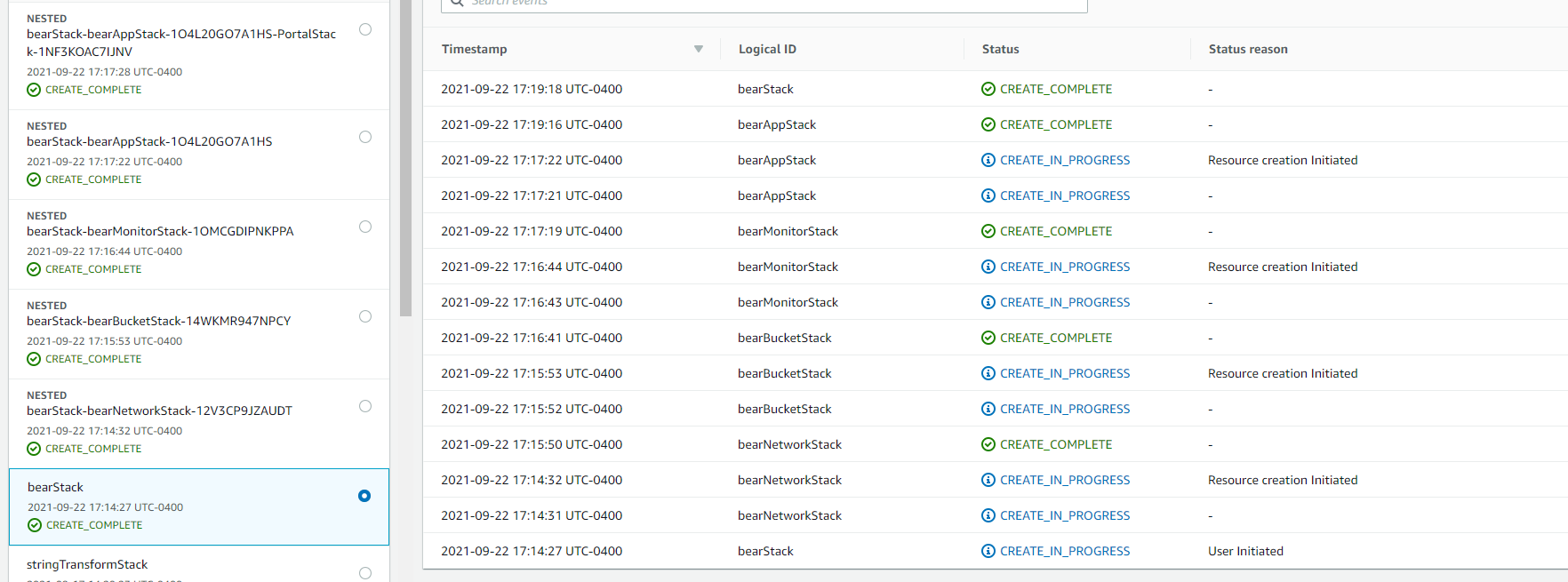
Condo Daily Report 9-22-21

Wednesday

* CloudFormation
  + Worked on dependency issue
    - Kept getting creation errors with the app stack
    - Took apart the app stack
    - Deployed the whole stack piece by piece
    - Individually with inputs the stack deployed correctly
    - This means it was an input issue in the bearstack
    - Found out the values I was passing were malformed and the import and export methods couldn’t understand them because they had ‘/’
    - Worked on finding the root of the problem
    - Worked through several different scenarios
    - Modified the input in the bear stack
    - Attempted to modify the input outside of the bearstack on a lower level
    - When I attempted to pass the stackId to the string macro it didn’t like it because it as an object
      * Couldn’t find a way to cast that ID to a string instead of a reference
      * Found somewhat of a workaround after I went to the drawing board by passing the reference as a parameter for the other stacks and having that parameter assert the string type
        + This is essentially the same thing as casting data types which cloudformation doesn’t allow
    - I then went in to each individual import and changed the string input parameter
      * This was easier said then done
        + The logic behind finding the right code structure was very hard to determine
        + It took me many tries and many builds to make this happen
        + The hardest part was, since we don’t have a ci/cd pipeline anytime I edited the code, I had to manually upload all the files again to the s3 bucket
        + So each time the code failed, which was often, I had to go back, fix the potential problem based off of a non specific error, save the file, upload the files to the s3 bucket which has versioning thank god, rebuild the stack which could take up to 5 minutes each build to wait for a success or another error code
    - Eventually I found the magic formula and tested it with the instance stack first
    - The instance stack accepted it and the values thrown were the same as I expected
    - I purposely made them fail so I could see desired output
      * This is one of the problems with cloudformation
      * I wish they had a form of unit testing, or build management or at least an IDE for the format of yaml so you can see your code die early
    - Once I saw them fail a specific way I adjusted the code to work for multiple use cases
    - This is the code snippet:
* !ImportValue  
   'Fn::Sub':  
   'Fn::Transform':  
   - Name: 'String'  
   Parameters:  
   InputString: !Sub  
   - '${bearNetworkStack}-PortalTableauSubnetAZ'  
   - {bearNetworkStack: !Ref networkStackName}  
   Operation: Replace  
   Old: "/"  
   New: ""
  + - Upon code review and multiple runs I got the bear stack to launch successfully with all the nested stacks
      * Final product had 1 main stack, 4 nested stacks, and 1 doubly nested stacks
    - 
* Sagemaker
  + Created sagemaker tableau connection using a aws cloudformation quickstart
    - Originally launched it with our original bearVPC
    - After I deleted this stack and remade one with a new vpc
  + Was able to connect the users
  + Logged into cognito
  + Verified myself to cognito
  + Ran into yarn package installation issues
  + Ran into a sagemaker endpoint issue, and ML models are necessary to make these endpoints