**ModalPopUp Component Documentation**

The **ModalPopUp** component is a React functional component used for uploading data files and filling necessary information required for processing.

**Purpose**

The purpose of this component is to facilitate users in uploading data files (in formats such as .xlsx, .xls, or .csv) and providing necessary information for processing those files.

Props

* **lat** (number, required): Latitude value used to fetch location-specific data.
* **lon** (number, required): Longitude value used to fetch location-specific data.

Component Structure

The **ModalPopUp** component consists of the following sections:

1. **Header Section**: Displays the "Upload Data" button to trigger the modal for uploading data.
2. **Modal Dialog**: Renders a modal dialog that appears when the "Upload Data" button is clicked. This modal allows users to provide necessary information and upload data files.
3. **AfterCalcErrorModal**: A separate modal dialog that appears in case of an error occurring after calculation. It provides users with information about the error and prompts them to try again or contact support.

State Management

The component utilizes React Hooks for managing state:

* **state**: Manages the internal state using the **useState** and **useReducer** hooks to manage various states such as loading, file selection, format correctness, etc.
* **userContext** and **MyContext**: Utilizes React Context API for managing user-related data and submission status.
* **inputFile**: Utilizes **useRef** hook to reference the file input element for handling file uploads.

External Dependencies

The component relies on several external dependencies:

* Axios: For making HTTP requests to fetch data and handle API calls.
* Moment.js: For parsing, validating, manipulating, and formatting dates.
* React Icons: Provides access to a collection of popular icons to be used within the application.
* React Toastify: For displaying toast notifications in the application.
* XLSX: For parsing Excel files and extracting data from them.

Component Methods

* **handleFileChange**: Handles the file change event when a user selects a file for upload. It validates the file format and extracts data from it.
* **closeModal**: Closes the modal dialog for uploading data.
* **closeAfterCalcErrorModal**: Closes the modal dialog that appears in case of an error occurring after calculation.
* **setLocation**: Opens the modal dialog for providing location information.
* **handleInfoSubmit**: Handles the submission of information and data file for processing.

External API Usage

The component interacts with external APIs to process data. It makes use of Axios to make HTTP requests to the following APIs:

* **https://api.opencagedata.com/geocode/v1/json**: Fetches location information based on latitude and longitude.
* **http://10.13.2.189:8000/api/close-calc**: Sends a request to close the calculation.
* **http://10.13.2.189:8000/api/calc-state**: Updates the calculation state in MongoDB.
* **http://10.13.2.189:2701/api/flask/quick/processing**: Initiates processing with the "quick" model.
* **http://10.13.2.189:2701/api/flask/premium/processing**: Initiates processing with the "premium" model.
* **http://10.13.2.189:2701/api/flask/premium-quick/processing**: Initiates processing with both "quick" and "premium" models.