Overall issues/difficulties:

* Using scenes with export will duplicate the canvas. Trying to use ES6 with the three.js scene would duplicate the entire scene for some reason.
* Visually displaying the angle correctly was not possible. SO I had to calculate them afterwards making it extremely messy
* Merging three.js with intelliJ locally. Instead I had to use a CDN for representation.
* JSON parsing, reading and file structuring were .

Things that were annoying:

* Resetting and starting the server in spring boot. In javascript using vite, it easily resets and refreshes the server with the correct code, whereas spring boot sometimes doesn't, which leads to me needing to close intellij as a whole and reopen it just for the changes to update.
* Three.js as a whole.
* Object formatting in blender. Something I realised at the very last minute, leading to me using the same object for all instances.

Overall takeaways:

The importance of the feasibility and requirements:

Overscoping the requirements, when sometimes it's best to follow the requirements. I was planning with an agile mindset, whilst implementing with a waterfall mindset. Remember that these are sprints. Ensure you have a working somewhat presentable product at the end, then expanding the scope in un-realistic ways. This is a prototype, not the final product, you can add the scopes in the next sprint. I do believe that this is also due to the fact that I tend to be a perfectionist; not moving on until I am satisfied; which leads to the lack of visual quality.

Alongside this, the importance of instructing users on what they are looking at and what to do. I was too focused on the quantity rather than the quality. For instance, having a good base prototype that tackles the ergonomics of the application, rather than the different things it can do.