

# HUGGING FACE SPACES

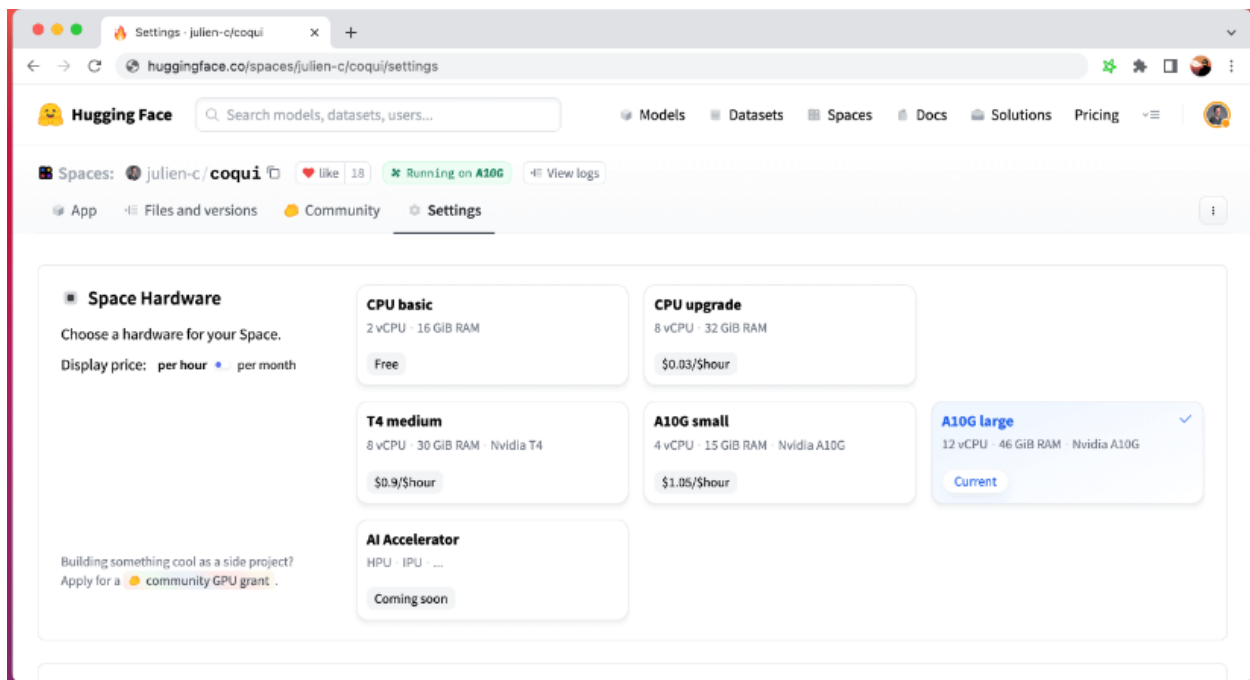
HuggingFace Spaces is a powerful yet user-friendly platform that enables users to build web applications with seamless access to the HuggingFace ecosystem.

**Hugging Face Spaces** offer a simple way to host ML demo apps directly on your profile or your organization's profile. This allows you to create your ML portfolio, showcase your projects at conferences or to stakeholders, and work collaboratively with other people in the ML ecosystem.

Spaces has built in support for two awesome SDKs that let you build cool apps in Python in a matter of minutes: **Streamlit** and **Gradio**, but you can also unlock the whole power of Docker and host an arbitrary Dockerfile. Finally, you can create static Spaces using JavaScript and HTML.

## Using GPU Spaces

You can upgrade your Space to use a GPU accelerator using the *Settings* button in the top navigation bar of the Space. You can even request a free upgrade if you are building a cool demo for a side project!



