**79. Lifecycle Hooks and Template Access**

* Presenter: Before concluding this module, one more thing related to our hooks here.
* We learned about @ViewChild, which we can use here in the cockpit to get access to a element from our, well DOM, from our template.
* Let's do the same here in the server element component html file where we have all these lifecycle hooks.
* Let's say I want to get access to this heading so I'll simply place a local reference named heading on it.
* Therefore, I'll add a new @ViewChild property make sure to import ViewChild from angular core.
* And with the data here I'll store this in uh header umm property which is of type element ref as we learned which also needs to be imported from angular core.
* And I want to select the heading local reference which I just placed on this div here.
* So if that we got access to it and now we would expect that we can't use this before we reached AfterViewInIt.
* So let's try it in ngOnInIt.
* Let me actually console log this header and there the native element.
* And now let's access something which is definitely available on a div, and that is the text content.
* If we do this, and I save this, let's view the console.
* If I reload the application here, you see this empty line here this is where I would've expected the text content.
* Can make this even clearer by outputting text content and then while rendering the text content, if there were any.
* There is no text content in here, as you can see an empty space.
* If I copy the exact same code and placed as an AfterViewInIt here.
* So after the view should have initialized.
* Now you will see that here when it's called the second time AfterViewInIt, we see Testserver.
* So that is the difference between, well the points of time where these hooks run.
* AfterViewInIt gives you access to the template elements.
* You can then access them and use their values and so on.
* Before this hook has been reached, you can't do that.
* You can't check the value of some element in your DOM because it hasn't been rendered yet.