Tasks:

- Develop a CoverNode that possesses a Smart Object, CoverStep SmartObject, that if an Agent were to occupy the space of the CoverNode, or is directly on top of it. The Agent will try to get a SmartObject from the Node, if it cannot, then the Agent just occupies the CoverNode. Otherwise, the Agent will perform the associated Interaction from the retrieved SmartObject.
- The SmartObject's Interaction has a string that is the name of some Animate State that is relevant to Agent types, including the eligible Agent occupying the CoverNode. THe Agent will retrieve the string to pass to its Animator controller. The Animator will then perform the appropriate Animation on the Agent.
- In the case for CoverStep, the string of the Animate State is retrieved, as well as the CoverStep parameters: the direction the Agent is to step; and the direction the Agent is to face.

Research:

- Revisiting Animation, since creating the Animation State in the FSM.

Learning Unity's Animation System

- Episode 1 Animator Fundamentals:
  Note: The Avatar and Sprite is a child of the Parent GameObject found in the hierarchy.
    - Unity's Built-in Animator Component:
        - Add this component to the GameObject Agent
        - There are 5 properties of this component:
            - **Controller**: Used to assign Animator Controller from Assets. Create an Animations Folder, and create the AnimatorController and add it to this field. Double clicking the custom AnimatorController, Unity will open the Animator window which consists of Two panes: The **Grid Pane** and the **Layers Pane**.
                - GridPane serves as a visual interface for developers to create, modify, and connect animation states.
                - An **Animation State** is a representation of an animation
            - **Avatar**:
            - **Apply Root Motion**: Want characters movement to be animation-based instead of script-based
            - **Update Mode**:
                - **Normal**: Alter the character's animation playback speed based on the current timescale. For example, if you cut the timescale in half, the speed animation will play half the speed, making it easy to create slow motion effects.
                - Animate Physics: Internally shifts the animator logic to the Fixed Update call, as Fixed Update in Unity is used for all things related to physics. So if the Agent or animation is

intended to interact with rigid body objects, this is likely what will be used.
        - Unscaled Time: The animator and the animations run independently from time scale. This is recommended to be used with UI animations, so even if the Game is running in a slow motion state, the User interface is still clean and snappy.
    - **Culling Mode**:
        - Culling is a performance saving feature builtin to Unity which only renders game objects that are in view of the camera.
        - **Always Animate**: Always perform the animate and the animation calculations, even when the model is out of view of the camera.
        - **Cull Update Transforms**: Unity continues to calculate the frames in the animation, so that when the character comes back into view, it's as if the animation never stopped
        - **Cull Completely**: Completely stops the animation when the character is removed from the camera view. When returning to the camera view, the animation resumes playing from where it was before being disabled.
  - Unity's Animator Controller
- Episode 2 Animator Controller (Transitions and Booleans)
  - Objective: Take a deeper look at the parameters and layers found in the animation controller and learn how to use them to blend animations together.
  - Parameters Tab in the left Pane of the Animator COntroller allows control over the Animation states.
  - To add a parameter, press "+" at the top right and you'll be presented with four types of parameters that you can use to control the Animation State Machine
    - Float:
    - Int:
    - Bool:
    - Trigger: Similar to a Boolean.
  - Can connect Animation States with Transitions in the Grid Pane. Can use bool parameters, and add conditions that are dependent on either of the four listed parameter types made earlier.
  - Use code to govern what value the bool parameter holds.
- Episode 3