Epic: Implement Efficient Cloud Operations Management for GCP Workloads

As a cloud operations team, we want to build a robust framework for managing, monitoring, and optimizing the operational aspects of our GCP infrastructure to ensure high availability, cost efficiency, and operational excellence.

User Story 1: Incident Response Automation

- **As a** Cloud Operations Engineer
- **I want to** automate incident response workflows for critical GCP services using Cloud Monitoring and Cloud Functions
- **So that I can** reduce the time to resolve incidents and minimize downtime

  Acceptance Criteria:

  - Incident response playbooks are integrated with Cloud Monitoring and automated using Cloud Functions.
  - Alerts trigger predefined actions such as restarting services or scaling instances automatically.
  - Incident logs are generated for every automated action taken, and notifications are sent to the operations team.
  - Alerts to have the right level of severity / impact defined with a predefined SLO for resolution.
  - Alerts to have self-healing capability and/or have self service functionality for app teams towards resolution.
  - Ops support team to receive only actionable / reactive alerts that need write access for resolution.
  - Consideration for Alert aggregation & SLO based alerting.
  - ‘Warning’ emails handled via pager duty or equivalent tool that can leveraged by SRE.

User Story 2: Health Monitoring of GCP Resources

- **As a** Site Reliability Engineer (SRE)
- **I want to** continuously monitor the health and performance of GCP resources (e.g., Compute Engine, Cloud SQL, GKE)
- **So that I can** detect and respond to performance degradation or resource failures before they affect users

Acceptance Criteria:

- Health checks are configured for all critical resources, including VMs, databases, and Kubernetes clusters.
- Metrics for CPU, memory, disk, and network utilization are visualized on a dashboard.
- Alerts are set for predefined thresholds, and notifications are sent via email or integration with incident management tools.

User Story 3: Autoscaling of Compute Resources

- **As a** Cloud Operations Engineer or Site Reliability Engineer (SRE)
- **I want to** implement autoscaling policies for my GCP workloads based on usage metrics
- **So that I can** ensure that my applications scale dynamically to meet demand while optimizing costs

Acceptance Criteria:

- Autoscaling policies are set up for Compute Engine instances and Kubernetes clusters.
- The system automatically scales up or down based on CPU and memory utilization thresholds.
- Notifications are sent when scaling events fail to occur, and resource usage approaches limits and capacity threshold.
- Alerts to have self-healing capability and/or have self service functionality for app teams towards resolution.
- Ops support team to receive only actionable / reactive alerts that need write access for resolution.

User Story 4: Incident Management and Root Cause Analysis

- **As a** Cloud Operations Engineer or Site Reliability Engineer (SRE)
- **I want to** manage incidents and conduct root cause analysis for outages or performance issues on GCP
- **So that I can** improve operational reliability and prevent future incidents

Acceptance Criteria:

- Incident management workflows are integrated with ServiceNow Incident Management tools

- Detailed logs and traces are available for each incident, enabling root cause analysis.
- Post-incident reviews and Post Mortems are generated with actionable recommendations for improvement.
- Performance metrics are available for key services and tooling available to search and filter logs by services, time and error type.
- Ability to understand and report dependencies between GCP resources/projects and applications for impact assessments.

User Story 5: Backup and Disaster Recovery Strategy

- **As a** Cloud Operations Engineer
- **I want to** implement a backup and disaster recovery (DR) strategy for critical GCP services
- **So that I can** ensure data integrity and service continuity in the event of failure

Acceptance Criteria:

- Backup policies are configured for storage, and Compute Engine instances.
- Disaster recovery plans are tested periodically
- Clearly defined and tested playbooks to be made available with less manual involvement & coordination.
- Clear recovery time (RTO) and recovery point objective is defined for each service.

User Story 6: Resource Limit Monitoring & Management

- **As a** Cloud Operations Engineer
- **I want to** implement a centralized access to all resource limits across my entire GCP projects
- **So that I can** monitor and request soft limit increase as well as plan around hard limits

Acceptance Criteria:

- Centralized view of resource limits across all resources, projects and regions
- Ability to alert before resource and service limits are breached
- Ability to auto request soft limit increases where appropriate