Day 4: Testing and Deployment, focusing on Test-Driven Development (TDD), creating reusable components, and deploying applications.

Problem Statement

Build and Deploy a Simple To-Do List Application: You are required to create a simple To-Do List application that allows users to add, remove, and mark tasks as completed. This application should be developed using **Test-Driven Development (TDD)** principles, featuring reusable components, and finally, it should be deployed to a platform like Netlify or Vercel.

User Stories

1. **As a user**, I want to add new tasks to my to-do list.
2. **As a user**, I want to remove tasks that I no longer need.
3. **As a user**, I want to mark tasks as completed so that I can track my progress.
4. **As a developer**, I want to write unit tests for each component before implementing the functionality.
5. **As a developer**, I want to create reusable components to maintain consistency across the application.

Assignment Structure

1. Setup (10 Minutes)

- Create a new React application using `create-react-app`.
- Install testing libraries, including `Jest` and `React Testing Library`.

2. Implement Test-Driven Development (TDD) (25 Minutes)

- **Introduction to TDD Principles**
  - Write unit tests for each component before implementing their functionality.
  - Create test cases for adding, removing, and toggling tasks. Ensure the tests fail initially (red phase).
- **Develop Components**
  - Create the following components:
    - **TaskInput**: Input field for adding tasks, including a button to submit.
    - **TaskList**: A list to display tasks with the ability to mark them as completed or remove them.
    - **TaskItem**: A single task item that includes a checkbox and a remove button.
- **Write Tests**

    - Ensure that the tests for all components pass (green phase). Each component should have at least 2 tests:
        - **TaskInput**: Check if the input value updates and if the task is added to the list upon submission.
        - **TaskItem**: Check if the task can be marked as completed and removed correctly.

3. Creating Reusable Components (15 Minutes)

- **Best Practices for Reusable Components**
    - Implement the Task components to be reusable.
    - Ensure that each component accepts props for customization (e.g., styles, callback functions).
- **Documentation**
    - Document each component's usage, including expected props and behavior.

4. Deploying the Application (10 Minutes)

- **Prepare for Deployment**
    - Optimize and bundle the application using `npm run build`.
- **Deployment**
    - Deploy the application to Netlify or Vercel.
    - Ensure that the deployed application works as expected.

5. Final Touches (5 Minutes)

- **Test the Live Application**
    - Access the live URL and test the application to ensure all functionalities are working.
    - Verify that all test cases run successfully in the deployed version.

Submission Requirements

- Submit the code via a Git repository link.
- Include a README.md file that provides:
    - Instructions on how to run the application locally and how to access the live version.
    - An overview of the testing approach and the components created.
    - Screenshots of the application and explanations of key functionalities.

Evaluation Criteria

- **Functionality**: All user stories should be implemented correctly with a seamless user experience.

- **Testing**: Comprehensive unit tests written for all components, demonstrating understanding of TDD principles.
- **Code Quality**: Clean, organized code adhering to best practices in React.
- **Reusability**: Effective implementation of reusable components.
- **Deployment**: Successful deployment with live testing confirming application functionality.

Final Note

This coding assignment is designed to assess your ability to implement TDD, create reusable components, and deploy a React application. Please manage your time effectively to ensure all components are completed within the allotted 60 minutes. Good luck!