План оптимізації витрат в Google Cloud Platform (GCP)

1. Аналіз поточних витрат

- **Інструменти GCP для моніторингу:**
    1. **Billing:** Основний інструмент для перегляду та аналізу витрат.
        - **Billing reports:** Для детального аналізу витрат за різними критеріями (проєктами, сервісами, регіонами, тегами).
        - **Cost breakdown:** Розбивка витрат на складові (обчислення, зберігання, мережа тощо).
        - **Cost tables:** Перегляд цін на ресурси.
    2. **Cloud Billing API:** Для автоматизації збору та аналізу даних про витрати. Інтеграція з іншими системами для створення кастомних звітів та дашбордів. (Наприклад в Grafana чи власний n8n)
    3. **BigQuery Billing Export:** Експорт даних про витрати до BigQuery для детальнішого аналізу та створення складних запитів.
    4. **Looker Studio:** Візуалізація даних про витрати з BigQuery. Створення інтерактивних дашбордів для моніторингу.
- **Визначення основних джерел перевитрат:**
    1. **Аналіз Billing reports:** Виявлення сервісів, які генерують найбільші витрати.
    2. **Аналіз розбивки витрат:** Оцінка пропорції витрат на обчислення, зберігання, мережу.
    3. **Фільтрація за проєктами та тегами:** Виявлення проблемних проєктів або ресурсів.
    4. **Аналіз історії витрат:** Виявлення аномальних витрат.
    5. **Використання BigQuery:** Створення запитів для аналізу витрат за різними параметрами, виявлення тенденцій та пікових значень використання ресурсів.

2. Оптимізація обчислювальних ресурсів

- **Compute Engine:**
    - **Правильний підбір розміру VM:** Аналіз використання процесора, пам'яті та дискового простору для підбору оптимального типу VM. (зазвичай стартуємо з мінімального рекомендованого і розширюємось)
    - **Custom machine types:** Створення власних типів VM для точного узгодження з потребами додатків.
    - **Використання Committed Use Discounts (CUD):** Придбання довгострокових зобов’язань для зниження витрат на постійне використання VM !!!.
    - **Preemptible VMs / Spot VMs:** Використання для завдань, які можуть бути перервані (пакетна обробка, тестування) зі значною знижкою.
    - **Регулярне видалення неактивних VM:** Автоматизоване відстеження та видалення неактивних VM.

- **GKE (Google Kubernetes Engine):**
    - **Авто-скейлінг:** Налаштування HPA (Horizontal Pod Autoscaling) та VPA (Vertical Pod Autoscaling) для динамічного масштабування pod’ів та вузлів кластера.
    - **Правильний підбір розміру вузлів:** Аналіз використання ресурсів контейнерами для оптимального розміру вузлів.
    - **Node auto-provisioning:** Автоматичне масштабування кількості вузлів кластера залежно від потреб.
    - **Використання Spot VMs в node pools:** Зниження витрат на ноди за рахунок використання spot VM.
- **Cloud Run:**
    - **Оптимізація коду:** Зниження навантаження на процесор і пам'ять для зменшення часу виконання запитів.
    - **Масштабування до 0:** Автоматичне масштабування до 0 для сервісів, що не використовуються.
    - **Використання мінімальних версій container image:** Зменшення часу холодного запуску.

3. Оптимізація зберігання даних

- **Cloud Storage:**
    - **Використання Storage Classes:** Перехід до Nearline, Coldline або Archive для рідко використовуваних даних. Standard використовуємо тільки для зберігання даних з дуже великою частотою запитів. Навіть для media зазвичай підходить Nearline.
    - **Object Lifecycle Management:** Автоматизація переміщення об’єктів між storage classes на основі їх віку.
    - **Компресія об'єктів:** Зменшення розміру об'єктів для зниження вартості зберігання та трафіку.
- **BigQuery:**
    - **Partitioning & Clustering:** Розбиття таблиць на секції та кластери для оптимізації запитів.
    - **Стиснення даних:** Використання формату стиснення Parquet або Avro.
    - **Правильний підбір типів даних:** Зменшення витрат на зберігання шляхом вибору оптимального типу даних.
    - **Уникання SELECT *:** Запит тільки потрібних стовпчиків.
    - **Оптимізація SQL запитів:** Перевірка та оптимізація запитів для зменшення часу виконання та обсягу оброблених даних.
- **Persistent Disks:**
    - **Використання SSD для критичних навантажень:** Економія та використання HDD для менш важливих даних.
    - **Оптимізація розміру дисків:** Аналіз фактичного використання для запобігання виділення зайвого дискового простору.

