**Documentation**

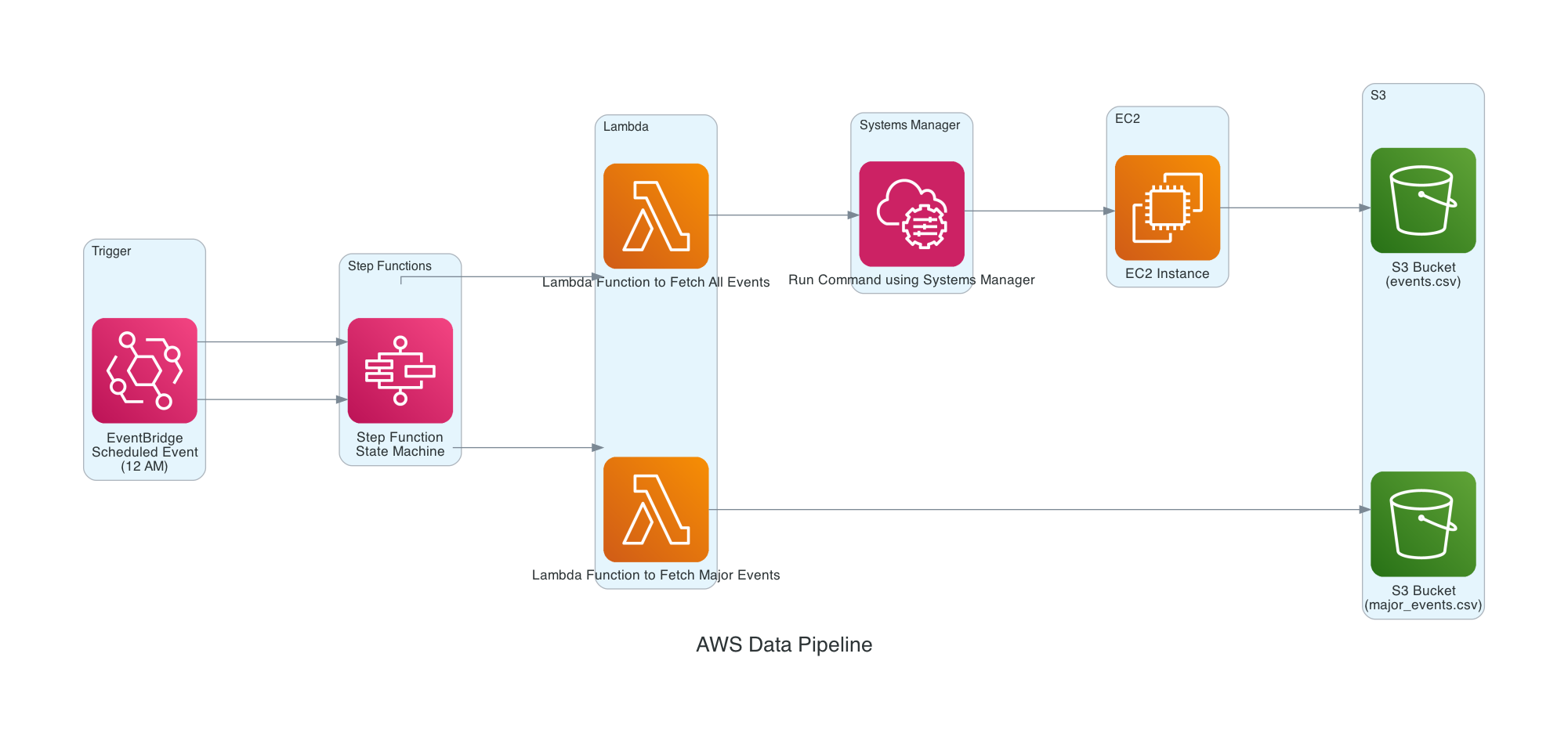
**Overview**

This data pipeline is designed to run a Python script on an EC2 instance and retrieve major events data from an S3 bucket every day at midnight. The Python script scrapes web data, processes it, and then stores the result in a CSV file in an S3 bucket. The Lambda function fetches major events data directly from another S3 bucket.

