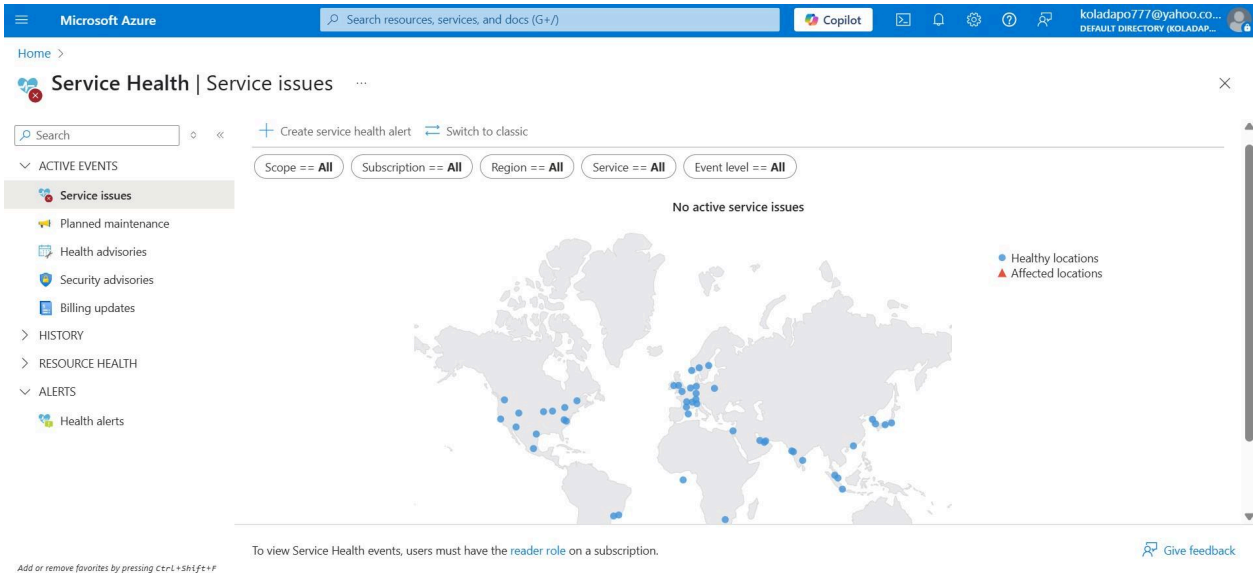
Project Report: Proactive Azure Monitoring with Service Health Alerts

This report summarises the key steps and outcomes of my project to implement proactive Azure monitoring using Service Health alerts. The objective was to establish a system for early notification of service incidents, planned maintenance, and health advisories affecting my Azure resources.

Task 1: Setting Up Service Health Monitoring

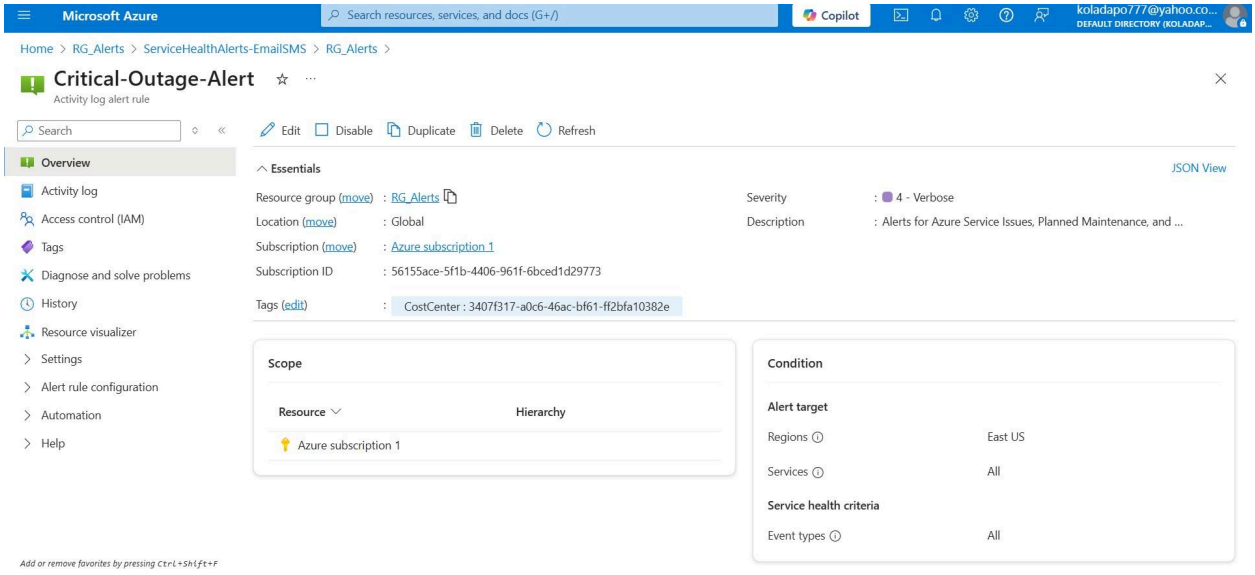
My initial step involved familiarising myself with the Azure Service Health dashboard. I navigated to the Service Health blade in the Azure portal and thoroughly explored its various sections. I reviewed the 'Service Issues' tab to understand active outages and performance degradations impacting Azure services. I then examined 'Planned Maintenance' to see upcoming scheduled downtimes relevant to my resources. The 'Health Advisories' section provided valuable insights into deprecations and recommendations. Finally, I delved into 'Resource Health' to grasp the granular health status of individual resources, which proved to be distinct from the broader service health view. This exploration provided a foundational understanding of the critical information available.



Screenshot of the Service Health Dashboard

Task 2 Step 1: Creating Targeted Health Alerts (Configure Alert Rules)

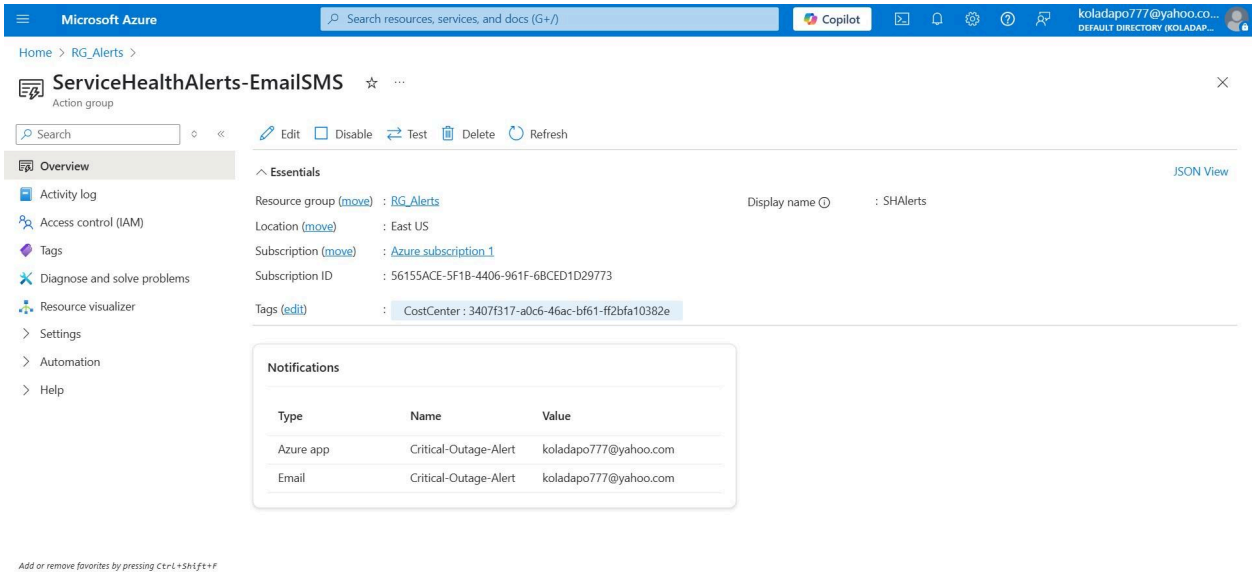
The core of this project was the creation of a targeted alert rule to ensure proactive notifications. I initiated the creation of a new service health alert within the Azure portal. I carefully defined the scope of the alert, selecting the specific Azure subscription, critical services, and relevant regions where my resources are deployed. For comprehensive coverage, I included all three event types: 'Service Issues (Outages)', 'Planned Maintenance', and 'Health Advisories', recognising that Azure Monitor inherently assigns appropriate severities based on these event types.



