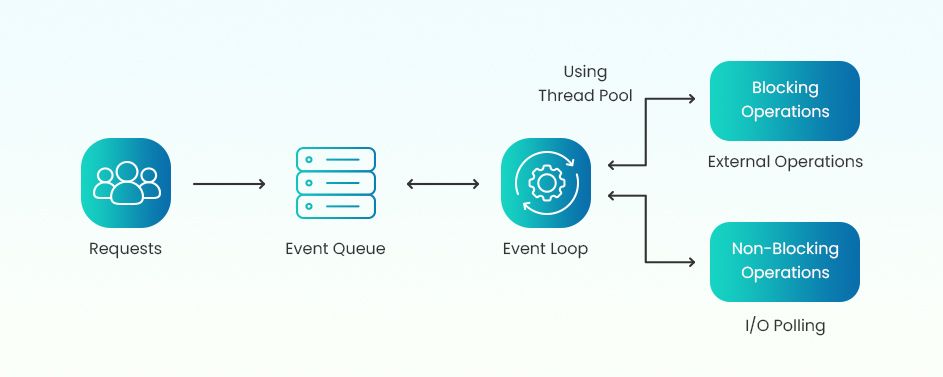
**Define NodeJS architecture**



**Request** : Incoming tasks or actions from users and vendors (e.g HTTP request).

**Event Queue** : Where all incoming requests are stored before being processed. Requests are passed to the Event Loop for further handling.

**Event Loop** : The core mechanism in Node.js that continuously checks the Event Queue for tasks. Decides whether tasks are blocking or non-blocking.

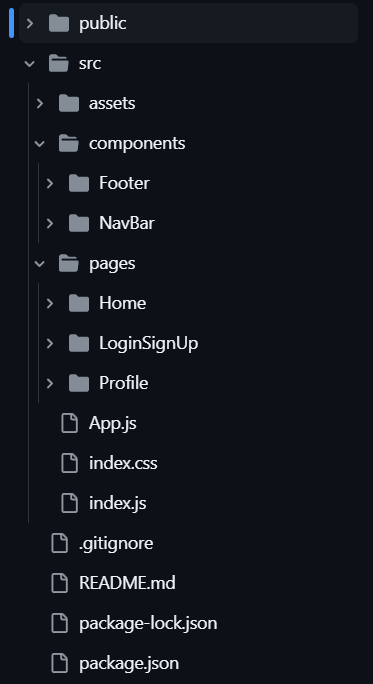
**Blocking Operations :**

* **Checkout** : This could involve payment processing, which typically requires waiting for completion.
* **Hire Personal Trainer :** Likely requires confirmations and may involve a waiting step for booking.
* **Sign in / Sign up / Sign out :** These often involve server communication and user authentication, making users wait.
* **Service / Product Uploading :** File uploads usually involve a waiting period.
* **Manage Orders :** Processing complex order details may require blocking behavior.

**Non-Blocking Operations :**

* **View Notification**
* **View Product / View Order History**
* **Product Search and Filter** **:** Could be asynchronous if done efficiently.
* **Add item to cart**
* **Favorite Products List management**

**Describe each components in the architecture**

****

* **Static File Delivery:** Files from the public folder are served to the client as needed, usually handled by a server like Express.js.
* **Routing (React):**
  + The entry point is index.js, which renders the App.js component.
  + App.js manages routing, often using libraries like React Router, to determine which page to display based on the URL.
* **Components Composition:**
  + pages (e.g., Home, LoginSignUp, and Profile) render their content using smaller components like NavBar and Footer.
  + Components are modular, enhancing reusability and separation of concerns.
* Other files like package.json, package-lock.json, .gitignore define dependencies and scripts for the Node.js project.