## 6. Application Development with Cloud Run

* This course introduces you to fundamentals, practices, capabilities and tools applicable to modern cloud-native application development using Google Cloud Run. learn how to on Google Cloud using Cloud Run.design, implement, deploy, secure, manage, and scale applications

## 6.1. Introduction to Cloud Run

* What is Cloud Run 5 minutes - <https://youtu.be/cTjzOfFL-Ow>
  + Graphical user interface, text

    Description automatically generatedGraphical user interface

    Description automatically generated with low confidence
  + Diagram

    Description automatically generatedGraphical user interface, timeline

    Description automatically generated
  + Chart, diagram, funnel chart

    Description automatically generated
* Typical use cases 6 minutes - <https://youtu.be/fwaOr4VW4IM>
  + Diagram

    Description automatically generated with medium confidenceDiagram

    Description automatically generated
  + Diagram

    Description automatically generatedDiagram

    Description automatically generated
  + Diagram

    Description automatically generatedGraphical user interface, application, Teams

    Description automatically generated
  + Graphical user interface, text, application, chat or text message

    Description automatically generated
* High availability 4 minutes - <https://youtu.be/P5CmiXKmd4g>
  + Graphical user interface, text, application

    Description automatically generatedTimeline

    Description automatically generated
  + Diagram

    Description automatically generatedDiagram

    Description automatically generated
  + Chart, waterfall chart

    Description automatically generated
* Concerns about serverless 2 minutes - <https://youtu.be/mq4CBrNcnrk>
  + A picture containing text

    Description automatically generatedGraphical user interface, text, application, chat or text message

    Description automatically generated
  + Graphical user interface, text, application

    Description automatically generated
* Positioning on Google Cloud 1 minute - <https://youtu.be/6sUaQIBolXE>
  + Graphical user interface

    Description automatically generated

## 6.2. Understanding Cloud Run

* Introduction to containers 3 minutes - <https://youtu.be/eqigDjhJ9II>
  + Graphical user interface, text, application, chat or text message

    Description automatically generatedGraphical user interface, application

    Description automatically generated
  + Diagram

    Description automatically generated with medium confidenceDiagram

    Description automatically generated
  + Chart

    Description automatically generatedGraphical user interface, application

    Description automatically generated
* Running your application on Cloud Run 5 minutes - <https://youtu.be/ZfT7J8D0I28>
  + Diagram

    Description automatically generatedText

    Description automatically generated with medium confidence
  + Diagram

    Description automatically generatedGraphical user interface, application

    Description automatically generated
  + Diagram, timeline

    Description automatically generatedChart

    Description automatically generated with medium confidence
  + Text

    Description automatically generatedDiagram

    Description automatically generated
  + Graphical user interface, application

    Description automatically generatedGraphical user interface, application

    Description automatically generated

### Hello Cloud Run [APPRUN] - https://www.cloudskillsboost.google/course\_sessions/888752/labs/187125

* + Objectives
    - Enable the Cloud Run API.
    - Create a simple Node.js application that can be deployed as a serverless, stateless container.
    - Containerize your application and upload to Container Registry (now called "Artifact Registry.")
    - Deploy a containerized application on Cloud Run.
    - Delete unneeded images to avoid incurring extra storage charges.
  + Overview
    - Cloud Run is a managed compute platform that enables you to run stateless containers that are invocable via HTTP requests. Cloud Run is serverless: it abstracts away all infrastructure management, so you can focus on what matters most — building great applications.
    - Cloud Run is built from Knative, letting you choose to run your containers either fully managed with Cloud Run, or in your Google Kubernetes Engine cluster with Cloud Run on GKE.
    - The goal of this lab is for you to build a simple containerized application image and deploy it to Cloud Run.
  + Reference
    - Basic Linux Commands
    - Below you will find a reference list of a few very basic Linux commands which may be included in the instructions or code blocks for this lab.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Command -->** | **Action** | **.** | **Command -->** | **Action** |
| **mkdir** (*make directory*) | create a new folder | . | **cd** (*change directory*) | change location to another folder |
| **ls** (*list* ) | list files and folders in the directory | . | **cat** (*concatenate*) | read contents of a file without using an editor |
| **apt-get update** | update package manager library | . | **ping** | signal to test reachability of a host |
| **mv** (*move* ) | moves a file | . | **cp** (*copy*) | makes a file copy |
| **pwd** (*present working directory* ) | returns your current location | . | **sudo** (*super user do*) | gives higher administration privileges |