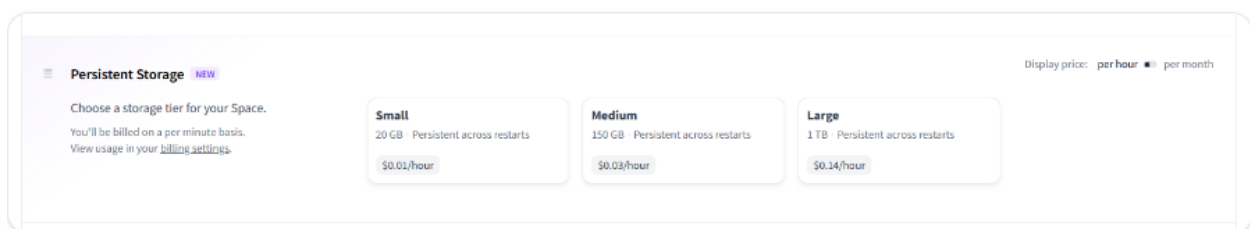
## Disk usage on Spaces

Every Space comes with a small amount of disk storage. This disk space is ephemeral, meaning its content will be lost if your Space restarts or is stopped. If you need to persist data with a longer lifetime than the Space itself, you can:

- [persistent storage upgrade](#)
- [dataset as a data store](#)

## Persistent storage

You can upgrade your Space to have access to persistent disk space from the **Settings** tab.



Here are the specifications for each of the different upgrade options:

Tier	Disk space	Persistent	Monthly Price
Free tier	50GB	No (ephemeral)	Free!
Small	20GB	Yes	\$5
Medium	150 GB	Yes	\$25
Large	1TB	Yes	\$100

## Dataset as DataStore

If you need to persist data that lives longer than your Space, you could use a [huggingface dataset repo](#).

## Gradio Spaces

**Gradio** provides an easy and intuitive interface for running a model from a list of inputs and displaying the outputs in formats such as images, audio, 3D objects, and more. Gradio now even has a **Plot output component** for creating data visualizations with Matplotlib, Bokeh, and Plotly!

Selecting **Gradio** as the SDK when **creating a new Space** will initialize your Space with the latest version of Gradio by setting the `sdk` property to `gradio` in your `README.md` file's YAML block. If you'd like to change the Gradio version, you can edit the `sdk_version` property.

## Create the Gradio interface

To create the Gradio app, make a new file in the repository called **app.py**, and add the following example code:

```

import gradio as gr
from transformers import pipeline

pipeline = pipeline(task="image-classification", model="julien-
n-c/hotdog-not-hotdog")

def predict(input_img):
    predictions = pipeline(input_img)
    return input_img, {p["label"]: p["score"] for p in predic
tions}

gradio_app = gr.Interface(
    predict,
    inputs=gr.Image(label="Select hot dog candidate", sources
=['upload', 'webcam'], type="pil"),
    outputs=[gr.Image(label="Processed Image"), gr.Label(labe
l="Result", num_top_classes=2)],
    title="Hot Dog? Or Not?",
)

if __name__ == "__main__":
    gradio_app.launch()

```

This Python script uses a [HuggingFace Transformers pipeline](#) to load the [julien-c/hotdog-not-hotdog](#) model, which is used by the Gradio interface.

## Embed Gradio Spaces on other webpages

You can embed a Gradio Space on other webpages by using either Web Components or the HTML `<iframe>` tag.

## Streamlit Spaces

**Streamlit** gives users freedom to build a full-featured web app with Python in a *reactive* way. Your code is rerun each time the state of the app changes. Streamlit is also great for data visualization and supports several charting libraries such as Bokeh, Plotly, and Altair. Read this [blog post](#) about building and hosting Streamlit apps in Spaces.

Selecting **Streamlit** as the SDK when [creating a new Space](#) will initialize your Space with the latest version of Streamlit by setting the `sdk` property to `streamlit` in your `README.md` file's YAML block.

