**Sprint 3**

This Sprint covers the following topics:

* Building apps with GCP
* Orchestrating containers with Kubernetes

**1. Logging into the GCP web console and checking the current billing status**

Goals:

* Understand how to access the GCP web console.
* Verify how much money you have spent already on your GCP account.

Tasks:

* Use your web browser and provided credentials to log into your GCP account.
* Navigate to the GCP dashboard and see how much money you have spent on the GCP services so far.
* Go to the billing service and find a billing forecast so you can manage your GCP spending.

**2. Publishing a Docker image of the application into Google Container Registry (GCR)**

Goals:

* Know how to publish a docker image to GCR.
* Gain an understanding of how to manage access to your images stored in GCR.

Tasks:

* Push the Docker image you created into GCR.
* Use the GCP web console or command line to verify the image has been published to GCR.
* Share your image's location with a team member; instruct them to download and run a container using the image.

**3. Publish a Docker image from a Jenkins server to GCR**

Goals:

* Allow the Jenkins Server to publish to GCR.
* Modify your existing Jenkins job to publish the Docker image into GCR on every commit.

Tasks:

* Create a GCP IAM service account for the Jenkins with the required access.
* Register the service account credentials as a Jenkins credential.
* Extend your existing Jenkins pipeline to publish the Docker image into GCR using the service account credentials from the previous task.

**4. Creating GCP cluster**

Goals:

* Create Google Kubernetes Engine (GKE) cluster that you will use to run your application.
* Install kubectl to communicate with a Kubernetes cluster.

Tasks:

* Create new GKE cluster using the GCP web console or the gcloud command line.
* The new cluster should have at least 2 compute nodes.
* Install kubectl tool.
* Use gcloud command line tool to authenticate kubectl with the cluster.
* Verify that you can connect to the GKE cluster using kubectl.

**5. Deploy your application to the GKE cluster**

Goals:

* Deploy the docker image from GCR to the Kubernetes cluster.
* Verify that your application has been successfully deployed.
* Access the application via a browser (optional).

Tasks:

* Create a deployment manifest defining a replica set for the application.
* Specify that you would like to run two replicas of the application.
* In the manifest, refer to the Docker image in GCR.
* Use kubectl to apply the deployment manifest.
* Use kubectl to verify that the application has been successfully deployed into GKE.
* Create a load balancer manifest targeting the replicas (optional).
* Use kubectl to apply the load balancer manifest (optional).