Frontend: EIDR Language Tool

The frontend component of the TEAM EMMY SDK project is developed using the React JavaScript library. It provides a dynamic and responsive user interface for managing title-related language data stored in AWS DynamoDB. The interface is designed to enhance user interaction with the dataset, enabling efficient exploration, modification, and management of records through an intuitive layout and real-time feedback mechanisms.

Key Features

1. **Dynamic Column Detection**
   The application automatically extracts unique keys from the dataset to generate table headers dynamically. This ensures flexibility, allowing the frontend to adapt to any structural variations in the data without requiring manual reconfiguration.

2. **Column Filtering and Sorting**
   Users can filter data per column using input fields embedded within each table header. Additionally, columns can be sorted in ascending or descending order by clicking on the column headers. The application supports both string-based and numeric sorting for improved data navigation.

3. **Column Visibility Control**
   The interface includes a column visibility modal that allows users to selectively display or hide columns. This feature enhances the usability of the interface when dealing with large datasets by enabling a focused view on relevant attributes.

4. **Modal Interfaces for Add/Edit Operations**
   Bootstrap-based modal dialogs are used to create or edit records. These modals are dynamically populated based on the table structure, and they provide form validation to ensure the integrity of input data (e.g., required fields like RecordID and Title).

5. **Real-Time Notifications**
   Feedback is provided to the user in the form of contextual notifications for various operations such as successful additions, updates, deletions, or errors (e.g., API failure or validation issues). This ensures transparency and enhances the user experience.

6. **Sidebar Navigation**
   A collapsible sidebar lists all visible columns and allows users to quickly scroll to a specific column. This feature is particularly helpful for datasets with a large number of attributes.

API Interactions from the Frontend

The React interface communicates with the backend through RESTful API endpoints. The following HTTP methods are used to support the full range of CRUD operations:

- **GET /api/items**: Retrieves all records from the database.
- **POST /api/items**: Adds a new record. Requires RecordID and Title as mandatory fields.
- **PUT /api/items/:RecordID/:Title**: Updates an existing record identified by a composite key of RecordID and Title.

- **DELETE /api/items/:RecordID/:Title**: Deletes a record using its composite key.
- **GET /api/table-info**: (Optional) Fetches metadata about the DynamoDB table schema to support frontend validation or introspection.

Backend Overview: Node.js Server and AWS DynamoDB Integration

The backend of the TEAM EMMY SDK web application is implemented using Node.js with the Express.js framework. It provides a secure and scalable REST API for interacting with AWS DynamoDB. This backend acts as the bridge between the frontend interface and the NoSQL storage layer.

Technology Stack

- **Express.js**
  Serves as the web application framework that facilitates routing and middleware integration.
- **AWS SDK for JavaScript (v3)**
  Provides support for communicating with DynamoDB using DynamoDBClient and DynamoDBDocumentClient, allowing seamless CRUD operations with structured JSON objects.
- **dotenv**
  Enables loading of sensitive AWS credentials and configuration settings from a .env file, thereby separating code from configuration and improving security.
- **CORS and Body Parser**
  These middleware modules enable cross-origin resource sharing (CORS) and automatic parsing of JSON request bodies, ensuring that the frontend can communicate effectively with the backend API.

Database Schema

The application uses a single DynamoDB table named TitleLanguageData, which is structured as follows:

- **Partition Key**: RecordID
- **Sort Key**: Title

These composite keys ensure that each record is uniquely identifiable and prevent duplication of titles under the same identifier.

API Endpoints

Method	Endpoint	Description
GET	/api/items	Retrieves all records from the database.
GET	/api/items/:recordID/:title	Retrieves a specific record identified by the composite key.
POST	/api/items	Adds a new record. Requires both RecordID and Title.
PUT	/api/items/:recordID/:title	Updates an existing record. The record must already exist.
DELETE	/api/items/:recordID/:title	Deletes a record based on its composite key.

GET	/api/table-info	Returns the DynamoDB table schema, useful for frontend introspection.

Security Considerations

To maintain security and good development practices:

- AWS credentials are stored securely using environment variables managed via a .env file.
- Secrets and sensitive configurations are never hard coded in the source code.
- Data validation checks are performed before creating or updating records to ensure schema integrity and avoid duplicate entries.