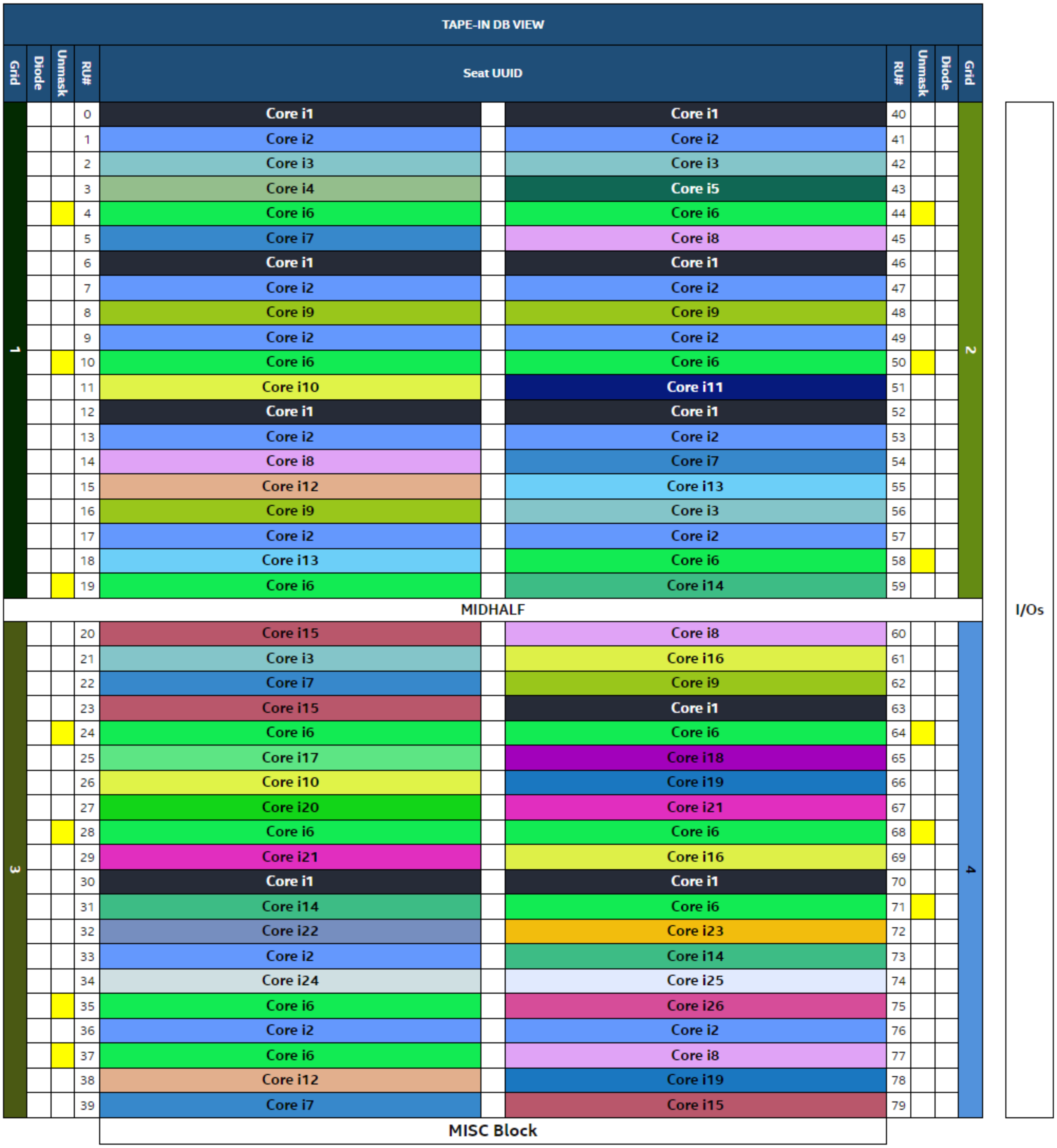
Generate Floor Plan from JSON Dataset

Objective: Create a JavaScript Application in a popular framework (React/Vue/Angular) that takes a JSON dataset containing product names and their repetition count. Generate a floor plan (RU Table) based on the given constraints.



