React.js Mini Project

Expense Management System

Course: Web Development / React.js
Project Type: Practical / Mini Project
Time Duration: As per instructor
Total Marks: As per evaluation scheme

Instructions to Students

1. Read all tasks carefully before starting the project.
2. This project must be developed using **React.js**.
3. The application should be simple, user-friendly, and beginner-level.
4. Proper component structure must be followed.
5. All functionalities must work correctly.

Project Title

Expense Management System Using React.js

Student Tasks (Functional Requirements)

Task 1: Create Project Interface

- Create a main screen with the title **“Expense Management System”**.
- Design a simple and clean layout.
- Divide the screen into:
    - Expense input section
    - Expense display section
    - Total expense section

Task 2: Add Expense Functionality

- Create a form to add a new expense.
- The form must include:
    - Expense Name
    - Expense Amount
    - Expense Date
    - Expense Category
- Add a button labeled **“Add Expense”**.
- When clicked, the expense should be added to the list.

Task 3: Input Validation

- Do not allow empty input fields.
- Ensure the amount entered is greater than zero.
- Display a simple error message for invalid input.

Task 4: Display Expense List

- Display all added expenses in a list format.
- Each expense should show:
    - Name
    - Amount
    - Date
    - Category
- If no expenses exist, show a message **“No expenses found”**.

Task 5: Delete Expense

- Add a **Delete** button for each expense.
- When clicked, the selected expense should be removed from the list.

Task 6: Edit Expense (Optional)

- Allow the user to edit an existing expense.
- Updated expense details should replace the old data.

Task 7: Calculate Total Expense

- Display the total amount of all expenses.
- The total should update automatically when:
    - An expense is added
    - An expense is deleted

Task 8: Filter Expenses

- Provide an option to filter expenses by:
    - Date
    - Category
- Display only the filtered expenses.

Task 9: Sort Expenses

- Allow sorting of expenses by:
    - Amount
    - Date
- Sorting should update the list instantly.

Task 10: Store Data Locally

- Save expense data in browser local storage.
- Expenses should remain after page refresh.

Task 11: Clear All Expenses

- Add a **Clear All** button.
- Ask for confirmation before removing all expenses.

Task 12: Responsive Design

- Ensure the application works on:
    - Mobile screens
    - Tablet screens
    - Desktop screens

Task 13: User Feedback Messages

- Show messages for:
    - Successful expense addition
    - Expense deletion
    - Errors or invalid input

Task 14: Accessibility Basics

- Use proper labels for input fields.
- Buttons should have clear names.
- Application should be keyboard accessible.

Evaluation Criteria (Suggested)

- Correct implementation of functionalities
- Proper use of React components and state
- Clean UI design
- Code readability
- Error handling and validation

Submission Guidelines

- Submit complete React project folder.
- Include:
    - Source code
    - Screenshots of application
    - SRS document (if required)
- Late submissions may not be accepted.

End of Question Paper

Best of luck.