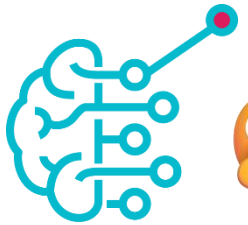


🤔 Which of the following is/are CORRECT?

A. If we want to use the model for prediction, we will pass in a list of input and these inputs correspond to the features that we use to build the model.

B. It is important to periodically retrain the model.

C. A model's performance on training data is always a good indicator of its performance on new, unseen data.

D. Re-training a model means starting from scratch with only new data.

E. You must always re-train a model if you want to make predictions on new data.

F. Overfitting occurs when a model performs well on training data but poorly on test data.

G. The prediction step does not require access to the original training data.

⭐ Multiple Choice

Temasek POLYTECHNIC
CREATING TOMORROW

School of Informatics & IT

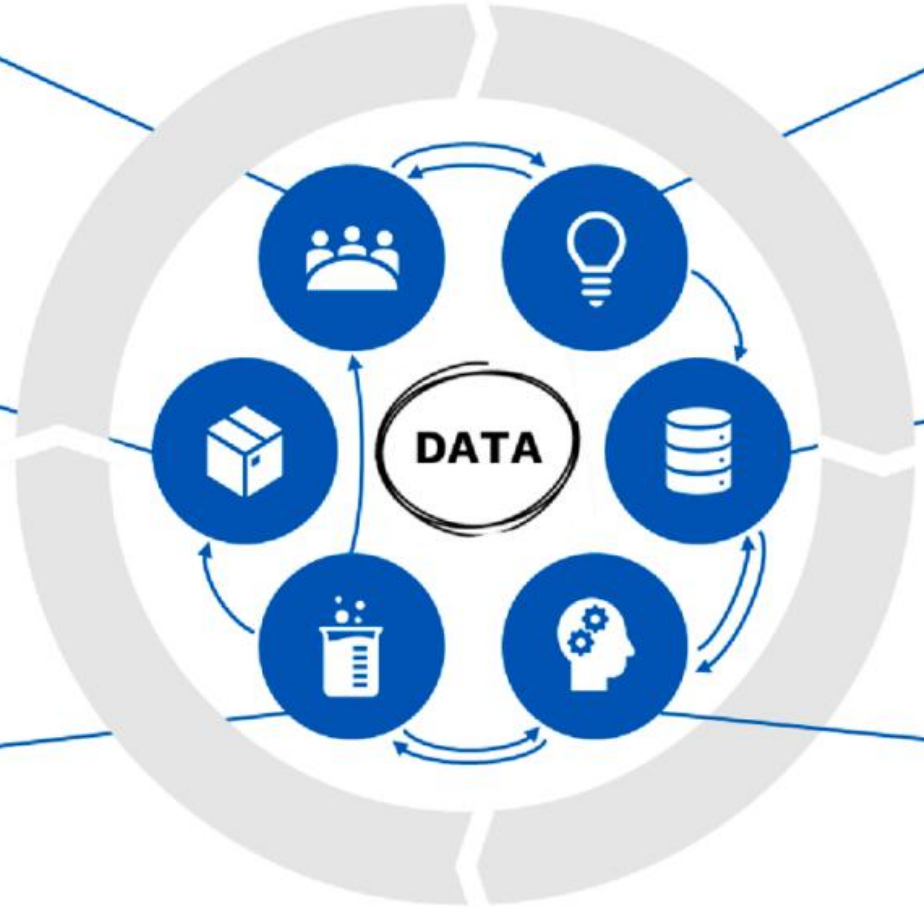
Model Deployment

Machine Learning / Data Science Life Cycle
Cross-Industry Standard Process for Data Mining (CRISP-DM)

1. Business Understanding
- Use case requirements and objectives
- Assess situation
- Development plan

2. Data Understanding
- Data sources identification
- Data gathering & integration
- First explorative & insights
- Data quality

6. Deployment
- Deployment plan
- Plan monitoring & maintenance
- Final reports and visualizations
- Project Review



3. Data Preparation
- Data cleaning & transformation
- Further data exploration
- Dimensionality reduction
- Features engineering

