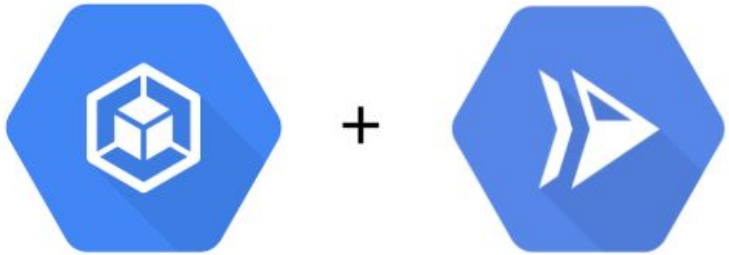
Understanding Google Cloud Platform

Day - 3

Deployment with Cloud build and Cloud Run

Agenda

- Recap of Day 2 (Cloud SQL, Instances, Auth Proxy, gcloud)
- Intro to Cloud Build
- Configuring the flask app for deployment
- Create Docker image with Cloud Build and Artifact Registry
- Deploying with Cloud Run
- Cloud Run vs. Cloud Build
- What’s next?



What is Cloud Build?

Cloud Build is a fully managed CI/CD platform that lets you build, test, and deploy applications at scale.

Key Features:

- Custom build pipelines using YAML
- Integration with GitHub, GitLab, Bitbucket
- Support for Docker, Maven, Gradle, Bazel, and more

Use Cases:

- Automate builds and tests for PRs
- Build container images and push to **Artifact Registry** or Docker Hub
- Deploy to Cloud Run, App Engine, or GKE

Configuring the flask app for deployment

A Flask blog application with MySQL database for data and uploading Image files.

Major changes from local to cloud:

- SQLite → PyMySQL
- Static Folder → gcloud bucket
- Server run with Gunicorn

```python
def upload_file(s_file, d_file_name):
    client = storage.Client()
    bucket = client.bucket(DB_BUCKET)
    blob = bucket.blob(d_file_name)
    blob.upload_from_file(s_file)
    return blob.public_url
```

Function to upload the binary file to cloud storage and return public URL of image

```python
DB_HOST = os.environ.get('DB_HOST')
DB_USER = os.environ.get('DB_USER')
DB_PASSWORD = os.environ.get('DB_PASSWORD')
DB_NAME = os.environ.get('DB_NAME')
DB_BUCKET = os.environ.get('DB_BUCKET')

def get_connection():
    return pymysql.connect(
        unix_socket = DB_HOST,
        user = DB_USER,
        password = DB_PASSWORD,
        database = DB_NAME,
        cursorclass = pymysql.cursors.DictCursor
    )
```

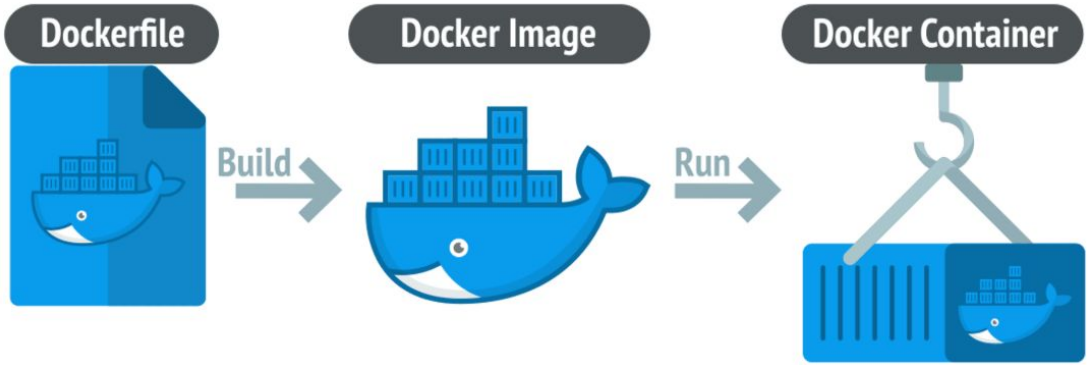
Function to configure cloud storage parameters using `pymysql` through environment variables

Creating Docker Image and containerization

A **Docker image** is a read-only, self-contained file that packages an application's code, libraries, dependencies, and operating system into a single unit.

A **Dockerfile** is the Docker image's source code.

A **Dockerfile** is a text file containing various instructions and configurations. When you run the cloud build submits command, the docker within uses this file to build the image itself.



What is Cloud Run?

Cloud Run is a fully managed compute platform for running containerized applications.

