AutoOps

Team Members
CSCE 5214.005
University of North Texas

Mounika Ponnam (11610822)
Sai Sarat Chandra Vytla (11588541)
Rahul Manikonda (11608670)

GitHub Link for the AutoOps Project :
sdaiproject/SDAI-Project-Final (github.com)

Video Link :

Increment-1 : SDAI-Video-Increment-1.mp4 - Google Drive

Increment-2 : SDA_Increment_2_video.mp4 - Google Drive

1. Project Introduction

The main aim of this project is to automate the provisioning of Azure cloud data platform, monitoring and detecting anomalies in the Azure cloud platform. In the increment-1 we have completed the provisioning of azure services feature. In this Increment-2 we are going to monitor the Azure platform by enabling the diagnostic settings for all the services. Whenever any abnormality is detected in the data platform, that particular log will be collected into some generic repository which makes it easy for us to track down all the details. Then sending emails to the corresponding stakeholders with all the error details. In addition to the error details, If we could send the corresponding steps to resolve the issue, It would be useful for the support team. In order to do that we have designed an AI/ML engine which takes error messages as input and gives us resolution steps. This makes stakeholders work much easier.

2. Background

Most of the organizations are moving towards AutoOps for automatic monitoring of Azure data platforms and resolving the issues that occur in data platforms. ServiceNow is using these AutoOps features for their platform. It logs all the tickets from the Azure data platform and provides resolutions by implementing AutoOps.

Learn How AIOps Delivers High Performance Business Services 24X7 (servicenow.com)

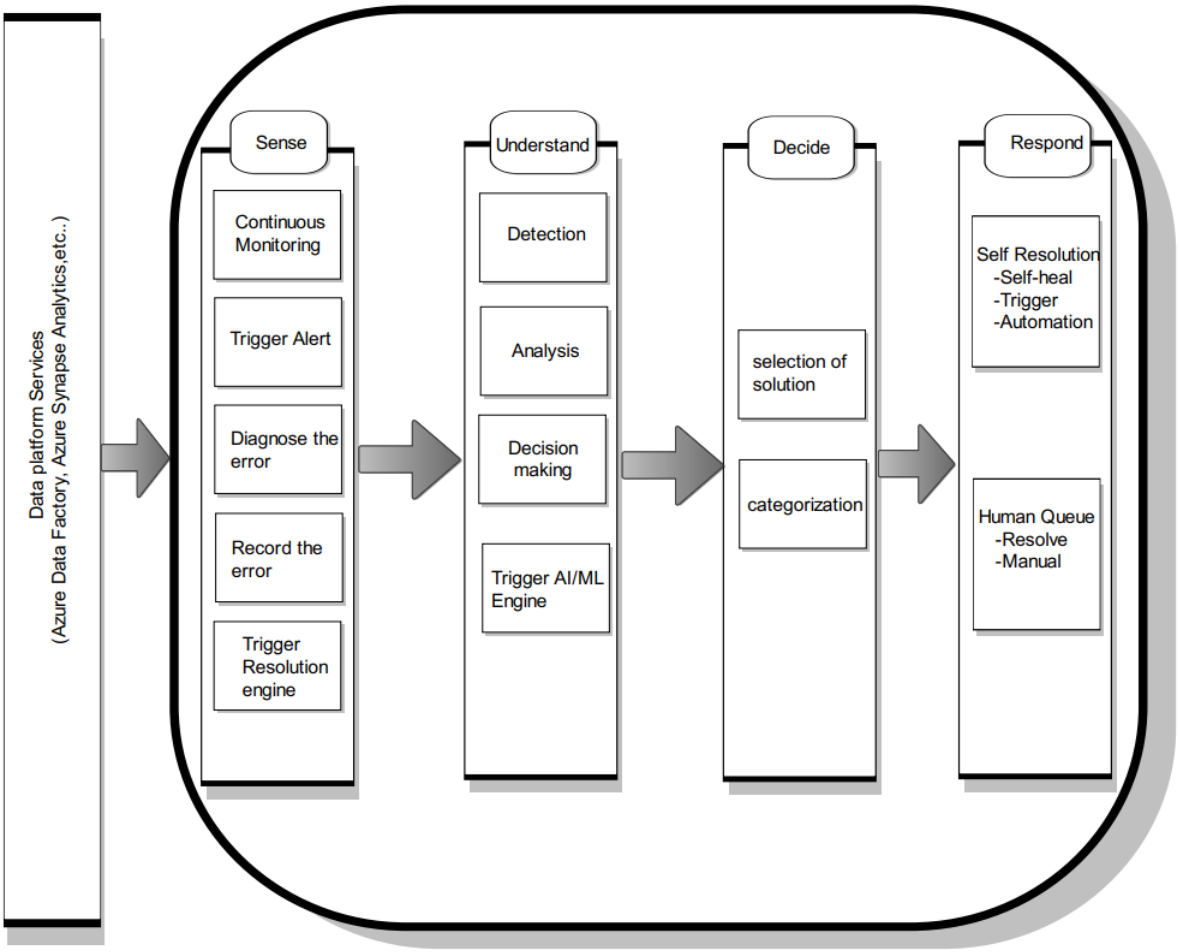
The blog below represents the multinational companies that are using this AutoOps for monitoring their data platform and how they are making use of all the features that were available in the AutoOps product. I developed my interest towards this project after going through all these details of the companies, how they are dealing with the AutoOps, and how it is helping the users with very minimal human intervention. Even though they did not provide much details on how to implement this product, I have done my own research and developed this product from scratch with the intention that it would be helpful for many users.

