**Lab Topics:**

1. Schedule / Tumbling / Event Grid Triggers
2. Dependency Triggers
3. Exponential Back-off Retries
4. Teams Webhook Notification
5. SLA Dashboard using Log Analytics + KQL

**Lab 1: Schedule, Tumbling, and Event Grid Triggers**

**Lab Objective:**

Implement all three trigger types in Azure Data Factory and observe how each type controls pipeline execution.

**Step A: Create a Schedule Trigger**

1. Go to **ADF Studio** > Author > **+** > **Pipeline**.
2. Create a simple pipeline named pl\_schedule\_trigger\_demo with a **Wait activity** of 1 minute.
3. **Publish** the pipeline.
4. Go to **Manage > Triggers** > **+ New**.
   * Name: trg\_schedule
   * Type: **Schedule**
   * Start Date: Select today
   * Recurrence: Every 1 Hour
5. Associate this trigger with the pipeline and publish.
6. Wait for the scheduled run or trigger manually for testing.

**Step B: Create a Tumbling Window Trigger**

1. Duplicate the pipeline and name it pl\_tumbling\_trigger\_demo.
2. Add a **Get Metadata** activity pointing to a blob folder.
3. Go to **Manage > Triggers** > **+ New**.
   * Name: trg\_tumbling\_window
   * Type: **Tumbling Window**
   * Window size: 15 minutes
   * Delay: 5 minutes (for late data)
4. Enable **Dependency**, and allow retries.
5. Link it to the pipeline.
6. Publish and observe window instances from Monitor pane.

**Step C: Event Grid Trigger on Blob Upload**

1. Go to **Azure Storage Account** > Events > **+ Event Subscription**.
2. Event Type: **Blob Created**
3. Endpoint Type: **Azure Data Factory Pipeline**
4. Choose pl\_event\_trigger\_demo pipeline.
5. Upload a file to the monitored container.
6. Verify pipeline is triggered automatically from Monitor tab.

**Lab 2: Dependency Trigger**

**Lab Objective:**

Run Pipeline B only after Pipeline A completes successfully.

**Steps:**

1. Create two pipelines:
   * pl\_A\_trigger (waits 30 sec)
   * pl\_B\_trigger (writes to blob using set variable and copy activity)
2. Publish both pipelines.
3. Go to **Manage > Triggers > + New**.
   * Type: **Tumbling Window**
   * Name: trg\_dependency
   * Window: 30 minutes
   * Configure dependency: Add pl\_A\_trigger as a dependency for pl\_B\_trigger
4. Ensure **Dependency conditions** are set to "Succeeded".
5. Trigger window manually to simulate dependency.

**Lab 3: Exponential Back-off Retry**

**Lab Objective:**

Simulate an unreliable REST endpoint and enable exponential retries.

**Steps:**

1. Create a pipeline pl\_retry\_demo.
2. Add a **Web activity** calling a mock REST endpoint that returns 500 error (use https://httpstat.us/500).
3. In Web activity settings:
   * Retry: 3
   * Retry Interval: 5 seconds
   * Exponential retry: Auto-enabled by increasing interval
4. Publish and run the pipeline.
5. Observe retry intervals in **Monitor > Activity runs**.

**Lab 4: Teams Webhook Notification**

**Lab Objective:**

Send pipeline failure alerts to a Microsoft Teams channel.

**Steps:**

1. In Microsoft Teams:
   * Go to a channel > Connectors > **Incoming Webhook**
   * Name it ADF Alerts and copy the URL.
2. In ADF, create a pipeline pl\_teams\_notify.
3. Add a **Fail activity** or force a failure using Divide by Zero in a Set Variable.
4. Add a **Web activity** after failure.
   * URL: [Teams Webhook URL]
   * Method: POST
   * Body:

{

"text": "ADF Pipeline Failed: `@pipeline().Pipeline` at `@utcNow()`"

}

1. Publish and trigger the pipeline.
2. Observe alert in Teams channel.

**Lab 5: SLA Dashboard with Log Analytics**

**Lab Objective:**

Build an SLA dashboard tracking pipeline completion within SLA times.

**Steps:**

1. Enable **Diagnostic Settings** for ADF:
   * Go to ADF > Diagnostic settings > Add diagnostic setting.
   * Choose **Send to Log Analytics**.
   * Select categories: PipelineRuns, ActivityRuns, TriggerRuns
   * Choose/create a Log Analytics Workspace.
2. Open **Log Analytics Workspace** > Logs.
3. Run this KQL query:

kusto

ADFPipelineRun

| where Status == "Succeeded"

| extend SLADeadline = datetime\_add("hour", 6, format\_datetime(Start, 'yyyy-MM-dd'))

| extend SLABreached = iff(End > SLADeadline, 1, 0)

| summarize Count = count(), Breaches = sum(SLABreached) by PipelineName

1. Pin it to a dashboard or export to Power BI for SLA visualizations.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |