In general, I am going to give you a method. You correct me if I am correct. Intention for this is that I am trying to understand if I learn React properly?

In this project, I am going to category the variables to 3 CLASSES

Class.1: nodeX, nodeY, nodeZ, startNode, endNode

Class.2: nodes, elements

Class.3: memberLengths

Class.1 is just giving information through “handle”s functions by using “axios.post” to backend as data to calculate something.

Class.2 is giving information through “handle”s functions by using “axios.post” to BE as data to calculate something. And getting some results by using “axios.get” from BE.

Class.3 is only getting the result from BE by “axios.get”

Process for sending only nodes data:

1. veriables

const [nodeX, setNodeX] = useState(''); // 🔴 //

    const [nodeY, setNodeY] = useState(''); // 🟡 //

    const [nodeZ, setNodeZ] = useState(''); // 🟢 //

1. sending to BE

const response = await axios.post('http://127.0.0.1:8000/api/nodes/', { // 🔵 //

                    x: parseFloat(nodeX),   // 🔴 //

                    y: parseFloat(nodeY),   // 🟡 //

                    z: parseFloat(nodeZ),   // 🟢 //

                });

1. getting data from FE

path('nodes/', NodeListCreateAPIView.as\_view(), name='nodes-list')

1. receiving data:

class NodeListCreateAPIView(APIView):

def post(self, request):

serializer = NodeSerializer(data=request.data)

1. converting data:

class NodeSerializer(serializers.ModelSerializer): # // 🔴 // 🟡 // 🟢 //

class Meta:

model = Node

1. saving data:

class Node(models.Model):

project = models.ForeignKey(Project, on\_delete=models.CASCADE, null=True, blank=True)

x = models.FloatField()

y = models.FloatField()

z = models.FloatField()

If you want me to **break down an app's code process like this**, you can ask in **one of the following ways**:

**💡 Option 1: General Request**

📝 **"Break down the process of [specific functionality] in my app, step by step, from FE to BE and back, just like you did before."**

**Example:**  
👉 *"Break down the process of sending and receiving element data in my app, step by step, from FE to BE and back, just like you did for nodes."*

**💡 Option 2: Specific Direction**

📝 **"Explain the full process of how [specific data] moves from [starting point] to [end point]. Include all steps, like variables, API calls, serializers, models, and responses."**

**Example:**  
👉 *"Explain the full process of how the member lengths are calculated and sent from BE to FE, including models, serializers, views, and API requests."*

**💡 Option 3: Debugging Focus**

📝 **"Analyze the process of [specific functionality] and find where things might be breaking. Explain the flow step by step."**

**Example:**  
👉 *"Analyze the process of calculating DOF in my app. Find where the issue is and break it down step by step."*

**💡 Option 4: Reverse Process Request**

📝 **"Explain how [specific data] is received and displayed in FE, step by step, in reverse order from MySQL to UI."**

**Example:**  
👉 *"Explain how node data is received and displayed in FE, step by step, from MySQL to UI in reverse order."*