Who's Who in AIOps: The 10 Most Innovative AIOps Companies (moogsoft.com)

List of Top 8 AIOps Companies & Their Products (wisdomplexus.com)

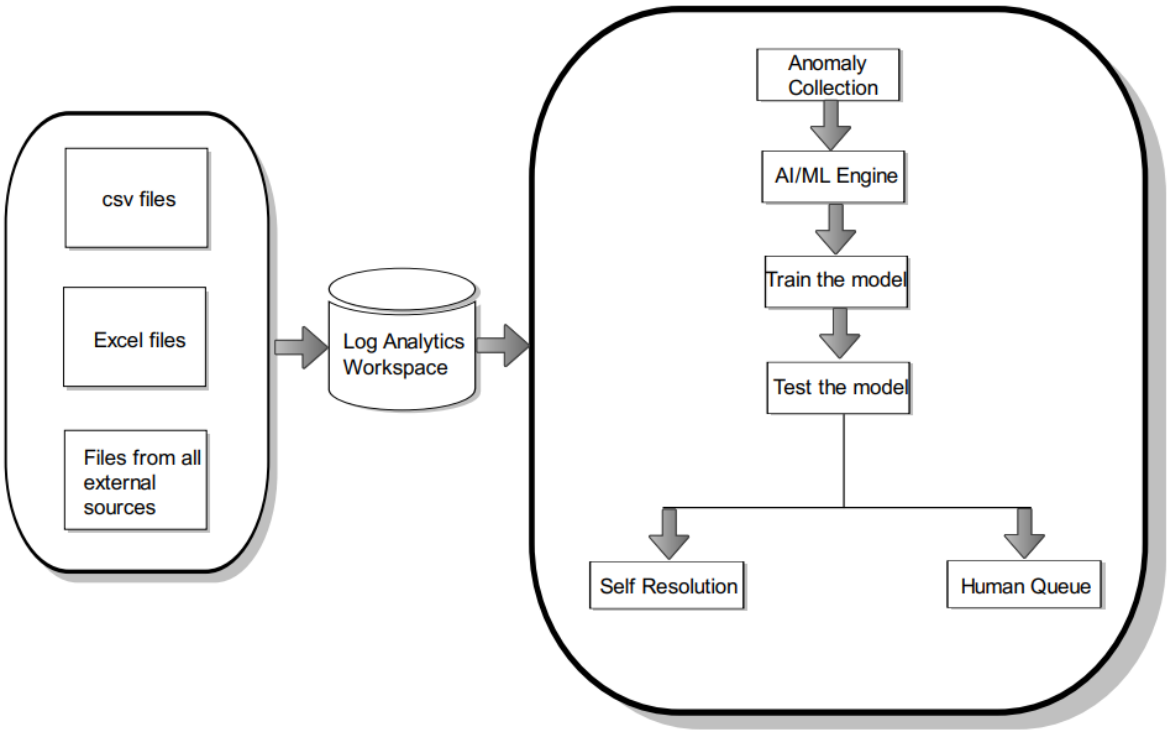
3. AutoOps Model

Architecture :