The pipeline leverages several AWS services:

1. **Amazon EventBridge**: Triggers the pipeline every day at midnight.
2. **AWS Step Functions**: Coordinates the steps of the pipeline.
3. **AWS Lambda**: Handles the state transitions of the EC2 instance, the execution of the Python script, and fetching major events data.
4. **Amazon EC2**: Runs the Python script.
5. **AWS Systems Manager**: Executes the Python script on the EC2 instance.
6. **Amazon S3**: Stores the resulting CSV files.

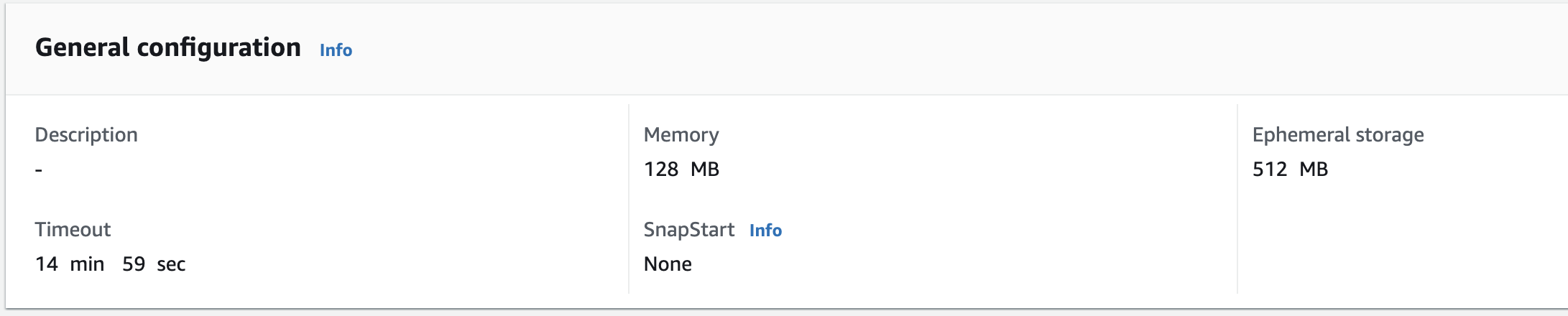
**Pipeline Flow**

****

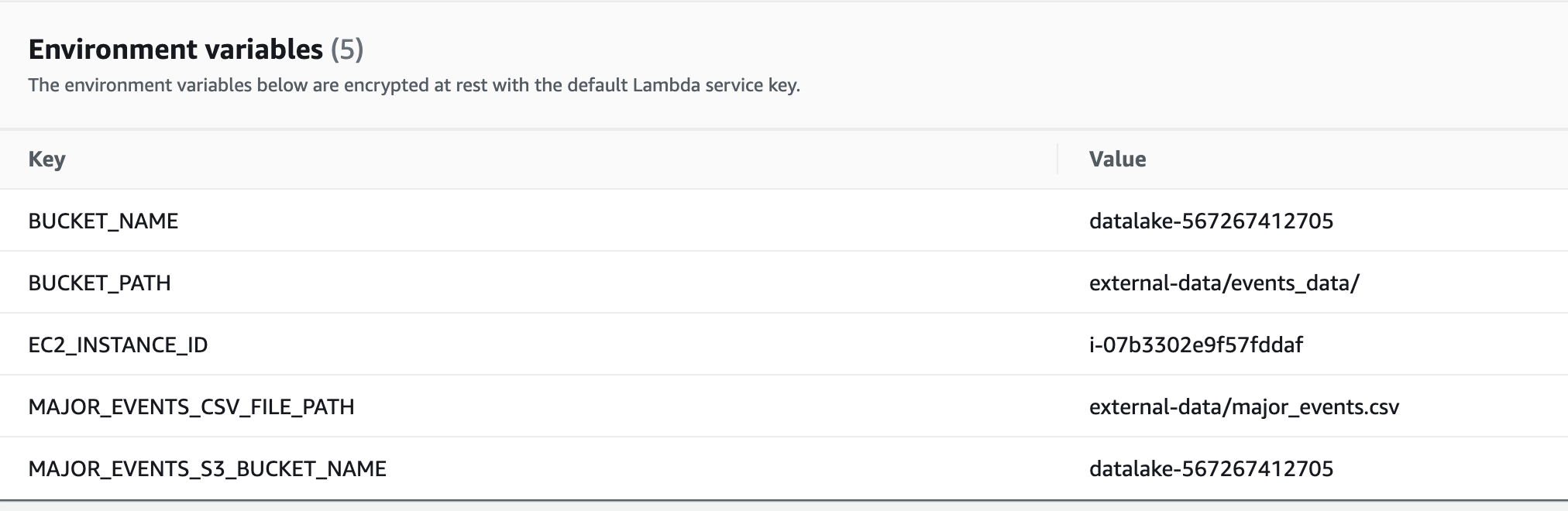
1. Amazon EventBridge triggers the pipeline every day at midnight by starting an execution of the Step Functions state machine.
2. The state machine triggers two Lambda functions.
3. The first Lambda function starts the EC2 instance, waits for it to enter the "running" state, uses AWS Systems Manager to execute the Python script on the EC2 instance, and then stops the EC2 instance after the script finishes executing and the resulting CSV file is stored in an S3 bucket.
4. The second Lambda function fetches major events data directly from the webpage and dumps it into s3 bucket.
5. The state machine execution completes.

