Laboratorio: Desarrollo de una Aplicación de Gestión de Tareas (Todo List)

Nombre: _____

Documento: _____

Objetivo: Crear una aplicación web de gestión de tareas utilizando React que incorpore los conceptos aprendidos en el Módulo 3.

Requerimientos:

1. Componentes y JSX:
    - Crear un componente principal `App`
    - Crear componentes para `TaskList`, `TaskItem`, y `TaskForm`
2. Props y Estados:
    - Utilizar props para pasar datos entre componentes
    - Implementar estados para manejar la lista de tareas y el formulario
3. Hooks:
    - Usar `useState` para manejar el estado de las tareas
    - Implementar `useEffect` para cargar tareas desde el almacenamiento local al iniciar la aplicación
4. Manejo de eventos y formularios:
    - Crear un formulario para añadir nuevas tareas
    - Implementar funcionalidad para marcar tareas como completadas
    - Añadir la opción de eliminar tareas
5. React Router:
    - Implementar rutas para:
        - Lista de todas las tareas
        - Tareas completadas
        - Tareas pendientes
6. Diseño responsive:
    - Utilizar CSS para crear un diseño que se adapte a diferentes tamaños de pantalla

Pasos a seguir:

1. **Configuración inicial:**
    - Crear un nuevo proyecto de React usando Create React App
    - Instalar las dependencias necesarias (react-router-dom)
2. **Crear los componentes básicos:**
    - `App.js`: Componente principal
    - `TaskList.js`: Lista de tareas
    - `TaskItem.js`: Tarea individual
    - `TaskForm.js`: Formulario para añadir tareas

3. **Implementar la funcionalidad básica:**
    - Añadir tareas
    - Marcar tareas como completadas
    - Eliminar tareas
4. **Agregar React Router:**
    - Configurar rutas en `App.js`
    - Crear componentes para diferentes vistas (AllTasks, CompletedTasks, PendingTasks)
5. **Estilizar la aplicación:**
    - Crear un archivo CSS para cada componente
    - Implementar un diseño responsive
6. **Persistencia de datos:**
    - Utilizar `localStorage` para guardar y cargar tareas

Código inicial para `App.js`:

```
import React, { useState, useEffect } from 'react';
import { BrowserRouter as Router, Route, Link, Switch } from
'react-router-dom';
import TaskList from './components/TaskList';
import TaskForm from './components/TaskForm';
import './App.css';

function App() {
  const [tasks, setTasks] = useState([]);

  useEffect(() => {
    const storedTasks = JSON.parse(localStorage.getItem('tasks')) || [];
    setTasks(storedTasks);
  }, []);

  useEffect(() => {
    localStorage.setItem('tasks', JSON.stringify(tasks));
  }, [tasks]);

  const addTask = (task) => {
    setTasks([...tasks, { ...task, id: Date.now(), completed: false }]);
  };

  const toggleTask = (id) => {
    setTasks(
      tasks.map((task) =>
        task.id === id ? { ...task, completed: !task.completed } : task
```

```
    )
  );
};

const deleteTask = (id) => {
  setTasks(tasks.filter((task) => task.id !== id));
};

return (
  <Router>
    <div className="App">
      <nav>
        <ul>
          <li>
            <Link to="/">Todas las tareas</Link>
          </li>
          <li>
            <Link to="/completed">Tareas completadas</Link>
          </li>
          <li>
            <Link to="/pending">Tareas pendientes</Link>
          </li>
        </ul>
      </nav>

      <TaskForm addTask={addTask} />

      <Switch>
        <Route path="/">
          <TaskList
            tasks={tasks}
            toggleTask={toggleTask}
            deleteTask={deleteTask}
          />
        </Route>
        <Route path="/completed">
          <TaskList
            tasks={tasks.filter((task) => task.completed)}
            toggleTask={toggleTask}
            deleteTask={deleteTask}
          />
        </Route>
        <Route path="/pending">
```

```
            <TaskList
              tasks={tasks.filter((task) => !task.completed)}
              toggleTask={toggleTask}
              deleteTask={deleteTask}
            />
          </Route>
        </Switch>
      </div>
    </Router>
  );
}

export default App;
```

Entrega:

1. Código fuente comentado de la implementación.
2. Informe breve que incluya:
    - Descripción de la estructura del sistema y decisiones de diseño.
    - Discusión sobre las ventajas y desventajas de estrategias, herramientas y hooks utilizados.
    - Propuesta de mejoras o alternativas para optimizar el rendimiento.