The full architecture of the AutoOps is depicted in the above diagram. First, we use terraform tools to automatically install Azure services such as webapps, function apps, data factory, storage account, synapse analytics, etc. into the Azure platform. All azure services operations are monitored by AutoOps. It will perform continuous monitoring and check the functionality of services like Azure synapse analytics, Azure Data Bricks, Data factory, etc. Any irregularity in the Azure data platform will cause a trigger, which will send all the error log information to the Azure log analytics workspace. Then we have designed a machine learning algorithm which takes all the error messages as input and sends us resolution steps as result. This makes the support members work much easier. They do not need to spend hours of time to search for the resolution of the issue that occurs in the azure data platform.

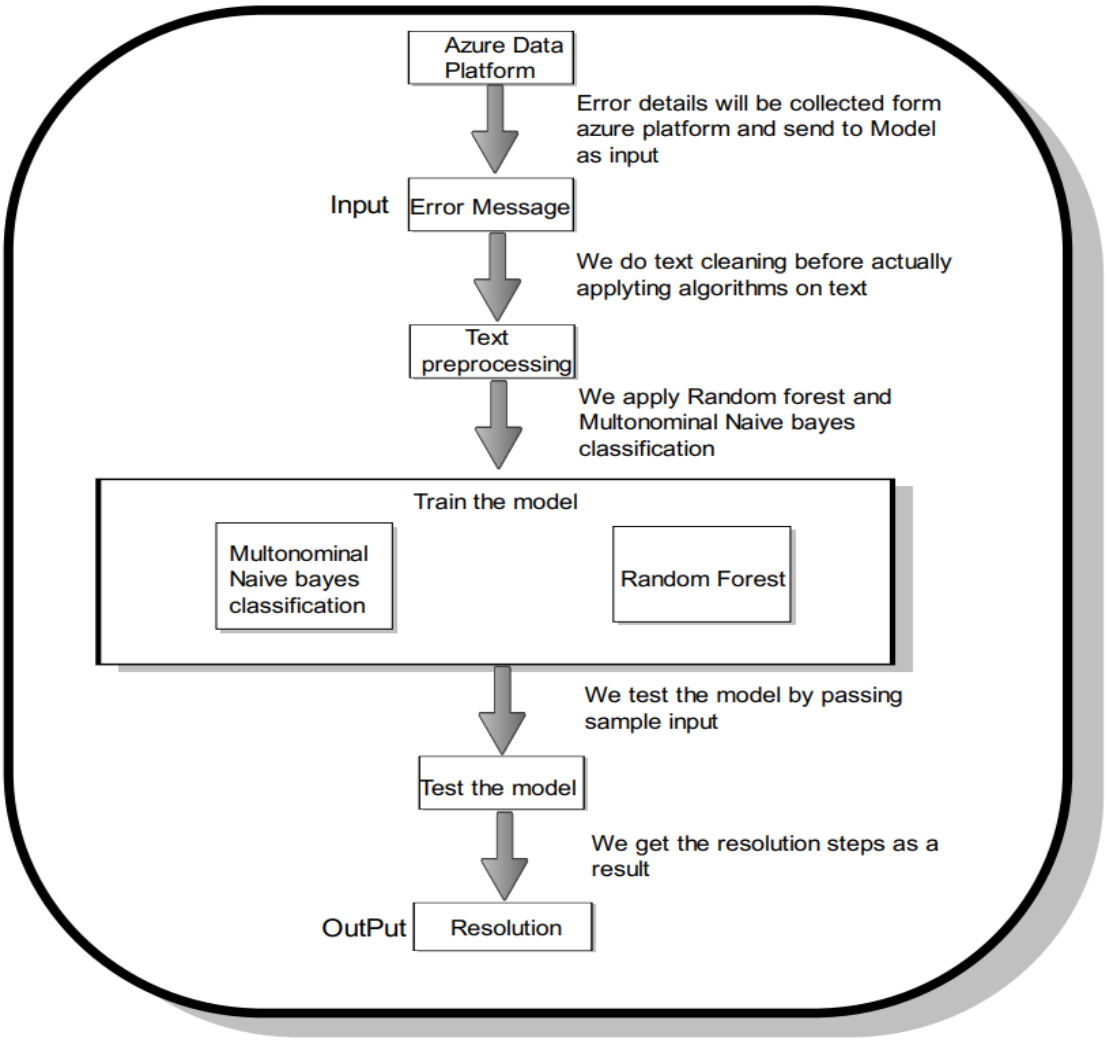
Flow Diagram:



The workflow for how AutoOps operates is shown in the diagram above. All of the error logs that are generated by the Azure Data Platform are initially gathered by AutoOps. The logs will be gathered in csv, excel, or any other format. The data will be delivered to some generic repository which is a log analytics workspace. This log analytics workspace is a service provided by Azure. Since it is an in-built service, it is very cost effective and we do not need to spend extra cost for storing all the error details. Log analytics workspace stores all the logs in a table format with all the column names. It makes it easy for users to read and understand the issue in the Azure data platform. We can avoid any third party application to collect all the logs because using another application costs us. We refer to it as automatic anomaly detection because it automatically identifies the error and gathers logs. After that, we communicate this abnormality to the Artificial Intelligence/Machine learning engine, which will provide us with the necessary resolution instructions to fix the problem. The resolution steps will be sent back if the AI/ML engine has a solution; otherwise, human troops will need to arrive and manually resolve the problem which is a tedious task for the support team. In order to overcome this problem, we have come up with the AutoOps solution which saves lots of time by automating several tasks in the Azure data platform.

4. Dataset

Two crucial columns in the dataset we used for our model are Error Message and Resolution. The precise anomalous information we obtain from the log analytics workspace after gathering it from the Azure data platform is contained in error messages. Resolutions are the actions taken to address problems that arise with the Azure data platform. Open - sourced websites like Kaggle and others have a ton of datasets available. I used the information from authorized Microsoft documents as the dataset for the model.
Dataset URL : [SDAI-Project-Final/Dataset.csv at main · sdaiproject/SDAI-Project-Final (github.com)]



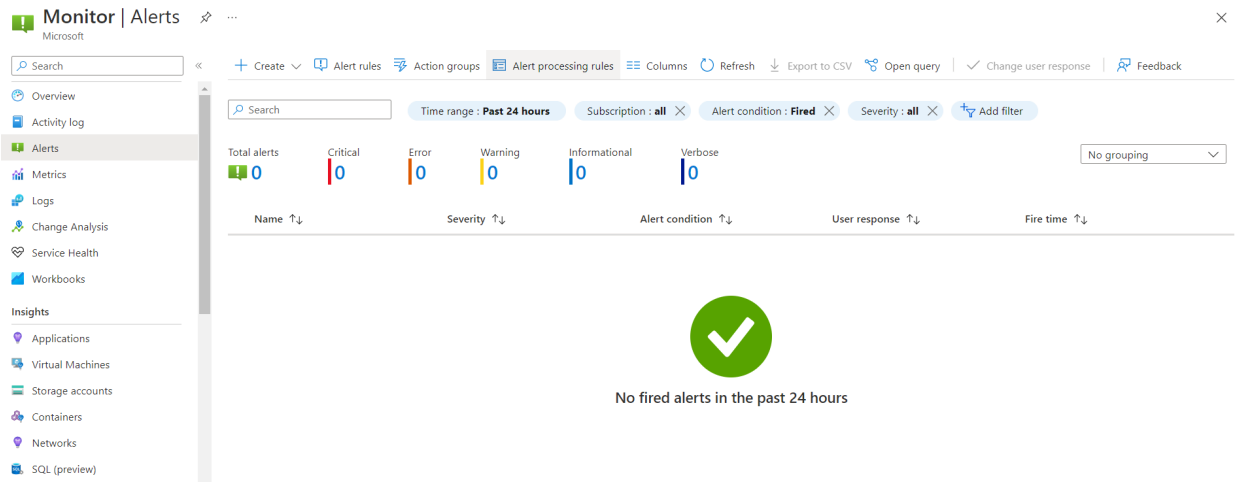
5. Implementation And Results

Monitoring Feature :