**Steps to Recreate the Pipeline**

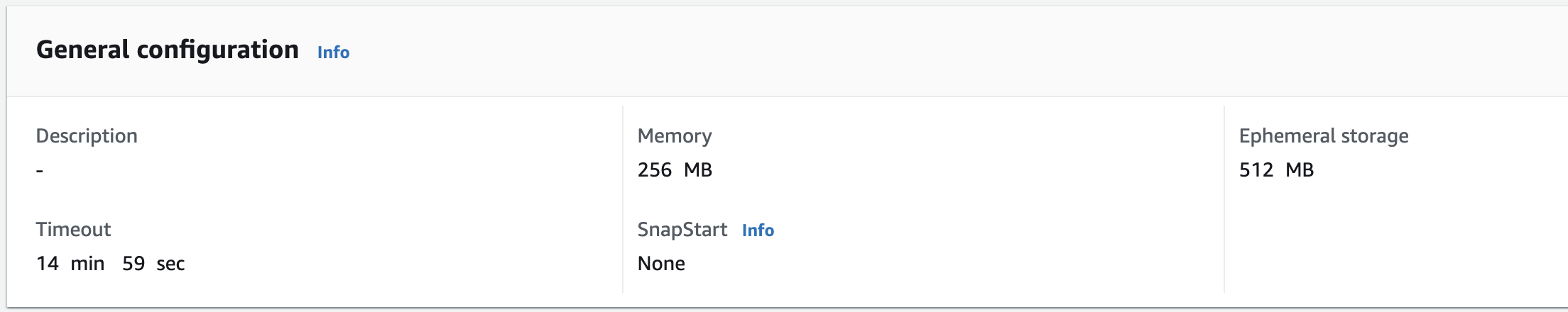
1. **Set up an EC2 instance**:
   * Log into the AWS Management Console and select 'EC2' from the 'Services' dropdown.
   * Click 'Launch Instance'.
   * Select an Amazon Machine Image (AMI).Choose the OS **Ubuntu**.
   * Choose an instance type. For this pipeline, a 't2.medium instance should be sufficient, but you may choose a larger one if you expect to process large amounts of data.
   * Configure the instance details. You can leave the default settings for most options, but ensure that the IAM role has the necessary permissions to execute the AWS Systems Manager (SSM) command and access the S3 bucket.
   * Add storage. The default 8GB should be enough for this pipeline, but you can add more if necessary.
   * Tag the instance. This can be useful for identifying the instance later.
   * Configure the security group. You can create a new security group and open the necessary ports. For the purpose of this pipeline, you do not need to open any additional ports.
   * Review the instance configuration and click 'Launch'.
   * Choose an existing key pair or create a new one, acknowledge that you have the private key, and then click 'Launch Instances'. The existing key pair event-**scraper.pem** file is in the code folder if you prefer to use it
2. **Install necessary Python libraries and AWS Systems Manager Agent on the EC2 instance**:
   * Connect to your EC2 instance using SSH. The command will be something like this **ssh -i "event-scraper.pem" ubuntu@ec2-35-93-4-186.us-west-2.compute.amazonaws.com**
   * Once connected, install the necessary Python libraries and firefox binaries by running all the commands present in the **commands.sh** file
3. **Upload your Python script to the EC2 instance**:
   * You can use SCP (Secure Copy Protocol) to upload the Python script from your local machine to the EC2 instance or use the native unix commands vi and touch to create the file and copy the contents of the local file to the EC2 Instance
4. **Create an S3 bucket**:
   * Go back to the AWS Management Console and select 'S3' from the 'Services' dropdown.
   * Click 'Create bucket'.
   * Provide a unique name for the bucket and select a region.
   * Leave the rest of the options as default and click 'Create bucket'.
5. **Create a Lambda function**:
   * From the AWS Management Console, select 'Lambda' from the 'Services' dropdown.
   * Click 'Create function'. Give the name **event\_fetcher**
   * Name the function, select 'Python 3.10' as the runtime, and choose or create a role with necessary permissions.(should have access to EC2, S3, SSM)



