

# [ML on GCP C7] Serving on Cloud MLE

2 hours

Free

[Rate Lab](#)

## Overview

*Duration is 1 min*

In this lab, you build an AppEngine app to serve your ML predictions. You get to modify the user-facing form and the python script and deploy them as an AppEngine app that makes requests to your deployed ML model.

## What you learn

In this lab, you:

- modify a simple UI form to get user input when making calls to your model
- build the http request to be made to your deployed ML model on Cloud MLE

## Setup

For each lab, you get a new GCP project and set of resources for a fixed time at no cost.

1. Make sure you signed into Qwiklabs using an **incognito window**.
2. Note the lab's access time (for example, **02:00:00** and make sure you can finish in that time block.

There is no pause feature. You can restart if needed, but you have to start at the beginning.


3. When ready, click


START LAB


4. Note your lab credentials. You will use them to sign in to Cloud Platform Console.

Open Google Console

**Caution:** When you are in the console, do not deviate from the lab instructions. Doing so may cause your account to be blocked. [Learn more.](#)

**Username**  
google2876526\_student@qwiklabs.n 

**Password**  
TG959yrKDX 

**GCP Project ID**  
qwiklabs-gcp-0855e773352d3560 

[New to labs? View our introductory video!](#)

5. Click **Open Google Console**.

6. Click **Use another account** and copy/paste credentials for **thislab** into the prompts.

If you use other credentials, you'll get errors or **incur charges**.

7. Accept the terms and skip the recovery resource page.

Do not click **End Lab** unless you are finished with the lab or want to restart it. This clears your work and removes the project.

# Start Cloud Shell

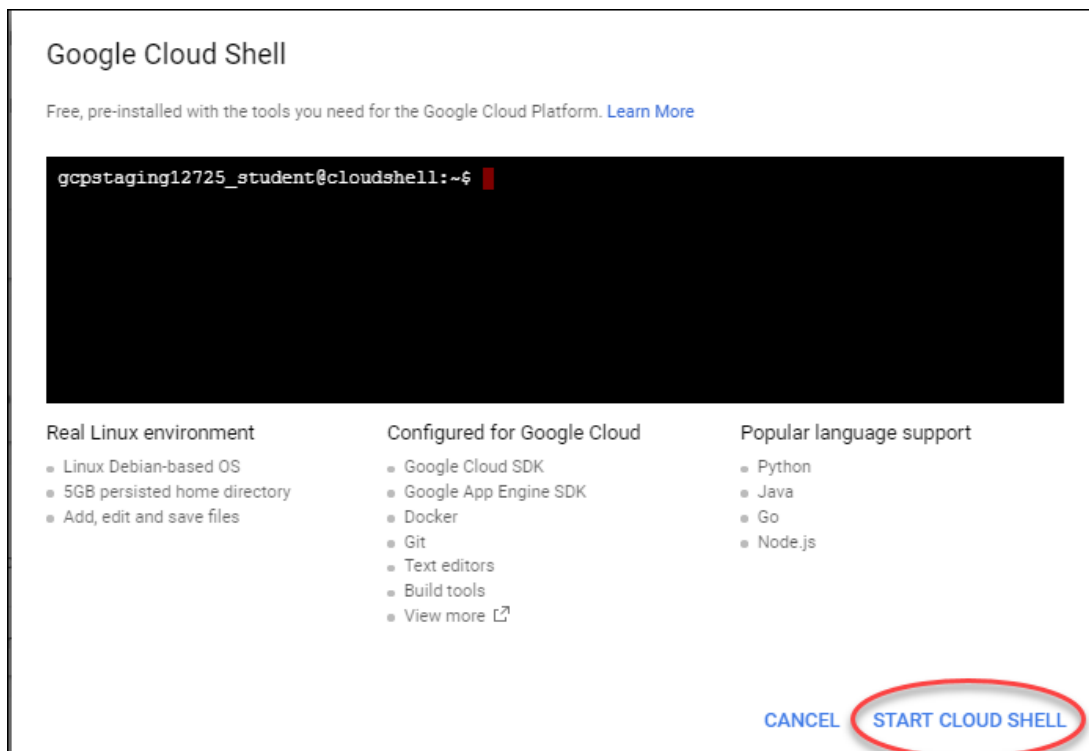
## Activate Google Cloud Shell

Google Cloud Shell provides command-line access to your GCP resources.