- **Snapshot для резервного копіювання:** Забезпечення ефективного резервного копіювання з мінімальними витратами.
- **Використання Regional PD для високої доступності:** Забезпечення високої доступності з оптимізацією витрат.

4. Оптимізація мережевих витрат

- **Трафік між сервісами:**
  - **Використання внутрішньої мережі:** Передача даних через внутрішню мережу VPC для зменшення витрат на зовнішній трафік. (в рамках проєкт має все ходити по внутрішньому VPC)
  - **Мінімізація трафіку:** Оптимізація додатків для зменшення обсягу даних, які передаються.
- **Трафік до зовнішніх систем:**
  - **Cloud CDN:** Кешування контенту на edge-серверах для зменшення трафіку до вихідних джерел.
  - **VPC Service Controls:** Забезпечення безпеки мережі та запобігання випадковим передачам даних.
  - **Оптимізація API:** Запити на вибірку тільки необхідних даних через API.
  - **Компресія даних:** Використання компресії для зменшення обсягу даних, що передаються через мережу.
- **Використання приватних з'єднань:**
  - **Cloud Interconnect / Cloud VPN:** Пряме з'єднання з on-premise інфраструктурою замість публічного інтернету.

5. Резервування та знижки!!!!

- **Committed Use Discounts (CUD):**
  - **Аналіз постійного використання:** Визначення ресурсів, які можна зарезервувати на 1 або 3 роки.
  - **Придбання CUD:** Забезпечення значної знижки на постійне використання обчислювальних ресурсів.
- **Sustained Use Discounts (SUD):**
  - **Аналіз використання VM:** Автоматичне застосування знижок за тривале використання VM.
- **Preemptible VMs / Spot VMs:**
  - **Використання для некритичних завдань:** Забезпечення значної економії на завданнях, що не потребують безперервної роботи.
  - **Регулярне оновлення завдань:** Здатність до відновлення після зупинки VM.

6. Автоматизація та контроль

- **Budgets:**
    - **Налаштування бюджетів:** Встановлення лімітів витрат для кожного проєкту.
    - **Відстеження витрат:** Моніторинг витрат у порівнянні з встановленими бюджетами.
- **Alerts:**
    - **Налаштування сповіщень:** Отримання повідомлень про перевищення бюджетів.
    - **Автоматичні дії:** Запуск автоматизованих дій при перевищенні витрат (наприклад, вимкнення ресурсів).
- **Recommender:**
    - **Використання рекомендацій:** Перевірка пропозицій Recommender щодо оптимізації витрат.
    - **Впровадження рекомендацій:** Актуалізація розмірів ресурсів, використання CUD/SUD.
- **Policy Controller:**
    - **Реалізація політик:** Автоматизований контроль за витратами на ресурси через встановлення обмежень.
- **Terraform / Deployment Manager:**
    - **Інфраструктура як код:** Автоматизація розгортання ресурсів з оптимізацією витрат.
- **Cloud Functions / Cloud Scheduler:**
    - **Автоматизація:** Регулярне виконання скриптів для управління бюджетами і ресурсами.

Висновок

Цей план забезпечує структурований підхід до оптимізації витрат у GCP. Його реалізація включатиме аналіз поточних витрат, впровадження оптимізації обчислювальних ресурсів, зберігання даних та мережі, використання знижок, автоматизацію процесів та постійний контроль.

Роль DevOPS в забезпеченні постійного моніторингу та оптимізації витрат в GCP потенційно може принести економію від 60% до 80% на регулярному використанні ресурсів.

Також, для забезпечення найбільшої ефективності, потрібно працювати із Legacy застосунками та переводити їх на мікросервіси в GKE та/або використовувати сервіси GCP такі як SQL, CloudRun, BigQuery.

Зберігання даних та використання для, наприклад, медіа SSD диски не є хорошою практикою. Статичні данні потрібно переводити на бакети з відповідним вибором класу. Робота з даними має мати регулярну основу.