* + In the 'Function code' section, write the code for the function. This function should be able to start and stop the EC2 instance, and run a command using AWS Systems Manager. It should also be able to fetch major events data from the S3 bucket. The code is in the **event\_fetcher\_lambda.py** file

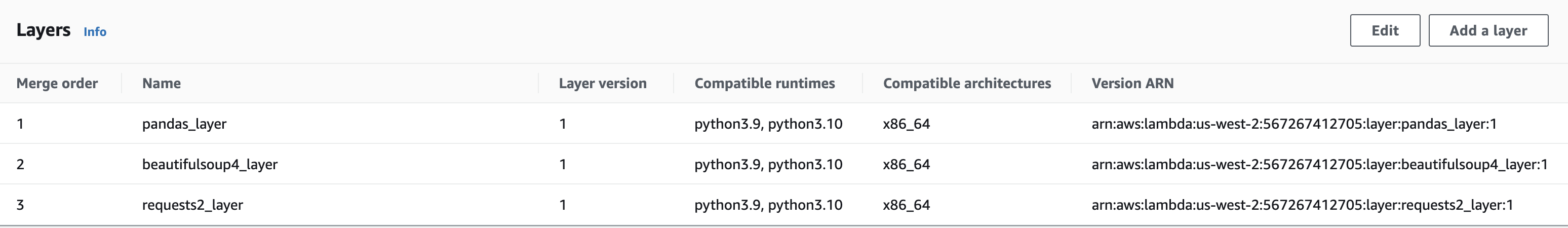


* + In the 'Environment variables' section, add any necessary environment variables. These are key-value pairs that you can dynamically pass to your function. Change the **EC2\_INSTANCE\_ID** environment variable to the actual ID of the EC2 instance created in the previous step
  + Create Another lambda function with the name **fetch\_major\_events** with the same python run time 3.10 and the same IAM role as earlier
  + In the code section copy the code of the file **fetch\_major\_events\_lambda.py**
  + Set the environment variables and configuration as below



