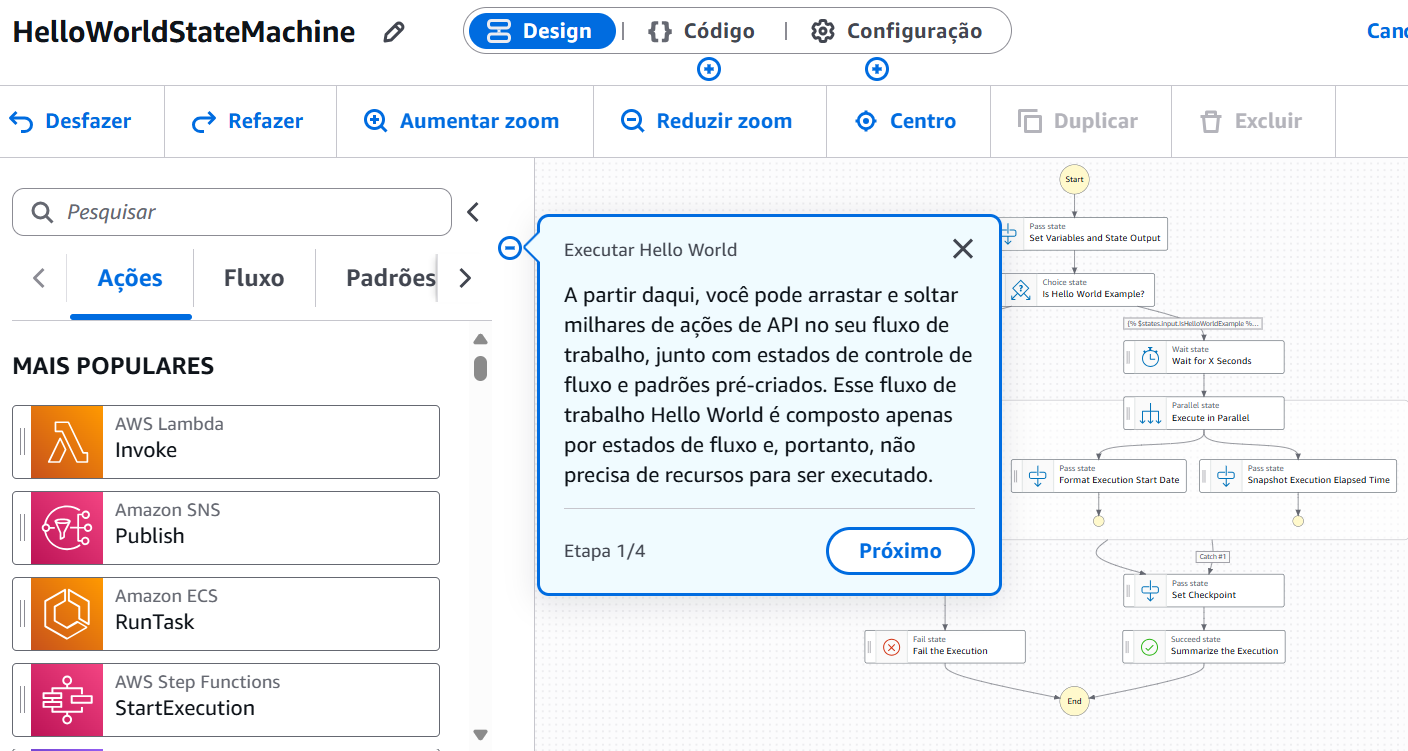
**Criando Hello Word State Machine**

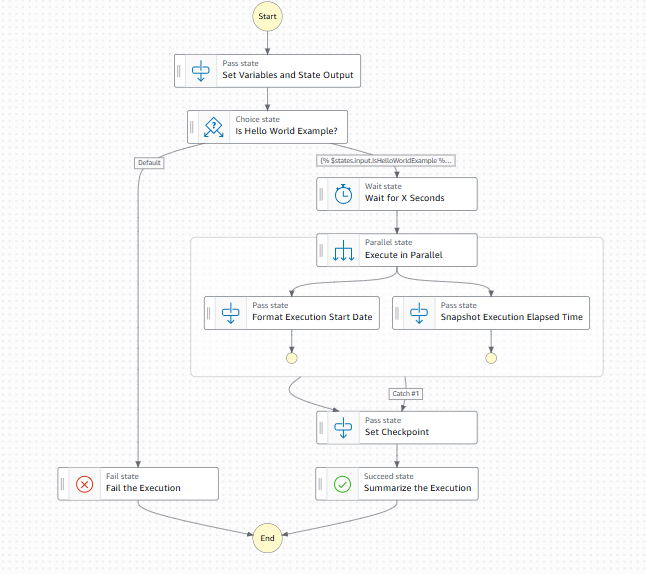


Interface gráfica do usuário, Texto, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.



