HLD Output 3 | Qwen3:4b

DIT Self Assessment Tool User

High-Level Design Document

1. Overview

This document outlines the high-level design of a UI component system that uses JSON-based definitions to describe UI elements and their behaviors. The system includes buttons with properties such as text, color, border color, and `OnSelect` actions. The goal is to provide a structured, reusable, and maintainable way to define UI elements and their interactions.

---

2. System Architecture

2.1 Data Structure

- **Root Object**: A JSON object containing a list of UI elements.
- **UI Element**: A JSON object representing a single UI component (e.g., button, label, etc.).
- **Properties**:
  - `Id`: Unique identifier for the UI element.
  - `Text`: Text displayed on the element.
  - `Color`: Color of the element (in RGBA format).
  - `BorderColor`: Border color of the element (in RGBA format).
  - `OnSelect`: Action to execute when the element is clicked (e.g., navigation, function call).

2.2 UI Component Types

- **Button**: A clickable element with a text label and visual styling.
- **Label**: A static text element with styling.
- **Input Field**: A text input element.

- **Image**: A visual element with a source and styling.

---

3. Core Components

3.1 UI Element Definition

Each UI element is defined as a JSON object with the following structure:

```
{
  "Id": "button1",
  "Text": "Submit",
  "Color": "RGBA(0, 0, 0, 1)",
  "BorderColor": "RGBA(204, 204, 204, 1)",
  "OnSelect": "Navigate('Line Manager Main Screen')"
}
```

3.2 UI Element Rendering

- UI elements are rendered based on their `Id`, `Text`, `Color`, and `BorderColor`.
- The `OnSelect` property defines the action to be executed when the element is clicked.

3.3 UI Element Interaction

- The `OnSelect` property is a string that represents a function or action to be executed.
- Supported actions include:
  - `Navigate('ScreenName')`: Navigate to a specific screen.
  - `TriggerFunction('FunctionName')`: Call a predefined function.
  - `ShowToast('Message')`: Display a toast message.

---

4. Data Flow

4.1 Input

- JSON data is provided as input, defining the UI elements and their properties.

4.2 Processing

- The JSON is parsed and validated.
- UI elements are rendered based on their properties.
- `OnSelect` actions are executed when the element is clicked.

4.3 Output

- Rendered UI elements.
- Execution of `OnSelect` actions (e.g., navigation, function calls).

---

5. Validation and Error Handling

5.1 Validation Rules

- All UI elements must have a unique `Id`.
- `OnSelect` must be a valid action string.
- `Color` and `BorderColor` must be in RGBA format.

5.2 Error Handling

- If a UI element is missing a required property (e.g., `Id`, `Text`), an error is logged.
- If an invalid `OnSelect` action is provided, it is flagged as an error.

---

6. Extensibility

6.1 Support for New UI Elements

- The system supports the addition of new UI element types (e.g., `InputField`, `Image`) by extending the JSON structure.

6.2 Support for New Actions

- New actions can be added to the `OnSelect` property by defining them in the system's action registry.

---

7. Use Cases

7.1 Navigation Between Screens

- Buttons with `OnSelect` set to `Navigate('ScreenName')` are used to move between screens.

7.2 Function Triggers

- Buttons can trigger predefined functions (e.g., `TriggerFunction('SaveData')`).

7.3 Toast Notifications

- Buttons can display toast messages using `ShowToast('Message')`.

---

8. Future Enhancements

- **Dynamic UI Generation**: Generate UI elements dynamically based on data.
- **UI State Management**: Support for UI states (e.g., enabled/disabled, focused).
- **UI Animation**: Support for animations and transitions between UI states.

---

9. Conclusion

This high-level design provides a structured and extensible way to define and manage UI elements with their associated behaviors. The system is based on a JSON data model, making it easy to maintain, extend, and integrate with other systems. The `OnSelect` property is a key feature that enables interaction and dynamic behavior in the UI.