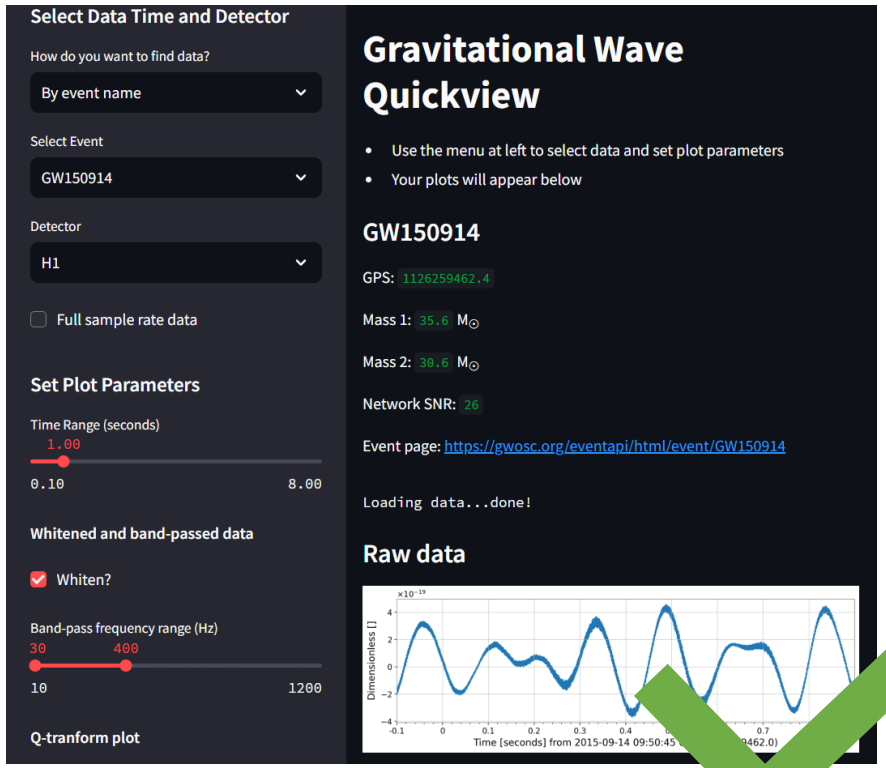
5. Evaluation
- Model testing
- Results evaluation
- Next steps determination

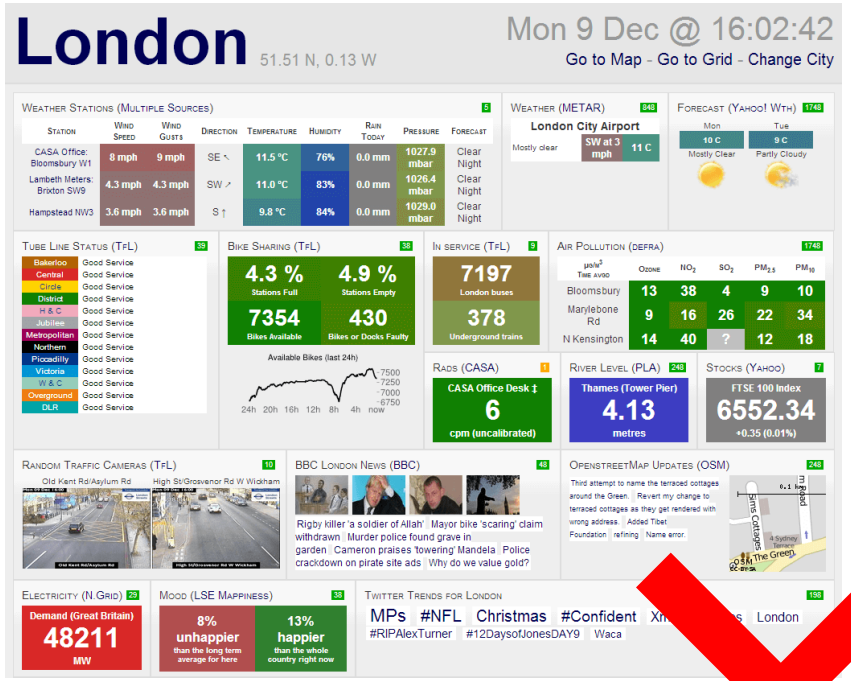
4. Modelling
- Model class selection & design
- Build test design
- Hyperparameter optimization
- Models training & validation

- https://www.edureka.co/blog/wp-content/uploads/2019/06/Data-Science-Project-Life-Cycle-Data-Science-Project-Edureka1-1.png

streamlit_app.py

- Design overall webpage aesthetics





- https://gw-quickview.streamlit.app/?ref=streamlit-io-gallery-favorites

streamlit_app.py

```
import joblib
model = joblib.load("model.pkl")
```

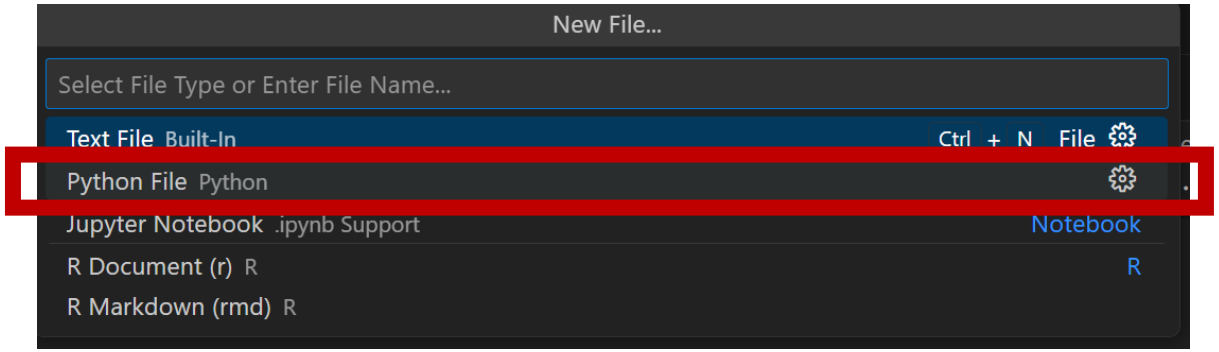
- Load trained model code

- Read new unseen data (user input data)
- Data pre-processing codes
- Prediction codes

Streamlit app development

- Code your **`app.py`** file **locally first**
  - You can code directly in **`*.py`**
    - Alternatively, code in **`*.ipynb`** first and then export it to **`*.py`**

  - Use `streamlit run app.py` to launch app locally and make adjustments
  - Once satisfied with the app, we will upload to Streamlit cloud for easier access



Streamlit template

- Ensure you have `joblib` library installed
  - Save trained model

```
joblib.dump(model, "model.pkl")
```

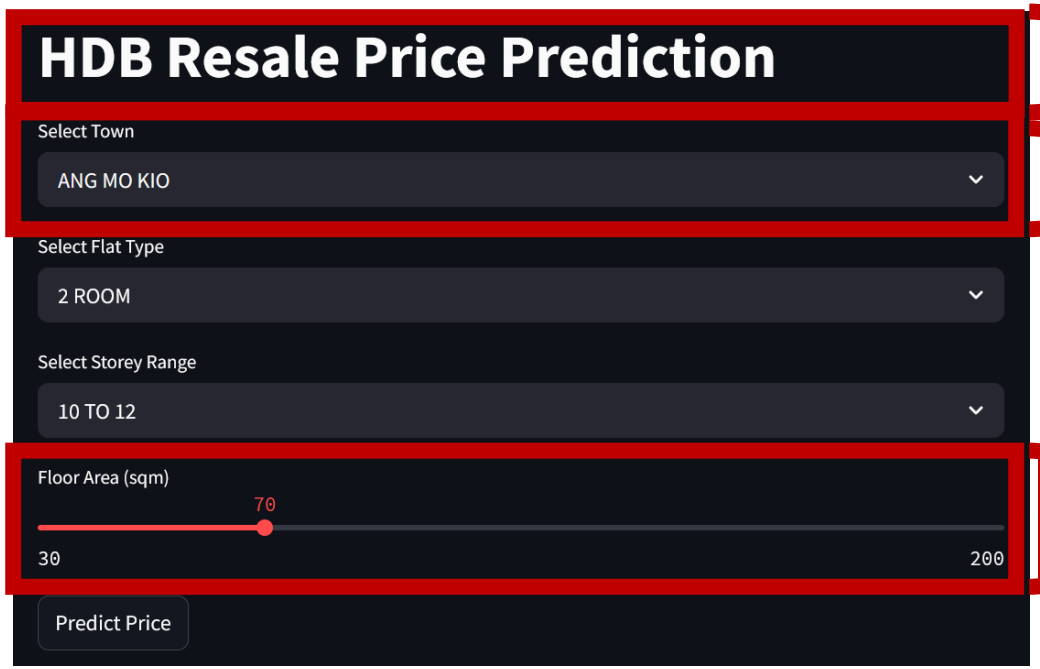
  - Load trained model

```
joblib.load("model.pkl")
```

- **Import all required libraries in the Python code**

```
import joblib
import streamlit as st
import numpy as np
import pandas as pd

## Load trained model
model = joblib.load("model.pkl")
```

Retrieving user inputs

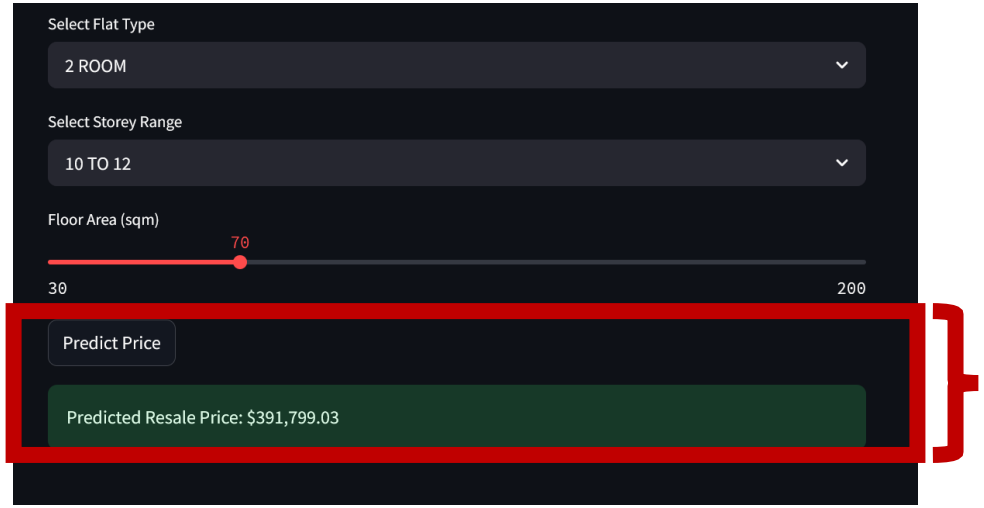



```
## Streamlit app
st.title("HDB Resale Price Prediction")
```

```
## User inputs
town_selected = st.selectbox("Select Town", towns)
```