##########################

#########Code#############

##########################

{

"Comment": "A Hello World example that demonstrates various state types in the Amazon States Language, and showcases data flow and transformations using variables and JSONata expressions. This example consists solely of flow control states, so no additional resources are needed to run it.",

"QueryLanguage": "JSONata",

"StartAt": "Set Variables and State Output",

"States": {

"Set Variables and State Output": {

"Type": "Pass",

"Comment": "A Pass state passes its input to its output, without performing work. They can also generate static JSON output, or transform JSON input using JSONata expressions, and pass the transformed data to the next state. Pass states are useful when constructing and debugging state machines.",

"Next": "Is Hello World Example?",

"Output": {

"IsHelloWorldExample": true,

"ExecutionWaitTimeInSeconds": 3

},

"Assign": {

"CheckpointCount": 0

}

},

"Is Hello World Example?": {

"Type": "Choice",

"Comment": "A Choice state adds branching logic to a state machine. Choice rules use the Condition property to evaluate expressions with custom JSONata logic, allowing for flexible branching.",

"Default": "Fail the Execution",

"Choices": [

{

"Next": "Wait for X Seconds",

"Condition": "{% $states.input.IsHelloWorldExample %}"

}

]

},

"Fail the Execution": {

"Type": "Fail",

"Comment": "A Fail state stops the execution of the state machine and marks it as a failure, unless it is caught by a Catch block.",

"Error": "Not a Hello World Example"

},

"Wait for X Seconds": {

"Type": "Wait",

"Comment": "A Wait state delays the state machine from continuing for a specified time.",

"Seconds": "{% $states.input.ExecutionWaitTimeInSeconds %}",

"Next": "Execute in Parallel",

"Assign": {

"CheckpointCount": "{% $CheckpointCount + 1 %}"

}

},

"Execute in Parallel": {

"Type": "Parallel",

"Comment": "A Parallel state can be used to create parallel branches of execution in your state machine.",

"Branches": [

{

"StartAt": "Format Execution Start Date",

"States": {

"Format Execution Start Date": {

"Type": "Pass",

"Output": {

"FormattedExecutionStartDate": "{% $fromMillis($toMillis($states.context.State.EnteredTime), '[M01]/[D01]') %}"

},

"End": true

}

}

},

{

"StartAt": "Snapshot Execution Elapsed Time",

"States": {

"Snapshot Execution Elapsed Time": {

"Type": "Pass",

"End": true,

"Output": {

"ElapsedTimeToSnapshot": "{% ($toMillis($now()) - $toMillis($states.context.Execution.StartTime)) / 1000 %}"

}

}

}

}

],

"Next": "Set Checkpoint",

"Catch": [

{

"ErrorEquals": [

"States.QueryEvaluationError"

],

"Next": "Set Checkpoint",

"Output": {

"ElapsedTimeToSnapshot": "Failed to calculate",

"FormattedExecutionStartDate": "Failed to format"

}

}

]

},

"Set Checkpoint": {

"Type": "Pass",

"Next": "Summarize the Execution",

"Assign": {

"CheckpointCount": "{% $CheckpointCount + 1 %}"

}

},

"Summarize the Execution": {

"Type": "Succeed",

"Comment": "A Succeed state indicates successful completion of the state machine.",

"Output": {

"Summary": "{% 'This Hello World execution began on ' & $states.input.FormattedExecutionStartDate & '. The state machine ran for ' & $states.input.ElapsedTimeToSnapshot & ' seconds before the snapshot was taken, passing through ' & $CheckpointCount & ' checkpoints, and has successfully completed.' %}"

