Deccan Education Society’s

FERGUSSON COLLEGE (AUTONOMOUS), PUNE-4

##### Department of Computer Science

# A

# Project Report

# On

# CLOUD RESOURCE GARBAGE COLLECTOR

**By**

* + - 1. **Siddesh Khade Roll No:226341**
      2. **Prathamesh Shinde Roll No:226346**

**[2023 – 2024]**

Deccan Education Society’s

FERGUSSON COLLEGE (AUTONOMOUS), PUNE-4

##### Department of Computer Science

# A

# Project Report

# On

# CLOUD RESOURCE GARBAGE COLLECTOR

## In partial fulfillment of requirements of the completion of S.Y.M.Sc (C.A.) Semester-III

Master of Science

Computer Applications

### Submitted By:

* + - 1. **Siddesh Khade Roll No:226341**
      2. **Prathamesh Shinde Roll No:226346**

## Under the Guidance of

Dr. Kavita A. Khobragade

**[2023– 2024]**

(CSA5312) Computer Applications Project-III

CERTIFICATE

This is to certify that the project entitled \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Completed by

1.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

in partial fulfillment of the requirement of the completion of M.Sc.(C.A.) Semester-III, has been carried out by team under my guidance satisfactorily during the academic year 2022-2023.

**Place**: Pune

**Date**: / /2022

(Name of Guide) (Dr. Kavita A. Khobragade)

**Project Guide Head, Computer Science Department**

**Internal Examiner: External Examiner:**

## ACKNOWLEDGEMENT

I would like to express my heartfelt gratitude to Dr. Kavita A. Khobragade for her invaluable guidance, unwavering support, and expert supervision throughout the duration of this project. Her extensive knowledge and insightful feedback have played a pivotal role in shaping the Cloud Resource Garbage Collector.

This project marks an exploration into the realm of cloud computing, addressing the critical issue of inefficient resource management. The development of the Cloud Resource Garbage Collector, a user-friendly desktop application, would not have been possible without Dr. Khobragade's mentorship.

I extend my appreciation to her for fostering an environment of learning and innovation. Her dedication to academic excellence and commitment to her students have been a constant source of inspiration.

I would also like to acknowledge the contributions of my peers and colleagues who provided valuable insights and feedback during the development process.

This project is a testament to the collaborative effort and the academic excellence fostered under the guidance of Dr. Kavita A. Khobragade.

**1. Siddesh Khade Roll No:226341**

**2. Prathamesh Shinde Roll No:226346**

|  |  |  |  |
| --- | --- | --- | --- |
| Sr. No. |  | Topic | Page Number |
| 1. |  | Introduction |  |
|  | 1.1 | Existing System & Problem Definition | 6 |
|  | 1.2 | Need for the New system | 7 |
|  | 1.3 | Overview of the Project | 8 |
| 2. |  | Analysis |  |
|  | 2.1 | Feasibility Study | 9 |
|  | 2.2 | Hardware and Software Requirements | 12 |
| 3 |  | Design |  |
|  | 3.1 | UML Diagram  Use case diagram,  Class Diagram,  Sequence/Activity diagram,  Deployment Diagram | 13 |
|  | 3.2 | Input and Output Screens and Reports | 17 |
| 4. |  | Testing |  |
|  | 4.1 | Importance of Testing | 18 |
|  | 4.2 | Types of Testing | 18 |
|  | 4.3 | Test Cases (with Expected and Actual Result) | 19 |
| 5 |  | Reports | 26 |
| 6 |  | Drawbacks / Limitations | 27 |
| 7 |  | Conclusion | 28 |
| 8 |  | Future Enhancements | 29 |
| 9 |  | References and Bibliography | 30 |

**1.INTRODUCTION**

**1.1 Existing System & Problem Definition:**The existing system for managing cloud resources requires manual intervention and can be time-consuming for organizations that deploy and manage multiple resources. Without an automated system, resources may be left unused or underutilized, resulting in increased costs and reduced efficiency.

In the current system, managing Cloud resources involves navigating through the Azure portal to identify and delete unnecessary resources. This process can be slow and tedious, especially when dealing with a large number of resources. It also requires a significant amount of human resources to manage, which can lead to human errors and inconsistencies.

In addition, managing resources using the Azure or AWS portal may not provide an accurate representation of the current state of the resources. It may be difficult to track changes made to resources over time and identify which resources are no longer being used.

Overall, the existing system for managing cloud resources is inefficient and can lead to increased costs and reduced efficiency for organizations. It requires manual intervention and can be slow and tedious, leading to errors and inconsistencies.

The proposed system aims to address these issues by providing a Graphical User Interface (GUI) tool that automates the process of managing Cloud resources. The tool will use Cloud REST APIs to query and manage resources based on the user's input. Users will be able to specify the necessary arguments, such as the resource group, resource type, or resource name, to identify the resources to be deleted.

The tool will also provide the ability to perform **dry-run** executions, allowing users to preview the changes that will be made without actually deleting any resources. This will help ensure that the user can review the resources to be deleted and avoid any unintentional deletions.

Overall, the proposed system will provide a more efficient and automated way of managing cloud resources using Graphical User Interface (GUI), reducing costs and improving overall system efficiency.

**1.2 Need for the New System:**A system for cloud resource garbage collector project is necessary for several reasons:

**1. Cost Optimization:** Unused or underutilized resources in the cloud can increase costs unnecessarily. A system for cloud resource garbage collector project can help identify and remove such resources, thereby reducing costs.

**2. Security:** Unused resources can be potential security risks, as they may contain sensitive data or be vulnerable to attacks. A system for cloud resource garbage collector project can help remove these resources and minimize security risks.

**3. Compliance:** Compliance requirements often mandate the removal of resources that are no longer needed. A system for cloud resource garbage collector project can help ensure compliance with these requirements by automating the process of identifying and removing such resources.

**4. Resource Management**: As cloud environments grow and become more complex, it becomes increasingly difficult to manage resources manually. A system for cloud resource garbage collector project can help automate the process of managing and cleaning up resources, thereby improving resource management.

**5. Performance:** Unused or underutilized resources can impact the performance of other resources in the cloud. A system for cloud resource garbage collector project can help improve performance by identifying and removing such resources.

Overall, a system for cloud resource garbage collector project is essential for optimizing costs, improving security and compliance, managing resources effectively, and improving performance in the cloud.

**1.3 Overview of the Project:**The cloud resource garbage collector project aims to automate the process of managing cloud resources. The project will provide a Graphical User Interface (GUI) tool that can be used to identify and delete unused or unnecessary resources, reducing costs and improving the overall efficiency of the system.

The tool will be developed using Azure and AWS REST APIs to query and manage resources based on the user's input. The user can provide necessary command line arguments such as resource group, resource type, or resource name to identify the resources to be deleted. The tool will then execute the necessary API calls to delete the specified resources.

The project will have the following features:

1. Graphical User Interface: This GUI simplifies resource management by providing a visual platform for users to specify arguments, select resources, and execute actions effortlessly and user-friendly experience.
2. Resource Identification: The user can identify resources based on resource group, resource type, or resource name.
3. Dry-run Execution: The tool will include the ability to perform dry-run executions, allowing the user to preview the changes that will be made without actually deleting any resources. This will help ensure that the user can review the resources to be deleted and avoid any unintentional deletions.
4. Error Handling: The tool will include error handling to handle any exceptions that may occur during resource deletion.
5. Logging: The tool will include logging functionality to track the resources that have been deleted, including the resource type, name, and deletion time.

**2.ANALYSIS**

**2.1 Feasibility Study:**

2.1.1 Technical Feasibility:

The proposed project to automate Azure cloud resource garbage collector using command line arguments is technically feasible. Here are some of the reasons why:

1. Cloud REST APIs: Cloud provides a rich set of REST APIs that allow developers to programmatically interact with resources. These APIs can be used to query and manage resources, making it possible to automate resource cleanup tasks using command line arguments.
2. Python Language: The proposed project can be implemented using Python, which is a popular programming language for building command line tools. Python has a rich set of libraries and frameworks that can be used to interact with Cloud REST APIs.
3. Authentication: Cloud requires authentication to access its resources and services. The proposed project can use to authenticate the user and obtain an access token. The access token can then be used to make API calls to Cloud REST APIs.
4. Resource Manager (RM): Resource Manager (RM) is a management layer that provides a unified API to manage resources. The proposed project can leverage RM to manage resources using GUI.
5. Error Handling and Logging: The proposed project can implement error handling and logging mechanisms to provide better visibility and control over resource cleanup tasks. Error handling can ensure that the project can recover from unexpected errors, while logging can provide a history of executed commands and their outcomes.

2.1.2 Economical Feasibility:

Economic feasibility is an important aspect of any software project, and the proposed cloud resource garbage collector project is no exception. The economic feasibility of the project can be evaluated by considering the costs involved in developing, deploying, and maintaining the system, as well as the potential benefits in terms of cost savings and increased efficiency

1. Deployment Costs: The deployment costs of the system will depend on the infrastructure needed to run the tool. The tool can be deployed on any system with an internet connection, and there are no specific hardware requirements. Therefore, deployment costs are expected to be minimal.

2. Maintenance Costs: Maintenance costs will depend on the frequency of updates and the need for ongoing support. As cloud is a rapidly evolving platform, this tool will need to be updated periodically to remain compatible with cloud APIs. However, this tool is expected to be relatively stable and require minimal maintenance.

3. Savings: The primary benefit of the system will be cost savings, as it will help identify and delete unused or unnecessary resources, reducing costs associated with maintaining these resources. Organizations with large and complex environments are likely to benefit the most from the system.

4. Increased Efficiency: In addition to cost savings, the proposed system will increase efficiency by automating the process of identifying and deleting resources, freeing up time for other tasks. This will allow organizations to focus on higher-priority tasks, leading to increased productivity and profitability.

Overall, the proposed cloud resource garbage collector project is economically feasible, as the development, deployment, and maintenance costs are expected to be relatively low, while the potential benefits in terms of cost savings and increased efficiency are significant.

2.1.3 Operational Feasibility:

The operational feasibility of the cloud resource garbage collector project using GUI is high due to several reasons.

1.The proposed system provides a straightforward and efficient way of managing cloud resources, which can be easily integrated into the existing operations of an organization. As the tool is GUI based, it can be integrated into the existing scripts and workflows, reducing the need for additional training or significant changes to the existing infrastructure.

2. The proposed system allows for greater control over resource management, which can help organizations to reduce costs and improve overall efficiency. By providing a way to easily identify and delete unused or unnecessary resources, the proposed system can help organizations to free up resources and reduce their costs. This can be particularly useful for organizations that deploy and manage multiple resources, where resources can become underutilized over time.

3. The proposed system offers a way to perform dry-run executions, allowing users to preview the changes that will be made before any resources are actually deleted. This feature can help organizations to avoid any unintentional deletions, reducing the risk of disrupting critical systems or workflows

2.2 Hardware and Software Requirement:

The proposed project can be executed on any standard computer system with the following minimum hardware requirements:

* CPU: Intel Core i3 or equivalent processor
* RAM: Minimum 4 GB or higher
* Storage: 100 GB or higher hard disk drive
* Network: Ethernet or Wi-Fi connectivity for internet access.

**Software Requirements:** The following software is required to execute the project:

* Python 3: The project is implemented in the Python language, and therefore, Python 3.x is required to execute the code.
* Python AWS SDK: To access AWS resources programmatically using Python,It can be installed using the command "pip install boto3".
* Python Azure SDK: The Python Azure SDK is required to access Azure resources programmatically using Python. It can be installed using the command "pip install azure".

**Development Environment:** The project can be developed using any integrated development environment (IDE) that supports Python development. Popular IDEs include PyCharm, Visual Studio Code, and Spyder.

**Dependencies:** The project has the following dependencies, which can be installed using the Python package manager, pip:

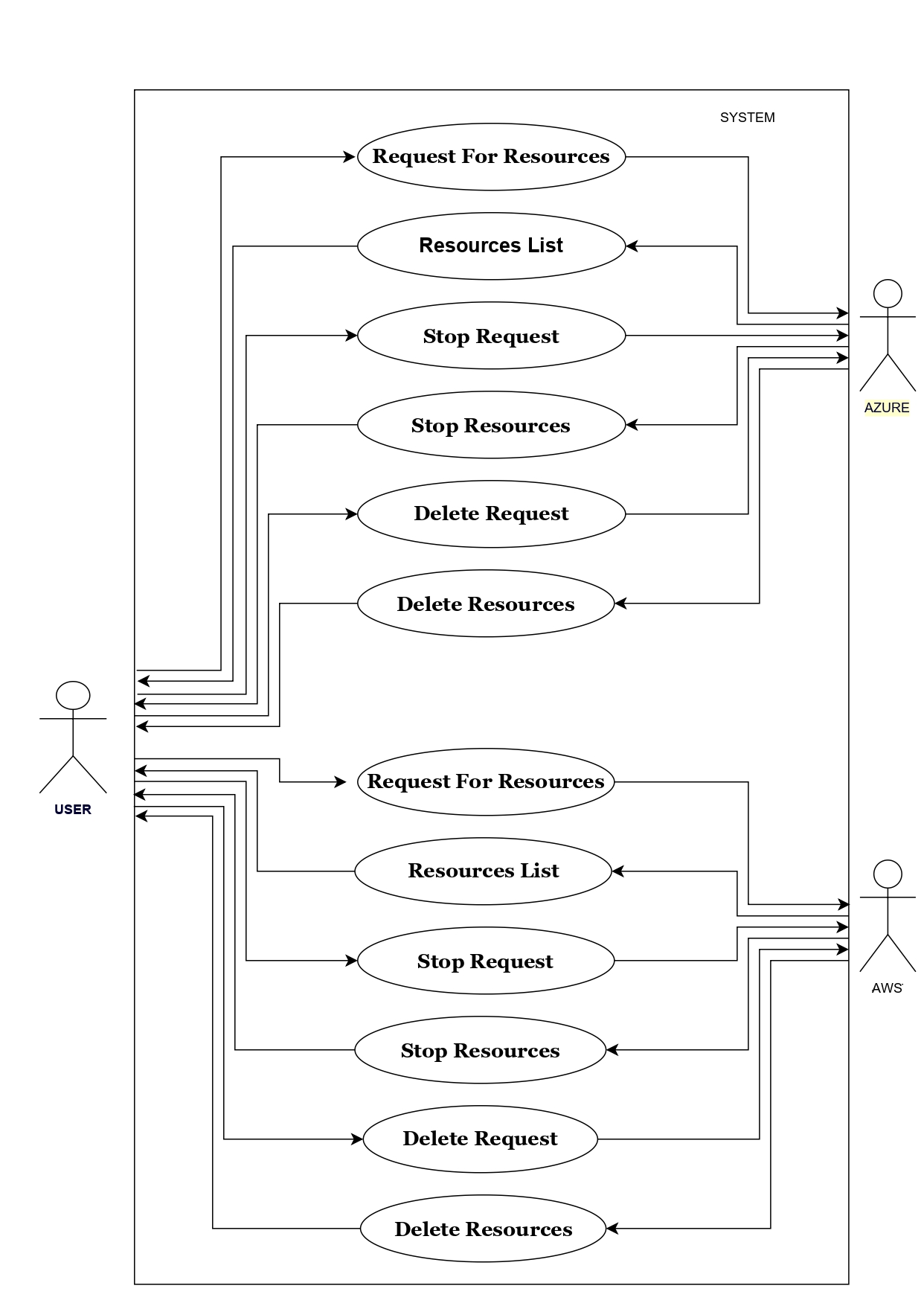
* Tkinter:This package is used for making GUI.
* Msrestazure:This is a Python package that provides Azure-specific extensions for the msrest library. msrest is a Python library for working with RESTful APIs.
* azure.mgmt.compute:This is a Python package that provides a client library for interacting with the Azure Compute service.
* azure.mgmt.network:This is a Python library for managing Azure Virtual Network resources.
* azure.identity:This is a Python package that provides Azure Active Directory authentication support for the Azure SDK.
* Boto3: This is a Python package that allowes developers to interact with AWS services,you need to set up authentication credentials for AWS account.

**Deployment:** The project can be deployed on any system that meets the above hardware and software requirements. It can be executed using the GUI, and the user can pass the required arguments to clean up the Cloud resources.

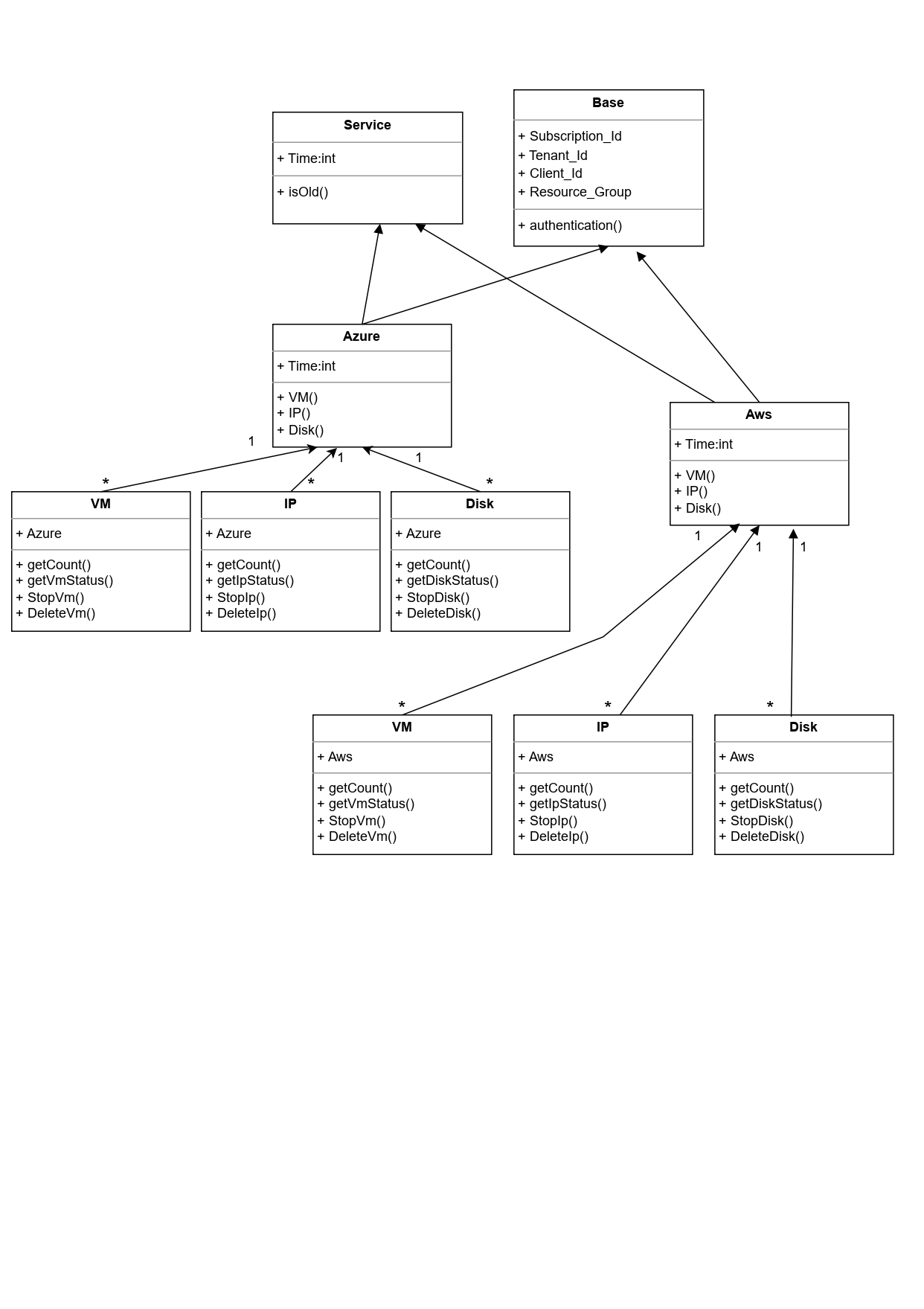
**3.DESIGN**

3.1 UML Diagram

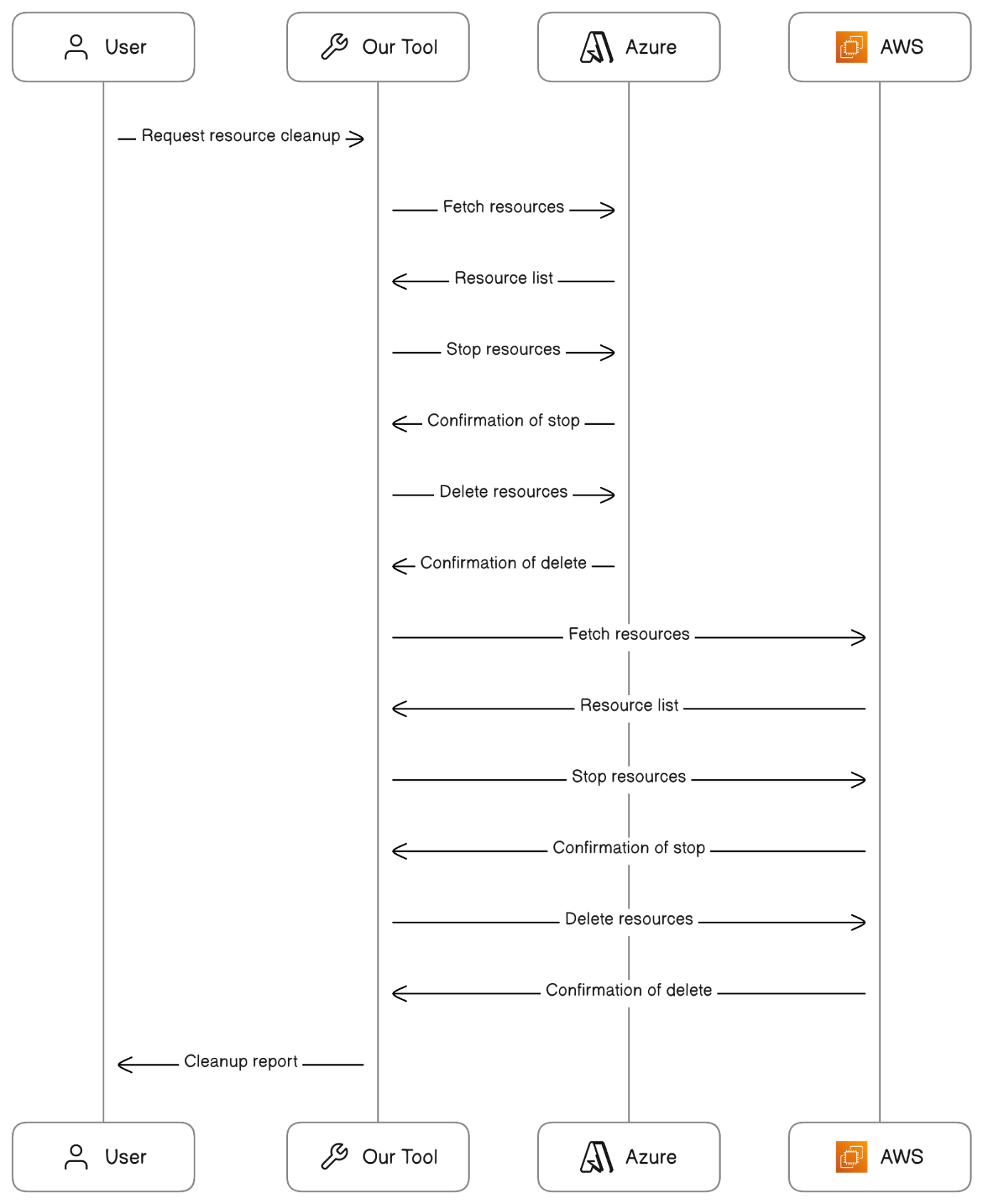
3.1.1 Use Case Diagram:



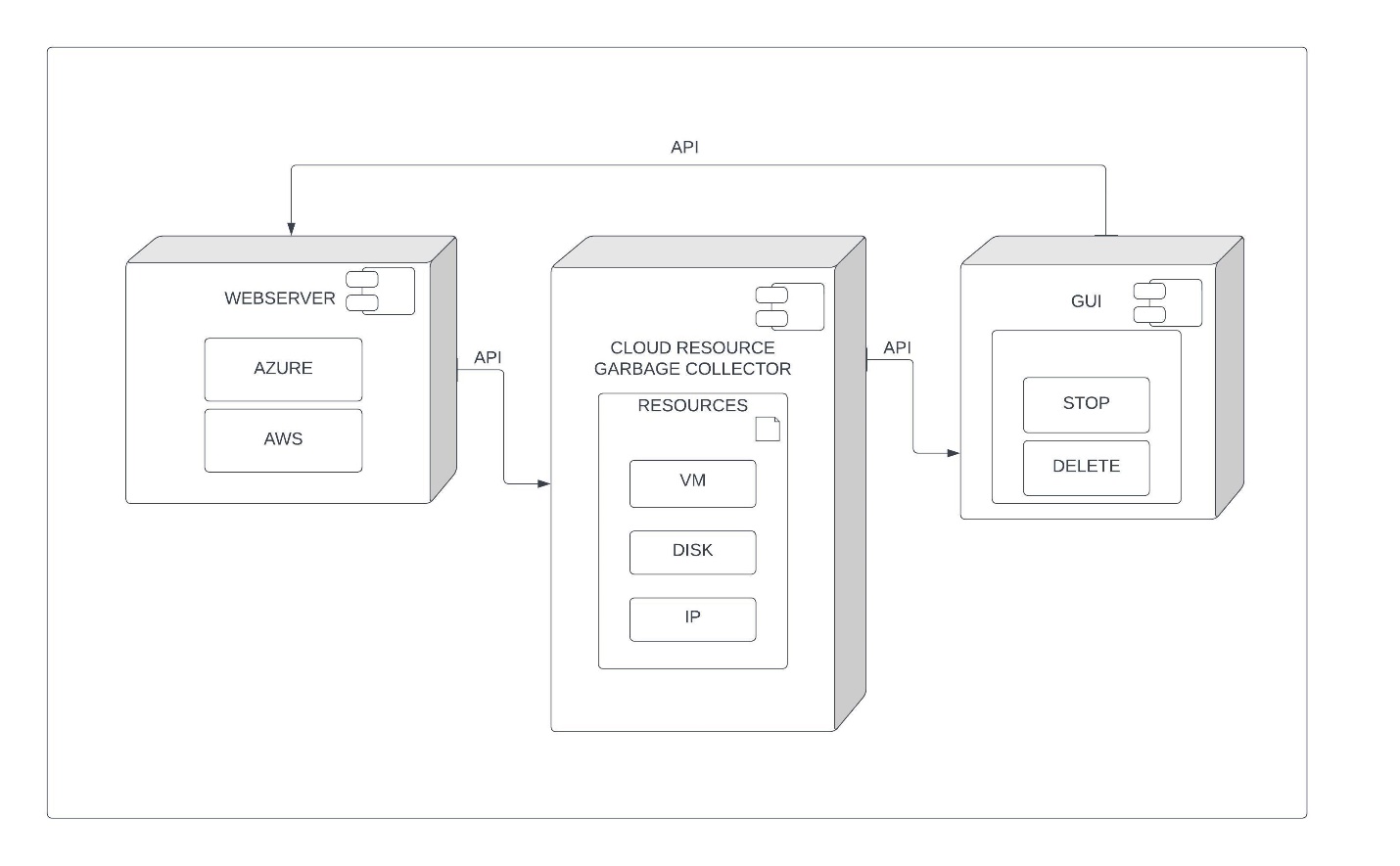
3.1.2 Class Diagram:

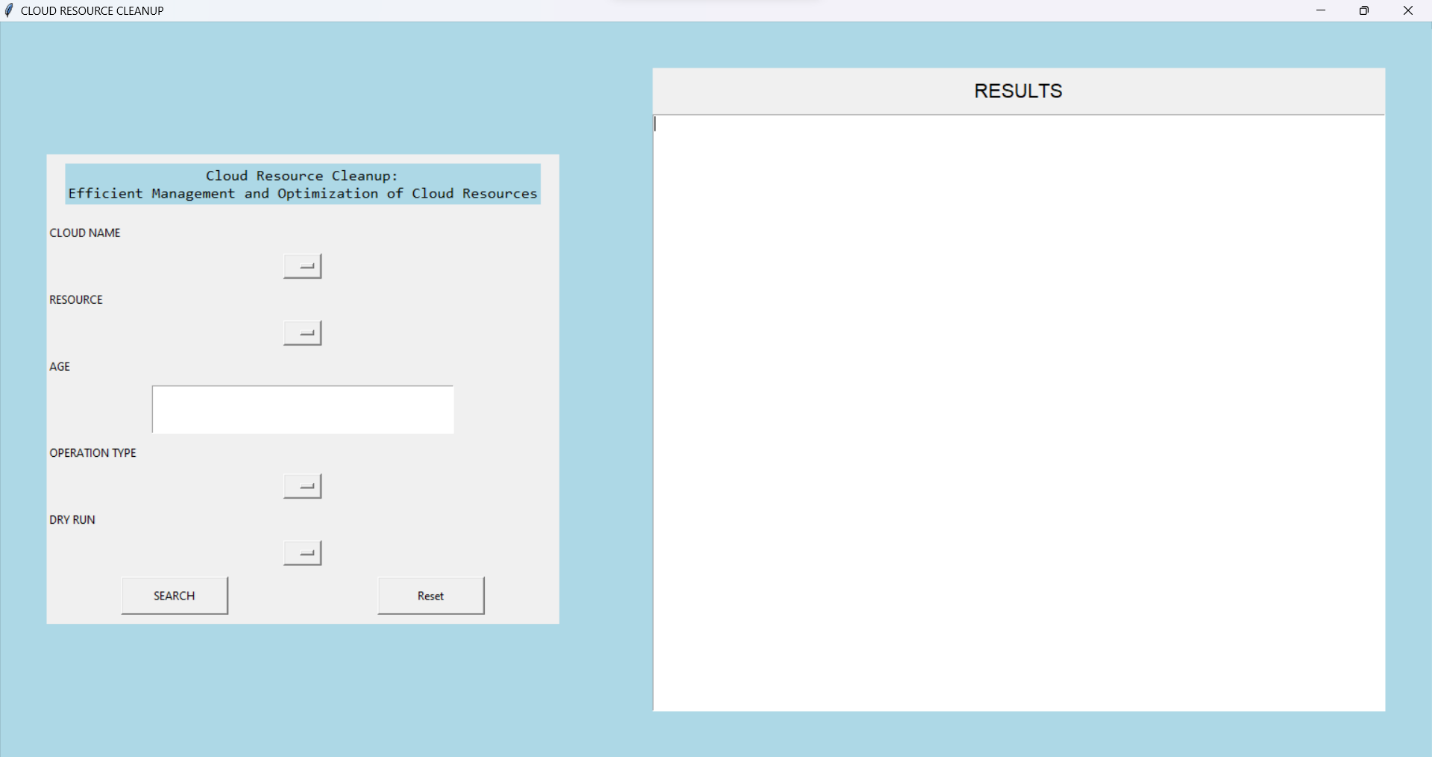


3.1.3 Sequence/Activity diagram:

****

3.1.4 Deployment Diagram:



3.2 Input and Output Screens And Reports:  
  
  


**4.TESTING**

4.1 Importance of Testing:

Testing is a crucial aspect of any software development project, including the creation of a Python script for cleaning up Azure cloud resources via command line arguments. Here are a few reasons why testing is important:

1. Catching bugs: Testing allows you to catch bugs in your code before it's deployed to production. By identifying and fixing issues early on, you can prevent potential downtime or data loss.

2. Ensuring functionality: Testing ensures that your script performs the functions it's supposed to. By verifying that your code is working as intended, you can be confident that your cleanup project will function properly.

3. Improving maintainability: Testing helps you to write more maintainable code. By creating unit tests that verify the functionality of your code, you can make changes to your code with greater confidence that you won't break anything.

4. Reducing risk: Automated testing reduces the risk of human error, which can be especially important in cloud resource cleanup projects where mistakes can be costly

4.2 Types of Testing:

System Testing: System testing involves testing the entire system as a whole to ensure that it meets the specified requirements. This type of testing could involve testing the cleanup process end-to-end, from the GUI to the final confirmation that the resources have been deleted.

* 1. Test Cases (with Expected and Actual Result)

1.**Expected:** Stop is only supported with VM not with IP or DISK.  
 **Result:**

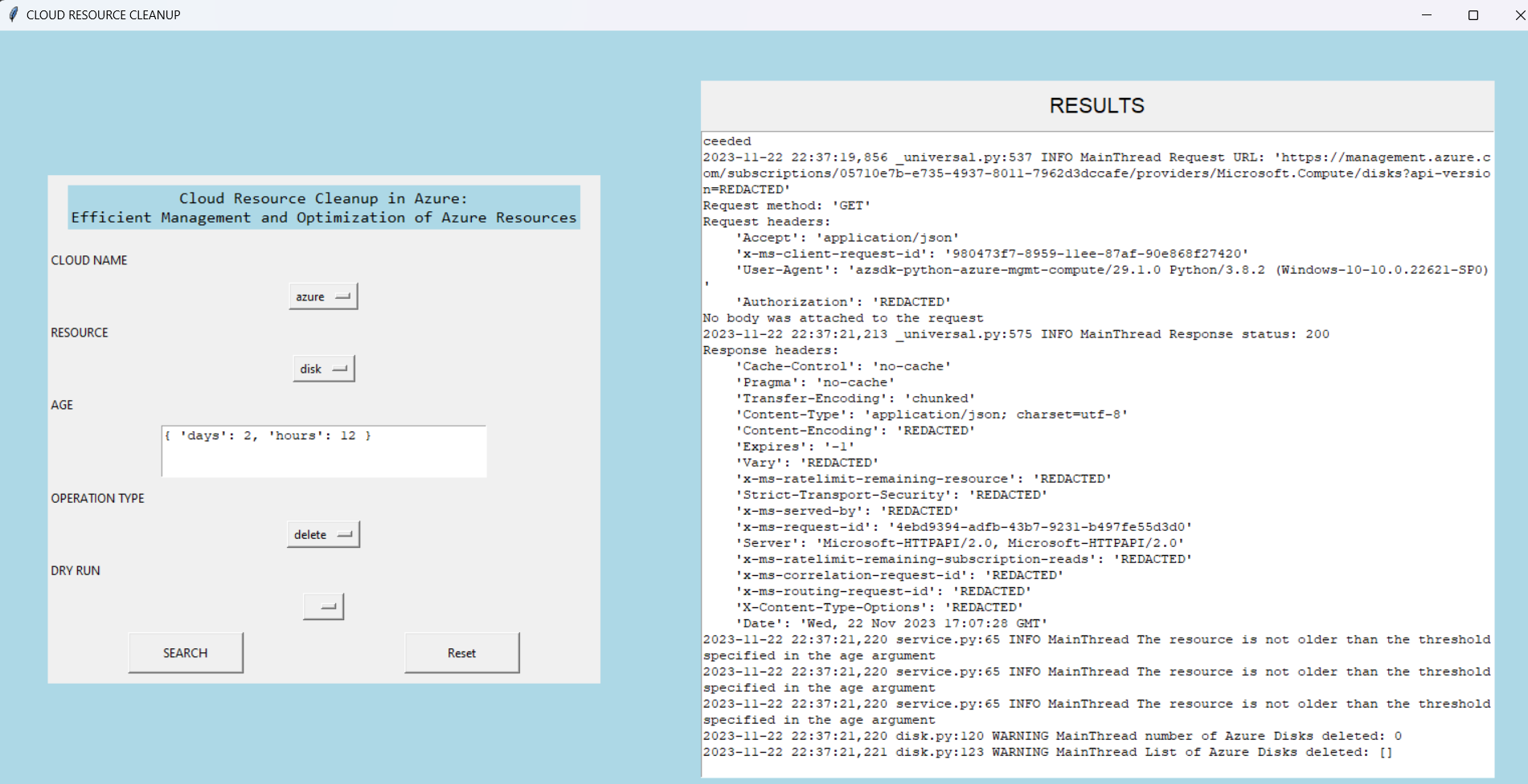


2.**Expected:** KeyPair option is only supported with AWS not with Azure.  
 **Result:**

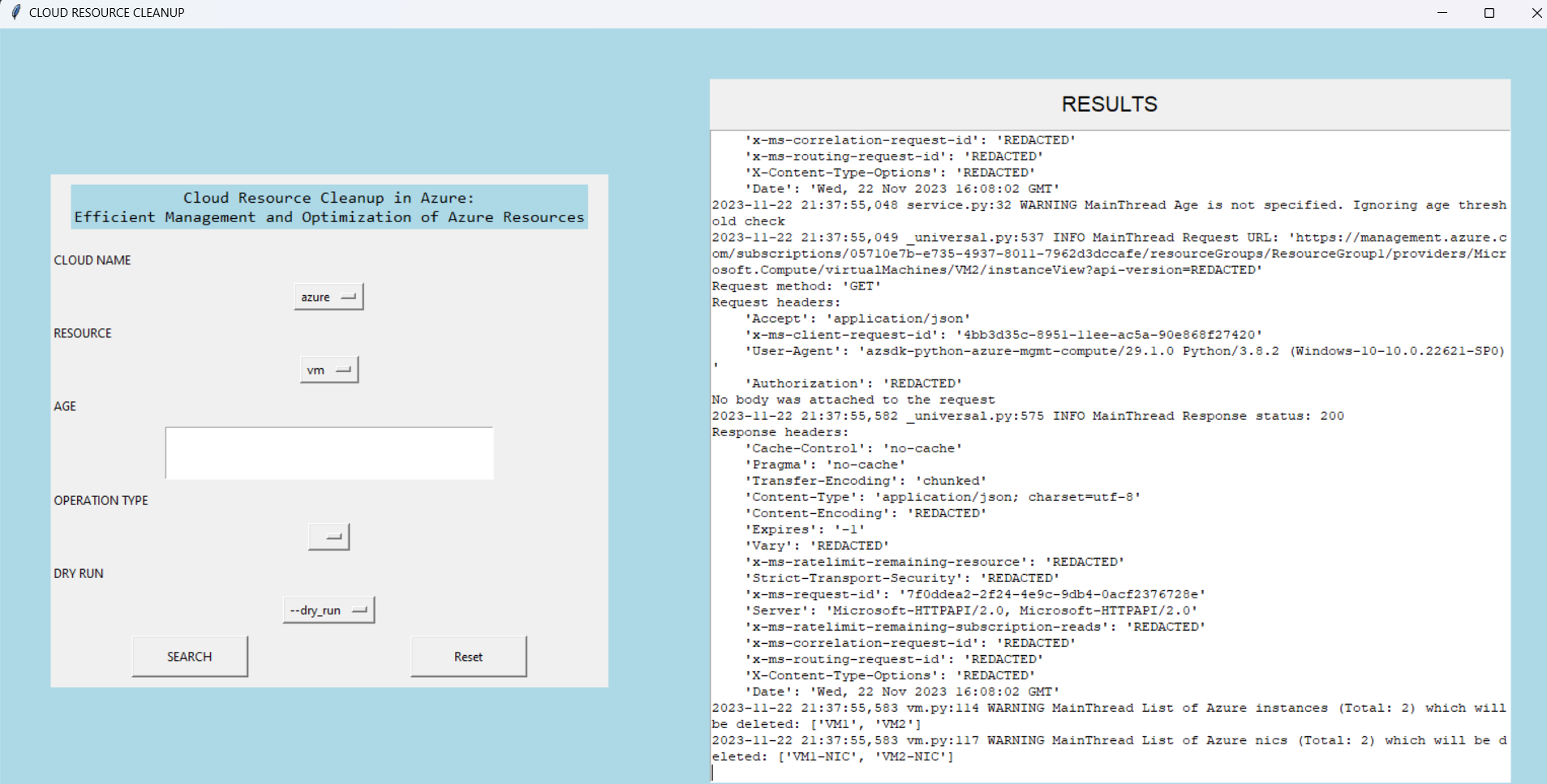
****

3.**Expected:** To check for resources older than 2 days and 12 hours.

