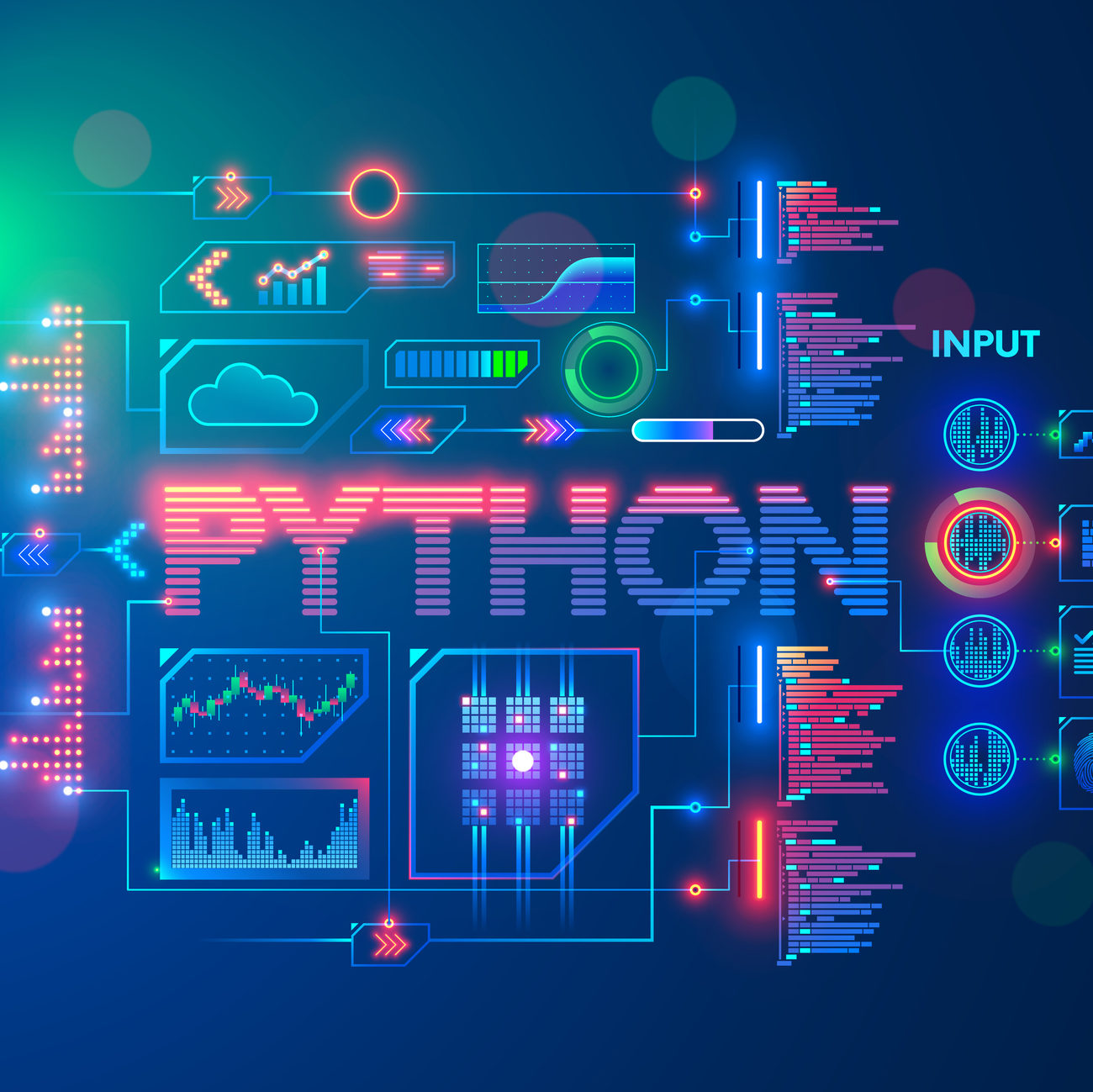
**Assignment 3**



**Submitted by:  
Name: Nikitha Donthi  
Student Id: 100953192  
Date: 06/12/2024**

**Part 1: Identify the two Problem**

**Problem Statement: Cloud Cost Optimization in Multi-Cloud Environments**

While cloud computing in the modern world is embraced by many organizations through multi-cloud strategies, which involve taking the best of all the cloud service providers' offerings, managing the cost across multiple platforms is a very complex process.

That is a problem because AWS, Azure, and GCP have different pricing models, cost control planes, and service catalogs. Without having a single place to view it all, businesses with erratic usage patterns and a mixed bag of services will experience inflated, often unexpected AWS bills.

