

## src\App.jsx

```
1 import { PipelineToolbar } from './toolbar';
2 import { PipelineUI } from './ui';
3 import { SubmitButton } from './submit';
4
5 function App() {
6   return (
7     <div>
8       <PipelineToolbar />
9       <PipelineUI />
10      <SubmitButton />
11    </div>
12  );
13 }
14
15 export default App;
16
```

## src\toolbar.jsx

```
1 // toolbar.js
2
3 import { DraggableNode } from './draggableNode';
4
5 export const PipelineToolbar = () => {
6
7   return (
8     <div style={{ padding: '10px' }}>
9       <div style={{ marginTop: '20px', display: 'flex', flexWrap: 'wrap', gap: '10px' }}>
10         <DraggableNode type='customInput' label='Input' />
11         <DraggableNode type='llm' label='LLM' />
12         <DraggableNode type='customOutput' label='Output' />
13         <DraggableNode type='text' label='Text' />
14       </div>
15     </div>
16   );
17 };
18
```

## src\ui.jsx

```

1 // ui.js
2 // Displays the drag-and-drop UI
3 // -----
4
5 import { useState, useRef, useCallback } from 'react';
6 import ReactFlow, { Controls, Background, MiniMap } from 'reactflow';
7 import { useStore } from './store';
8 import { shallow } from 'zustand/shallow';
9 import { InputNode } from './nodes/inputNode';
10 import { LLMNode } from './nodes/llmNode';
11 import { OutputNode } from './nodes/outputNode';
12 import { TextNode } from './nodes/textNode';
13
14 import 'reactflow/dist/style.css';
15
16 const gridSize = 20;
17 const proOptions = { hideAttribution: true };
18 const nodeTypes = {
19   customInput: InputNode,
20   llm: LLMNode,
21   customOutput: OutputNode,
22   text: TextNode,
23 };
24
25 const selector = (state) => ({
26   nodes: state.nodes,
27   edges: state.edges,
28   getNodeID: state.getNodeID,
29   addNode: state.addNode,
30   onNodesChange: state.onNodesChange,
31   onEdgesChange: state.onEdgesChange,
32   onConnect: state.onConnect,
33 });
34
35 export const PipelineUI = () => {
36   const reactFlowWrapper = useRef(null);
37   const [reactFlowInstance, setReactFlowInstance] = useState(null);
38   const {
39     nodes,
40     edges,
41     getNodeID,
42     addNode,
43     onNodesChange,
44     onEdgesChange,
45     onConnect
46   } = useStore(selector, shallow);
47
48   const getInitNodeData = (nodeID, type) => {
49     let nodeData = { id: nodeID, nodeType: `${type}` };
50     return nodeData;
51   }
52
53   const onDrop = useCallback(
54     (event) => {

```

```

55     event.preventDefault();
56
57     const reactFlowBounds = reactFlowWrapper.current.getBoundingClientRect();
58     if (event?.dataTransfer?.getData('application/reactflow')) {
59         const appData = JSON.parse(event.dataTransfer.getData('application/reactflow'));
60         const type = appData?.nodeType;
61
62         // check if the dropped element is valid
63         if (typeof type === 'undefined' || !type) {
64             return;
65         }
66
67         const position = reactFlowInstance.project({
68             x: event.clientX - reactFlowBounds.left,
69             y: event.clientY - reactFlowBounds.top,
70         });
71
72         const nodeID = getNodeID(type);
73         const newNode = {
74             id: nodeID,
75             type,
76             position,
77             data: getInitNodeData(nodeID, type),
78         };
79
80         addNode(newNode);
81     }
82 },
83 [reactFlowInstance]
84 );
85
86 const onDragOver = useCallback((event) => {
87     event.preventDefault();
88     event.dataTransfer.dropEffect = 'move';
89 }, []);
90
91 return (
92     <>
93     <div ref={reactFlowWrapper} style={{width: '100vw', height: '70vh'}}>
94         <ReactFlow
95             nodes={nodes}
96             edges={edges}
97             onNodesChange={onNodesChange}
98             onEdgesChange={onEdgesChange}
99             onConnect={onConnect}
100             onDrop={onDrop}
101             onDragOver={onDragOver}
102             onInit={setReactFlowInstance}
103             nodeTypes={nodeTypes}
104             proOptions={proOptions}
105             snapGrid={[gridSize, gridSize]}
106             connectionLineType='smoothstep'
107         >
108             <Background color="#aaa" gap={gridSize} />
109             <Controls />
110             <MiniMap />

```

```
111         </ReactFlow>
112     </div>
113 </>
114 )
115 }
116
```

## src\submit.jsx

```
1 // submit.js
2
3 export const SubmitButton = () => {
4
5     return (
6         <div style={{display: 'flex', alignItems: 'center', justifyContent: 'center'}}>
7             <button type="submit">Submit</button>
8         </div>
9     );
10 }
11
```