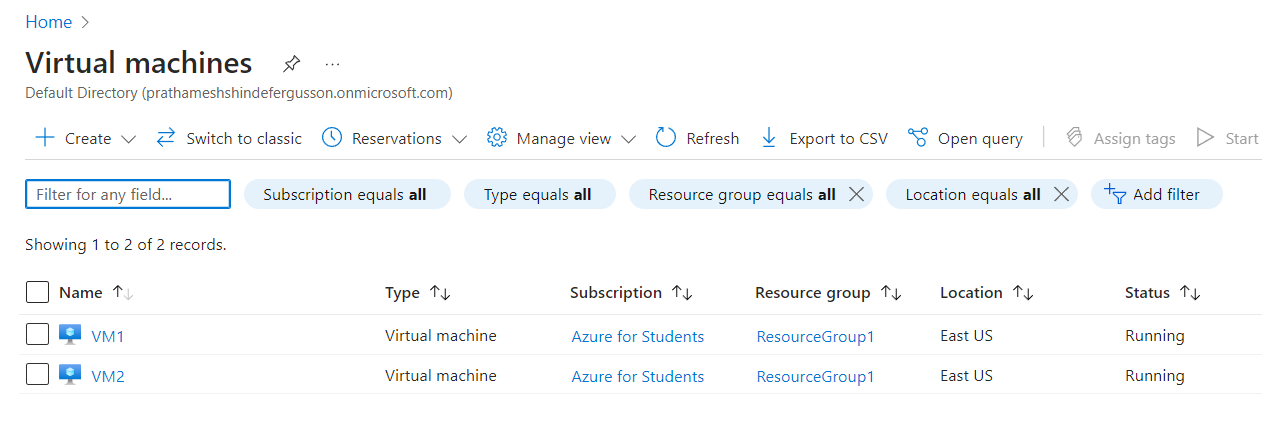
**Result:**

****

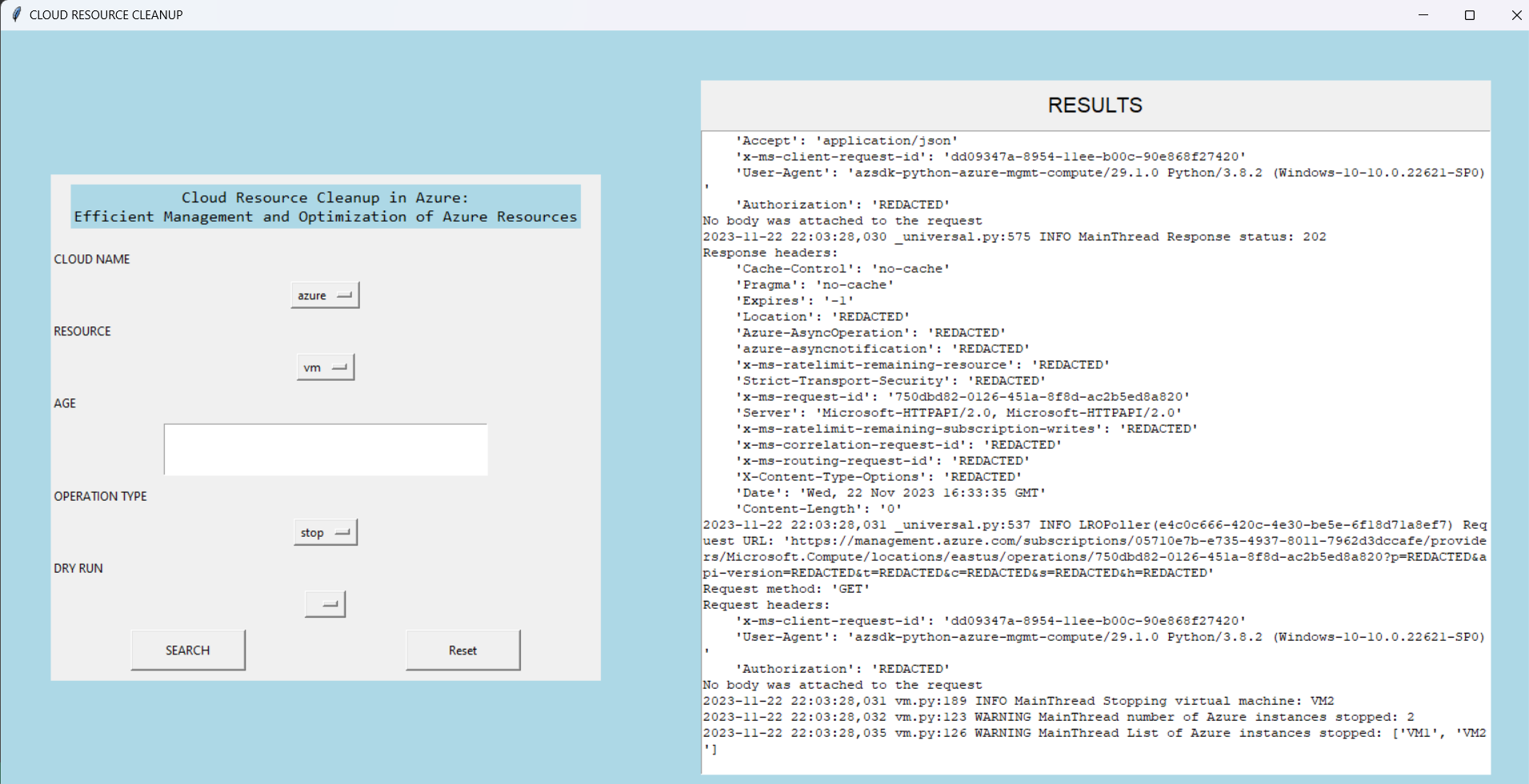
4.**Expected:** To check is there Azure VM running:  
 **Result:**



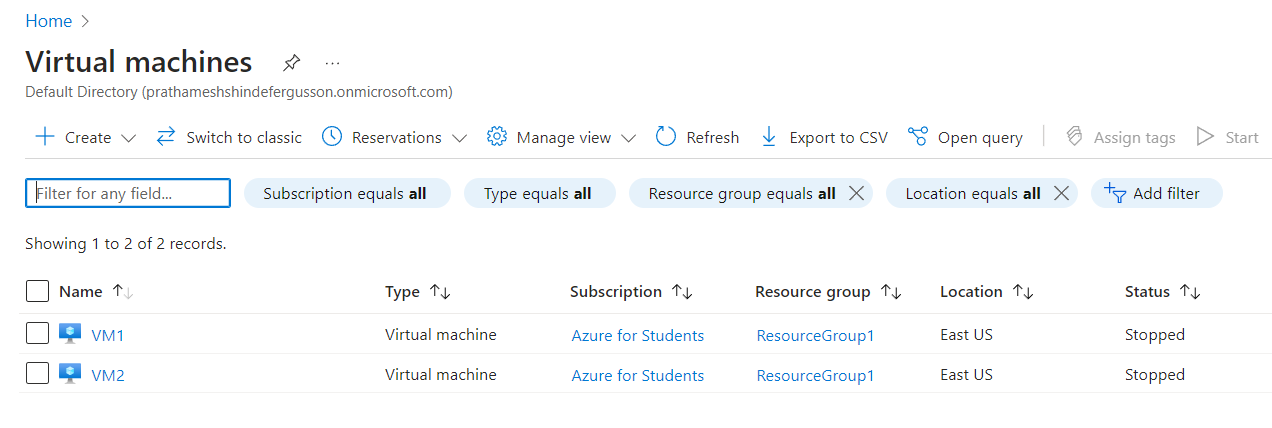
**Verify through Azure portal are there 2 VM running (VM1 and VM2).**

