

1. Setup Google Cloud Platform (GCP)

Create a GCP Account (if not already created)

1. Go to Google Cloud Console
2. Sign in and activate your **Free Tier** (\$300 free credits for new users).

Enable Required APIs

1. Open **Google Cloud Console** and navigate to **APIs & Services > Library**.
2. Enable the following APIs:
 1. **Cloud Build API**
 2. **Artifact Registry API**
 3. **Compute Engine API**
 4. **Cloud Run API** (optional, if deploying on Cloud Run)

Install and Configure gcloud CLI (if using local machine)

1. Install **gcloud SDK**: Installation Guide



```
sh
gcloud auth login

• Set your GCP Project:
sh
gcloud config set project YOUR_PROJECT_ID
```

The image shows a terminal window with a dark background. The first terminal block shows the prompt 'sh' and the command 'gcloud auth login'. The second terminal block is preceded by a bullet point '• Set your GCP Project:' and shows the prompt 'sh' and the command 'gcloud config set project YOUR_PROJECT_ID'. Both terminal blocks have 'Copy' and 'Edit' icons in the top right corner.

2. Create and Push a Docker Application to GitHub

Create a Simple Flask (Python) Application

1. Create a Simple Flask (Python) Application

- Create a project directory:

Copy Edit

```
mkdir gcp-docker-app && cd gcp-docker-app
```

- Create `app.py`:

python

Copy Edit

```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def home():
    return "Hello, GCP with Docker!"

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=8080)
```

- Create a `requirements.txt` file:

nginx

Copy Edit

```
flask
gunicorn
```

Create a Dockerfile

Create a `Dockerfile` in the project directory:

```
dockerfile Copy Edit

# Use official Python image
FROM python:3.9-slim

# Set the working directory
WORKDIR /app

# Copy the project files
COPY . /app

# Install dependencies
RUN pip install --no-cache-dir -r requirements.txt

# Expose the port
EXPOSE 8080

# Command to run the application
CMD ["gunicorn", "-b", "0.0.0.0:8080", "app:app"]
```

3. Push Code to GitHub

- Initialize Git and push the project to a GitHub repository:

```
sh Copy Edit

git init
git add .
git commit -m "Initial commit"
git branch -M main
git remote add origin https://github.com/YOUR_USERNAME/gcp-docker-app.git
git push -u origin main
```

3. Build and Push Docker Image to GCP Artifact Registry

1. Enable Artifact Registry

```
sh Copy Edit  
gcloud services enable artifactregistry.googleapis.com
```

2. Create a Docker Repository

```
sh Copy Edit  
gcloud artifacts repositories create my-docker-repo \  
  --repository-format=Docker \  
  --location=us-central1
```

3. Authenticate Docker with GCP

```
sh Copy Edit  
gcloud auth configure-docker us-central1-docker.pkg.dev
```

4. Build and Tag Docker Image

```
sh Copy Edit  
docker build -t us-central1-docker.pkg.dev/YOUR_PROJECT_ID/my-docker-repo/gcp-docker-app:  
◀
```

5. Push Image to Artifact Registry

```
sh Copy Edit  
docker push us-central1-docker.pkg.dev/YOUR_PROJECT_ID/my-docker-repo/gcp-docker-app:latest  
◀
```

4. Deploy Docker Application on GCP

Option 1: Deploy on Cloud Run (Fully Managed, Free Tier Eligible)

```
sh
gcloud run deploy gcp-docker-app \
  --image=us-central1-docker.pkg.dev/YOUR_PROJECT_ID/my-docker-repo/gcp-docker-app:latest \
  --platform=managed \
  --region=us-central1 \
  --allow-unauthenticated
```

- This will provide a **public URL** for your application.

5. Test Your Application

- If deployed on **Cloud Run**, visit the provided **Cloud Run URL**.
- If deployed on **VM**, visit http://VM_EXTERNAL_IP:8080.

6. Clean Up Resources (To Avoid Charges)

- Delete Cloud Run Service:

```
sh
gcloud run services delete gcp-docker-app --region=us-central1
```

- Delete VM:

```
sh
gcloud compute instances delete gcp-docker-vm --zone=us-central1-a
```

- Delete Docker Image:

```
sh
gcloud artifacts repositories delete my-docker-repo --location=us-central1
```