IMPLEMENTATION STEPS AND REFERENCE SCREENSHOTS:

PART1:

1. Create AWS and Azure accounts.
2. Create IAM user in AWS with S3 full access and downloaded the csv file
3. Now in the google cloud shell > upload the PART1 file using the wget command and make all the sh files executable using chmod commands
4. Now prepare the AWS environment  
   A screen shot of a computer program

   Description automatically generated
5. Set the AWS credentials and assign the gcp project id to the variable and set the project in the google cloud shell  
   A screen shot of a computer program

   Description automatically generated
6. Execute the commands to enable the Kubernetes, Container Registry, and Cloud SQL APIs  
   enable these, for terraform to communicate to the cloud provider using API from the cloudshell.  
   A black screen with many small colored text

   Description automatically generated with medium confidence
7. Now goto the terraform files and start running the terraform commands to initialize and deploy the resources.  
   init, plan, apply  
   A screen shot of a computer program

   Description automatically generated

A black and white screen

Description automatically generated

PART2:

1. On AWS side create a new IAM user with s3 full access to access this bucket from the GCP.

A screenshot of a computer

Description automatically generated

1. Now create app user with pwd in SQL server at GCP  
   A screenshot of a computer

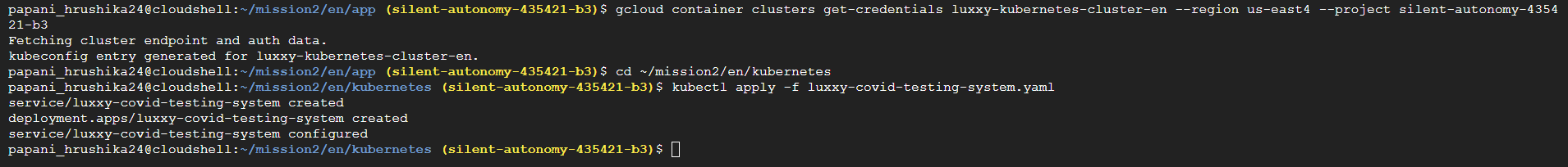
   Description automatically generated
2. Now connect to the cloud shell(GCP). And upload the PART2 folder.  
   here we are using the SQL server instead of running the database on the virtual machine as we are exploring the automation, Easy scalable, high availability, no need a database team to automatic and less maintenance(no patching), more secure.  
   mainly this architecture is suitable for major workloads and unknown incoming traffic(not suitable here for small project as it is high cost – just for the hands on).
3. Follow the commands as shown below to first connect to the SQL server > use the dbcovidtesting database inside the SQL server > source command helps us creating a table inside the database > you can also use desc command to see all the columns created inside the database  
   A screenshot of a computer program

   Description automatically generated  
   A screen shot of a computer screen

   Description automatically generated
4. Make sure to edit the Kubernetes deployment file (**luxxy-covid-testing-system.yaml**) and update the variables below  
   with your <PROJECT\_ID> on the Google Container Registry path,   
   AWS Bucket name,   
   AWS Keys (**Access key ID** and **Secret access key**),  
   Cloud SQL Database Private IP.  
   A screenshot of a computer

   Description automatically generated
5. Now connect to the GKE via console  
   click on connect and run the command in the clous shell.
6. Deploy the application Luxxy in the Cluster using below commands

cd ~/mission2/en/kubernetes

kubectl apply -f luxxy-covid-testing-system.yaml  


1. Under **GKE** > **Workloads** > **Exposing Services**, get the application Public IP  
   A screenshot of a computer

   Description automatically generated  
   and your application is here!

PART 3:

1. upload the PART3 folder in GCP in the home directory.
2. Connect to the SQL server with the public IP and change to the required database of the SQL server and import the dump on the cloud SQL.  
   A screen shot of a computer

   Description automatically generated
3. You can check if the data got imported correctly.  
   A screen shot of a computer

   Description automatically generated  
   here pdf column stores only the metadata of the PDF name
4. Now in AWS:
5. Connect to the shell and download the mission3 file which consists of the PDF files  
   A screenshot of a computer

   Description automatically generated
6. Navigate to the pdf files folder and use s3 sync command to sync all the files to the s3.  
   A screen shot of a computer screen

   Description automatically generated
7. Now navigate to your application.  
   you can view all the details under the view guest results.  
   and cloud able to add the details by clicking on the add guest result.  
   A screenshot of a hotel

   Description automatically generated  
   A screenshot of a computer

   Description automatically generated  
   A screenshot of a computer

   Description automatically generated
8. A screenshot of a computer

   Description automatically generated
9. You can see the link for the PDF file for S3 bucket > filename > file is signed with the signature key > only the application has the access to the S3 bucket, has access to the PDF files.  
   