From the GCP Console click the **Cloud Shell** icon on the top right toolbar:

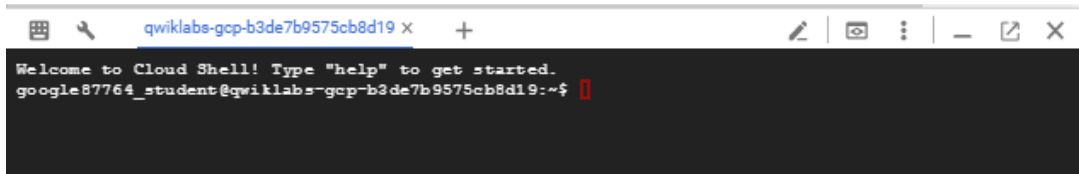


Then click **START CLOUD SHELL**:



You can click **START CLOUD SHELL** immediately when the dialog comes up instead of waiting in the dialog until the Cloud Shell provisions.

It takes a few moments to provision and connects to the environment:



The Cloud Shell is a virtual machine loaded with all the development tools you'll need. It offers a persistent 5GB home directory, and runs on the Google Cloud, greatly enhancing network performance and authentication.

Once connected to the cloud shell, you'll see that you are already authenticated and the project is set to your *PROJECT\_ID*:

```
gcloud auth list
```

Output:

```
Credentialed accounts:
-
<myaccount>@<mydomain>.com
(active)
```

**Note:** `gcloud` is the powerful and unified command-line tool for Google Cloud Platform. Full documentation is available on [Google Cloud gcloud Overview](#). It comes pre-installed on Cloud Shell and supports tab-completion.

```
gcloud config list project
```

Output:

```
[core]
project = <PROJECT_ID>
```

## Copy trained model

### Step 1

Set necessary variables and create a bucket:

```
REGION=us-central1
BUCKET=$(gcloud config
get-value project)
TFVERSION=1.7
gsutil mb -l ${REGION}
gs://${BUCKET}
```

## Step 2

Copy trained model into your bucket:

```
gsutil -m cp -R
gs://cloud-training-
demos/babyweight/trained_model
gs://${BUCKET}/babyweight
```



# Deploy trained model

## Step 1

Set necessary variables:

```
MODEL_NAME=babyweight
MODEL_VERSION=ml_on_gcp
MODEL_LOCATION=$(gsutil ls
gs://${BUCKET}/babyweight/exp
| tail -1)
```



## Step 2

Deploy trained model:

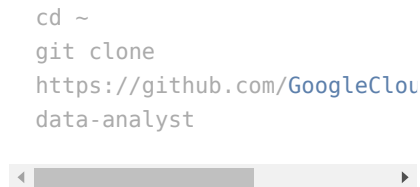
```
gcloud ml-engine models
create ${MODEL_NAME} --
regions $REGION
gcloud ml-engine versions
create ${MODEL_VERSION} --
model ${MODEL_NAME} --
origin ${MODEL_LOCATION} -
-runtime-version
$TFVERSION
```

# Code for your frontend

## Step 1


Clone the course repository:

```
cd ~  
git clone  
https://github.com/GoogleClou  
data-analyst
```



## Step 2

You can use the Cloud Shell code editor to view and edit the contents of these files.

Click on the (  ) icon on the top right of your Cloud Shell window to launch Code

Editor.

Once launched, navigate to the `~/training-data-analyst/courses/machine_learning/deepdive/06_structured/labs/serving` directory.