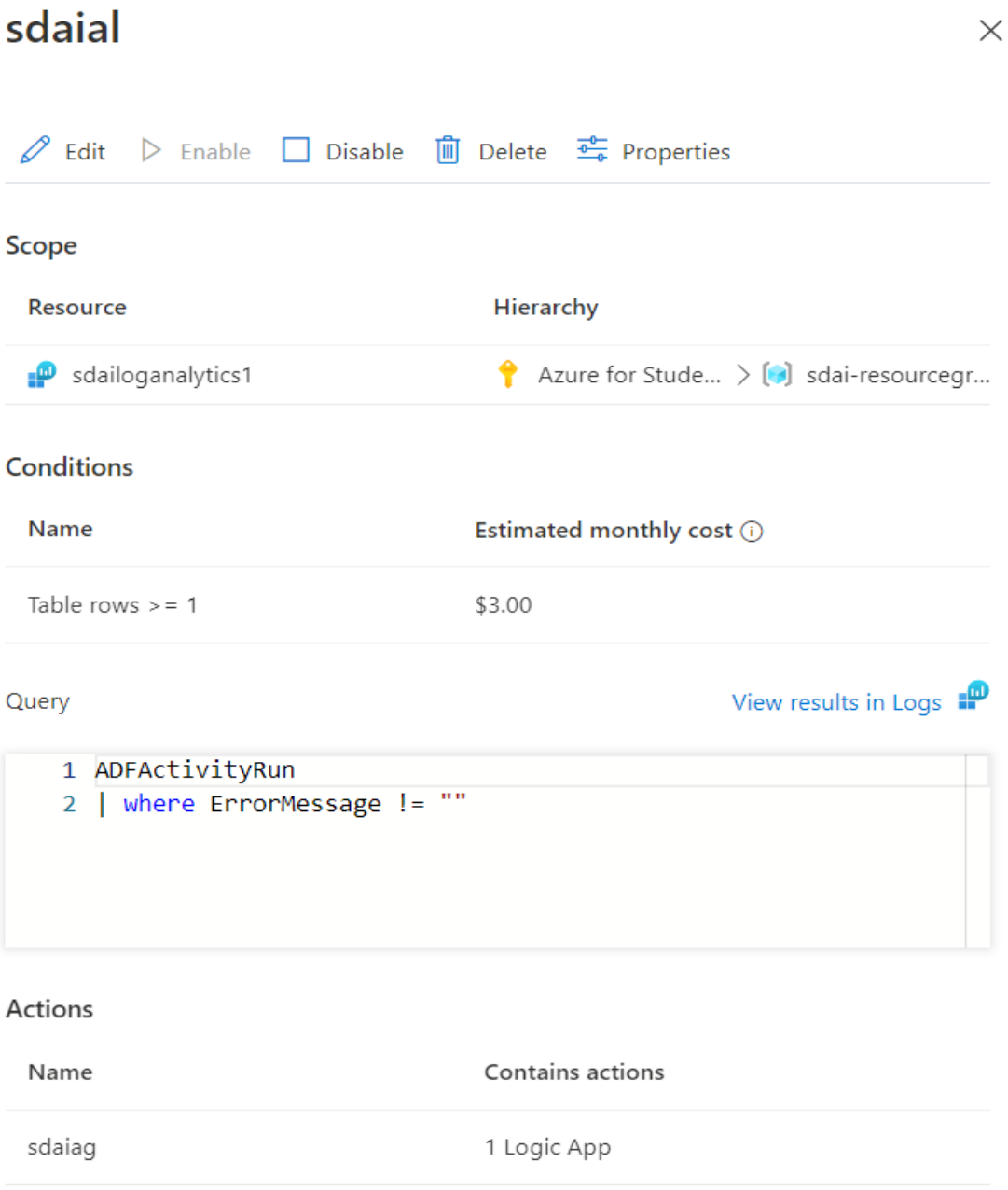
In this feature, we need to configure diagnostic settings for all the services in Azure, so that we can collect all the logs at a single place. The monitor is one of the services in Azure which is in-built. This is cost effective and easy to use. We can send log details to multiple resources, for now we have selected the destination store as only a log analytics workspace because it is easy to use, and we can filter out all the error details using the language called Kusto Query language.

