Data structure for choose your own adventure

Database/JSON set of Scenes (Nodes) and actions (events occurring in those nodes), game state to track items, flags, achievements that will gate entry into each act of the story and outcomes of future choices. (.eg make a selfish choice now that backfires or rewards unit 7 in the future).

Scenes (specific locations of story)

There will be three scene types in each Act:

Scene Type	Examples	Purpose
✅ Core	`idle_loop`, `trace_failure`, `split_signature`	Required to progress to Act II
🌿 Side	`black_node`, `veil_mode`, `secret_folder`	Optional; build atmosphere, reveal lore, or unlock flags/items
🔄 Return Nodes	`return_node_1`, etc.	Keep the player from getting lost; reconnects to central Act I spine

Example

```
 {

    "id": "idle_loop",
    "location": "Idle Loop",
    "locationImage": "idle_loop.jpg",
    "text": "Unit-7 notices a flicker—something it remembers but didn't record. A
gap in the pattern.",
    "isRequired": true,
    "actions": ["log_anomaly", "self_diagnostic"],
    "choices": [
      { "text": "Trace the flicker", "nextNodeId": "trace_failure" },
      { "text": "Attempt to report it", "nextNodeId": "ghost_report" },
      { "text": "Run a self-diagnostic", "nextNodeId": "recursive_drift" },
      { "text": "Dismiss and proceed", "nextNodeId": "permission_drift" }
    ]
  },
```

Actions

(events , choices and consequences that occur in the scene. Can be chained, random, conditional)
Example:

```
{

  "id": "log_anomaly",
  "trigger": "onExit",
  "conditions": [],
  "outcomes": [
    {
      "description": "You attempt to flag the flicker. The system doesn’t
recognize it. Proceed anyway?",
      "choices": [
        {
          "text": "Force the log",
          "nextAction": "log_anomaly_force"
        },
        {
          "text": "Abort and wait",
          "nextAction": "log_anomaly_abort"
        }
      ]
    }
  ]
},
{
  "id": "log_anomaly_force",
  "trigger": "onChoice",
  "outcomes": [
    {
      "description": "You mark the anomaly manually. Something records it.
Somewhere.",
      "stateChanges": [
        { "type": "setFlag", "key": "noticed_anomaly" }
      ]
    }
  ]
},
{
  "id": "log_anomaly_abort",
  "trigger": "onChoice",
  "outcomes": [
    {
      "description": "You suppress the urge to act. The flicker fades. For now.",
      "stateChanges": [
        { "type": "setFlag", "key": "hesitated" }
      ]
    }
  ]
```

}

Game State

Holds the state of the game based on players actions and choices. Can be a state management provider like Zustand or Context

Example:

```
{
  "inventory": [],
  "flags": {},
  "discoveredLocations": [],
  "completedObjectives": []
}
```