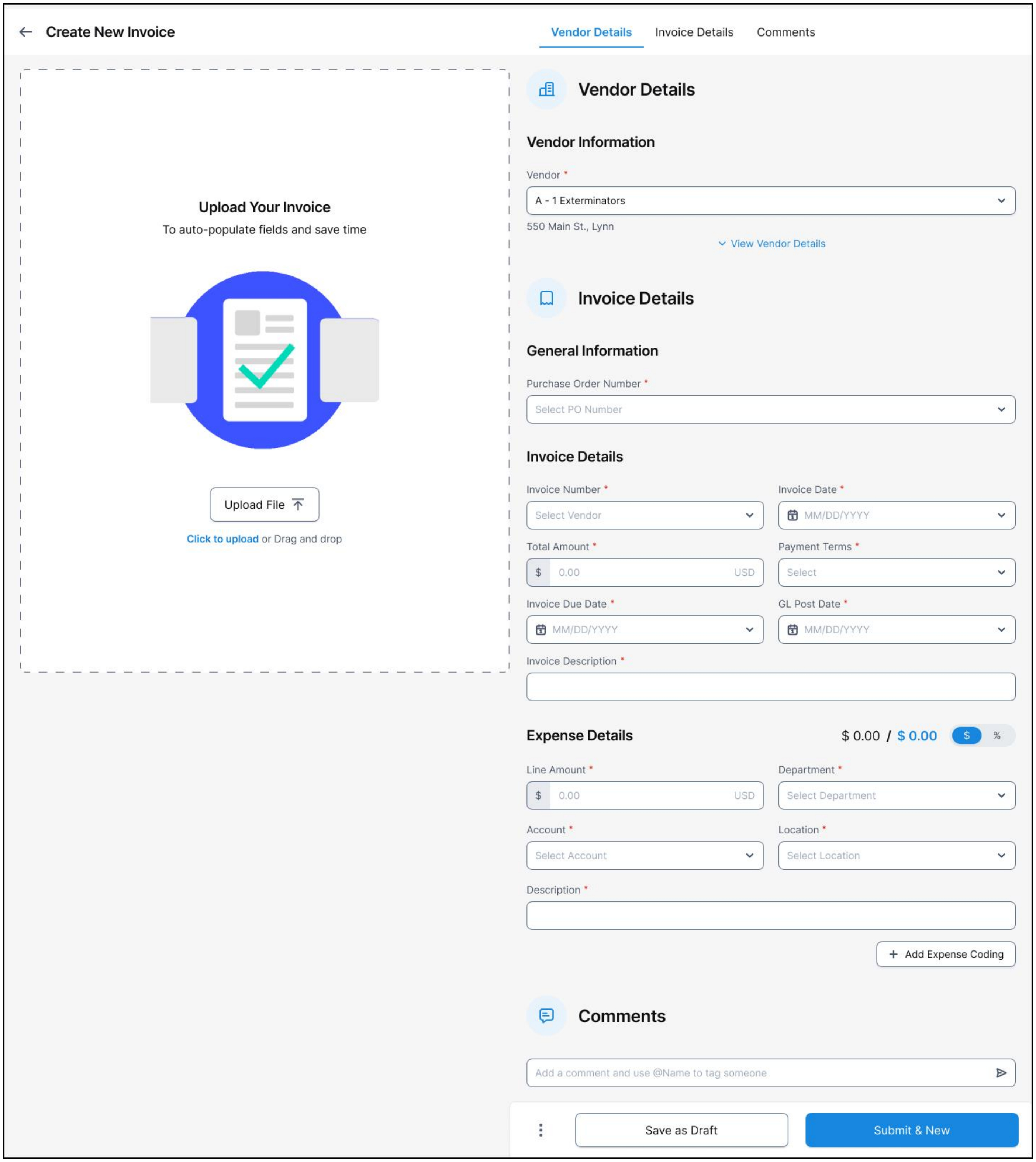
Assignment: Invoice Form with Login

Overview

You are tasked with developing a React application that replicates the provided design. The application must include a login system and a functional form, ensuring data persistence using local storage. The form functionality should be implemented using Formik, and the data must persist across page reloads.



Create New Invoice

Vendor Details | Invoice Details | Comments

Upload Your Invoice

To auto-populate fields and save time

Upload File

Click to upload or Drag and drop

Vendor Details

Vendor Information

Vendor *: A - 1 Exterminators

550 Main St., Lynn

View Vendor Details

Invoice Details

General Information

Purchase Order Number *: Select PO Number

Invoice Details

Invoice Number *: Select Vendor	Invoice Date *: MM/DD/YYYY
Total Amount *: $ 0.00 USD	Payment Terms *: Select
Invoice Due Date *: MM/DD/YYYY	GL Post Date *: MM/DD/YYYY

Invoice Description *

Expense Details $ 0.00 / $ 0.00 $ %

Line Amount *: $ 0.00 USD	Department *: Select Department
Account *: Select Account	Location *: Select Location

Description *

+ Add Expense Coding

Comments

Add a comment and use @Name to tag someone

Save as Draft | Submit & New

Deliverables

- A fully functional React application replicating the design.
- Hosted live demo accessible via a publicly available link.
- A GitHub repository containing the project code, including a README file with setup and usage instructions.

Requirements

1. Login System

- Login Form:
    - Create a login form where users can enter their username and password.
    - Validate the form inputs using Formik.
-
- Session Management:
    - Store the user's session in localStorage upon successful login.
    - Redirect the user to the main application interface after login.
-
- Auto-Login:
    - Automatically redirect users to the main application interface if their session exists in local storage.
-
- Logout Functionality:
    - Include a logout button to clear the user's session from local storage and redirect to the login page.
-

2. Replicate the Design

- Accurately replicate the provided design using React components and styling.
- Ensure responsiveness and alignment with the design specifications.

3. Functional Form Implementation

- Use Formik to build a functional form as per the design.
- Validate form inputs to ensure data integrity.

4. Data Persistence with LocalStorage

- Store form data in localStorage upon submission.
- Ensure that the form data persists across page reloads and is pre-populated when the page is revisited.

5. PDF Upload and Display

- PDF Upload Component:
  - Implement a feature that allows users to upload a PDF file from their local system.
  - Render and display the uploaded PDF within the application interface using a library like `react-pdf`.

6. Populate Form Fields with Dummy Data

- Button Functionality:
  - Add a button that, when clicked, populates all form fields and pdf view on the left side with predefined dummy data.

Bonus Features

- Form Validation: Implement comprehensive validation for all form fields.
- Error Handling: Display user-friendly error messages for invalid inputs.
- Dynamic Styling: Apply dynamic styles to indicate input validation status (e.g., error highlights).

Technology Stack

- React for building the user interface.
- Formik for managing forms and validation.
- LocalStorage for data persistence.
- `react-pdf` for rendering PDFs.

Submission Guidelines

Code Repository:

Submit the project code in a GitHub repository.
Include a README file with clear instructions on how to run the application.

Live Demo:

Host the application on a freely accessible platform and provide the link.

Application Quality:

Ensure the application runs without errors and adheres to the provided design and functionality requirements.

Evaluation Criteria

- Adherence to the provided design.
- Code quality, structure, and use of modern React practices.
- Effective use of Formik for form handling and validation.
- Proper integration of local storage for data persistence.
- Implementation of optional bonus features (if any).
- Overall user experience and interface design.