It involves developing a Python solution to allow users to collect gathering, analyze, and optimize cloud costs across a range of cloud platforms. In cases where the objective would involve fetching cost data, the tools are set to support cost data from several cloud services so that one can have a proper comparison to deliver results one looks out for in providing value to organizations and helping to decrease waste with effective cost-cutting.

**Relevance to Cloud Infrastructure and Hosting**

This problem is of direct relevance to cloud infrastructure, development, and hosting because this is more about the proper management of resources in the cloud. Cloud cost management refers to keeping the infrastructure appropriate while expenses are within or at a lower level of the budget. In scaling with cloud usages, organizations often face challenges in monitoring these costs, especially when they draw services from multiple providers.

Without having one centralized tool or strategy on how to manage the cost, the businesses end up in overprovisioning the resources for the utilization of expensive services that may not be utilised or in missing opportunities that are related to the cost savings, including the rightsizing of instances and optimizing storage.

**Value of Learning Experience**

This project will be of great value for hands-on experience with the best practices in cloud cost management, which is a very essential skill in cloud computing. In the future where organizations will be adopting multi-cloud environments, professionals who can navigate and optimize the costs will be sought after. This project will also enhance my skills in using APIs to collect data from different cloud platforms and further develop my skills in performing data analysis and visualization to produce actionable insights.

It might be a good addition soon to include advanced features, such as predictive cost modeling, or integrating into the existing cloud-native cost management tooling, like AWS Cost Explorer or Azure Cost Management.

**Libraries and Tools**

The libraries and tools that will be used for this project are:

* **boto3:** AWS SDK for Python for interacting with AWS services pulling cost data.
* **Google Cloud Client Library for Python:** Enables gathering data from Google Cloud services.
* **Azure SDK for Python:** This is going to help in integrating with Microsoft Azure and pulling cost-related information.
* **Pandas: For data manipulation and analysis.**
* **Matplotlib/Seaborn**: in case the analysis of costs could be better depicted with a bar chart, pie chart, or other forms of graphical analysis.
* **Requests:** To handle HTTP requests to cloud cost APIs, if needed.

**Challenges and** **Complexity**

I would imagine that a couple of challenges might be the integration of cost data from multiple cloud providers involves a complex task. Each cloud provider differs in API structure and is subject to authentication and authorization. Every cloud provider uses its method of billing or cost tracking, which probably requires some processing to reformat into a common state to allow comparison. Besides that, handling edge cases, where APIs return either empty or incomplete data, and rate limiting set by cloud providers will require careful planning for the solution to be reliable and efficient. However, even with these challenges, this is a problem that I would argue feels suitable for my level of comprehension about cloud computing with Python programming. The skills being developed in solving this problem will be directly applicable in real-world scenarios, so I look forward to it.

**Part 3: Reflect on the Solution**

Although I was unable to meet with you for direct feedback, I undertook a comprehensive review and revision process to refine the Python solution to the cloud cost optimization problem. In this section, I will explain the changes I made after testing and reflecting on the initial version of the code.

**1. Self-Assessment and Debugging**

Upon reviewing my code, I identified a few areas where the solution could be improved, particularly regarding error handling and code optimization. Some of the issues I addressed include:

**Error Handling:**

Initially, my code did not handle potential issues such as empty data sets or connection problems with APIs. I added exception handling to ensure that the program gracefully handles these situations.

**Edge Cases:**

I tested the program with different inputs, including empty lists and very large data sets. This helped me to improve the code's robustness by adding safeguards to ensure it handles such cases properly. For example, I added a check to ensure that the generate\_report function doesn't break if the analysis data is empty.

**2. Code Optimization**

To make the code more efficient, I refactored certain parts where I recognized redundancies or inefficiencies. For example, I combined the analysis and visualization steps into a single function to improve the workflow and reduce function calls.

* **Optimized Analysis and Visualization**: Instead of calling two separate functions for analysis and report generation, I integrated them into a more streamlined approach:

**3. Improved Documentation and Comments**

To ensure that the code is easier to understand for others (or for future revisions), I enhanced the documentation and added detailed comments explaining key sections of the code. This was particularly important for explaining the logic behind the cost analysis and data fetching process.

**4. Final Testing and Reflection**

Before finalizing the solution, I ran multiple tests, including those with varied data and edge cases. After the revisions, the program was able to handle a wider range of inputs and errors. I also reviewed the code for clarity, optimizing readability and ensuring that all functions adhered to the established coding standards.

**Conclusion of Revisions**

The revised solution provides a more robust, reliable, and user-friendly Python script for automating cloud cost optimization. The added error handling, improved documentation, and code optimization ensure that the tool performs as expected under different conditions. Although direct feedback from the professor was not possible, these revisions were made based on self-reflection and testing, ensuring a higher-quality solution\