## Step 3

Open the **application/main.py** and **application/templates/form.html** files and notice the *#TODOs* within the code. These need to be replaced with code. The next section tells you how.

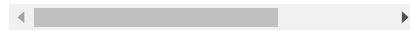
# Modify main.py

## Step 1

Open the `main.py` file by clicking on it. Notice the lines with `# TODO` for setting credentials and the api to use.

Set the credentials to use Google Application Default Credentials (recommended way to authorize calls to our APIs when building apps deployed on AppEngine):

```
credentials =  
GoogleCredentials.get_applica
```



Specify the api name (ML Engine API) and version to use:

```
api =  
discovery.build('ml',  
'v1',  
credentials=credentials)
```

## Step 2

Scroll further down in `main.py` and look for the next `#TODO` in the method `get_prediction()`. In there, specify, using the **parent** variable, the name of your trained model deployed on Cloud MLE:

```
parent =  
'projects/%s/models/%s' %  
(project, model_name)
```

## Step 3

Now that you have all the pieces for making the call to your model, build the call request by specifying it in the **prediction** variable:

```
prediction =  
api.projects().predict(body=i  
name=parent).execute()
```



## Step 4

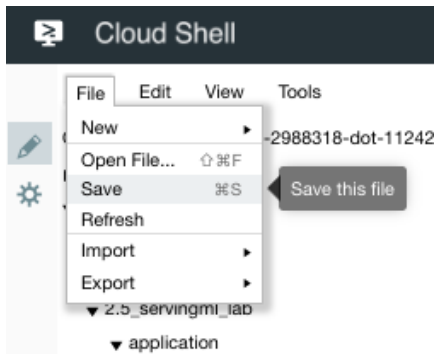
The final `#TODO` (scroll towards bottom) is to get `gestation_weeks` from the form data and cast into a float within the **features** array:

```
features['gestation_weeks']
=
float(data['gestation_weeks'])
```



## Step 5

Save the changes you made using the **File > Save** button on the top left of your code editor window.



## Modify form.html

form.html is the front-end of your app. The user fills in data (features) about the mother based on which we will make the predictions using our trained model.

## Step 1

In code editor, navigate to the application/templates directory and click to open the form.html file.

## Step 2

There is one *#TODO* item here. Look for the div segment for **Plurality** and add options for other plurality values (2, 3, etc).

```
<md-option
value="2">Twins</md-
option>
<md-option
```



```
value="3">Triplets</md-  
option>
```

## Step 3

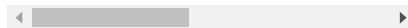
Save the changes you made using the **File > Save** button on the top left of your code editor window.

# Deploy and test your app

## Step 1

In Cloud Shell, run the `deploy.sh` script to install required dependencies and deploy your app engine app to the cloud.

```
cd training-data-  
analyst/courses/machine_learn  
  
./deploy.sh
```



Note: Choose a region for App Engine when prompted and follow the prompts during this process

## Step 2

Go to the url `https://<PROJECT-ID>.appspot.com` and start making predictions.

*Note: Replace <PROJECT-ID> with your Project ID.*

## End your lab

When you have completed your lab, click **End Lab**. Qwiklabs removes the resources you've used and cleans the account for you.

You will be given an opportunity to rate the lab experience. Select the applicable number of stars, type a comment, and then click **Submit**.

The number of stars indicates the following:

- 1 star = Very dissatisfied
- 2 stars = Dissatisfied
- 3 stars = Neutral
- 4 stars = Satisfied
- 5 stars = Very satisfied

You can close the dialog box if you don't want to provide feedback.

For feedback, suggestions, or corrections, please use the **Support** tab.

Last Tested Date: 12-11-2018

Last Updated Date: 12-11-2018

©2019 Google LLC All rights reserved. Google and the Google logo are trademarks of Google LLC. All other company and product names may be trademarks of the respective companies with which they are associated.