****

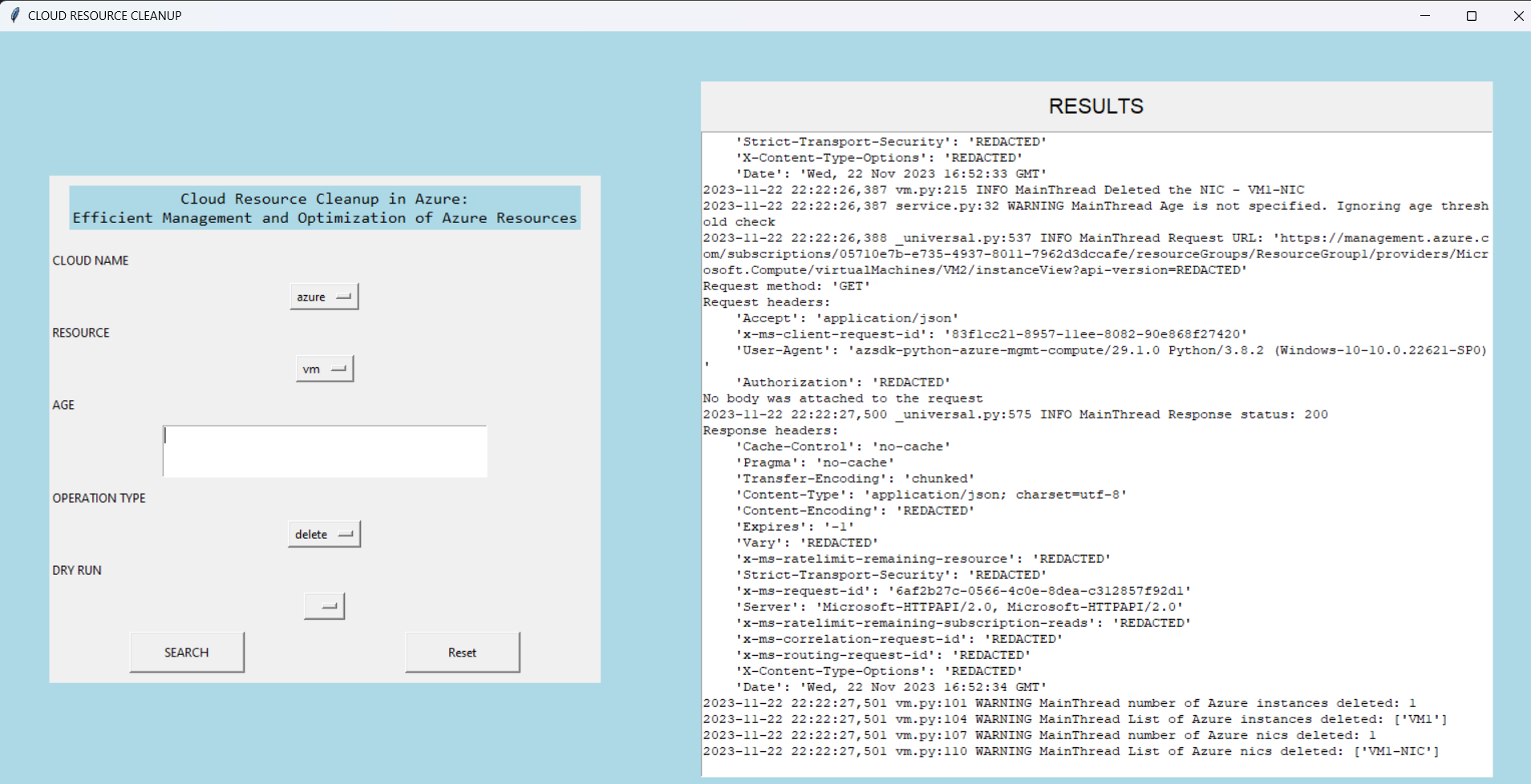
5.**Expected:** To delete both VM.  
 **Result:**

****

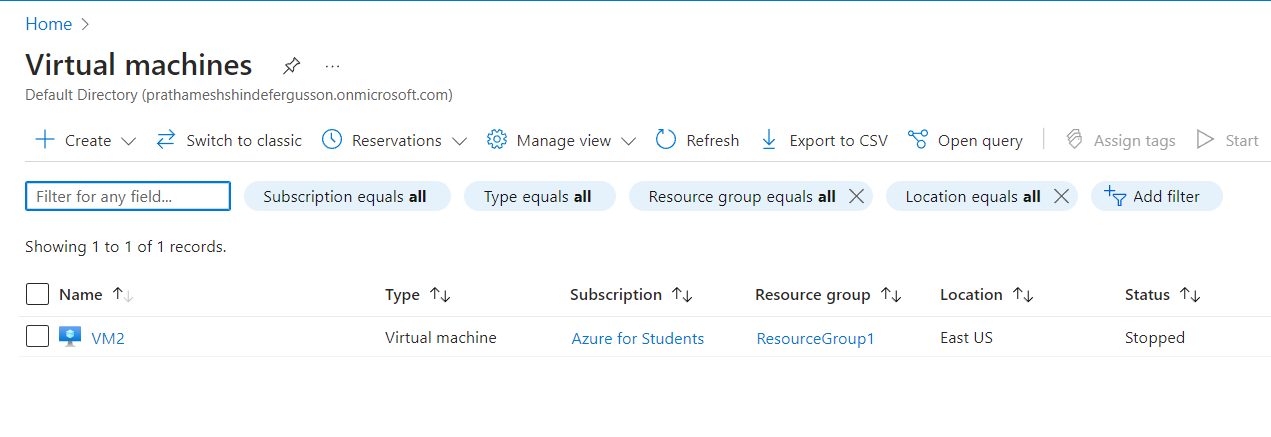
**Verify through Azure portal does it succesfully stopped or not?**



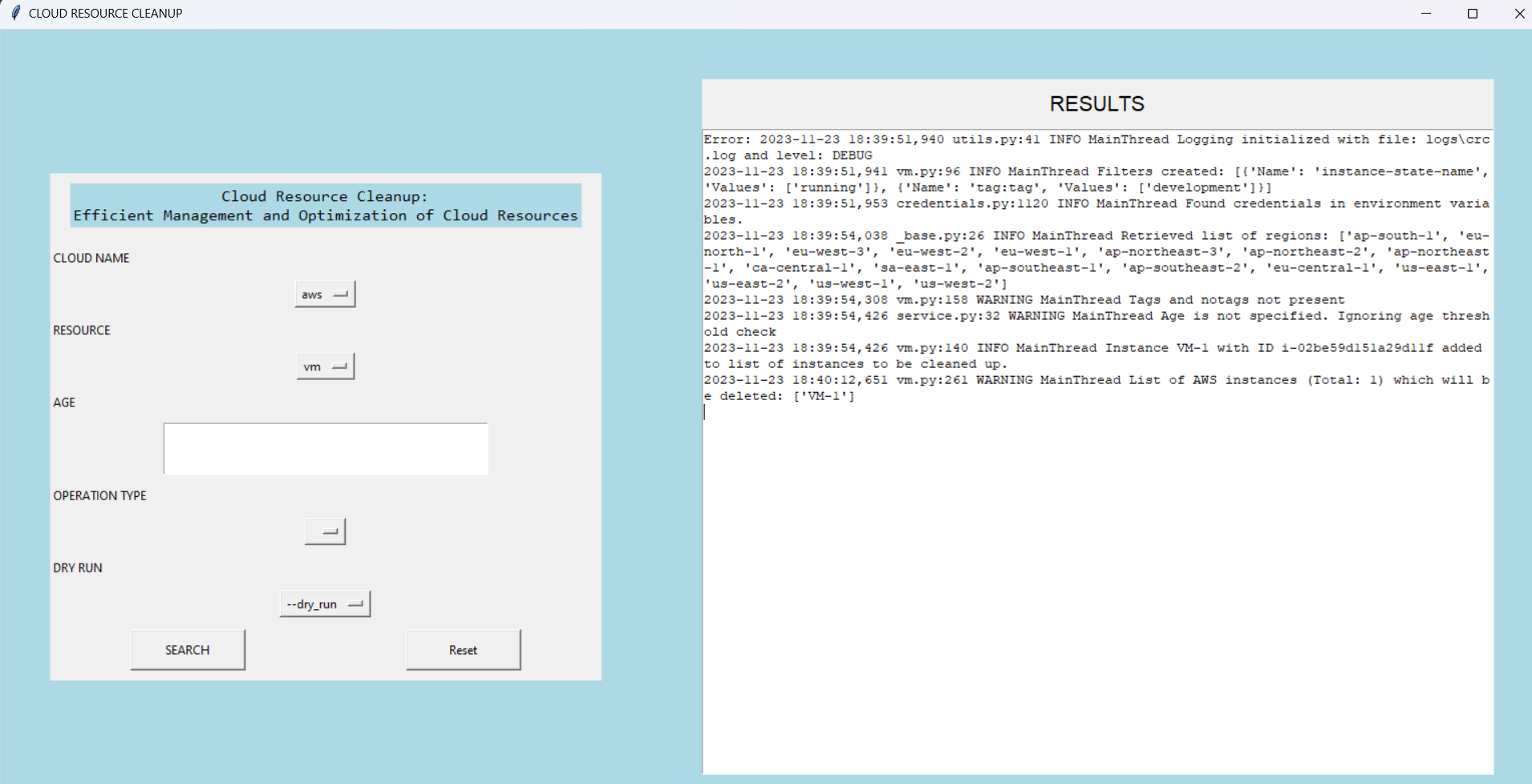
6.**Expected:** Delete VM1.  
 **Result:**