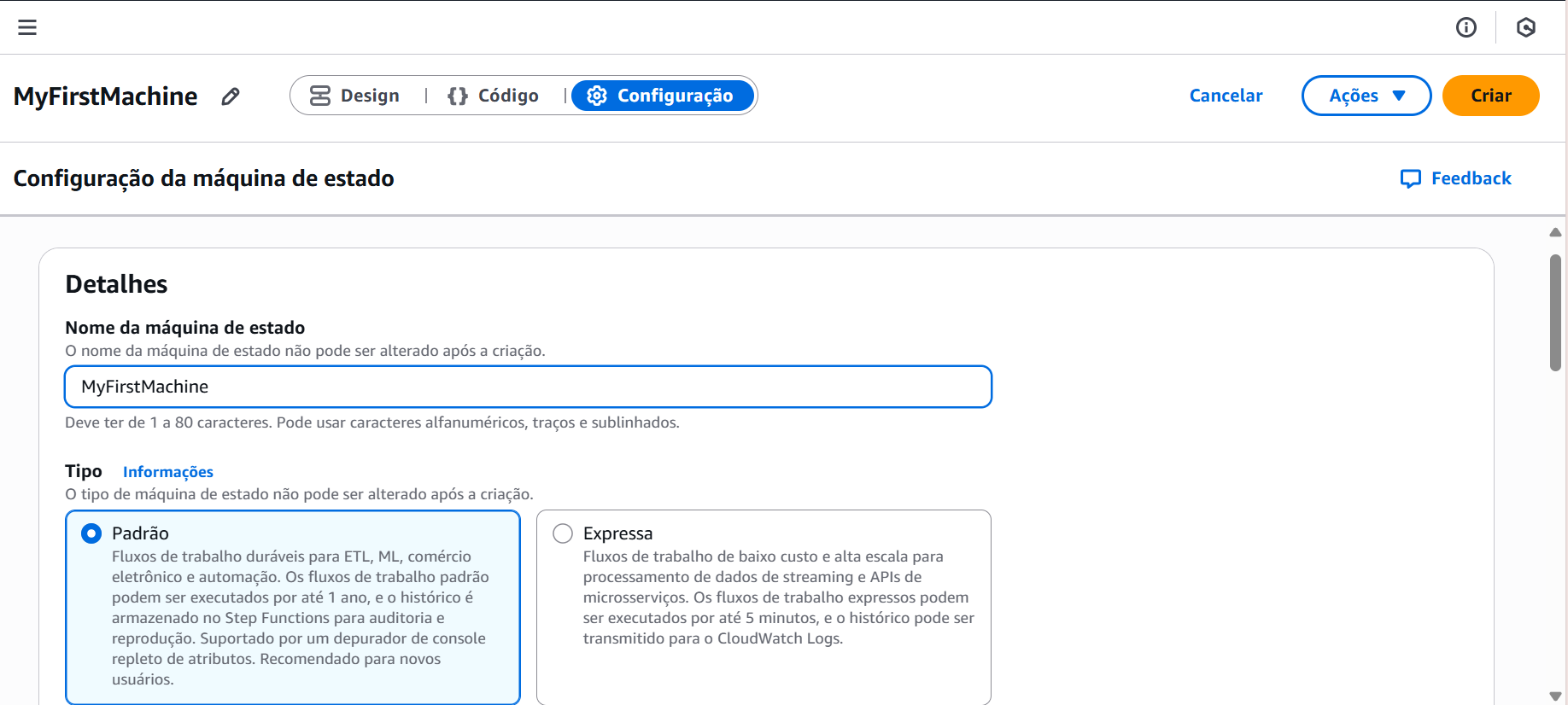
}

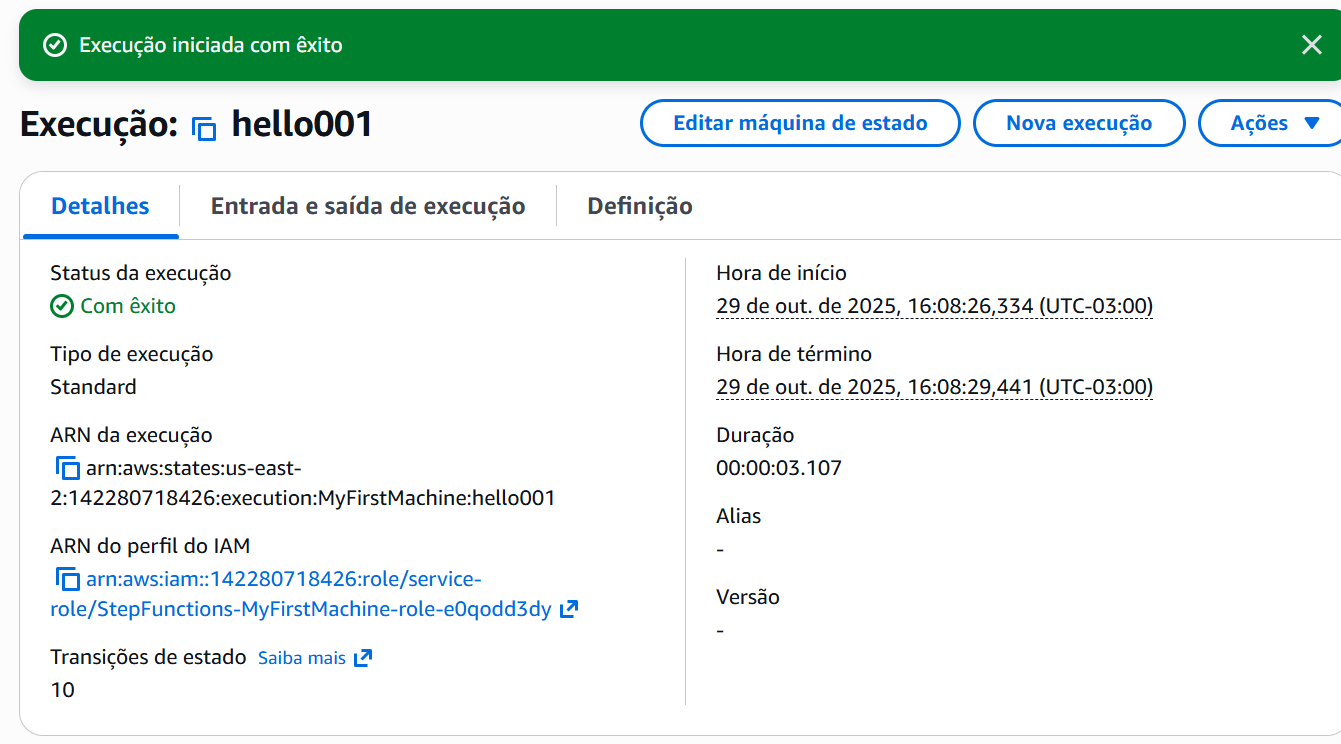
}

}

}

Criando a primeira máquina





Interface gráfica do usuário, Texto, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

