Title: Advanced Angular Custom Pipes Lab Exercise: Data Filtering and Sorting

Objective: In this lab exercise, you will create advanced custom pipes in Angular to handle data filtering and sorting. You will develop custom pipes to filter and sort data based on user input, providing a dynamic and responsive user experience.

Requirements:

- Proficiency in Angular framework.
- Angular CLI installed.
- Knowledge of TypeScript.
- Code editor (e.g., Visual Studio Code).

Scenario: You are developing a data management application that displays a list of items. Users should be able to filter and sort the items based on various criteria such as name, date, and category. You are required to implement custom pipes to filter and sort the data efficiently.

Tasks:

1. **Create Angular Project:**
    - Generate a new Angular project using Angular CLI.
    - Navigate into the project directory.
2. **Create Custom Pipes:**
    - Create a custom pipe named `FilterPipe`. This pipe should filter the data based on user-defined criteria (e.g., name, date, category).
    - Create another custom pipe named `SortPipe`. This pipe should sort the data based on the selected sorting criteria (e.g., ascending, descending).
3. **Implement Custom Pipes:**
    - Implement the `FilterPipe` to filter the data based on user-defined criteria such as name, date range, and category.
    - Implement the `SortPipe` to sort the filtered data based on the selected sorting criteria (e.g., alphabetical order, date, category).
4. **Integrate Custom Pipes:**
    - Integrate the custom pipes into the data management application.
    - Apply the `FilterPipe` to dynamically filter the displayed data based on user input.

- Apply the `SortPipe` to sort the filtered data based on the selected sorting criteria.

5. **Test and Validate:**
   - Test the application to ensure that the custom pipes are functioning as expected.
   - Verify that the data is filtered and sorted dynamically based on user input.
   - Test edge cases and boundary conditions to ensure robustness.
6. **Enhancements (Optional):**
   - Add additional features or improvements to the custom pipes (e.g., support for complex filtering criteria, customizable sorting options).
   - Implement pagination functionality to handle large datasets more efficiently.

Submission:

- Submit the source code of your Angular project along with a detailed report documenting the implementation of the custom pipes, any additional features or enhancements, and test results.

Note: Ensure that you follow best practices for Angular development, including proper folder structure, component separation, and code organization. Use HTML and CSS to enhance the visual presentation of the data management application and the custom pipes. Additionally, consider implementing unit tests for the custom pipes to ensure reliability and maintainability.