This will create a repository with a `README.md` that contains the following properties in the YAML configuration block:

```
sdk: streamlit
sdk_version: 1.25.0
# The latest supported version
```

## Create the Streamlit app

To create the Streamlit app, make a new file in the repository called **app.py**, and add the following code:

```
import streamlit as st
from transformers import pipeline
from PIL import Image

pipeline = pipeline(task="image-classification", model="julien-c/hotdog-not-hotdog")

st.title("Hot Dog? Or Not?")

file_name = st.file_uploader("Upload a hot dog candidate image")

if file_name is not None:
    col1, col2 = st.columns(2)

    image = Image.open(file_name)
```

```
col1.image(image, use_column_width=True)
predictions = pipeline(image)

col2.header("Probabilities")
for p in predictions:
    col2.subheader(f"{ p['label'] }: { round(p['score'] *
100, 1)}%")
```

This Python script uses a [HuggingFace Transformers pipeline](#) to load the **julien-c/hotdog-not-hotdog** model, which is used by the Streamlit interface.

## Embed Streamlit Spaces on other webpages

```
<iframe
  src="https://NimaBoscarino-hotdog-streamlit.hf.space?embed=
true"
  title="My awesome Streamlit Space"
></iframe>
```

## Static HTML Spaces

Spaces also accommodate custom HTML for your app instead of using Streamlit or Gradio. Set `sdk: static` inside the `YAML` block at the top of your Spaces **README.md** file. Then you can place your HTML code within an **index.html** file.

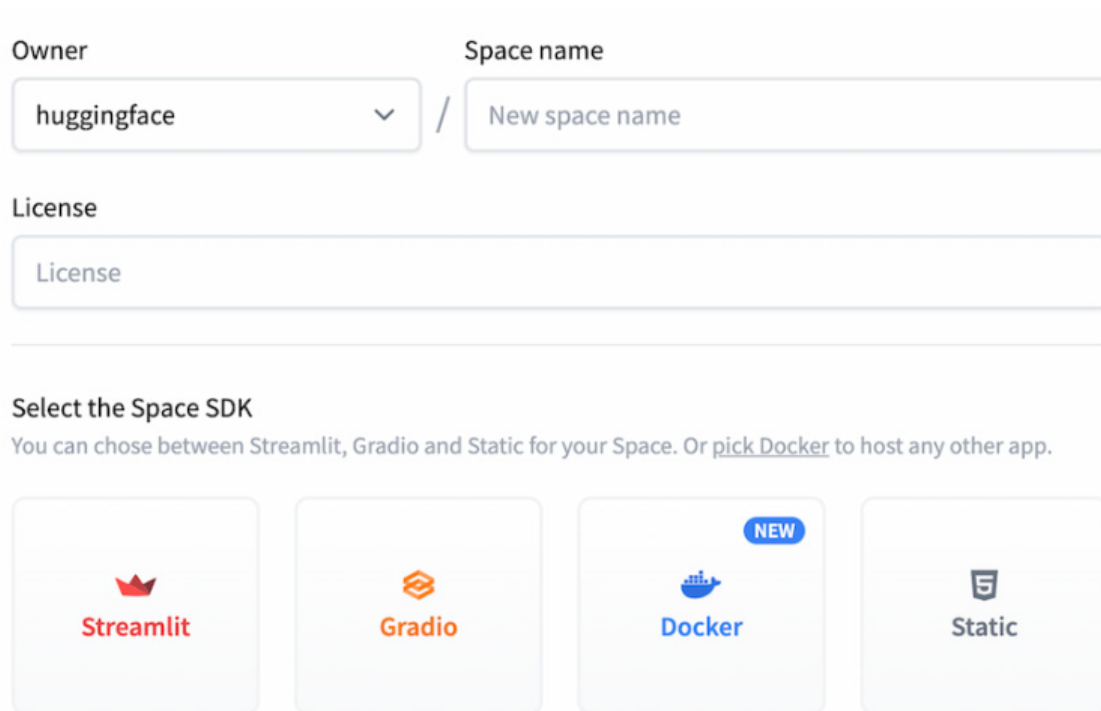
## Docker Spaces

Spaces accommodate custom **Docker containers** for apps outside the scope of Streamlit and Gradio. Docker Spaces allow users to go beyond the limits of what was previously possible with the standard SDKs. From FastAPI and Go endpoints

to Phoenix apps and ML Ops tools, Docker Spaces can help in many different setups.