Anomaly Collection Feature :

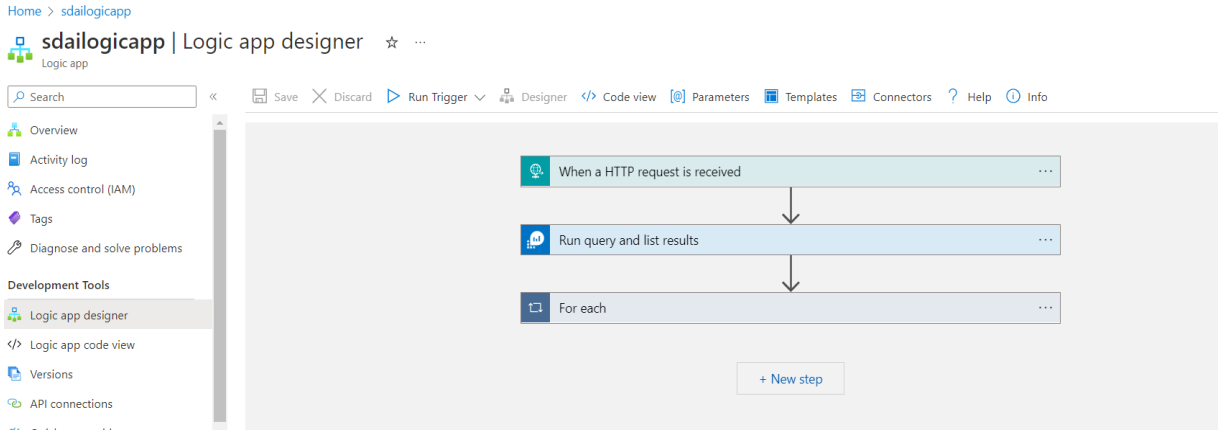
We then set up alerts to stakeholders and intimate the details of error and the root cause of the issue, from which service the error has occurred.

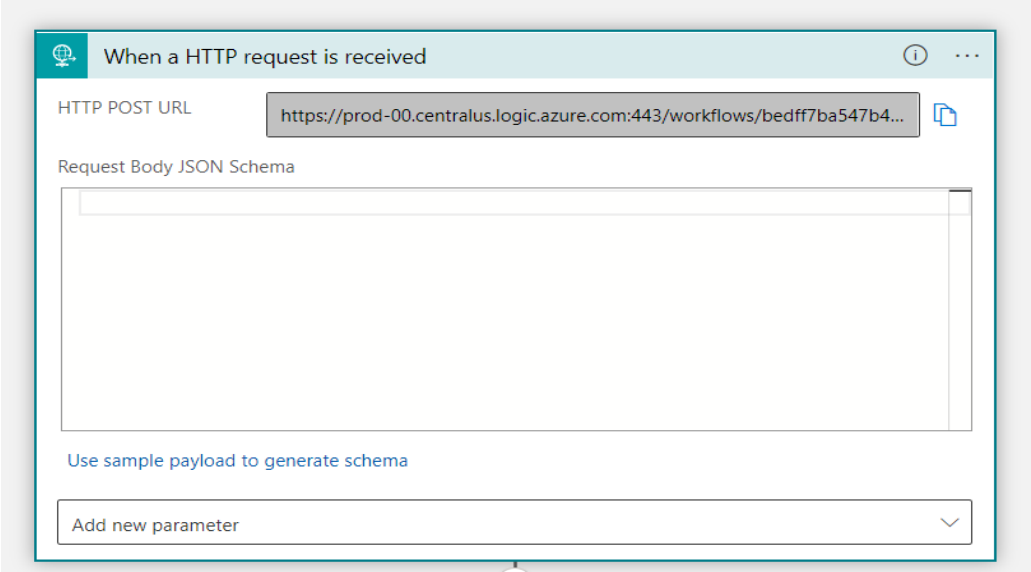


The alert contains three main parts. One is Scope which means the repository where we have stored the error logs. So, we have selected our scope as a Log analytics workspace. Then second we need to mention the condition, here we need to write a Kusto Query to filter out logs which contain error messages. If the error message is empty, that means, the pipeline is successfully run, but if the error message is not empty in the log analytics workspace, that means an error has occurred in the pipeline, and through a Kusto query we are fetching the details. Since we are collecting logs related to only the data factory, we have written ADF which means Azure Data Factory. The third component in alert is Logic app, it is the in-built azure service. We can write multiple activities inside a logic app and it executes them in a sequence. Logic is very useful to send the mails or any text messages to the stakeholders.

