To-Do List Data Flow

1. Introduction

This document provides a detailed Data Flow Diagram (DFD) representation of the To-Do List React application.

It explains how state management, user input handling, and dynamic UI updates are managed.

2. System Overview

The React application consists of the following key components:

- App.js: The main file that initializes the application and manages tasks.
- useState Hook: Maintains the list of tasks and user input.
- Input Field: Accepts new tasks from the user.
- Button: Adds tasks to the list when clicked.
- Task List: Displays all added tasks dynamically.

3. Data Flow Diagram (DFD)

Level 0 (Context Diagram)

At the highest level, the system consists of external users interacting with the to-do list UI.

External Entities:

- User: Enters tasks and adds them to the list.

Processes:

- React Application: Manages task input, updates state, and renders the task list.

Data Stores:

- useState Hook: Stores the list of tasks.

```
  +--------------------+
  |     User           |
  +--------------------+
           |
           v
  +--------------------+
  |  React App (UI)   |
  +--------------------+
           |
           v
  +--------------------+
  | useState (State)  |
  +--------------------+
```

Level 1 DFD (Task Addition Process)

This level breaks down how user input is processed and added to the task list.

```
+--------------------+        +--------------------+
|  User Enters Task | -----> |  Update useState   |
+--------------------+        +--------------------+
        |                             |
        v                             v
+--------------------+        +--------------------+
|  User Clicks Btn  | -----> |  Add Task to List  |
+--------------------+        +--------------------+
```

Process Explanation:

1. The user types a task in the input field.
2. The `newTask` state updates as the user types.
3. When the "Add Task" button is clicked, the task is added to the list.

Level 2 DFD (State Update and UI Changes)

```
+----------------------+        +----------------------+
|  Append Task to List | -----> |  Re-render UI        |
+----------------------+        +----------------------+
         |                        |
         v                        v
+----------------------+    +----------------------+
|  Display Updated List | -----> |  Show New Tasks    |
+----------------------+    +----------------------+
```

Process Explanation:

1. The new task is appended to the `tasks` array using `setTasks([...tasks, newTask])`.
2. React detects the state change and re-renders the UI.
3. The updated list of tasks is displayed to the user.

4. Explanation of Data Flow

1. The user interacts with the input field to type a task.
2. The component updates state using `useState`.
3. Clicking "Add Task" adds the task to the list and triggers a re-render.
4. The task list updates dynamically as new tasks are added.

5. Conclusion

This document outlines the detailed data flow in a simple React-based To-Do List application.

The application dynamically manages tasks using React state management and user interactions.