diagram and dataflow so that the person who taken on the project understand what is happeinng

README for Admin Frontend

Project Overview

This project is an admin frontend for managing and analyzing academic results. It consists of two main pages: the front page for uploading Excel or CSV files and the result analysis page for viewing and modifying the uploaded data. The frontend communicates with a backend server to store and retrieve data.

Features

1. **File Upload**: Upload Excel or CSV files via drag-and-drop or file browsing.
2. **Data Analysis**: Analyze the uploaded data and display it in a tabular format.
3. **Data Modification**: Modify school name, course, semester, and examination month.
4. **Backend Integration**: Send the processed data to the backend server for storage.

Project Structure

java

```java
adminfrontend/
├── public/
│   └── index.html
├── src/
│   ├── components/
│   │   ├── frontPage/
│   │   │   ├── AdminFrontPage.js
│   │   │   ├── FileUpload.js
│   │   │   └── FetchDataButton.js
│   │   └── resultAnalysisPage/
│   │       └── ResultAnalysisPage.js
│   ├── App.css
│   ├── App.js
│   └── index.js
├── .gitignore
├── package.json
└── README.md
```

Installation and Setup

1. **Clone the Repository**:

   sh

   ```sh
   git clone <repository-url>
   cd adminfrontend
   ```

2. **Install Dependencies**:

   ```sh
   npm install
   ```

3. **Run the Application**:

   ```sh
   npm start
   ```

4. **Backend Server**:
   Ensure the backend server is running at `http://localhost:4000`.

Components

1. `App.js`

The main component that manages the state and renders either the `AdminFrontPage` or `ResultAnalysisPage` based on the current state.

2. `AdminFrontPage.js`

Handles file uploads and sends the uploaded data to the backend server. It uses the `FileUpload` component for the drag-and-drop functionality.

3. `FileUpload.js`

Provides drag-and-drop functionality for file uploads, allowing only Excel and CSV files.

4. `FetchDataButton.js`

A button component to fetch data from the backend server (optional).

5. `ResultAnalysisPage.js`

Displays and allows modification of the uploaded data. It also sends the modified data back to the backend server.

Data Flow and Block Diagram

Data Flow

1. **File Upload**:
   - The user uploads an Excel or CSV file via the `FileUpload` component.
   - `AdminFrontPage` processes the file and sends the data to the backend.

  - Upon successful upload, the data is stored in the **`data`** state and the view switches to **`ResultAnalysisPage`**.
2. **Data Analysis**:
   - **`ResultAnalysisPage`** displays the data in a table.
   - The user can modify specific fields (school name, course, semester, exam month).
   - Clicking "Create" sends the modified data back to the backend.

Block Diagram

```plaintext
+-------------------+          +-------------------+          +-------------------+
|                   |          |                   |          |                   |
|  AdminFrontPage   +---------->  ResultAnalysisPage+---------->    Backend        |
|                   |          |                   |          |                   |
+---------+---------+          +---------+---------+          +---------+---------+
          ^                              ^                              ^
          |                              |                              |
          |                              |                              |
          |                              |                              |
+---------+---------+          +---------+---------+          +---------+---------+
|                   |          |                   |          |                   |
|   FileUpload      |          | FetchDataButton   |          |    Database       |
|                   |          |                   |          |                   |
+-------------------+          +-------------------+          +-------------------+
```

Explanation:

1. **AdminFrontPage**:
   - Handles file uploads and processes the data.
   - Sends the data to the backend server.
2. **FileUpload**:
   - Component for drag-and-drop file upload functionality.
3. **ResultAnalysisPage**:
   - Displays and allows modification of the uploaded data.
   - Sends modified data back to the backend server.
4. **FetchDataButton**:
   - Optional component to fetch data from the backend server.
5. **Backend**:
   - Stores and manages the data.
   - Interacts with a database to persist the data.

Running the Project

1. **Start the Frontend**:

   ```sh
   npm start
   ```

2. **Ensure the Backend Server is Running**: