

# GithubActions: How to create .NET8 WebAPI Docker image and upload to Google Cloud Artifacts Registry

You can see this example source code in this github repo:

[https://github.com/luiscoco/GithubActions\\_dotNET8WebAPI\\_Create\\_DockerImage\\_Upload\\_to\\_GoogleCloud\\_Artifacts\\_Registry](https://github.com/luiscoco/GithubActions_dotNET8WebAPI_Create_DockerImage_Upload_to_GoogleCloud_Artifacts_Registry)

## 0. Prerequisite

We have to create a new repo in Google Cloud Artifacts Registry

The screenshot shows the Google Cloud Artifact Registry interface. At the top, there's a navigation bar with 'Google Cloud', 'My First Project', a search bar containing 'artifact', and various icons. Below the navigation is a header with 'Artifact Registry', 'Repositories', '+ CREATE REPOSITORY', and other buttons like 'EDIT REPOSITORY', 'DELETE', 'SETUP INSTRUCTIONS', 'REFRESH', 'SHOW INFO PANEL', and 'LEARN'. A modal window titled 'Turn on vulnerability scanning' is open, explaining that the registry is not being monitored for known vulnerabilities and offering to turn it on. The main area shows a table with columns: Name, Format, Type, Location, Description, Labels, Version policy, Encryption, Encryption key, Immutable image tags, Created, and Updated. A filter bar at the top of the table allows entering a property name or value. The message 'No rows to display' is shown.

We set the input data

The screenshot shows the 'Create repository' form in the Google Cloud Artifact Registry interface. The left sidebar has 'Repositories' and 'Settings' sections. The main form has fields for 'Name' (set to 'myfirstrepo'), 'Format' (set to 'Docker'), 'Mode' (set to 'Standard'), 'Location type' (set to 'Region'), and 'Region' (set to 'europe-southwest1 (Madrid)'). There's also a 'Release Notes' section at the bottom.

**Labels**

+ ADD LABEL

**Encryption**

This resource is encrypted with a Google-managed key by default. If you need to manage your encryption, you can use a customer-managed key instead. [Learn more](#)

Google-managed encryption key  
No configuration required

Customer-managed encryption key (CMEK)  
Manage via [Google Cloud Key Management Service](#)

Immutable image tags [PREVIEW](#) ?

**Cleanup policies** [PREVIEW](#)

Define policies to automatically clean up artifacts.

Delete artifacts  
Artifacts will be deleted according to cleanup policy criteria.

Dry run  
Artifacts will not be deleted. Cleanup policies will be evaluated, and test delete events sent to Cloud Audit Logging. [Learn more](#)

**ADD A CLEANUP POLICY**

**CREATE** CANCEL

**Turn on vulnerability scanning**

Your registry is not being monitored for known vulnerabilities. GCP offers automatic vulnerability monitoring of all images pushed or pulled within the last 30 days at a cost of \$0.26 per image.

**TURN ON** LEARN MORE

**Filter** Enter property name or value

Name	Format	Type	Location	Description	Labels	Version policy	Encryption	Encryption key	Immutable image tags
myfirstrepo	Docker	Standard	europe-southwest1 (Madrid)				Google-managed	-	False

## 1. Create a Service Account in Google Cloud Platform

Go to the GCP Console: Open the Google Cloud Console and log in to your account.

**IAM & Admin**

**Service accounts** + CREATE SERVICE ACCOUNT

**Service accounts for project "My First Project"**

A service account represents a Google Cloud service identity, such as code running on Compute Engine VMs, App Engine apps, or systems running outside Google. [Learn more about service accounts](#).

Organization policies can be used to secure service accounts and block risky service account features, such as automatic IAM Grants, key creation/upload, or the creation of service accounts entirely. [Learn more about service account organization policies](#).

**Filter** Enter property name or value

Email	Status	Name	Description	Key ID	Key creation date	OAuth 2 Client ID	Actions
extreme-axon-381209@appspot.gserviceaccount.com	Enabled	App Engine default service account	No keys			11679376569859336056	⋮
812224746189-compute@developer.gserviceaccount.com	Enabled	Compute Engine default service account	No keys			103072955633293544031	⋮

**Select Your Project:** Make sure you have the correct project selected in which you want to create the service account.

**Navigate to IAM & Admin:** In the left-hand menu, click on "IAM & Admin", then select "Service Accounts".

**Create Service Account:** Click on "Create Service Account" and fill in the necessary details:

**Name:** Give your service account a name.

**ID:** This is filled automatically based on the name but can be customized.

**Description:** (Optional) Add a description for your service account.

The screenshot shows the Google Cloud IAM & Admin interface. On the left sidebar, under 'Service Accounts', the 'Service Accounts' tab is selected. In the main area, a step-by-step guide for creating a service account is displayed:

- 1 Service account details**
  - Service account name \***: myserviceaccountdotnetwebapi
  - Display name for this service account
  - Service account ID \***: myserviceaccountdotnetwebapi
  - Email address: myserviceaccountdotnetwebapi@extreme-axon-381209.iam.gserviceaccount.com
  - Service account description**
  - Describe what this service account will do
- 2 Grant this service account access to project (optional)**
- 3 Grant users access to this service account (optional)**

At the bottom of the form are 'DONE' and 'CANCEL' buttons.

**Grant Access:** Assign the service account appropriate roles. For Docker images push we can assing the role: "**Artifact Registry Writer**"

Other similar roles could be: "**Storage Admin**" or "**Artifact Registry Administrator**"

The screenshot shows the Google Cloud IAM & Admin interface for creating a service account. On the left sidebar, 'Service Accounts' is selected. The main area is titled 'Create service account' and shows step 2: 'Grant this service account access to project (optional)'. It lists 'Artifact Registry Writer' as the role assigned. Buttons for 'CONTINUE', 'DONE', and 'CANCEL' are at the bottom.

IAM & Admin

My First Project

Create service account

Service account details

Grant this service account access to project (optional)

Role: Artifact Registry Writer

Access to read and write repository items.

+ ADD ANOTHER ROLE

CONTINUE

DONE CANCEL

Do not forget to set the project ID (for this example: extreme-axon-381209) in the **Service account admin role**: [extreme-axon-381209@appspot.gserviceaccount.com](mailto:extreme-axon-381209@appspot.gserviceaccount.com) App Engine default service account

The screenshot shows the Google Cloud IAM & Admin interface for creating a service account. The left sidebar is titled 'Service Accounts' under 'IAM & Admin'. The main area is titled 'Create service account' with a back arrow. It lists three steps: 'Service account details' (checked), 'Grant this service account access to project (optional)', and 'Grant users access to this service account (optional)'. Step 3 is highlighted with a yellow background. A 'Service account users role' dropdown contains 'extreme-axon-381209@appspot.gserviceaccount.com'. A 'Service account admins role' dropdown contains the same email address. Buttons for 'DONE' and 'CANCEL' are at the bottom.

The screenshot shows the Google Cloud IAM & Admin Service accounts list for the project 'My First Project'. The left sidebar is titled 'Service Accounts'. The main area shows a table of service accounts:

Email	Status	Name	Description	Key ID	Key creation date	OAuth 2 Client ID	Actions
<a href="#">extreme-axon-381209@appspot.gserviceaccount.com</a>	Enabled	App Engine default service account	No keys	1167937656985893360			⋮
<a href="#">compute@developer.gserviceaccount.com</a>	Enabled	Compute Engine default service account	No keys	1030729556332935440			⋮
<a href="#">myserviceaccountdotnetwebapi@extreme-axon-381209.iam.gserviceaccount.com</a>	Enabled	myserviceaccountdotnetwebapi	No keys	1168422964106845138			⋮

A red box highlights the last row, which corresponds to the service account created in the previous step.

Be cautious with permissions to follow the principle of least privilege.

**Create Key:** After creating the service account, click on it to open its details. Under the "Keys" tab, click "Add Key", then select "Create new key".

Choose "JSON" as the key type and click "Create". This will download the JSON key file to your computer.

Service accounts for project "My First Project"

A service account represents a Google Cloud service identity, such as code running on Compute Engine VMs, App Engine apps, or systems running outside Google. [Learn more about service accounts.](#)

Organization policies can be used to secure service accounts and block risky service account features, such as automatic IAM Grants, key creation/upload, or the creation of service accounts entirely. [Learn more about service account organization policies.](#)

Email	Status	Name	Description	Key ID	Key creation date	OAuth 2 Client ID	Actions
<a href="#">extreme-axon-381209@appspot.gserviceaccount.com</a>	Enabled	App Engine default service account	No keys	1167937656985893360			⋮
<a href="#">compute@developer.gserviceaccount.com</a>	Enabled	Compute Engine default service account	No keys	1030729556332935440			⋮
<a href="#">myserviceaccountdotnetwebapi@extreme-axon-381209.iam.gserviceaccount.com</a>	Enabled	myserviceaccountdotnetwebapi	No keys	116842296410684513824			⋮

IAM & Admin

Service Accounts

myserviceaccountdotnetwebapi

DETAILS PERMISSIONS KEYS METRICS LOGS

### Service account details

Name: myserviceaccountdotnetwebapi SAVE

Description SAVE

Email: myserviceaccountdotnetwebapi@extreme-axon-381209.iam.gserviceaccount.com

Unique ID: 116842296410684513824

### Service account status

Disabling your account allows you to preserve your policies without having to delete it.

Enabled DISABLE SERVICE ACCOUNT

### Advanced settings

The screenshot shows the Google Cloud IAM & Admin interface. On the left, a sidebar lists various sections: IAM, Identity & Organization, Policy Troubleshooter, Policy Analyzer, Organization Policies, Service Accounts (which is currently selected), Workload Identity Federation, Workforce Identity Federation, Labels, and Tags. The main area is titled 'myserviceaccountdotnetwebapi'. It has tabs for DETAILS, PERMISSIONS, KEYS (which is highlighted with a red box), METRICS, and LOGS. Under the KEYS tab, there's a warning about service account keys being a security risk if compromised. Below that, it says 'Add a new key pair or upload a public key certificate from an existing key pair.' There are two options: 'Create new key' (highlighted with a red box) and 'Upload existing key'. At the bottom of the KEYS section, there are fields for 'Key creation date' and 'Key expiration date'.

## Create private key for "myserviceaccountdotnetwebapi"

Downloads a file that contains the private key. Store the file securely because this key can't be recovered if lost.

### Key type

JSON

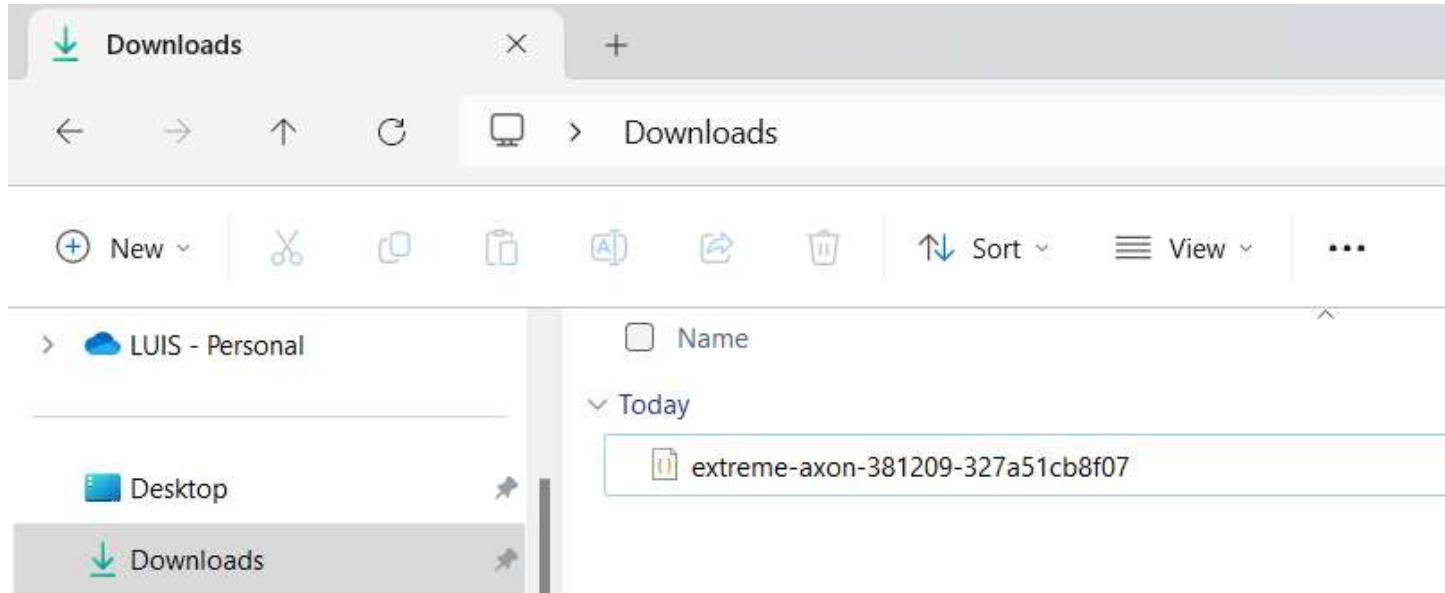
Recommended

P12

For backward compatibility with code using the P12 format

CANCEL

CREATE



## 2. Add the Key as a Secret in your GitHub Repository

**Go to Your GitHub Repository:** Open your GitHub repository in a web browser.

**Navigate to Settings:** Click on "Settings" in the top menu of your repository.

**Access Secrets:** In the left-hand sidebar, click on "Secrets", then select "Actions".

**Add a New Secret:** Click on "New repository secret".

**Name Your Secret:** Enter **GOOGLE\_CLOUD\_CREDENTIALS** as the name.

**Paste the Key Content:** Open the **JSON** key file you downloaded from GCP in a text editor, copy all its contents, and paste them into the secret's value field in GitHub.

**Save the Secret:** Click "Add secret" to save your new secret.

The screenshot shows the 'General' settings for a GitHub repository. The left sidebar lists 'General', 'Code and automation', 'Security', and 'Secrets and variables'. The main area shows the repository name 'GithubActions\_dotNET8WebAPI\_Cre...' and a 'Default branch' set to 'main'. A 'Social preview' section is at the bottom.

## Secrets and variables

### Actions

#### Codespaces

#### Dependabot

The screenshot shows the GitHub repository settings page for 'luiscoco/GithubActions\_dotNET8WebAPI\_Create\_DockerImage\_Upload\_to\_GoogleCloud\_Artifacts\_Registry'. The 'Actions' section is highlighted in the sidebar. In the main area, there are sections for 'Environment secrets' (which says 'This repository has no environment secrets.') and 'Repository secrets'. A green button labeled 'New repository secret' is visible in the 'Repository secrets' section.

The screenshot shows the 'Actions secrets / New secret' creation page. The 'Name' field is set to 'GOOGLE\_CLOUD\_CREDENTIALS'. The 'Secret' field contains a JSON service account key. At the bottom, there is an 'Add secret' button.

```
{
  "type": "service_account",
  "project_id": "extreme-axon-381209",
  "private_key_id": "c3f6e3a95070ea41660264c9ae7beac04a543dc",
  "private_key": "-----BEGIN PRIVATE KEY-----\nMIIEvglBADANBgkqhkiG9w0BAQEFAASCBkgwggSkAgEAAoIBAQCP0ZjUtm/Vhu0TnSQbT5BubVdgz0NmPytONizGg9Bx\nqq3GghC2ORuSxlw04vsMGi8D1jk/qcwjFN4x\nph3qfPaLUXHUYN5mNL+B6GCiYEjJ22CyzkTEkEkoUh+EBOQv+EB5Jw\nRqVA5Wsvni4WB0j6xAzXjhCrVigqkT25+whYcl6xCfftUwEdqaJhx7S4Tom5XwsAnlOCnfhGNgZnTJ/rRjz6y1NOAVzw6Z\no2qCyHWWA6vHTTjt8xc76QSIIUBFLdYCGciadH\nN1rhRq0v7PACTrW5y65lliA6VSNd1Fy8hVeO6qf7DpLk.laeuu3drhp7d4A9\nsEKNntsBN8StrAgMBAECggEAPKfir2EMI//AEUnduvR4mcCtMODPxMQ77tcZxGw89Crd\nvkB1pyKST+IG9zRH88AaQx9\n9zd0dAD10oOlnu/5t23BRUJutF1MrVdoC6d3UcsW\nn983hEIBhCMzGEHRA\ny+zRWGbd+uV45Detr03SuJlke1RyvJu3MZCa\ny5ClnngRV\nnfCRsmj8YAkAlb+RLgsp7gTn3ETtUy2X5Sz7dLEPFL6Mfp81OOrSydQS\nOo5xoz\nrh4Y4Kqk75hiwwKTWFN\n-----END PRIVATE KEY-----"
}
```

Now, your GitHub Actions workflow can use this **secret** to authenticate with Google Cloud services. In your workflow file, you can reference this secret as `${{ secrets.GOOGLE_CLOUD_CREDENTIALS }}`.

### 3. Create the main.yml file for Github actions workflow

Below is the `main.yml` file tailored for your requirements.

This workflow assumes you have already set up Google Cloud credentials as secrets in your GitHub repository

```
name: Build and Push Docker Image

on:
  push:
    branches:
      - main

env:
  PROJECT_ID: extreme-axon-381209
  IMAGE_NAME: my-dotnetwebapi
  REPOSITORY: europe-southwest1-docker.pkg.dev/extreme-axon-381209/myfirstrepo
  TAG: latest

jobs:
  build-and-push:
    runs-on: ubuntu-latest

    steps:
      - name: Checkout code
        uses: actions/checkout@v4

      - name: Authenticate to Google Cloud
        uses: google-github-actions/auth@v2
        with:
          credentials_json: ${{ secrets.GOOGLE_CLOUD_CREDENTIALS }}

      - name: Configure Docker for Google Cloud Artifact Registry
        run: |
          echo '${{ secrets.GOOGLE_CLOUD_CREDENTIALS }}' | gcloud auth activate-service-account
          gcloud auth configure-docker europe-southwest1-docker.pkg.dev --quiet

      - name: Build Docker image
        run: |
          docker build -t ${{ env.REPOSITORY }}/{{ env.IMAGE_NAME }}:${{ env.TAG }} .

      - name: Push Docker image
        run: |
          docker push ${{ env.REPOSITORY }}/{{ env.IMAGE_NAME }}:${{ env.TAG }}

      - name: Verify the image was pushed
        run: |
          gcloud artifacts docker images list ${{ env.REPOSITORY }}/{{ env.IMAGE_NAME }}
```

### 4. Verify the Docker image uploaded to Google Cloud

## Navigate to Google Cloud Artifacts Registry repo

The screenshot shows the Google Cloud Platform dashboard for a project named "My First Project". The left sidebar has a red box around the "Cloud overview" item. The main content area has a search bar at the top with the word "artifacts". Below it, under "PRODUCTS & PAGES", there is a section for "Artifact Registry" which is also highlighted with a red box.

We can see inside the repo the uploaded Docker image

This screenshot shows the "Repositories" page in the Google Cloud Artifact Registry. The left sidebar has a red box around the "Artifact Registry" item. The main table lists one repository: "myfirstrepo" (Docker, Standard type, located in europe-southwest1 (Madrid)). A red box highlights the repository name in the table.

This screenshot shows the "Images for myfirstrepo" page. The left sidebar has a red box around the "Repositories" item. The main table lists one image: "my-dotnetwebapi" (Created 2 minutes ago, Updated 1 minute ago). A red box highlights the image name in the table.

This screenshot shows the "Digests for my-dotnetwebapi" page. The left sidebar has a red box around the "Repositories" item. The main table lists one digest: "38fd2294fa66" (Tags latest, Created 1 minute ago, Updated 1 minute ago). A red box highlights the digest ID in the table.

This screenshot shows the Google Cloud Artifact Registry interface. The URL in the address bar is `38fd2294fa66c`. The 'PULL' tab is selected. The page displays various metadata about the Docker image, including its digest, size, and creation date. A 'Tags' section shows 'latest'.

This screenshot shows the same Google Cloud Artifact Registry interface as above, but with a red box highlighting the 'Pull by tag' command in the 'RUN IN CLOUD SHELL' section. The command is:

```
$ docker pull \
  europe-southwest1-docker.pkg.dev/extreme-axon-381209/myfirstrepo/my-dotnetwebapi:latest
```

We run this command to pull the image and to

```
docker pull europe-southwest1-docker.pkg.dev/extreme-axon-381209/myfirstrepo/my-dotnetwebapi:1
```

We verified the downloaded image in Docker Desktop

This screenshot shows the Docker Desktop interface. The left sidebar has 'Images' selected. The main area shows a table of images. One row is highlighted, showing the full path: `europe-southwest1-docker.pkg.dev/extreme-axon-381209/myfirstrepo/my-dotnetwebapi`.

Name	Tag	Status	Created	Size	Actions
<code>europe-southwest1-docker.pkg.dev/extreme-axon-381209/myfirstrepo/my-dotnetwebapi</code>	latest	In use	4 minutes ago	221.04 MB	[More]

Also we can see the image with the command

`docker images`

```
PowerShell para desarrolladores
+ PowerShell para desarrolladores | ⌂ ⌂ ⌂
PS C:\New folder\GithubActions_dotNET8WebAPI_Create_DockerImage_Upload_to_GoogleCloud_Artifacts_Registry> docker images
REPOSITORY                                     TAG      IMAGE ID      CREATED        SIZE
europe-southwest1-docker.pkg.dev/extreme-axon-381209/myfirstrepo/my-dotnetwebapi   latest    54d69a0d46f4  35 minutes ago  221MB
PS C:\New folder\GithubActions_dotNET8WebAPI_Create_DockerImage_Upload_to_GoogleCloud_Artifacts_Registry>
```

We run the image in our Docker Desktop

```
docker run -p 8080:8080 europe-southwest1-docker.pkg.dev/extreme-axon-381209/myfirstrepo/my-do
```

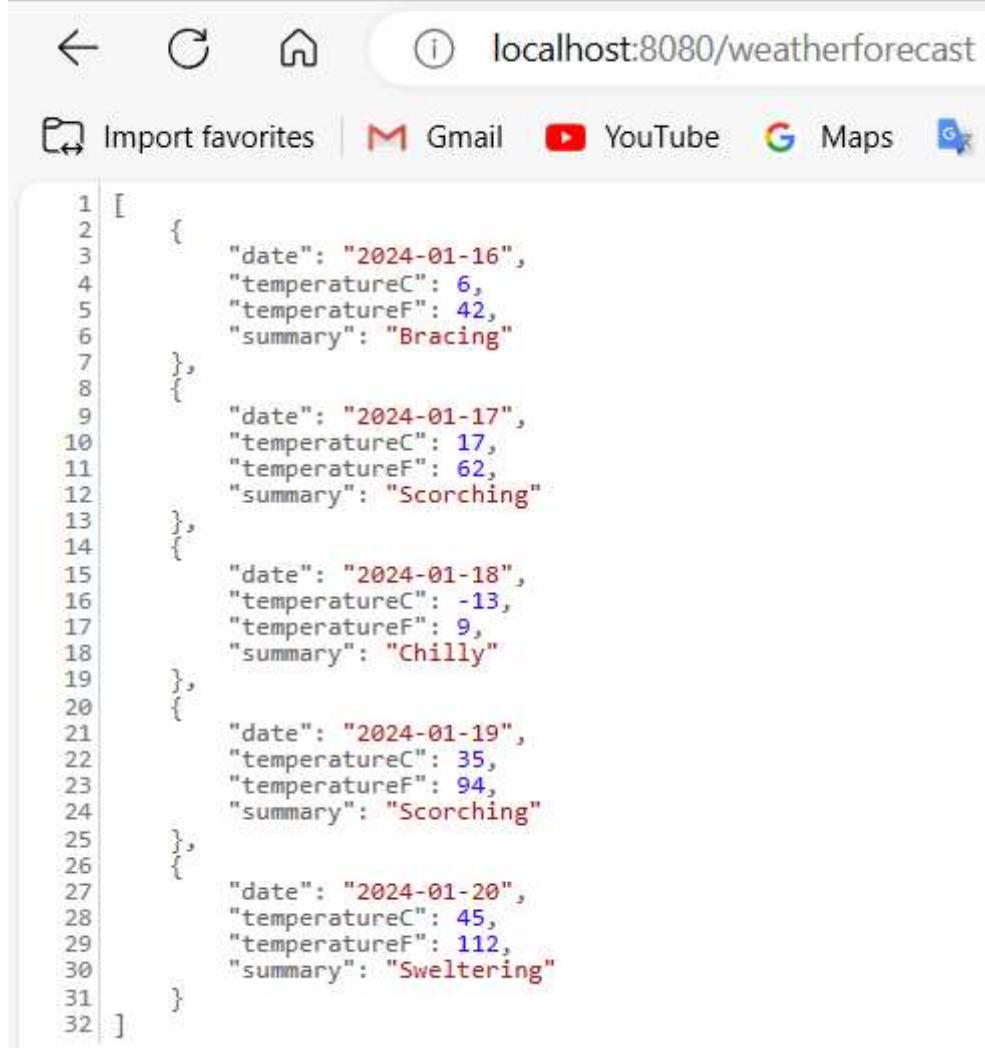
We see the running image with the command