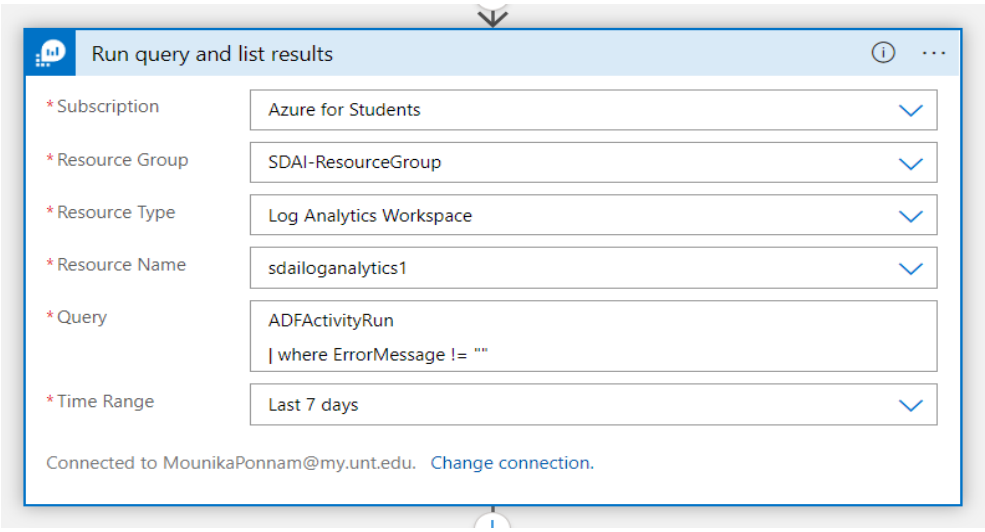


The above diagram shows the structure of the alerts, what all the components in it, and what all the things we need to select the particular component to make Azure alerts work. In the logic app we have written 3 activities in a sequence. This gets triggered whenever some error details are logged into log analytics workspace and then using monitor activity it fetches the log details dynamically from the log analytics workspace, then it sends to stakeholders or support team to let them know, some error has occurred in the data platform. It is much easier to execute multiple activities in a sequence.

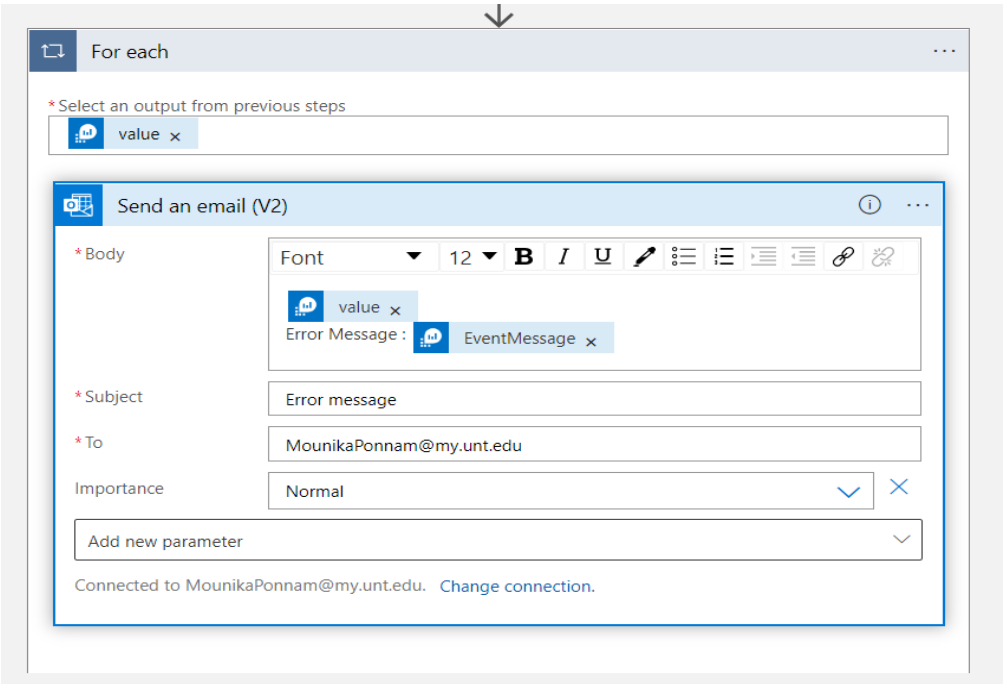




First activity in the Azure logic app is HTTP Request received. The alert connects this HTTP Request and the log analytics workspace, so as soon as an error occurs in the log analytics workspace, the logic app automatically starts to trigger as there is a connection established between them. The URL for this trigger automatically generates when we create this activity in the Azure logic app. The next activity is to add the monitor service in the Azure logic app service. There is an in-built connector in the Azure LogicApp which will automatically connect the Azure monitor and Log analytics workspace. But it will authenticate using multi factor authentication. We need to provide our tenant id of our azure account, then it will ask for our username and password of our Azure account. If those credentials are successfully authenticated it will allow us to use the in-built connector of the Azure logic app. If we want to create new connections or change existing connections, it will allow us because it is more flexible.



