Homework_Lesson_37

Задание:

1. Создать конфигурацию для GCP Storage. Добавить модуль, который будет отвечать за доп. параметры бакета (имя, размер, файлы для закачки)
2. Используйте созданный модуль в основной конфигурации для развертывания инфраструктуры.
3. Настройте удаленное хранение tfstate в Terraform, используя GCP Storage в связке с CloudSQL
4. Перенесите свою существующую инфраструктуру на удаленное хранение стейта.
5. Создать в ручном режиме любой ресурс (лучше всего бакет либо ВМ) и импортируйте его в свою конфигурацию.

Создадим бакет для хранения состояния Terraform (tfstate) и добавим создание Cloud SQL экземпляра

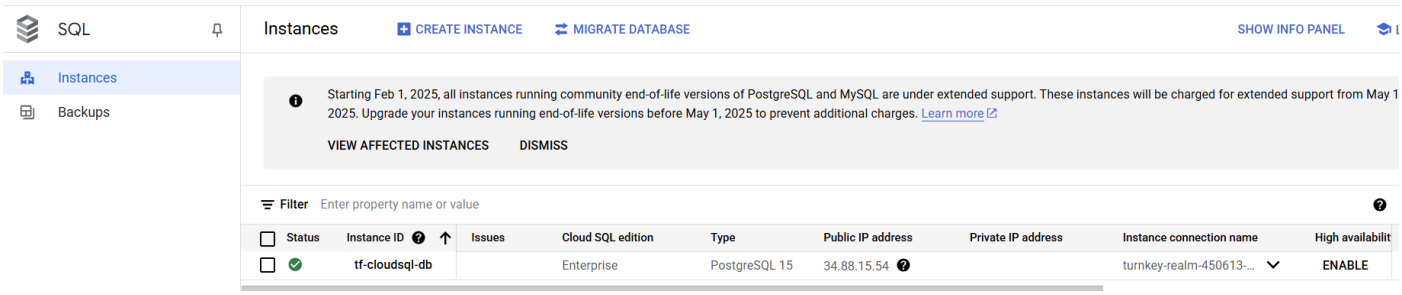
```
gcpa4607@tms:~/HW37$ terraform apply
Terraform used the selected providers to generate the following execution plan. Resource actions are
indicated with the following symbols:
 + create
Terraform will perform the following actions:
 # google_sql_database.appdb will be created
 + resource "google_sql_database" "appdb" {
    + charset       = (known after apply)
    + collation     = (known after apply)
    + deletion_policy = "DELETE"
    + id            = (known after apply)
    + instance      = "tf-cloudsql-db"
    + name          = "appdb"
    + project       = (known after apply)
    + self_link     = (known after apply)
  }

 # google_sql_database_instance.db will be created
 + resource "google_sql_database_instance" "db" {
    + available_maintenance_versions = (known after apply)
    + connection_name            = (known after apply)
    + database_version           = "POSTGRES_15"
    + deletion_protection        = true
    + dns_name                   = (known after apply)
    + encryption_key_name        = (known after apply)
    + first_ip_address           = (known after apply)
    + id                         = (known after apply)
    + instance_type              = (known after apply)
    + ip_address                 = (known after apply)
    + maintenance_version        = (known after apply)
    + master_instance_name       = (known after apply)
    + name                       = "tf-cloudsql-db"
    + private_ip_address         = (known after apply)
    + project                    = (known after apply)
    + psc_service_attachment_link  = (known after apply)
    + public_ip_address          = (known after apply)
    + region                     = "europe-north1"
    + self_link                  = (known after apply)
    + server_ca_cert             = (known after apply)
    + service_account_email_address  = (known after apply)

    + replica_configuration (known after apply)
```

```
    + settings {
        + activation_policy    = "ALWAYS"
        + availability_type    = "ZONAL"
        + connector_enforcement = (known after apply)
        + disk_autoresize      = true
        + disk_autoresize_limit = 0
        + disk_size            = (known after apply)
        + disk_type            = "PD_SSD"
        + pricing_plan         = "PER_USE"
        + tier                 = "db-f1-micro"
        + user_labels          = (known after apply)
        + version              = (known after apply)

        + backup_configuration (known after apply)

        + ip_configuration (known after apply)

        + location_preference (known after apply)
      }
  }

# google_sql_user.usersvc will be created
+ resource "google_sql_user" "usersvc" {
    + host                 = (known after apply)
    + id                   = (known after apply)
    + instance             = "tf-cloudsql-db"
    + name                 = "appuser"
    + password             = (sensitive value)
    + project              = (known after apply)
    + sql_server_user_details = (known after apply)
  }

# module.app_bucket.google_storage_bucket.bucket will be created
+ resource "google_storage_bucket" "bucket" {
    + force_destroy            = true
    + id                       = (known after apply)
    + labels                   = (known after apply)
    + location                 = "EUROPE-NORTH1"
    + name                     = "my-unique-app-bucket-1234"
    + project                  = (known after apply)
    + public_access_prevention    = (known after apply)
    + self_link                = (known after apply)
    + storage_class            = "STANDARD"
    + uniform_bucket_level_access = (known after apply)
    + url                      = (known after apply)

    + soft_delete_policy (known after apply)

    + versioning (known after apply)

    + website (known after apply)
  }
```

```
# module.app_bucket.google_storage_bucket_object.initial["welcome.txt"] will be created
+ resource "google_storage_bucket_object" "initial" {
    + bucket       = "my-unique-app-bucket-1234"
    + content      = (sensitive value)
    + content_type = (known after apply)
    + crc32c       = (known after apply)
    + detect_md5hash = "different hash"
    + id           = (known after apply)
    + kms_key_name = (known after apply)
    + md5hash      = (known after apply)
    + media_link   = (known after apply)
    + name         = "welcome.txt"
    + output_name  = (known after apply)
    + self_link    = (known after apply)
    + source       = "welcome.txt"
    + storage_class = (known after apply)
  }

Plan: 5 to add, 0 to change, 0 to destroy.
```

```
module.app_bucket.google_storage_bucket.bucket: Creating...
google_sql_database_instance.db: Creating...
module.app_bucket.google_storage_bucket.bucket: Creation complete after 2s [id=my-unique-app-bucket-1234]
module.app_bucket.google_storage_bucket_object.initial["welcome.txt"]: Creating...
module.app_bucket.google_storage_bucket_object.initial["welcome.txt"]: Creation complete after 0s [id=my-
unique-app-bucket-1234-welcome.txt]
google_sql_database_instance.db: Still creating... [10s elapsed]
```

```
Apply complete! Resources: 5 added, 0 changed, 0 destroyed.

Outputs:

bucket_name = "my-unique-app-bucket-1234"
bucket_url = "gs://my-unique-app-bucket-1234"
db_connection_name = "turnkey-realm-450613-c7:europe-north1:tf-cloudsql-db"
gcpa4607@tms:~/HW37$
```



В процессе Terraform создал бакет и загрузил в него начальный welcome.txt