Screenshot of the configured Alert Rule

Task 2 Step 2: Creating Targeted Health Alerts (Setup Notifications)

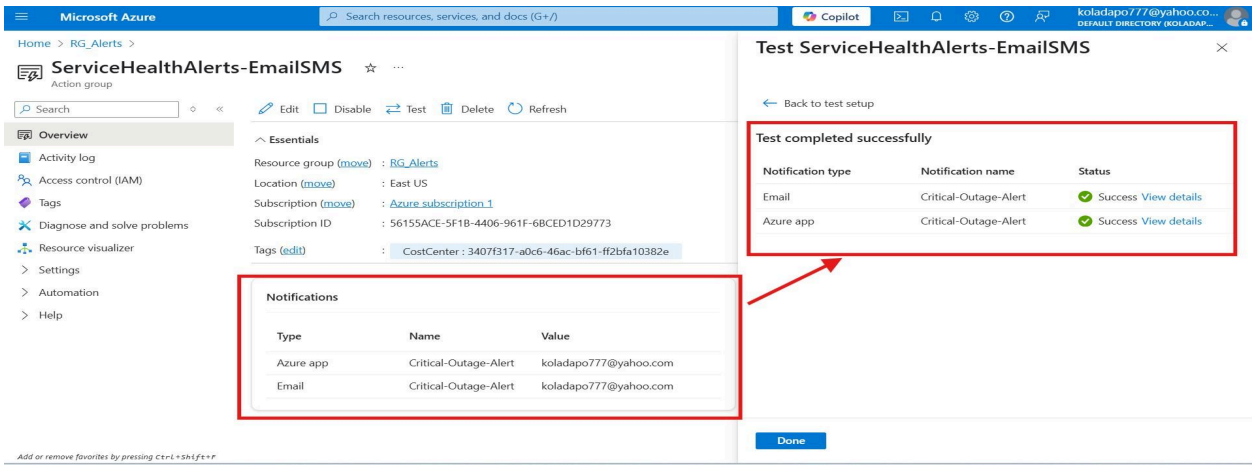
A crucial part of this task was establishing an Action Group to handle notifications. I created a new Action Group, which I named **`ServiceHealthAlerts-EmailSMS`**. Within this action group, I configured the primary notification channels: my email address and mobile app. I also ensured this Action Group was stored within my dedicated monitoring resource group, **`RG_Alerts`**. Finally, I named the alert rule **`Critical-Outage-Alert`** and ensured it was enabled upon creation, ready to monitor my selected Azure services.