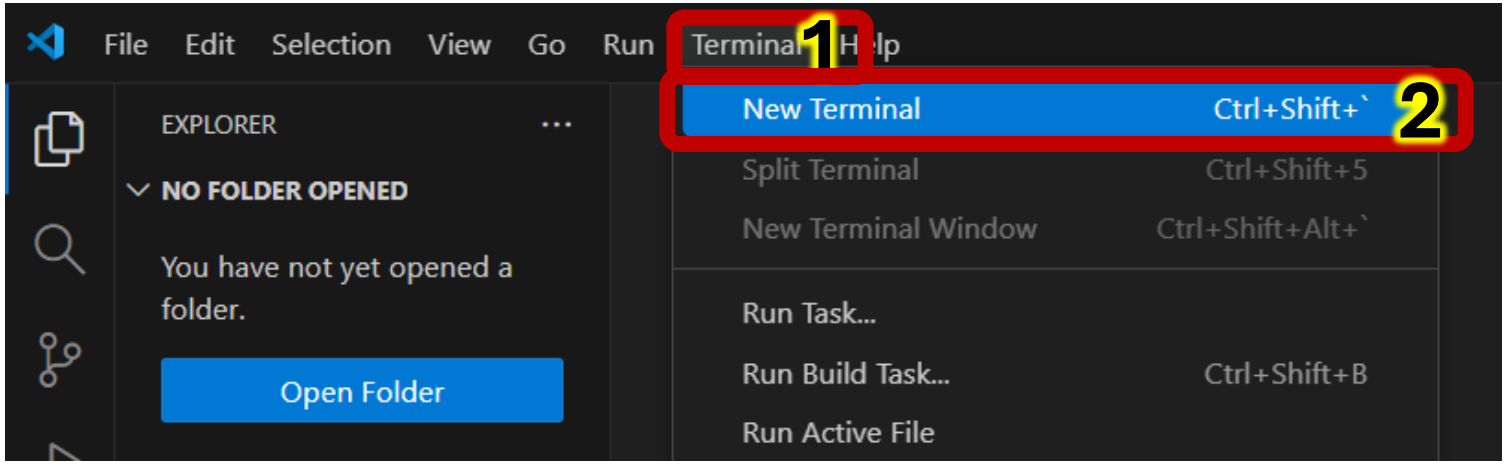
```
## User inputs
floor_area_selected = st.slider("Floor Area (sqm)",
                                min_value=30,
                                max_value=200,
                                value=70)
```

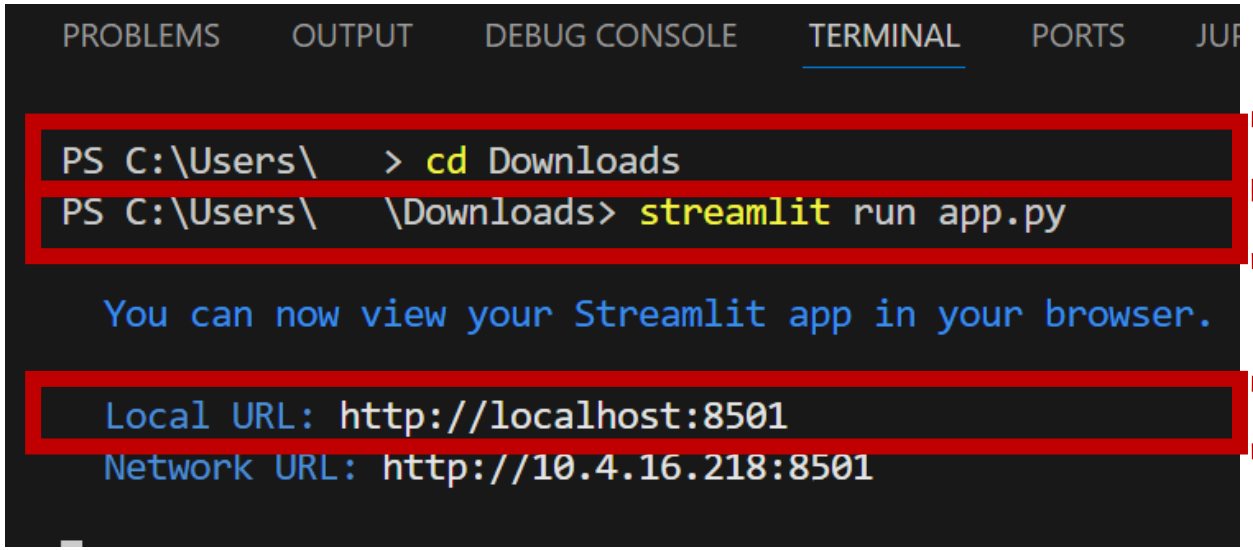
Predicting user input




```
## Predict button
if st.button("Predict HDB price"):

    ## Create dict for input features
    input_data = {
        'town': town_selected, ...
    }

    ## Convert input data to a DataFrame
    df_input = pd.DataFrame({
        'town': [town_selected], ...
    })

    ## Data preprocessing
    ...


    ## Predict
    y_unseen_pred = model.predict(df_input)[0]
    st.success(f"Predicted Price: ${y_unseen_pred:,.2f}")
```