In this activity we need to select our subscription for the Azure account, which is "Azure for Student" in our case. Then we have to mention our Resource Group of the log analytics workspace. Then we need to select the Resource type, which is Log analytics workspace. There will be another option "Application Insights", but it will give only visual representation in the form of a graph, as we need to get a detailed error message, we have selected Log analytics workspace over the Application Insights option. Then we need to select the Resource name of the log analytics workspace. Then we need to write a Kusto query to collect if the Data factory pipeline fails. Here we have written a filter to sort logs based on ErrorMessage, which means if the error message column is not empty then we are fetching details of those details. We can even select the time range of how many days we need to get logs.



The final activity in the Logic app is to send the Error details to the support team or stakeholders. This can be done using the in-built connector of Outlook in Logic App. This connector can be created simply by having an outlook account. We just need to write our mail id corresponding to the “TO” field. Then inside the body, we can write past the dynamic content that comes from the previous log analytics output activity. We can send detailed error messages. We can modify the details as per need. Then we can write the subject of this mail, and we can even define the importance of this mail. These are the three activities that we defined in the logic app workflow. There are numerous connectors present in the logic app which makes our job easier to establish the connection between the Azure services without looking for any third party applications.

6. Project Management

6.1 Work completed:

- **Deployment :** The attractiveness of the AutoOps effort is that individuals no need to wait weeks or even months for the delivery of Azure data services. We can set up our platform with Terraform in a short amount of time without the need for human intervention. As a result, the project is rather distinctive.

- **Monitoring**: Every single service in Azure has configurable diagnostic options. In doing so, it will gather all service logs into a general repository that we refer to as an Azure log analytics workspace. This makes it simple for the developer to look at the specifics of the anomaly and allows them to concentrate more closely on the performance.

- **Anomaly collection** : For the AutoOps project, this also offers the advantage of eliminating the requirement for developers to use any third-party resources in order to save input data from the Azure data platform. The "Log analytics Workspace" which is a native Azure service can be used to collect all trace and metric data on the Azure data platform. This service's price is also reasonable.

- **AI/ML Engine** : We use all of the typical error messages and their accompanying resolutions to train the model. We will post our model into the Azure ML workspace once it has improved accuracy. This will provide us with the necessary resolution steps that will enable us to fix the problem with the Azure data platform.

6.2 MEMBER CONTRIBUTION TABLE:

Name	Student Id	Responsibility	Contribution
Mounika Ponnam	11610822	Terraform code for Azure services was written, features for deployment, monitoring, and anomaly detection were done, and drafted a document.	34
Sai Sarat Chandra Vytla	11588541	designing Terraform scripts, creating logic app processes to automate the full AutoOps process, prepared documents.	33
Rahul Manikonda	11608670	strengthening the machine learning model, deployment of codes aided in creating diagrams and documents.	33

7. References/Bibliography :

7.1 Azure services provisioning :

Deploying Azure resources using Terraform | Mashford's Musings (mashfords.com)

Deploying Azure Resources using Terraform | by Arun Kumar Singh | TechBull | Medium

How to deploy an Azure resource using Terraform when it is not available in the AzureRM official provider :: my tech ramblings — A blog for writing about my techie ramblings

How to Use the Terraform Azure Provider to Deploy Cloud Resources (petri.com)

7.2 Monitoring Platform :

Azure monitoring | Dynatrace

Monitor Azure resources with Azure Monitor - Azure Monitor | Microsoft Learn

Sources of data in Azure Monitor - Azure Monitor | Microsoft Lear

7.3 Anomaly Collection :

Learn How AIOps Delivers High Performance Business Services 24X7 (servicenow.com)

Who’s Who in AIOps: The 10 Most Innovative AIOps Companies (moogsoft.com)