## src\draggableNode.jsx

```
1 // draggableNode.js
2
3 export const DraggableNode = ({ type, label }) => {
4   const onDragStart = (event, nodeType) => {
5     const appData = { nodeType }
6     event.target.style.cursor = 'grabbing';
7     event.dataTransfer.setData('application/reactflow', JSON.stringify(appData));
8     event.dataTransfer.effectAllowed = 'move';
9   };
10
11   return (
12     <div
13       className={type}
14       onDragStart={(event) => onDragStart(event, type)}
15       onDragEnd={(event) => (event.target.style.cursor = 'grab')}
16       style={{
17         cursor: 'grab',
18         minWidth: '80px',
19         height: '60px',
20         display: 'flex',
21         alignItems: 'center',
22         borderRadius: '8px',
23         backgroundColor: '#1C2536',
24         justifyContent: 'center',
25         flexDirection: 'column'
26       }}
27       draggable
28     >
29       <span style={{ color: 'fff' }}>{label}</span>
30     </div>
31   );
32 };
33
```

## src\nodes\inputNode.jsx

```
1 // inputNode.js
2
3 import { useState } from 'react';
4 import { Handle, Position } from 'reactflow';
5
6 export const InputNode = ({ id, data }) => {
7   const [currName, setCurrName] = useState(data?.inputName || id.replace('customInput-', '
input_'));
8   const [inputType, setInputType] = useState(data.inputType || 'Text');
9
10  const handleNameChange = (e) => {
11    setCurrName(e.target.value);
12  };
13
14  const handleTypeChange = (e) => {
15    setInputType(e.target.value);
16  };
17
18  return (
19    <div style={{width: 200, height: 80, border: '1px solid black'}}>
20      <div>
21        <span>Input</span>
22      </div>
23      <div>
24        <label>
25          Name:
26          <input
27            type="text"
28            value={currName}
29            onChange={handleNameChange}
30          />
31        </label>
32        <label>
33          Type:
34          <select value={inputType} onChange={handleTypeChange}>
35            <option value="Text">Text</option>
36            <option value="File">File</option>
37          </select>
38        </label>
39      </div>
40      <Handle
41        type="source"
42        position={Position.Right}
43        id={` ${id}-value` }
44      />
45    </div>
46  );
47 }
48
```



## src\nodes\llmNode.jsx

```
1 // llmNode.js
2
3 import { Handle, Position } from 'reactflow';
4
5 export const LLMNode = ({ id, data }) => {
6
7   return (
8     <div style={{width: 200, height: 80, border: '1px solid black'}}>
9       <Handle
10         type="target"
11         position={Position.Left}
12         id={` ${id}-system` }
13         style={{top: ` ${100/3}%` }}
14       />
15       <Handle
16         type="target"
17         position={Position.Left}
18         id={` ${id}-prompt` }
19         style={{top: ` ${200/3}%` }}
20       />
21       <div>
22         <span>LLM</span>
23       </div>
24       <div>
25         <span>This is a LLM.</span>
26       </div>
27       <Handle
28         type="source"
29         position={Position.Right}
30         id={` ${id}-response` }
31       />
32     </div>
33   );
34 }
35
```

## src\nodes\outputNode.jsx

```
1 // outputNode.js
2
3 import { useState } from 'react';
4 import { Handle, Position } from 'reactflow';
5
6 export const OutputNode = ({ id, data }) => {
7   const [currName, setCurrName] = useState(data?.outputName || id.replace('customOutput-', '
output_'));
8   const [outputType, setOutputType] = useState(data.outputType || 'Text');
9
10  const handleNameChange = (e) => {
11    setCurrName(e.target.value);
12  };
13
14  const handleTypeChange = (e) => {
15    setOutputType(e.target.value);
16  };
17
18  return (
19    <div style={{width: 200, height: 80, border: '1px solid black'}}>
20      <Handle
21        type="target"
22        position={Position.Left}
23        id={` ${id}-value` }
24      />
25      <div>
26        <span>Output</span>
27      </div>
28      <div>
29        <label>
30          Name:
31          <input
32            type="text"
33            value={currName}
34            onChange={handleNameChange}
35          />
36        </label>
37        <label>
38          Type:
39          <select value={outputType} onChange={handleTypeChange}>
40            <option value="Text">Text</option>
41            <option value="File">Image</option>
42          </select>
43        </label>
44      </div>
45    </div>
46  );
47 }
48
```

## src\nodes\textNode.jsx

```
1 // textNode.js
2
3 import { useState } from 'react';
4 import { Handle, Position } from 'reactflow';
5
6 export const TextNode = ({ id, data }) => {
7   const [currText, setCurrText] = useState(data?.text || '{{input}}');
8
9   const handleTextChange = (e) => {
10     setCurrText(e.target.value);
11   };
12
13   return (
14     <div style={{width: 200, height: 80, border: '1px solid black'}}>
15       <div>
16         <span>Text</span>
17       </div>
18       <div>
19         <label>
20           Text:
21           <input
22             type="text"
23             value={currText}
24             onChange={handleTextChange}
25           />
26         </label>
27       </div>
28       <Handle
29         type="source"
30         position={Position.Right}
31         id={` ${id}-output` }
32       />
33     </div>
34   );
35 }
36
```