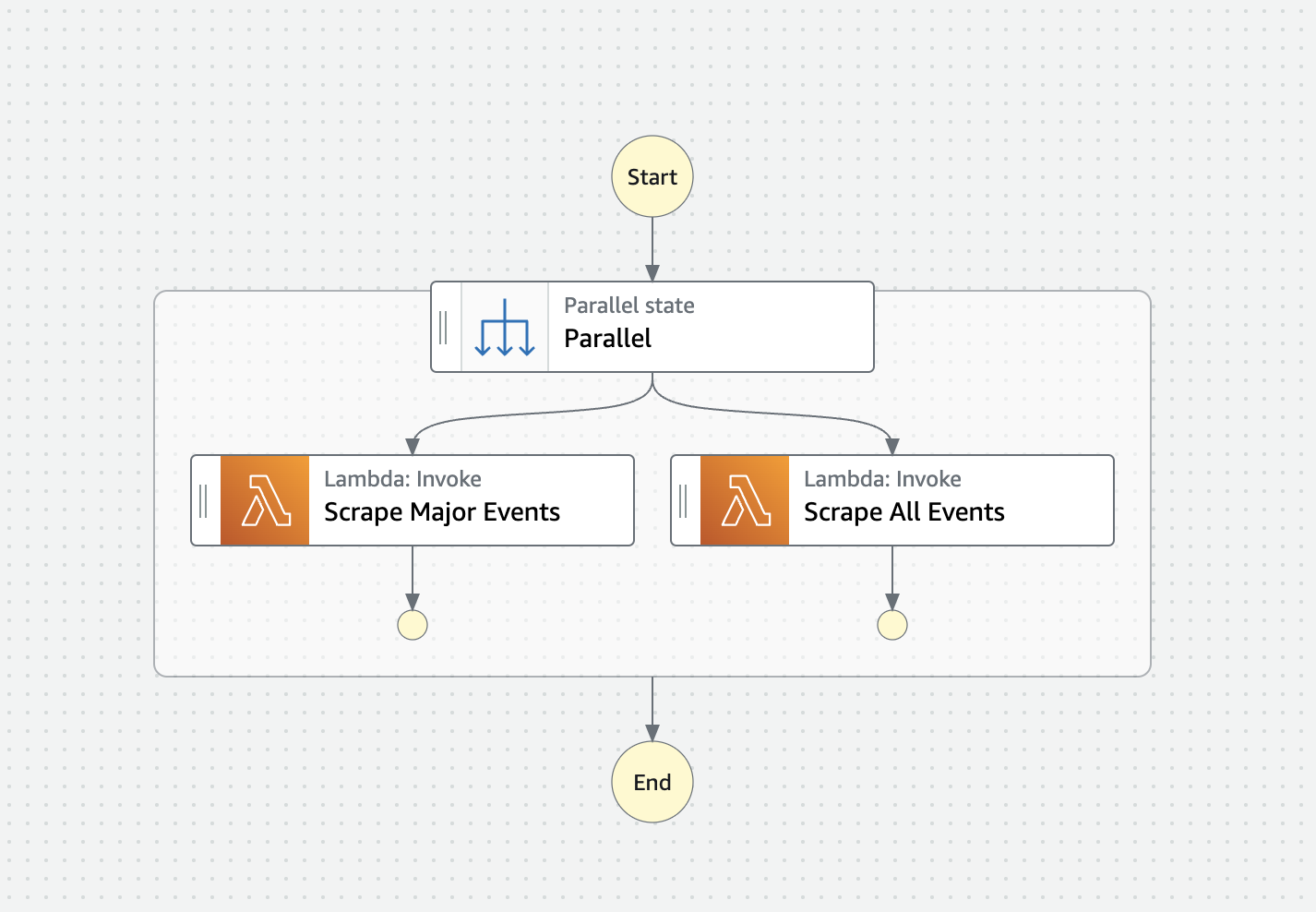


* + In the 'Layers' section, add any necessary layers. These are ZIP archives that contain libraries, a custom runtime, or other dependencies. You can create a layer, or use a layer published by AWS or other AWS customers. The Layers are 3 zip files which are in the folder supplied to you.



* + Click 'Deploy' to save the function.

1. **Create a Step Functions state machine**:
   * From the AWS Management Console, select 'Step Functions' from the 'Services' dropdown.
   * Click 'Create state machine'.
   * In the 'Type' section, select 'Standard'.
   * In the 'Define state machine' section, write the state machine definition in Amazon States Language. It should call both the lambdas created in the previous step in parallel



* + In the 'Choose or create an IAM role for executions' section, choose or create a role with necessary permissions.(Access to Lambda)
  + Review the state machine configuration and click 'Create state machine'.

1. **Create an EventBridge rule**:
   * From the AWS Management Console, select 'EventBridge' from the 'Services' dropdown.
   * Click ‘Schedules' from the left hand pane under the scheduler section
   * Click Create schedule and give all the necessary details
   * In the select Target section, Select the step function which was created in the previous step
   * In the 'Define pattern' section, select 'Schedule'.
   * In the 'Cron expression' field, enter **0 0 \* \* ? \*** to schedule the rule to run every day at midnight.
   * Save to create the rule which triggers periodically

**Key Considerations**

- Ensure the EC2 instance has the necessary IAM role and permissions to execute the SSM command and access the S3 bucket.

- Make sure the Lambda function, Step Functions state machine, and EventBridge rule are properly configured to handle errors and retries.

- Monitor the execution of your pipeline regularly to catch and address any issues. You can set up CloudWatch alarms to notify you if there are any failures.