Screenshot of my configured Action Group Overview Page

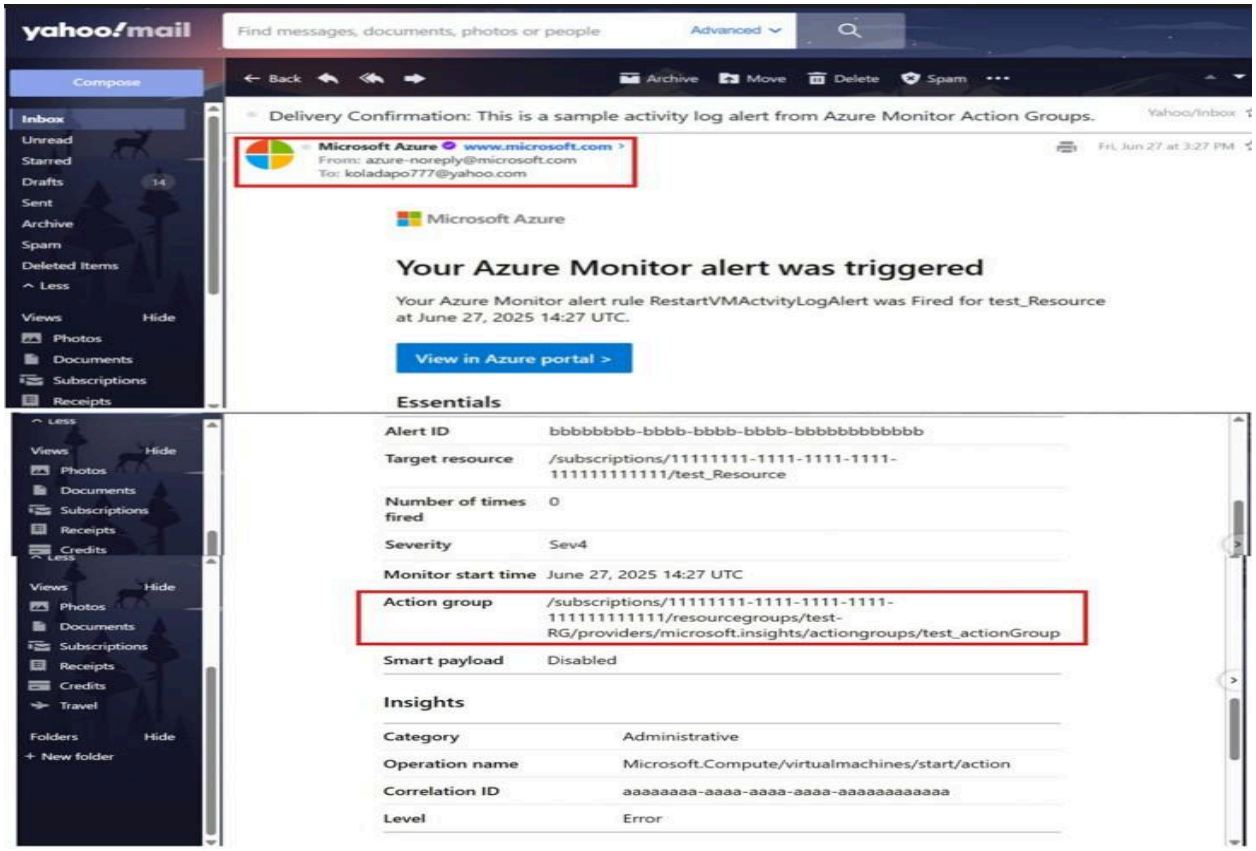
Task 3: Testing Alert Workflows (Trigger a Test Notification)

To validate the effectiveness of my setup, I proceeded to test the alert delivery mechanism. My primary validation involved triggering a test notification directly from the Action Group. I navigated to my **`ServiceHealthAlerts-EmailSMS`** Action Group and used the "Test action group" feature to send a test notification via email and Azure App message.