* + Task 1. Enable the Cloud Run API and configure your Shell environment:
    - From Cloud Shell, enable the Cloud Run API :
      * gcloud services enable run.googleapis.com
    - Set the compute region
      * gcloud config set compute/region us-central1
    - Create a LOCATION environment variable
      * LOCATION="us-central1"
  + Task 2. Write the sample application
    - In this task, you will build a simple express-based NodeJS application which responds to HTTP requests.
    - In Cloud Shell create a new directory named helloworld, then move your view into that directory:
      * mkdir helloworld && cd helloworld
    - Create a package.json file, then add the following content to it:
      * {
      * "name": "helloworld",
      * "description": "Simple hello world sample in Node",
      * "version": "1.0.0",
      * "main": "index.js",
      * "scripts": {
      * "start": "node index.js"
      * },
      * "author": "Google LLC",
      * "license": "Apache-2.0",
      * "dependencies": {
      * "express": "^4.17.1"
      * }
      * }
    - Most importantly, the file above contains a start script command and a dependency on the Express web application framework.
    - Next, in the same directory, create a index.js file, and copy the following lines into it:
      * const express = require('express');
      * const app = express();
      * const port = process.env.PORT || 8080;
      * app.get('/', (req, res) => {
      * const name = process.env.NAME || 'World';
      * res.send(`Hello ${name}!`);
      * });
      * app.listen(port, () => {
      * console.log(`helloworld: listening on port ${port}`);
      * });
    - This code creates a basic web server that listens on the port defined by the PORT environment variable. Your app is now finished and ready to be containerized and uploaded to Container Registry.
    - You can use many other languages to get started with Cloud Run. You will find instructions for Go, Python, Java, PHP, Ruby, Shell scripts, and others here: https://cloud.google.com/run/docs/quickstarts/build-and-deploy
  + Task 3. Containerize your app and upload it to Artifact Registry
    - To containerize the sample app, create a new file named Dockerfile in the same directory as the source files, and add the following content :
      * # Use the official lightweight Node.js 12 image.
      * # https://hub.docker.com/\_/node
      * FROM node:12-slim
      * # Create and change to the app directory.
      * WORKDIR /usr/src/app
      * # Copy application dependency manifests to the container image.
      * # A wildcard is used to ensure copying both package.json AND package-lock.json (when available).
      * # Copying this first prevents re-running npm install on every code change.
      * COPY package\*.json ./
      * # Install production dependencies.
      * # If you add a package-lock.json, speed your build by switching to 'npm ci'.
      * # RUN npm ci --only=production
      * RUN npm install --only=production
      * # Copy local code to the container image.
      * COPY . ./
      * # Run the web service on container startup.
      * CMD [ "npm", "start" ]
    - Now, build your container image using Cloud Build by running the following command from the directory containing the Dockerfile. (Note the $GOOGLE\_CLOUD\_PROJECT environmental variable in the command, which contains your lab's Project ID):
      * gcloud builds submit --tag gcr.io/$GOOGLE\_CLOUD\_PROJECT/helloworld
    - Cloud Build is a service that executes your builds on GCP. It executes a series of build steps, where each build step is run in a Docker container to produce your application container (or other artifacts) and push it to Cloud Registry, all in one command.
    - Once pushed to the registry, you will see a SUCCESS message containing the image name (gcr.io/[PROJECT-ID]/helloworld). The image is stored in Artifact Registry and can be re-used if desired.
    - List all the container images associated with your current project using this command :
      * gcloud container images list
    - To run and test the application locally from Cloud Shell, start it using this standard docker command :
      * docker run -d -p 8080:8080 gcr.io/$GOOGLE\_CLOUD\_PROJECT/helloworld
    - In the Cloud Shell window, click on Web preview and select Preview on port 8080.
      * If the docker command cannot pull the remote container image then try running this :
        + gcloud auth configure-docker
  + Task 4. Deploy to Cloud Run
    - Deploying your containerized application to Cloud Run is done using the following command adding your Project-ID:
      * gcloud run deploy --image gcr.io/$GOOGLE\_CLOUD\_PROJECT/helloworld --allow-unauthenticated --region=$LOCATION
    - The allow-unauthenticated flag in the command above makes your service publicly accessible.
    - When prompted:
      * confirm the service name by pressing Enter
    - On success, the command line displays the service URL & visit in browser
    - Congratulations! You have just deployed an application packaged in a container image to Cloud Run. Cloud Run automatically and horizontally scales your container image to handle the received requests, then scales down when demand decreases. In your own environment, you only pay for the CPU, memory, and networking consumed during request handling.
    - For this lab you used the gcloud command-line. Cloud Run is also available via Cloud Console.
    - From the Navigation menu, in the Compute section, click Cloud Run and you should see your helloworld service listed:
      * Graphical user interface, text, application, email

        Description automatically generated
  + Task 5. Clean up
    - While Cloud Run does not charge when the service is not in use, you might still be charged for storing the built container image.
    - You can either decide to delete your GCP project to avoid incurring charges, which will stop billing for all the resources used within that project, or simply delete your helloworld image using this command :
      * gcloud container images delete gcr.io/$GOOGLE\_CLOUD\_PROJECT/helloworld
    - To delete the Cloud Run service, use this command :
      * gcloud beta run services delete helloworld

### Autoscaling and on-demand containers 5 minutes - https://youtu.be/qAYurXK8N3c

* + Diagram

    Description automatically generated Diagram

    Description automatically generated
  + Diagram

    Description automatically generated Timeline

    Description automatically generated
  + Graphical user interface, application

    Description automatically generated Graphical user interface, application, website

    Description automatically generated
  + Graphical user interface

    Description automatically generated A picture containing graphical user interface

    Description automatically generated
  + Graphical user interface

    Description automatically generated
* Quiz
  + Pricing on Cloud Run is determined by: (select all that apply)
    - A small fee per one million requests served
      * Correct! You will, however, get your first 2 million requests free per month. If you go over that, you will pay according to your “tier,” which is location-dependent.
    - Network egress from the container
      * Correct! Data egress over the network incurs charges.
    - The system resources a container uses while handling a request or during startup/shutdown
      * Correct! The price of container time depends upon CPU and memory. A container with more vCPU and memory is more expensive to run.
  + Cloud Run can only pull images from:
    - Artifact Registry
      * Correct! If you use other container registries, you will need to push them to Container Registry first before you use them.
  + You should complete all work before you return a response to a web request for the following reason:
    - Because Cloud Run shuts down and throttles idle containers so any tasks you run in the background after returning a response might not finish.
      * Correct! If you return your HTTP response before all tasks are completed, you run the risk of losing your data when the container is throttled.