[00:00:01.11] - Speaker 1
All right, this lesson allows me to actually demonstrate to you today how I built a prompt with using AI to generate a feature. So in this example, I'm going to show you a prompt that led to— this is the initial prompt that led to the development of an institutional hierarchy generator that I have available.

[00:00:32.17] - Speaker 1
On blackboard.tools/ih-generator. So I asked my Cursor environment, my Cursor AI agent to create a Python script that number one, accepts a DOC or DOCX file from the user. Number two, parses an outline structure. So it uses the outline structure that is native to Microsoft and showing a top level equals parent, subitems equals children. Then I said split each line on the pipe value and remove surrounding whitespace values.

[00:01:18.07] - Speaker 1
Then classify the line as node name, pipe, node ID, and then an optional option of a node description. Then I said build a hierarchy and write, write to hierarchy node generator in the format node name or node ID equals external node key, node name equals name, description equals node description, and parent node key is the parent or the a level above the current spot. Uh, it'll make sense when I show you the video, but basically this is creating an SIS, Institutional Hierarchy Feed, file that you can process in your Blackboard environment, and it should process every single line. And I included a sample doc of the test file of what the input, and then I gave it an output, a sample of output. So I gave it everything I could think of to make it, allow it to make that decision.

[00:02:29.27] - Speaker 1
And so when we get past this point, we get to this area here that's Blackboard Tools Institutional Hierarchy Generator. So let me just click on this real quick. And so what it is, is a— you can have download a sample DOCX or DOC file. And I think even if I flip back over here, you can drag and drop it here and then tell the system to generate the hierarchy. So let's show you what that looks like.

[00:03:13.13] - Speaker 1
So here you can see, and I'm going to pause this real quick. This is a Microsoft outline. And you can see here, University of St. Thomas, University Parent node, Academic Units, and then I have an, a node ID. Then underneath the Academic Units are various different schools. Um, then under that school are various departments.

[00:03:39.26] - Speaker 1
So this is— so if you used, pulled into Microsoft Word and, and built this out, this is how, uh, if you created it this way, this is the input that we would take. Let me take a sip of— so What happens then is— and there you can see I'm highlighting just to call attention to that. I'm going to go— whoop, I can't go full screen here, that doesn't help. Um, so here you can see everything going on here.

[00:04:25.28] - Speaker 1
So now what's going to happen is we've built the structure. I'm going to go and add this file. So I'm actually going to upload this file into my hierarchy generator, and then it's going to generate that hierarchy. Then I'm going to download the generated file. Now let's take a look at that generated file.

[00:04:43.24] - Speaker 1
We're going to open it so we can take a look here. And here we go. So you can see the system has automatically generated the external node keys, the name, and the description. And then you can see where I needed to, for example, in— and let me just pause this and scroll over.

[00:05:13.14] - Speaker 1
Here you can see, for example, Communication is under the School of BSG, which is, I think, um, Business Sciences and Government. I don't know, Biology Sciences and Government, or Geology. I'm sorry, can't remember. But anyway, so Communication and History are under that, and that's their parent node key. And so that parent node key is is coming from the node in the breakdown there.

[00:05:49.24] - Speaker 1

And so it actually does that breakdown. The reason why we did— why I did this is this is one of the— I've had so many different people who have said this is like the biggest problem for them when they're tackling institutional hierarchy nodes is just trying to figure out how to do it. And if I could— if you could tell them to just go and build an outline in Microsoft Word and then be able to process it, it makes things so much easier. So this is how, uh, and what I recommended. And so we use this and we've used it successfully.

[00:06:30.16] - Speaker 1
So when you're tackling a situation like this where you're writing scripting or things like that, you're also tackling automation and you're looking at evaluation, evaluating how you need to do things. And so I wanted to share with you my decision framework on how to decide what to automate, what things you might want to look at from an AI perspective. Your mind, hopefully so far, is going and looking at and thinking, all of these different things that you might be able to go do when you go home and back to your institution. And so now you need to kind of be able to say, well, what is best? What would be best for me?

[00:07:21.22] - Speaker 1
So, and the— so I created this matrix based off of that book that I shared with you that Kevin recommended. Last year. So let's go from on these 4 different quadrants here. So high frequency and simple tasks like enrollment syncs, availability toggles, account enable and disables. Those are easy calls and they can really be automated rather simply.

[00:07:49.26] - Speaker 1
And they are, don't really have to, you know, you don't really have to think about, you know, all the different aspects of it. They're normally pretty much, you know, cut and dry type of situations. So if you're going to tackle it, or if you were going to tackle something, I'd recommend maybe looking at those first.

[00:08:10.00] - Speaker 1
Then you may have more complex issues, maybe like high-frequency and more complex processes, rolling over things between terms with copies, bulk imports, maybe permission updates, things that might need to be automated, but AI is going to do a very good job of managing the automation and managing to deal with the complexities of that instead of you just having to manually do that yourself. Then you've got your low frequency and simple processes that you may just want to generate like single restores.

[00:09:00.01] - Speaker 1
Those kind of things. They don't have to be automated, but scripting could be useful.

[00:09:07.22] - Speaker 1
This is kind of talking about data patching, but that's not really something that it would do there for us. But then you've got LTI reconfigurations, changes to or testing of workflows, and those kind of things are low frequency and may be complex. So these are various different ways that you can use AI. Maybe not in Blackboard— some of this may not work in Blackboard, but it's the way that you can utilize it in your day-to-day work as a Blackboard administrator.

[00:09:42.05] - Speaker 1
And so, um, let me go ahead and briefly touch on course copy automations, and then we'll switch to our, uh, what I believe will be the last, uh, lesson for this series. So I mentioned course copy automation was one of those things that I wrote a couple of, as one of my steps to AI coding. And so I was able to build a tool and create the ability to copy content between two courses. Now, most of us know that you can copy content and pull content into Ultra courses now, but you have to be enrolled in those courses or you have to have like a system administrator access. So this script will take on those copy jobs and allow you to, or allow the user to be able to make those copies happen without having to have the administrative access.

[00:10:43.02] - Speaker 1

So it's got an interactive mode where you can type in the source course and the destination course and just make the copy happen. You can provide a source and destination when you run the command line prompt. Or you can actually use a CSV file and have the system batch process the course copies and allow for a more granular copying process. So that's really the biggest benefit for that and really has empowered a lot of our instructional designers who support faculty and programs where they have to do these course copies. So here's an example of me running the course copy process just from a command line prompt.

[00:11:34.16] - Speaker 1
And you can see if you just run the command, it gives you an output of what the, what it's looking for, the various ways you're looking for. So here I'm running it in verbose mode, course destination copy. It goes and checks the courses and validates the instructors. And says, you know, is this the right course you want to copy? Then you say yes, and it'll go ahead and submit the copy request.

[00:12:00.21] - Speaker 1
It will continue to pull back and check the status, and once it gets a statement that the copy is completed, it will do that. After about 30 to 60 seconds, if the copy is not done, it'll just give you the code and say, hey, go check the logs.

[00:12:20.24] - Speaker 1
So again, here's providing the source and destination courses in the feed file or in the command prompt. Same process, pretty much. And so this is how that works. Relatively simple, uh, but a nice course copy automation script that, uh, alleviates and, and provides some simple support, uh, for instructional designers to be able to make course copies relatively available. So let's touch on our final lesson next.