**Pixel Tech Frontend Developer Assignment**

**Task Overview:**

The objective of this assignment is to create an **Interactive Payment Provider Workflow** using the **React Flow** library with **TypeScript**. Users should be able to add, manage, and visualize payment provider nodes such as Google Pay, Apple Pay, Stripe, etc., and perform interactions like dragging, deleting, and saving the workflow. Additionally, the workflow should support **node resizing**, **connection validation**, and **undo/redo** functionality.

**Main Features:**

**1.** **Add Payment Provider Node:**

* 1. Provide a dropdown or list from which users can add payment providers (e.g., **Google Pay**, **Stripe**, or **Apple Pay**) to the canvas.
  2. Ensure that each payment provider node is unique (no duplicates).

**2.** **Delete Node:**

* 1. Provide a delete button (usually an “X” icon) on each payment provider node for easy removal.

**3.** **Drag and Drop:**

* 1. Allow users to drag and reposition nodes within the canvas.
  2. Ensure smooth interaction for dragging nodes in real-time.

**4.** **Node Resizing:**

* 1. Implement the ability to resize nodes for better customization.
  2. Users should be able to increase or decrease the size of the payment provider nodes.

**5.** **Node Connection Validation:**

* 1. Ensure logical connections between nodes. For example, a **Payment Initialized** node must connect to a **payment provider node** (like **Stripe** or **Google Pay**).
  2. Invalid connections should provide visual feedback (e.g., error styling or tooltips).

**6.** **Conditional styling:**

* 1. Apply dynamic styles to nodes based on conditions. For example, change the color of the **Payment Initialized** node if the payment amount exceeds a threshold.

**7.** **Display Payment Initialization:**

* 1. Allow users to add a node representing a payment being initialized.
  2. Display relevant data such as the **payment amount** on the node.

**8.** **Undo/Redo:**

* 1. Implement **undo/redo** functionality for node addition, deletion, dragging, and resizing.
  2. Allow users to undo or redo recent actions to make the workflow more flexible.

**9.** **Save and Load Workflow:**

* 1. Provide functionality to **save** the current state of the workflow (including all nodes and connections) to **local storage**.
  2. Enable users to **load** a saved workflow when returning to the application, restoring all nodes, connections, and layouts.

**10.** **Validation:**

* 1. Ensure that users cannot add **duplicate** payment provider nodes. If a user attempts to add a node that already exists, show an error message or block the action.

**11.** **Auto Layout:**

* 1. Implement a feature that automatically arranges nodes to avoid overlapping or clutter.
  2. Provide an option for users to manually trigger this auto-layout functionality if needed.

|  |  |
| --- | --- |
|  |  |
|  |  |

**Bonus Features:**

**1.** **Export/Import:**

* 1. Provide functionality to export the workflow as a **JSON** file and re-import it for restoration.

**2.** **Zoom and Pan:**

* 1. Allow users to zoom in/out and pan around the canvas to manage large workflows.

**3.** **Highlight Connected Nodes:**

* 1. When a user selects a node, highlight all connected nodes and edges to visualize the workflow.

**Deliverables:**

1. A **GitHub repository** with the project code.
2. A README.md file containing:
   1. Setup instructions.
   2. A brief description of how you implemented core features like node resizing, undo/redo, and validation.
3. An explanation of how you implemented features such as **auto-layout** and **node connection validation**.

# **Visual Example:**



|  |  |
| --- | --- |
|  | **For the** **reference, please go through this video:** [**331101642-5d054bbc-c0e4-4341-bd62-9deb546356ed (1).mp4**](https://1drv.ms/v/s!AmUx1eLY8xtTgadT4n1Bwvv_aqJDOA?e=X2Jv1f) |
|  | ***Once you have completed the assignment must fill this form:-*** <https://forms.office.com/r/zucPTuiFpx> |