TAPE-IN DB VIEW

Grid	Diode	Unmask	RU#	Seat UUID		Seat UUID	RU#	Unmask	Diode	Grid
1			0	Core i1		Core i1	40			2
			1	Core i2		Core i2	41			
			2	Core i3		Core i3	42			
			3	Core i4		Core i5	43			
		🟨	4	Core i6		Core i6	44	🟨		
			5	Core i7		Core i8	45			
			6	Core i1		Core i1	46			
			7	Core i2		Core i2	47			
			8	Core i9		Core i9	48			
			9	Core i2		Core i2	49			
		🟨	10	Core i6		Core i6	50	🟨		
			11	Core i10		Core i11	51			
			12	Core i1		Core i1	52			
			13	Core i2		Core i2	53			
			14	Core i8		Core i7	54			
			15	Core i12		Core i13	55			
			16	Core i9		Core i3	56			
			17	Core i2		Core i2	57			
			18	Core i13		Core i6	58	🟨		
		🟨	19	Core i6		Core i14	59			
MIDHALF										
3			20	Core i15		Core i8	60			4
			21	Core i3		Core i16	61			
			22	Core i7		Core i9	62			
			23	Core i15		Core i1	63			
		🟨	24	Core i6		Core i6	64	🟨		
			25	Core i17		Core i18	65			
			26	Core i10		Core i19	66			
			27	Core i20		Core i21	67			
		🟨	28	Core i6		Core i6	68	🟨		
			29	Core i21		Core i16	69			
			30	Core i1		Core i1	70			
			31	Core i14		Core i6	71	🟨		
			32	Core i22		Core i23	72			
			33	Core i2		Core i14	73			
			34	Core i24		Core i25	74			
		🟨	35	Core i6		Core i26	75			
			36	Core i2		Core i2	76			
		🟨	37	Core i6		Core i8	77			
			38	Core i12		Core i19	78			
			39	Core i7		Core i15	79			

MISC Block

I/Os

Requirements:

Your task is to create a table using any front-end framework (React/Vue/Angular) as shown in the image above. The table (RU Table) should have the features described below:

1. Go through the JSON dataset containing product names and their repetition count to get a clear understanding of the data structure.
2. Make necessary modifications to the data for optimal table rendering.
3. Generate a table (RU Table) floor plan on the Front-end consisting of 80 RUs (Rack Units) divided into 4 grids. Feel free to use different components to render various sections of the table.
4. Add “**Diode**” and “**Unmask**” columns on both sides of the floor plan grid.
    a. The “Diode” cell will be interactive. When clicked, the cell should change its color to blue.
    b. The “Unmask” cell is also interactive. When clicked, all cells containing products with the same name within the “Unmask” column should change their color to yellow. (Example: When user clicks on Core i1 Unmask column all respective Core i1 Unmask cells should change to yellow)
5. Render the products on the floor plan based on the repetition count.
6. **Implement an algorithm** to ensure that the constraints below are met.
7. Other columns such as I/Os, Grid, RU# in the table are self-explanatory.
8. MIDHALF row should be rendered after 20 RUs to create a grid like structure.

Constraints:

- Core i4/i5 products can only be placed in grids 1 and 2.
- No two products can be adjacent to each other. (Example: Core i1 in grid 1 and RU#0 cannot be next to another Core i1 in grid 1 RU#1, but can be placed at RU#2 in grid 1)

Evaluation Criteria:

1. Correct implementation of JSON parsing.
2. Proper placement of products based on constraints.
3. Efficient use of space (avoiding vertical placement of similar products).
4. Code readability and structure.
5. Proper documentation and comments.
6. Effective use of FE framework concepts such as props, states, and components.