`docker ps`

And also we can see the image in Docker Desktop

Name	Image	Status	Port(s)	Last started	CPU (%)	Actions
nifty_golick	europe-southwest1-docker.pkg.dev/extre... 52fd48deb89f	Running	8080:8080	9 seconds ago	0%	[More Options]

We can verify the running Docker container



The screenshot shows a browser window with the URL `localhost:8080/weatherforecast` in the address bar. The page displays a JSON array of weather forecast data. Each item in the array contains a date, temperature in Celsius, temperature in Fahrenheit, and a summary. The data is color-coded: dates in red, temperatures in blue, and summaries in pink.

```
[{"date": "2024-01-16", "temperatureC": 6, "temperatureF": 42, "summary": "Bracing"}, {"date": "2024-01-17", "temperatureC": 17, "temperatureF": 62, "summary": "Scorching"}, {"date": "2024-01-18", "temperatureC": -13, "temperatureF": 9, "summary": "Chilly"}, {"date": "2024-01-19", "temperatureC": 35, "temperatureF": 94, "summary": "Scorching"}, {"date": "2024-01-20", "temperatureC": 45, "temperatureF": 112, "summary": "Sweltering"}]
```

## 5. Modify the main.yml for creating a new repo if it doesn't exist

As prerequisite for creating a new Artifact Registry repo it is mandatory to assign to the Service Account the permission **Artifact Registry Administrator**

We also set the permission **Artifact Registry Writer** to push Docker image to the Artificat Registry rrepo

**Service account details**

Service account name: newserviceaccount

Service account ID \*: newserviceaccou

Email address: newserviceaccou@extreme-axon-381209.iam.gserviceaccount.com

Service account description:

CREATE AND CONTINUE

**Grant this service account access to project (optional)**

**Grant users access to this service account (optional)**

DONE CANCEL

**Service account details**

**Grant this service account access to project (optional)**

Grant this service account access to My First Project so that it has permission to complete specific actions on the resources in your project. [Learn more](#)

Role: Artifact Registry Administrator

IAM condition (optional): + ADD IAM CONDITION

Administrator access to create and manage repositories.

Role: Artifact Registry Writer

IAM condition (optional): + ADD IAM CONDITION

Access to read and write repository items.

+ ADD ANOTHER ROLE

CONTINUE

We also set the Service account admins role, we select our Projec ID role

**Service account details**

**Grant this service account access to project (optional)**

**Grant users access to this service account (optional)**

Grant access to users or groups that need to perform actions as this service account. [Learn more](#)

Service account users role: Service account admins role

Grant users the permission to administer this service account

extreme-axon-381209@appspot.gserviceaccount.com

DONE CANCEL

IAM & Admin

Service accounts

+ CREATE SERVICE ACCOUNT

DELETE

MANAGE ACCESS

REFRESH

LEARN

Service accounts for project "My First Project"

A service account represents a Google Cloud service identity, such as code running on Compute Engine VMs, App Engine apps, or systems running outside Google. [Learn more about service accounts](#).

Organization policies can be used to secure service accounts and block risky service account features, such as automatic IAM Grants, key creation/upload, or the creation of service accounts entirely. [Learn more about service account organization policies](#).

Filter	Enter property name or value	Status	Name ↑	Description	Key ID	Key creation date	OAuth 2 Client ID	Actions
<input type="checkbox"/>	Email	Enabled	App Engine default service account		No keys	116793765698589336056		⋮
<input type="checkbox"/>	✉️ extreme-axon-381209@appspot.gserviceaccount.com	Enabled	Compute Engine default service account		No keys	103072955633293544031		⋮
<input type="checkbox"/>	✉️ B12224746189-compute@developer.gserviceaccount.com	Enabled	newserviceaccount		No keys	108754526796940298234		⋮
<input type="checkbox"/>	✉️ newserviceaccount@extreme-axon-381209.iam.gserviceaccount.com	Enabled						⋮

Then We create a new Key to copy it in our Github Secrets and store it with the variable name **GOOGLE\_CLOUD\_CREDENTIALS**

We click on the Key tab and we create a new JSON key file

IAM & Admin

Service accounts

newserviceaccount

DETAILS

PERMISSIONS

KEYS

METRICS

LOGS

Service account details

Name: newserviceaccount

SAVE

Description

SAVE

Email: newserviceaccount@extreme-axon-381209.iam.gserviceaccount.com

Unique ID: 108754526796940298234

Service account status

Disabling your account allows you to preserve your policies without having to delete it.