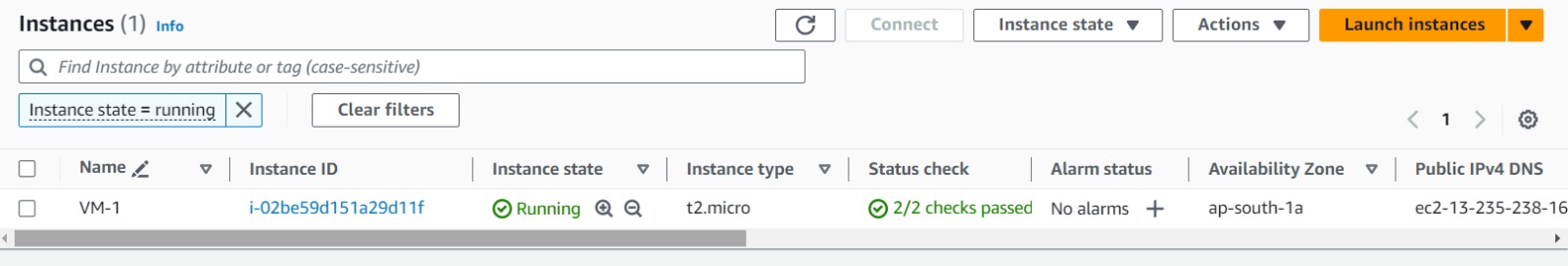
****

**Verify through Azure portal does it Succesfully delete or not?**

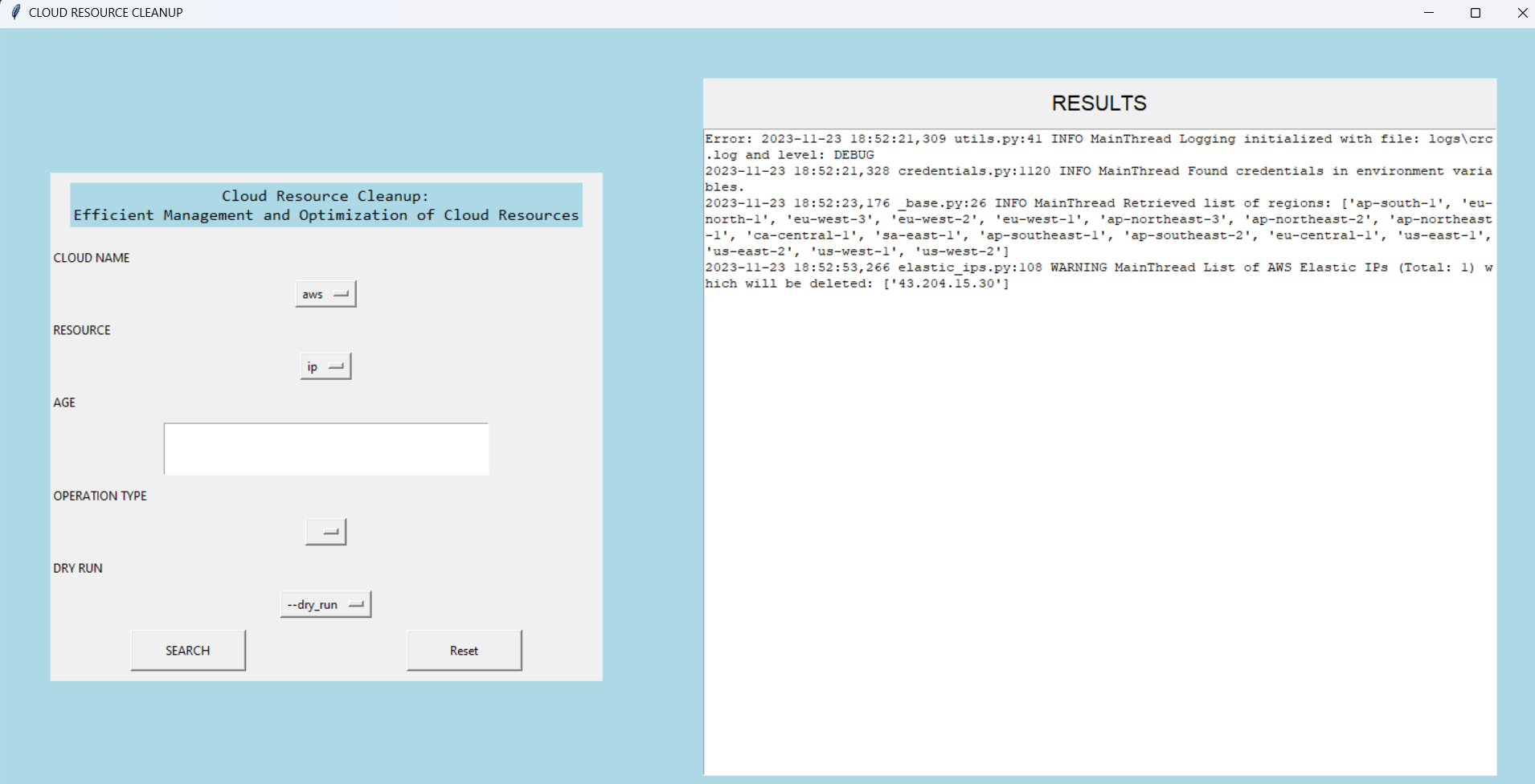


**7.Expected: To dry\_run Available Virtual Machine in AWS.  
 Result:**

****

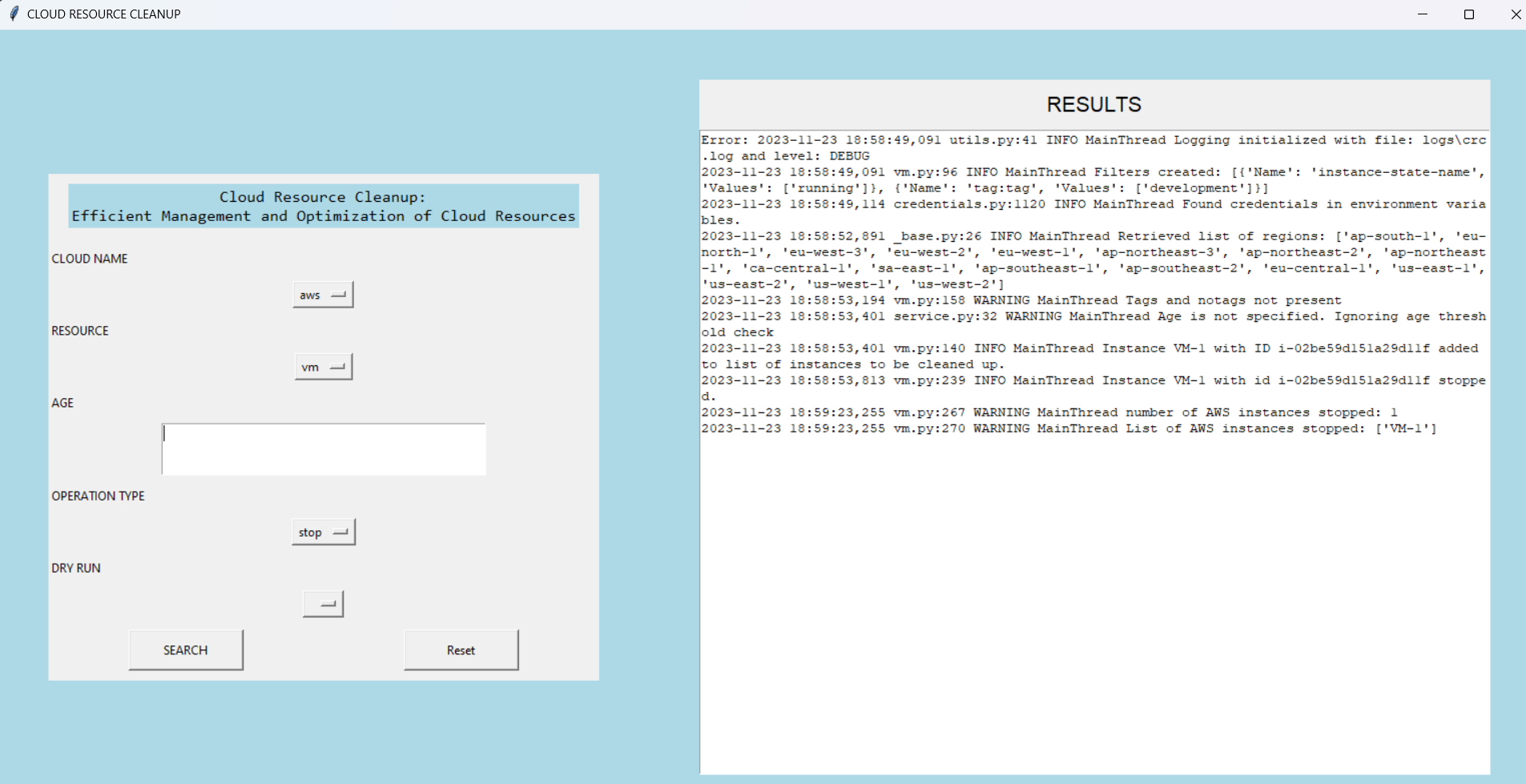
**Verify through AWS portal to cross check.  
**

