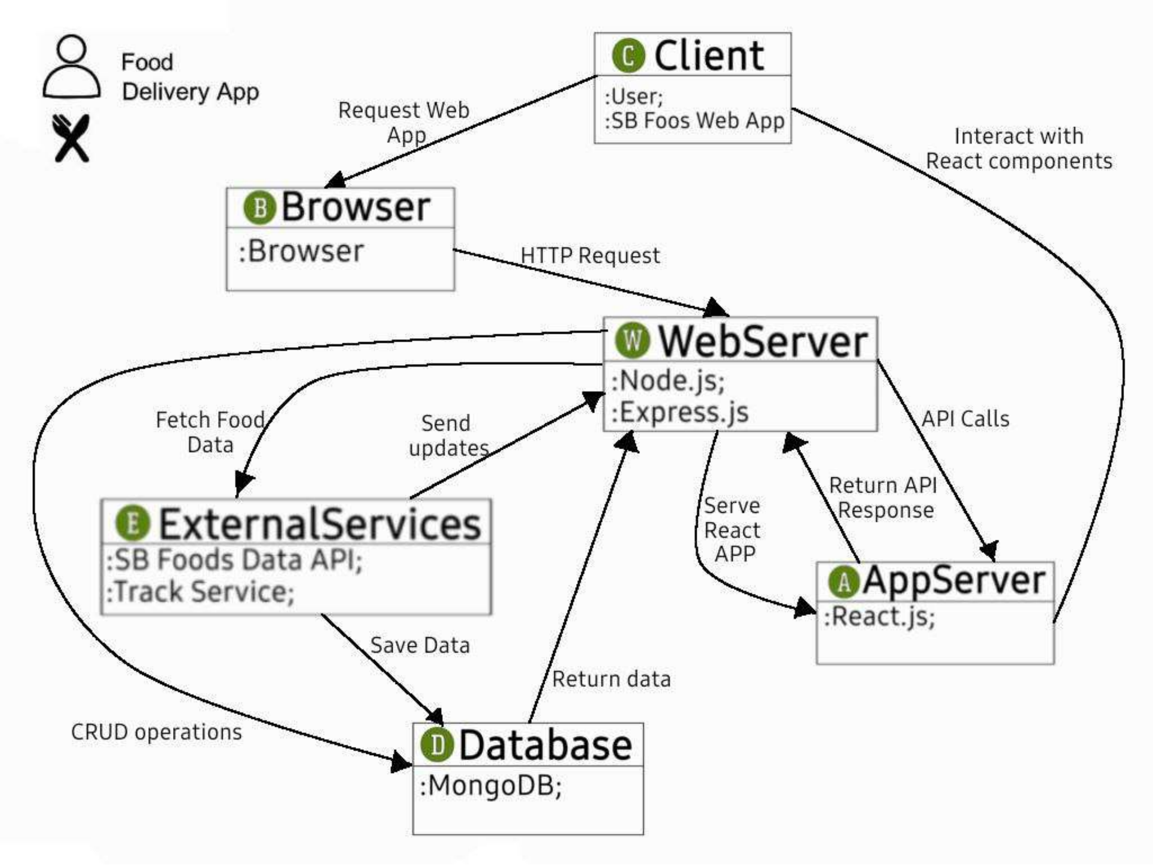
Notes:

- Client Note: The client interacts with the React.js application, which is served by the Node.js and Express.js backend.

- Database Note: MongoDB is used to store user information, food details, order details

- External Services Note: External services are used for fetching menu items, order confirmation and order status updates to users.

Solution Architecture Diagram:


Food
Delivery App
C Client
:User;
:SB Foos Web App
Request Web App
Interact with React components
B Browser
:Browser
HTTP Request
W WebServer
:Node.js;
:Express.js
Fetch Food Data
Send updates
API Calls
Return API Response
Serve React APP
E ExternalServices
:SB Foods Data API;
:Track Service;
A AppServer
:React.js;
Save Data
Return data
CRUD operations
D Database
:MongoDB;