Key Features:

- Deploy any container image
- Automatically scales from zero to N based on traffic
- Pay only for what you use (per request)
- HTTPS, traffic splitting, and revision control out-of-the-box

Use Cases:

- Running REST APIs or backend services
- Event-driven microservices
- Low-maintenance container hosting

App Deployment Architecture - Cloud Run

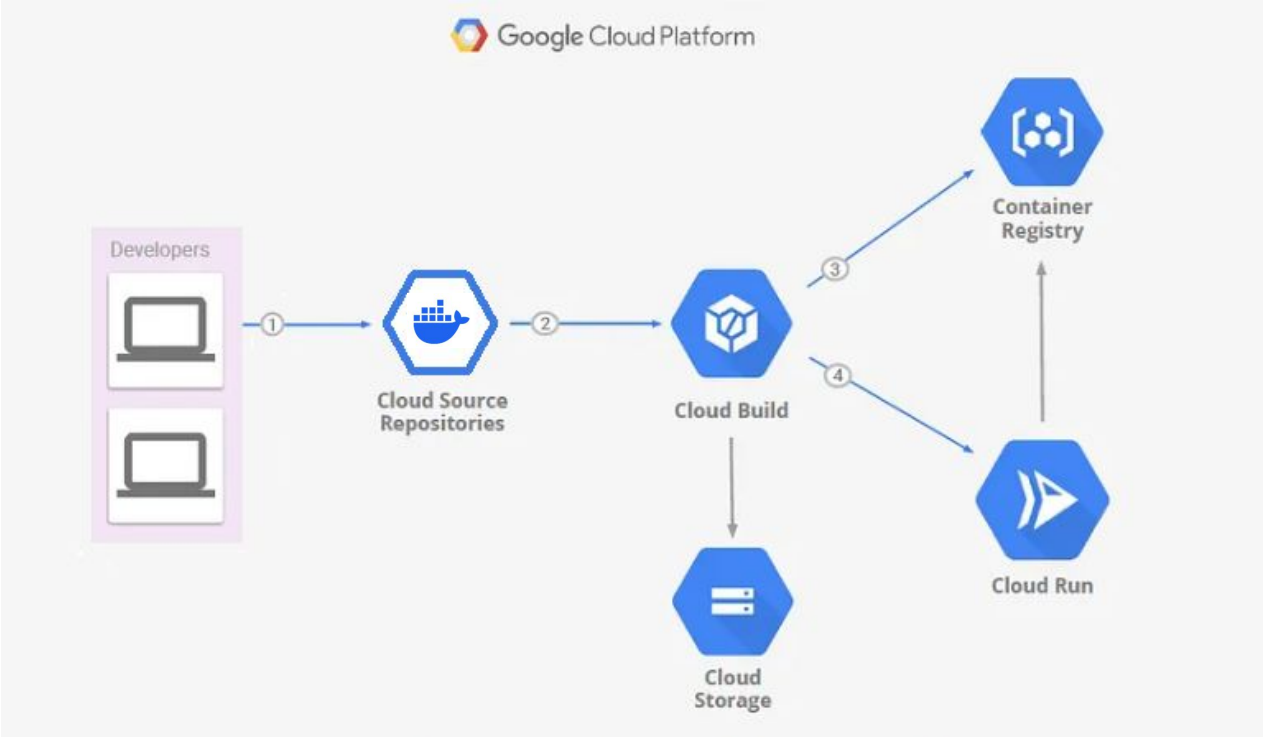


Image source: https://meridian.com/cloud-run-architecture

Cloud Run vs. Cloud Build

Feature	Cloud Build	Cloud Run
Purpose	Build and CI/CD	Run containerized applications
Type	Developer tool (CI/CD)	Serverless compute platform
Input	Source code or container image	Container image
Output	Container/image/artifact	Live running application
Scaling	Not applicable	Auto-scales based on HTTP traffic
Pricing	Based on build minutes used	Based on CPU and memory used per request

Exploring Beyond Cloud Run

Need	Recommended Service	Why
Stateful or complex apps	**Google Kubernetes Engine (GKE)**	Full orchestration and networking
GPU-based workloads	**GKE / AI Platform**	Cloud Run does not support GPUs
Background processing	**Cloud Functions / GKE Jobs / Workflows**	Event-driven or scheduled jobs
Multi-container apps	**GKE or Cloud Composer**	Cloud Run supports only 1 container
Lower cold start time	**App Engine (Standard or Flex)**	Warm instances, built-in scaling
Custom networking or ports	**GKE**	Full control over networking stack

Thank You!

That’s all for the workshop