## Create a new Docker Space

Start by **creating a brand new Space** and choosing **Docker** as our SDK.



The screenshot shows the 'Create a new Space' form on the Hugging Face website. It includes fields for 'Owner' (set to 'huggingface'), 'Space name' (placeholder 'New space name'), and 'License' (placeholder 'License'). Below these is a section titled 'Select the Space SDK' with a descriptive text: 'You can chose between Streamlit, Gradio and Static for your Space. Or pick Docker to host any other app.' There are four buttons: 'Streamlit' with a red crown icon, 'Gradio' with an orange cube icon, 'Docker' with a blue ship icon and a 'NEW' badge, and 'Static' with a grey shield icon.

## Create the app

```
from fastapi import FastAPI

app = FastAPI()

@app.get("/")
def read_root():
    return {"Hello": "World!"}
```

## Create the Dockerfile

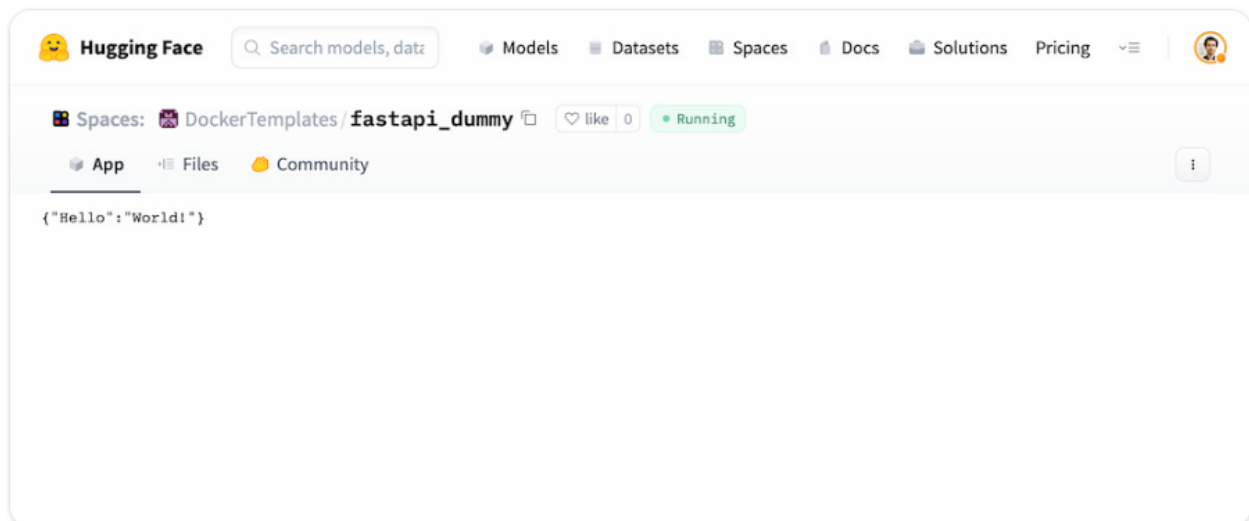
```
FROM python:3.9

add -m -u 1000 user
WORKDIR /app

COPY --chown=user ./requirements.txt requirements.txt
RUN pip install --no-cache-dir --upgrade -r requirements.txt

COPY --chown=user . /app
CMD ["uvicorn", "main:app", "--host", "0.0.0.0", "--port", "7860"]
```

When the changes are saved, the Space will rebuild and your demo should be up after a couple of seconds! [Here](#) is an example result at this point.



## Conclusion:

**HuggingFace Spaces** is a powerful and accessible platform that can greatly simplify the process of sharing and collaborating on machine learning models and applications.