**8.Expected: To dry\_run elastic IPs in AWS  
 Result:**

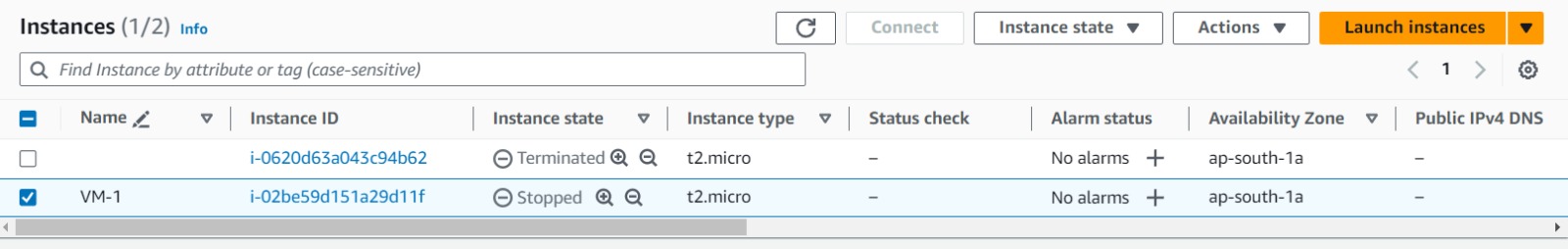
****

**9.Expected: To stop and delete Virtual machine in AWS**

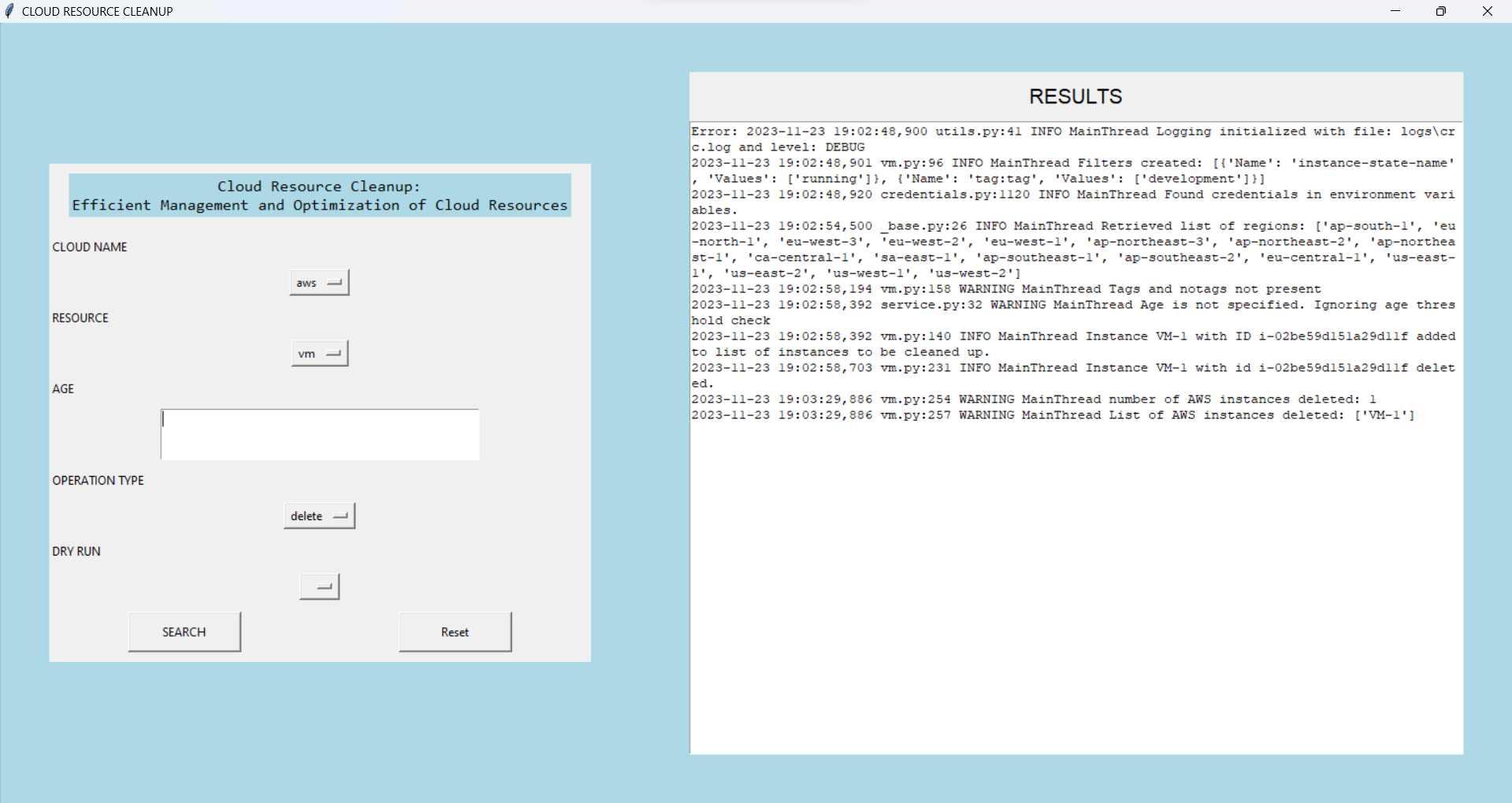
**STOP:**

****

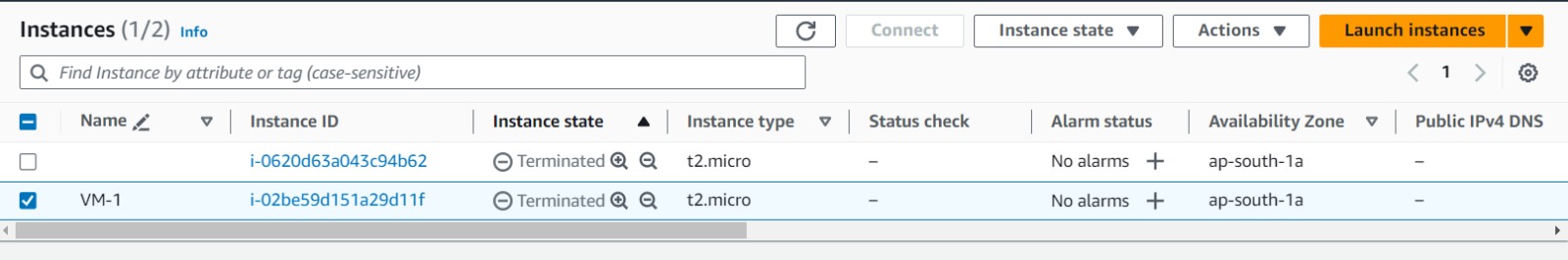
**Verify throughAWS portal does It Succesfully STOP or not?**

****

**DELETE:**

****

**Verify throughAWS portal does It Succesfully DELETE or not?**

****

**5.REPORTS**

**Report 1:** Resource Usage Report The resource usage report provides an overview of the resources deployed in Coud. This report can help identify underutilized resources that can be deleted to reduce costs. The report includes the following information:

* Number of resources
* Resource Name
* Timestamp

**Report 2:** Unused Resource Report The unused resource report provides a list of resources that have not been used in a specified period. This report can help identify resources that can be safely deleted to reduce costs. The report includes the following information:

* Number of resources
* Resource Name
* Timestamp

**Report 3:** Resource Deletion Report The resource deletion report provides a summary of the resources that were deleted using the tool. This report can help track the changes made to the resources and ensure that the correct resources were deleted. The report includes the following information:

* Number of resources
* Resource Name
* Deletion Timestamp
* Deletion Status

**6.DRAWBACKS AND LIMITATION:**

1. **Limited functionality:** The project is limited to managing AWS and Azure resources using GUI. It cannot be used to manage resources in other cloud platforms.
2. **Potential for accidental deletions:** The GUI can be powerful, but it also has the potential for user error. The user must be careful when providing input arguments to avoid accidentally deleting important resources.
3. **Dependency on REST APIs:** The project relies heavily on REST APIs to manage resources. Any changes or updates to these APIs can impact the functionality of the tool.
4. **Limited support:** The project may have limited support or maintenance, as it is a custom tool developed by a team. Any issues or bugs may take longer to address.
5. **No support for automatic scheduling:** The tool cannot be scheduled to automatically run on a regular basis. This means that manual intervention is required to ensure that resources are properly managed.

**7.CONCLUSION:**

1. The AWS Azure documnetation provide guidance on how to delte resources and to reduce costs, unused resources can accrue charges while unattached, so it is important to reguraly identify and delte them to manage costs.
2. The project has some limitations, including limited functionality, potential for accidental deletions, dependency on REST APIs, limited support, and no support for automatic scheduling.
3. Overall, the proposed system will provide a more efficient and automated way of managing cloud resources using a Graphical User Interface (GUI), reducing costs and improving overall system efficiency.
4. To avoid accidental deletions, the tool will provide the ability to perform dry-run executions, allowing users to preview the changes that will be made without actually deleting any resources. This will help ensure that the user can review the resources to be deleted and avoid any unintentional deletions.

**8. FUTURE ENHANCEMENT**

1. Cost visibility prior to resource deletion: Before removing a cloud resource, conduct a cost analysis to gauge its financial impact. This step ensures informed decision-making and empowers users to optimize expenses. By understanding the associated costs, efficient resource utilization becomes more achievable.
2. Cron or Windows Task Scheduler: Cron and Windows Task Scheduler are tools that allow you to schedule tasks to run at specific intervals or times. We can use these tools to schedule our Python scripts to run automatically at certain intervals to clean up resources.
3. Multi-cloud support: Extending the tool to support other cloud providers in addition to AWS and Azure would provide a more comprehensive solution for organizations that utilize multiple clouds.

**9.REFRENCES AND BIBLIOGRAPHY**

1. **Microsoft Azure. Azure SDK for Python-**

This documentation provides information on the Azure SDK for Python, which includes libraries and tools that developers can use to build applications that interact with Azure services. This resource was useful in understanding how to use Python to interact with Azure services, including the Azure Resource Manager APIs used in this project.

[*https://docs.microsoft.com/en-us/azure/developer/python/*](https://docs.microsoft.com/en-us/azure/developer/python/)

1. **Microsoft Azure. Azure Resource Manager REST API-**

This documentation provides information on the Azure Resource Manager REST API, which is the API used to manage Azure resources. This resource was useful in understanding how to use the API to query and manage resources in Azure and how to integrate it with Python.

[*https://docs.microsoft.com/en-us/rest/api/resources/*](https://docs.microsoft.com/en-us/rest/api/resources/)

1. **Azure-Samples.Azure SDK for Python Samples-**

This GitHub repository provides code samples for using the Azure SDK for Python. This resource was useful in understanding how to use the SDK to interact with Azure services and how to integrate it with Python.

[*https://github.com/Azure-Samples?q=sdk-python*](https://github.com/Azure-Samples?q=sdk-python)

1. **AWS Boto3 SDK for python-**

Boto3 is the Amazon Web Services (AWS) Software Development Kit (SDK) for Python, allowing Python developers to write software that makes use of services like Amazon S3 and Amazon EC

[*https://boto3.amazonaws.com/v1/documentation/api/latest/index.html*](https://boto3.amazonaws.com/v1/documentation/api/latest/index.html)

1. **Tkinter for GUI-**

Tkinter is a Python interface to the Tcl/Tk GUI toolkit, which provides a robust and platform-independent windowing toolkit for Python programmers.The Tkinter widget set allows for constructing graphical user interfaces (GUIs) in Python

[*https://docs.python.org/3/library/tkinter.html*](https://docs.python.org/3/library/tkinter.html)

1. **Stack Overflow-**

Stack Overflow is a community-driven question and answer website for developers. This resource was useful in finding solutions to specific programming issues and in troubleshooting problems encountered during the project.

[*https://stackoverflow.com/*](https://stackoverflow.com/)