Interface (States & Input blocks)

Overview

This Vue component, named `StateTable`, provides a user interface for managing state files and input files. It allows users to view, search, filter, and interact with these files in a tabular format.

Key Features

1. Tabbed interface for States and Inputs
2. Searchable and filterable table
3. Expandable rows with detailed file content
4. Ability to start workflows
5. File status management
6. File deletion

Component Structure

Data Properties

- `activeTab`: Controls which tab is currently active (0 for States, 1 for Inputs)
- `pagination`: Controls table pagination
- `tabs`: Defines the available tabs and their configurations
- `apiBaseUrl`: Base URL for API requests
- `search`: Stores the current search term
- `workerStatusOptions`: Available status options for workers
- `selectedStatus`, `selectedGroup`: For filtering in respective tabs
- `rows`: Stores the main data displayed in the table

Computed Properties

- `filteredData`: Returns the filtered and searched data for the table
- `filteredWorkerFields`: Filters specific fields from worker data for display

Methods

1. `selectTab(index)`: Switches between States and Inputs tabs
2. `fetchApiData(endpoint)`: Fetches data from the API
3. `refreshFiles()`: Refreshes the file list
4. `startWorkflow(user_code, payload)`: Initiates a new workflow
5. `checkWorkflowStatus(workflowID, resolve, reject)`: Checks the status of a running workflow
6. `fetchFiles()`: Fetches and processes the list of files
7. `fetchFileContent(filePath, row)`: Retrieves detailed content for a specific file
8. `startWorkflowForRow(row, activeTab)`: Starts a workflow for a specific row
9. `toggleExpand(row)`: Expands/collapses a row to show/hide details
10. `saveStatus(row)`: Saves the current status of a file
11. `uploadFile(jsonData, fileName, path, currentFileIndex, totalFiles)`: Uploads a file to the server
12. `deleteFile(row)`: Deletes a file from the server

Usage Guide

Viewing Files

1. The component displays two tabs: "States" and "Inputs"
2. Each tab shows a table with relevant information about the files
3. Use the search bar to filter files by any field
4. Use the status/group dropdown (depending on the active tab) to filter files

Interacting with Files

1. Click the '+' button on a row to expand and view detailed information
2. In the expanded view:
   - For States: You can view worker details and their individual states
   - For Inputs: You can view the structure of the input file

File Actions

1. **Start Workflow**:
   - Click "Start Workflow" in the expanded view to initiate a workflow for that file
   - For States, it uses the `workflow-manager` user code
   - For Inputs, it uses the `generate-state` user code
2. **Save State** (States tab only):
   - Click "Save State" to update the file on the server with any changes
3. **Delete State** (States tab only):
   - Click "Delete State" to remove the file from the server

Managing Worker and Item States

1. In the expanded view, each worker and item has a status dropdown
2. Click on the status to open the dropdown and select a new status

Refreshing Data

- Click the refresh button (circular arrow icon) to fetch the latest data from the server

API Integration

The component interacts with several API endpoints:

1. File listing: `${apiBaseUrl}/api/v1/explorer/view/?path=components/states_files.json` or `${apiBaseUrl}/api/v1/explorer/view/?path=components/inputs_files.json`
2. File content: `${apiBaseUrl}/api/v1/explorer/view/?path=${filePath}`
3. Workflow start: `${apiBaseUrl}/workflow/api/workflow/run-workflow/`
4. Workflow status check: `${apiBaseUrl}/workflow/api/workflow/${workflowID}/`
5. File upload: `${apiBaseUrl}/api/v1/explorer/upload/`
6. File deletion: `${apiBaseUrl}/api/v1/explorer/delete/?path=${filePath}&is_dir=false`

Notes for Developers

- The component uses custom icons (`IconRefresh`, `IconCopy`, `IconDownload`, `IconTabActive`) which should be properly imported
- Authorization is handled by the `authorization()` function from `@/utils/customFetch`
- The component relies on the Quasar framework for UI components (e.g., `q-table`, `q-input`, `q-select`)

Revision #2
Created 30 August 2024 14:39:46 by Anton Petrushkaneki
Updated 30 August 2024 14:41:01 by Anton Petrushkaneki