Testing out `streamlit_app.py` locally





Change directory to where the `app.py` and `model.pkl` files are

```
streamlit run app.py
```

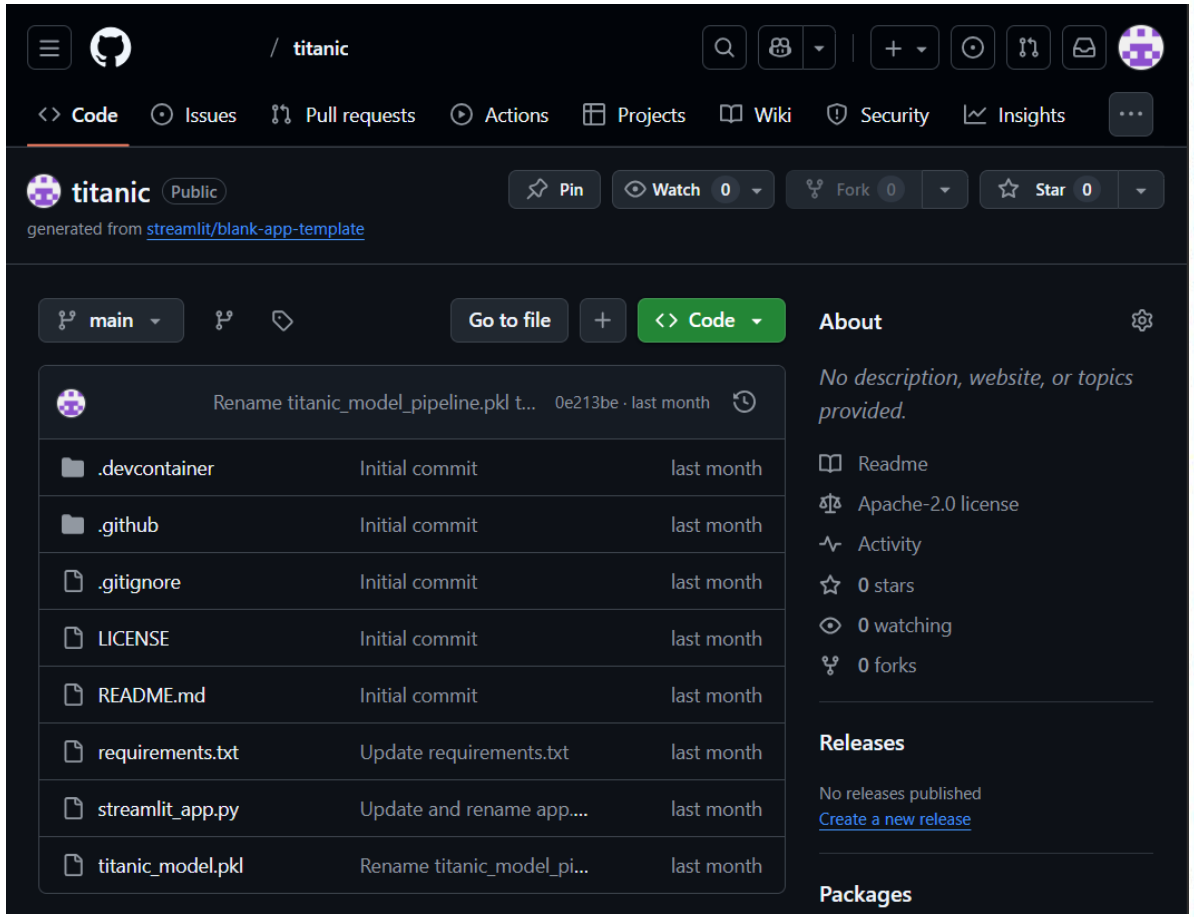
Hold **Ctrl** and click on the Local URL to view your app

