Logistics Application Documentation

Introduction:

The logistics application is a simple system designed to manage product information, suppliers, and warehouses. It allows users to add products, suppliers, and warehouses, and retrieve information about them. The application is built using the Laravel framework for the backend and React for the frontend.

Tech Stack:

- Backend Framework: Laravel
- Frontend Framework: React

Database Design:

The application uses a database to store information about products, suppliers, and warehouses. The database design includes the following tables:

1. Table: users

- Columns:
  1. id
  2. name
  3. email
  4. email_verified_at
  5. password
  6. remember_token
  7. created_at
  8. updated_at
  9. api_token

2. Table: suppliers

- Columns:
  1. id
  2. name
  3. address
  4. created_at
  5. updated_at

3. Table: warehouses

- Columns:
  1. id
  2. supplier_id
  3. name
  4. address
  5. created_at
  6. updated_at

4. Table: products

- Columns:
  1. id
  2. name
  3. price
  4. supplier_id
  5. warehouse_id
  6. created_at
  7. updated_at

5. Table: oauth_clients

- Columns:
  1. id
  2. client_id
  3. created_at
  4. updated_at

6. Table: oauth_auth_codes

- Columns: (columns not specified)

7. Table: oauth_personal_access_clients

- Columns:
  1. id
  2. client_id
  3. created_at
  4. updated_at

8. Table: failed_jobs

- Columns:
  1. id
  2. uuid
  3. connection
  4. queue
  5. payload
  6. exception
  7. failed_at

9. Table: password_reset_tokens

- Columns:
  1. email
  2. token
  3. created_at

10. Table: personal_access_tokens

- Columns:
  1. id
  2. tokenable_type
  3. tokenable_id
  4. name
  5. token
  6. abilities
  7. last_used_at
  8. expires_at
  9. created_at

11. Table: migrations

- Columns:
  1. migration
  2. updated_at
  3. id
  4. batch

12. Table: oauth_access_tokens

- Columns:
  1. id
  2. user_id
  3. client_id
  4. name
  5. scopes
  6. revoked
  7. created_at
  8. updated_at
  9. expires_at

Backend API Endpoints:

The backend of the application provides the following API endpoints:

1. `**POST /api/addproduct**`: Adds a new product to the database.
   - Request Body: name (string), price (number), supplier_id (number), warehouse_id (number)

2. `**POST /api/addsupplier**`: Adds a new supplier to the database.

- Request Body: name (string), address (string)

3. `**POST /api/addwarehouse**`: Adds a new warehouse to the database.
   - Request Body: supplier_id (number), address (string)

4. `**POST /api/login**`: Authenticates the user and generates a session token for subsequent requests.
   - Request Body: username (string), password (string)

5. `**GET /api/getallproducts**`: Retrieves information about all products in the database.
   - Response: Array of product objects

6. `**GET /api/getsupplier**`: Retrieves information about a specific supplier by its ID.
   - Request Parameters: supplier_id (number)
   - Response: Supplier object

7. `**GET /api/getallwarehouses**`: Retrieves information about all warehouses in the database.
   - Response: Array of warehouse objects

Frontend Structure:

The frontend of the application is built using React and follows the component-driven development approach. The components are organized as follows:

1. `**App**`: The main component that serves as the entry point of the application.
   - Contains the routing logic and renders other components based on the current route.

2. `**Login**`: Renders the login form and handles user authentication.
   - Sends a login request to the backend and receives a session token upon successful authentication.

3. `**AddProduct**`: Renders a form for adding a new product.
   - Sends a POST request to the `/api/addproduct` endpoint to add the product to the database.

4. `**AddSupplier**`: Renders a form for adding a new supplier.
   - Sends a POST request to the `/api/addsupplier` endpoint to add the supplier to the database.

5. `**AddWarehouse**`: Renders a form for adding a new warehouse.

  - Sends a POST request to the `/api/addwarehouse` endpoint to add the warehouse to the database.

6. `**Home**`: Retrieves and displays a list of all products.
   - Sends a GET request to the `/api/getallproducts` endpoint to retrieve product information.

Retrieves and displays a list of all warehouses.
   - Sends a GET request to the `/api/getallwarehouses` endpoint to retrieve warehouse information

Retrieves and displays information about a specific supplier.
   - Sends a GET request to the `/api/getsupplier` endpoint with the supplier ID to retrieve supplier details.