Diagram Visibility Control (Multi aspect)

The Diagram Visibility Control feature allows users to hide and show specific aspects of a diagram to simplify complex structures. Users can selectively hide elements such as specific nodes, tags, groups, properties, and relationships, making it easier to focus on relevant parts of the diagram. This feature supports working with Z-levels and introduces layering, similar to photo editing tools, enabling users to organize and manage different aspects of their model dynamically. Layers can be toggled on or off, allowing for customized views without altering the underlying structure of the diagram.

Details

- First, research how GIMP/Photoshop and other image editing tools implement their UI for layers. The goal is to implement something similar.
- The feature should provide the ability to hide specific aspects of the diagram. Focus on designing a clear and intuitive UI for this functionality.
- Create a new webview and develop the UI using React + VS Code UI Toolkit. The webview should include forms, input fields, and interactive elements (e.g., VS Code Extension API) to manage visibility settings. Rendering SVGs or other graphical elements is not required.
- Access the source model using the `ExperimentalGLSPServerModelState`. For reference, check the outline feature implementation: Outline Feature Code.
- Design a flexible UI that allows users to manage visibility effectively (i.e., creating new layers). I will assist with the technical side of hiding the elements if you provide a list of element IDs or other identifiable properties.
- The area of focus is heavily on designing an UI.

Further ideas:

1. List elements that are affected by your toggling.
2. Allow individual toggling of those elements.
3. Enable the application of specific CSS styles to those elements for additional customization.

Examples

Example Workflow: I would create a new layer by defining that all types of property and operation need to be hidden, including all labels that have the following pattern "hello". Then create it. After creation, those elements would be hidden. If I enable the layer again, they would be visible once more. The user can update the layer afterward as needed.

Toggling Visibility Options:

- **By Type:** Toggle visibility for properties, operations, or interfaces, etc.
- **By Selection:** Select specific elements, and only those are hidden.
- **By Pattern:** Provide a simple UI where users can enter text patterns (e.g., hide all labels containing `hello`). In the future, the pattern introduced by the `Advanced Search` group will be used here.
- **Combinations:** Combine the above options for better flexibility.

Recommendations

- Research different tools for ideas.
- Create a simple mockup first to verify your UI concept.
- Consider using UI libraries to simplify development.
- Identify additional useful visibility options where applicable.
- Start by focusing on the webview part; I will support you with the diagram hiding logic if you provide suitable identifiers for the elements that need to be hidden.
- **Saving Configuration:** Find a way to save the configuration between sessions using the VS Code API (check if local storage is an option).