Enabled

DISABLE SERVICE ACCOUNT

IAM & Admin

Service accounts

newserviceaccount

DETAILS

PERMISSIONS

KEYS

METRICS

LOGS

Keys

⚠️ Service account keys could pose a security risk if compromised. We recommend you avoid downloading service account keys and instead use the Workload Identity Federation. You can learn more about the best way to authenticate service accounts on Google Cloud [here](#).

Add a new key pair or upload a public key certificate from an existing key pair.

Block service account key creation using [organization policies](#). Learn more about setting organization policies for service accounts.

ADD KEY

Create new key

Key creation date

Upload existing key

Key expiration date

# Create private key for "newserviceaccount"

Downloads a file that contains the private key. Store the file securely because this key can't be recovered if lost.

## Key type

JSON

Recommended

P12

For backward compatibility with code using the P12 format

CANCEL

CREATE

In our github repo we navigate to the **Settings->Secrets option->Actions**

The screenshot shows a GitHub repository settings page for 'luiscoco / GitHubActions\_dotNET8WebAPI\_Create\_DockerImage\_Upload\_to\_GoogleCloud\_Artifacts\_Registry'. The 'Actions' tab is selected in the navigation bar. The main content area shows the 'General' settings, including fields for 'Repository name' (set to 'GitHubActions\_dotNET8WebAPI\_Cre...'), 'Template repository', 'Require contributors to sign off on web-based commits', and 'Default branch' (set to 'main'). On the left sidebar, the 'Security' section is expanded, showing options like 'Code security and analysis', 'Deploy keys', and 'Secrets and variables'. The 'Secrets and variables' option is highlighted with a red box.

The screenshot shows the GitHub repository settings for a project named 'dotnetwebapi'. In the left sidebar, the 'Secrets and variables' tab is selected and highlighted with a red box. The main area displays a table of repository secrets. One secret, 'GOOGLE\_CLOUD\_CREDENTIALS', is listed with a red box around its name. At the top right of the secrets table, there is a green button labeled 'New repository secret' which is also highlighted with a red box.

Below is the complete `main.yml` file based on your initial workflow, with the added functionality to check for the existence of a Google Cloud Artifact Registry repository and create it if it does not exist.

This workflow includes steps for authentication, checking and creating the repository, building and pushing the Docker image, and verifying the push.

```

name: Build and Push Docker Image

on:
  push:
    branches:
      - main

env:
  PROJECT_ID: extreme-axon-381209
  IMAGE_NAME: my-dotnetwebapi
  REPOSITORY: myfirstrepo
  TAG: latest

jobs:
  build-and-push:
    runs-on: ubuntu-latest

    steps:
      - name: Checkout code
        uses: actions/checkout@v4

      - name: Authenticate to Google Cloud
        uses: google-github-actions/auth@v2
        with:
          credentials_json: ${{ secrets.GOOGLE_CLOUD_CREDENTIALS }}

      - name: Configure Docker for Google Cloud Artifact Registry

```

```
run: |
  echo '${{ secrets.GOOGLE_CLOUD_CREDENTIALS }}' | gcloud auth activate-service-account
  gcloud auth configure-docker europe-southwest1-docker.pkg.dev --quiet

- name: Check if Google Cloud Artifact Registry repository exists
  id: check-repo
  run: |
    if gcloud artifacts repositories describe ${{ env.REPOSITORY }} --location=europe-sout
      echo "Repository exists"
    else
      echo "::set-output name=repo_exists::false"
    fi

- name: Create Google Cloud Artifact Registry repository if it does not exist
  if: steps.check-repo.outputs.repo_exists == 'false'
  run: |
    gcloud artifacts repositories create ${{ env.REPOSITORY }} --repository-format=docker
    echo "Repository created"

- name: Build Docker image
  run: |
    docker build -t europe-southwest1-docker.pkg.dev/${{ env.PROJECT_ID }}/${{ env.REPOSITO
  run: |
    docker push europe-southwest1-docker.pkg.dev/${{ env.PROJECT_ID }}/${{ env.REPOSITORY

- name: Verify the image was pushed
  run: |
    gcloud artifacts docker images list europe-southwest1-docker.pkg.dev/${{ env.PROJECT_I
```