### **Step-by-Step Guide to Deploy Budget Management System on Google Cloud**

This guide will walk you through **cloning, configuring, and deploying** the **Budget Management System** on **Google Cloud** using **Terraform**.

## **🔹 Step 1: Set Up Your Environment**

### **1.1 Install Required Tools**

Ensure you have the following installed: ✅ **Google Cloud SDK** → Install Guide  
✅ **Terraform** → Install Guide  
✅ **PostgreSQL Client** → Install using:

sudo apt update && sudo apt install -y postgresql-client

## **🔹 Step 2: Clone the Project Repository**

git clone https://github.com/srivats09-git/budget-management-system.git

cd budget-management-system

## **🔹 Step 3: Set Up Google Cloud Project**

### **3.1 Authenticate with Google Cloud**

gcloud auth login

Follow the on-screen instructions to authenticate.

### **3.2 Set Your GCP Project**

Replace your-project-id with your **actual** GCP project ID (alert-synapse-450214-e2).

gcloud config set project alert-synapse-450214-e2

gcloud config get-value project

Confirm it shows:

alert-synapse-450214-e2

### **3.3 Enable Required GCP Services**

gcloud services enable cloudsql.googleapis.com run.googleapis.com secretmanager.googleapis.com iam.googleapis.com cloudresourcemanager.googleapis.com

## **🔹 Step 4: Configure Terraform**

### **4.1 Navigate to Terraform Directory**

cd budget-management/terraform

### **4.2 Update terraform.tfvars**

Edit the file:

nano terraform.tfvars

Ensure it contains:

hcl

project\_id = "alert-synapse-450214-e2"

region = "us-west2"

db\_password = "StrongRandomPassword123!" # Change this to a secure password

Save (CTRL + X, then Y, then ENTER).

### **4.3 Remove Any Old State Files**

rm -rf .terraform terraform.tfstate\*

### **4.4 Initialize Terraform**

terraform init

terraform refresh

### **4.5 Apply Terraform to Create Infrastructure**

terraform apply -auto-approve

This will: ✅ Create a **Cloud SQL PostgreSQL instance**✅ Deploy **Cloud Run**✅ Set up **IAM roles and permissions**

After completion, Terraform will output:

makefile

Outputs:

cloud\_run\_url = "https://budget-management-xyz.run.app"

db\_instance\_name = "budget-db"

🚀 **Save this Cloud Run URL**.

## **🔹 Step 5: Store Database Password in Secret Manager**

echo "StrongRandomPassword123!" | gcloud secrets create db-password --data-file=-

gcloud secrets add-iam-policy-binding db-password \

--member="serviceAccount:budget-app-sa@alert-synapse-450214-e2.iam.gserviceaccount.com" \

--role="roles/secretmanager.secretAccessor"

## **🔹 Step 6: Deploy Flask Application on Cloud Run**

### **6.1 Build and Push Docker Image**

gcloud builds submit --tag gcr.io/alert-synapse-450214-e2/budget-app

### **6.2 Deploy to Cloud Run**

gcloud run deploy budget-management \

--image gcr.io/alert-synapse-450214-e2/budget-app \

--platform managed \

--region us-west2 \

--allow-unauthenticated

🚀 **Your API is now live at the Cloud Run URL!**

## **🔹 Step 7: Verify Deployment**

### **7.1 Check Cloud SQL Instance**

gcloud sql instances list

Expected output:

pgsql

NAME DATABASE\_VERSION LOCATION

budget-db POSTGRES\_13 us-west2

### **7.2 Check Cloud Run Service**

gcloud run services list

Expected output:

nginx

NAME REGION URL

budget-management us-west2 https://budget-management-xyz.run.app

### **7.3 Test API**

curl -X POST -H "Content-Type: application/json" \

-d '{"message": "add aop name \"FY2025\" amount 1000000"}' \

https://budget-management-xyz.run.app/api/chat

✅ Expected response:

json

{"response": "AOP FY2025 created with amount $1,000,000"}

## **🔹 Step 8: Monitor and Debug**

### **8.1 View Logs**

gcloud run logs read budget-management

### **8.2 SSH into Cloud SQL**

gcloud sql connect budget-db --user=postgres

## **🎯 Summary of Deployment Steps**

| **Step** | **Command** |
| --- | --- |
| **1. Clone the project** | git clone https://github.com/srivats09-git/budget-management-system.git && cd budget-management-system |
| **2. Set GCP Project** | gcloud config set project alert-synapse-450214-e2 |
| **3. Enable Cloud APIs** | gcloud services enable cloudsql.googleapis.com run.googleapis.com secretmanager.googleapis.com |
| **4. Configure Terraform** | cd budget-management/terraform && nano terraform.tfvars |
| **5. Initialize & Apply Terraform** | terraform init && terraform apply -auto-approve |
| **6. Store DB Password in Secret Manager** | `echo "password" |
| **7. Build & Deploy App on Cloud Run** | gcloud builds submit --tag gcr.io/alert-synapse-450214-e2/budget-app → gcloud run deploy budget-management --image gcr.io/alert-synapse-450214-e2/budget-app --platform managed --region us-west2 --allow-unauthenticated |
| **8. Verify Deployment** | curl -X POST -H "Content-Type: application/json" -d '{"message": "add aop name \"FY2025\" amount 1000000"}' https://budget-management-xyz.run.app/api/chat |
| **9. View Logs** | gcloud run logs read budget-management |