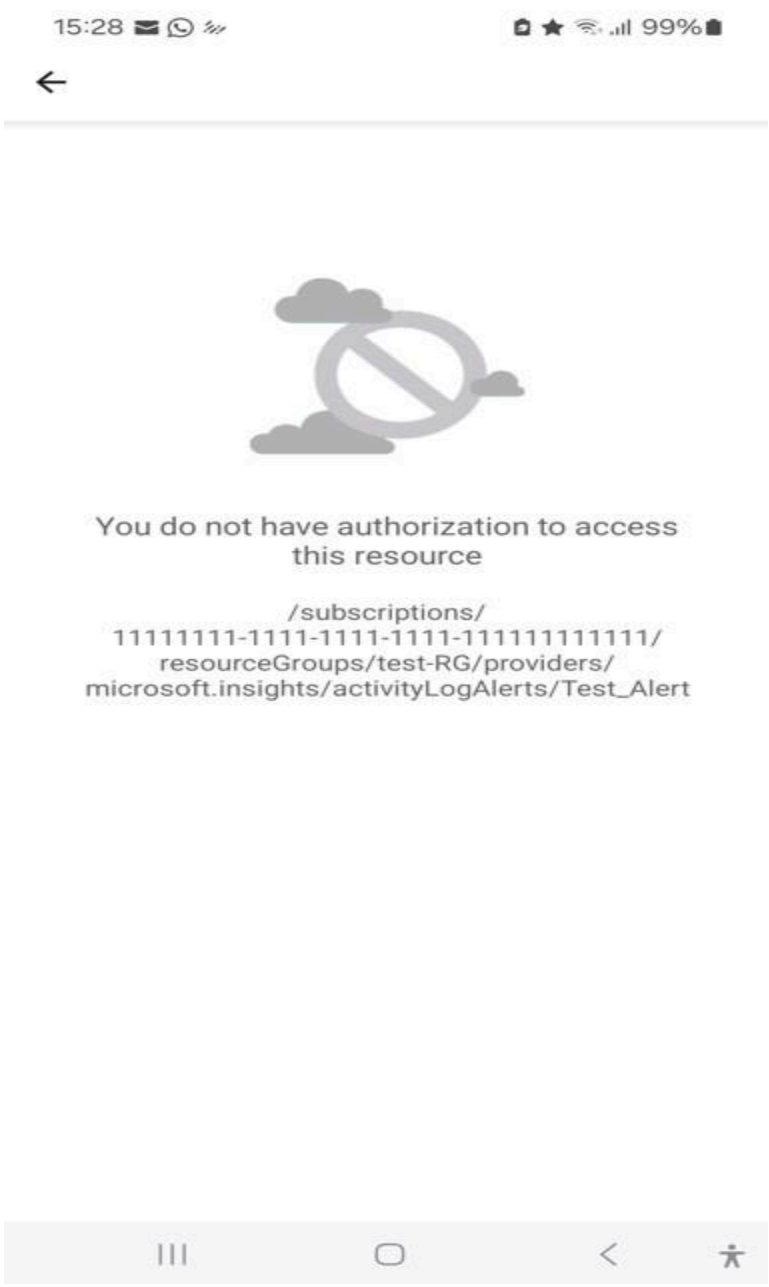


Screenshot of validation for successful test of Email & Azure App Notifications

Upon initiating the test, I successfully received both the test email and the mobile app Notification on my configured contact details. This crucial step confirmed that my Action Group was correctly configured and that notifications would be delivered reliably when an actual Service Health event occurs. While other testing options exist, this direct Action Group test provided immediate and conclusive validation of the notification pathway.



Screenshot of successful delivery of Email Test Alert Notification



Screenshot of successful delivery of Mobile App Test Alert Notification

Real-World Applications and Deeper Insights

Beyond the immediate goal of receiving alerts, this proactive monitoring setup has significant implications for real-world operations:

- **Proactive Issue Management:** Instead of waiting for users to report issues or manually discovering outages, I will now be informed immediately about service incidents. This shifts my operational stance from reactive to proactive.
- **Minimising Downtime Impact:** Rapid notification means I can initiate my incident response protocols faster. This significantly reduces the mean time to resolution (MTTR) for service-related disruptions, thereby minimising the impact on my applications and users.
- **Informed Strategic Planning:**
    - Alerts for 'Planned Maintenance' enable me to schedule my application maintenance windows or notify my users in advance, preventing unexpected disruptions.
    - 'Health Advisories' provide crucial lead time for deprecations or upcoming changes, allowing me to plan necessary architectural updates or migrations without last-minute scrambling.
- **Ensuring Compliance and SLAs:** By being immediately aware of service outages, I can accurately track downtime against my service level agreements (SLAs) with my customers or internal departments. This also aids in demonstrating due diligence for regulatory compliance requirements.
- **Enhancing Collaboration:** By adding relevant team members or stakeholders to the Action Group, I can ensure that everyone who needs to be aware of service health impacts receives timely information, fostering better cross-functional collaboration during incidents.
- **Foundation for Automation:** The Action Group is not limited to just sending notifications. It can trigger various automated actions, such as:
    - Calling an Azure Function to automatically log an incident in an IT Service Management (ITSM) system (e.g., ServiceNow, PagerDuty).
    - Triggering an Azure Logic App or Power Automate flow to send custom notifications to Microsoft Teams, Slack, or other communication channels.
    - Executing runbooks to attempt automated remediation for certain predictable resource health issues (though this is more advanced).

This proactive monitoring system provides not just alerts, but crucial intelligence that empowers faster, more informed decision-making and sets the stage for advanced automation in my cloud operations.

Conclusion

Through these steps, I have successfully implemented a proactive Azure monitoring solution using Service Health alerts. I am now equipped to receive automated notifications for service incidents, planned maintenance, and health advisories, enabling quicker awareness and response to potential impacts on my Azure resources. This setup is a vital component of my strategy for maintaining the availability and performance of my cloud environment.