Uploading files to GitHub

1. Create GitHub account to store the model and code files
2. Create **PUBLIC** repository and upload the following
   - `requirements.txt`
   - `streamlit_app.py`
   - `model.pkl`
   - Supporting files

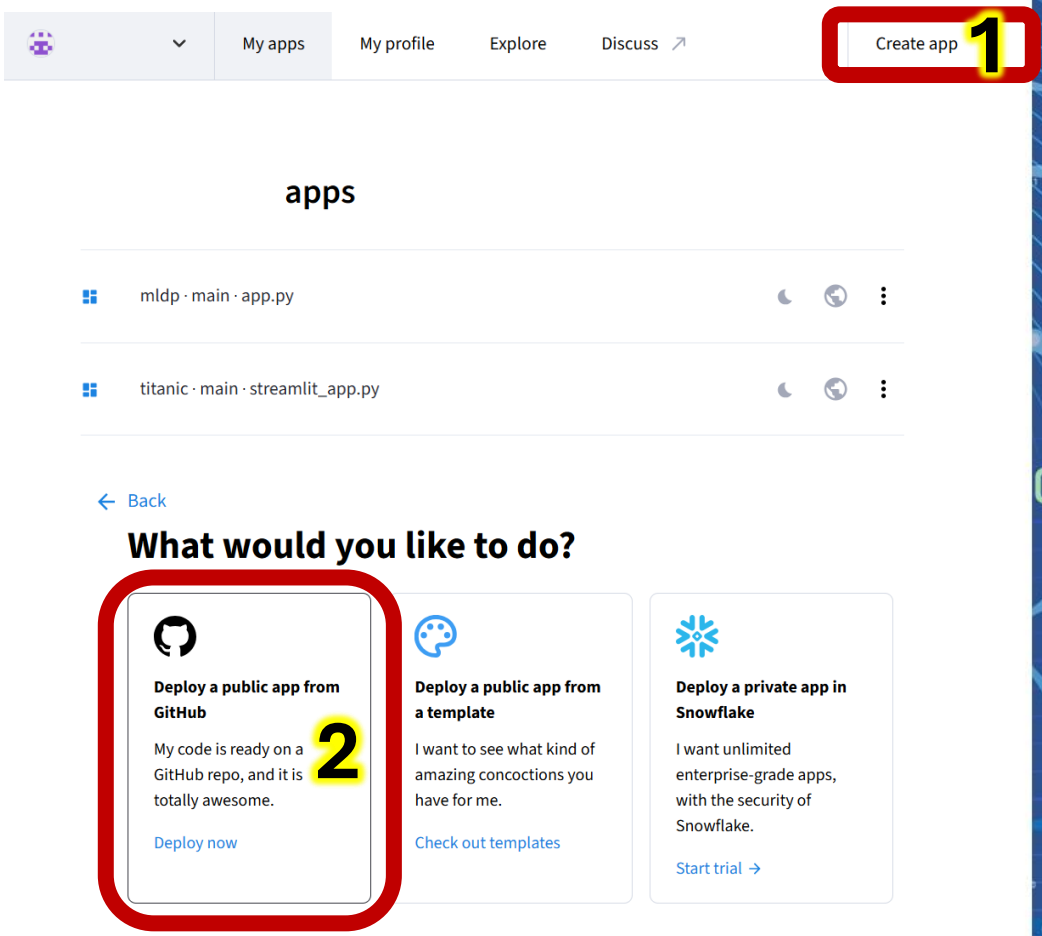
Files required to be uploaded to GitHub

- `requirements.txt`
  - State all required libraries in Python code so that Streamlit can install the libraries and run the code
- `streamlit_app.py`
  - Streamlit app (user interface) code
- `model.pkl`
  - Trained model
- Supporting files
  - Background image
  - Dataset (if you want to show additional features in your app)



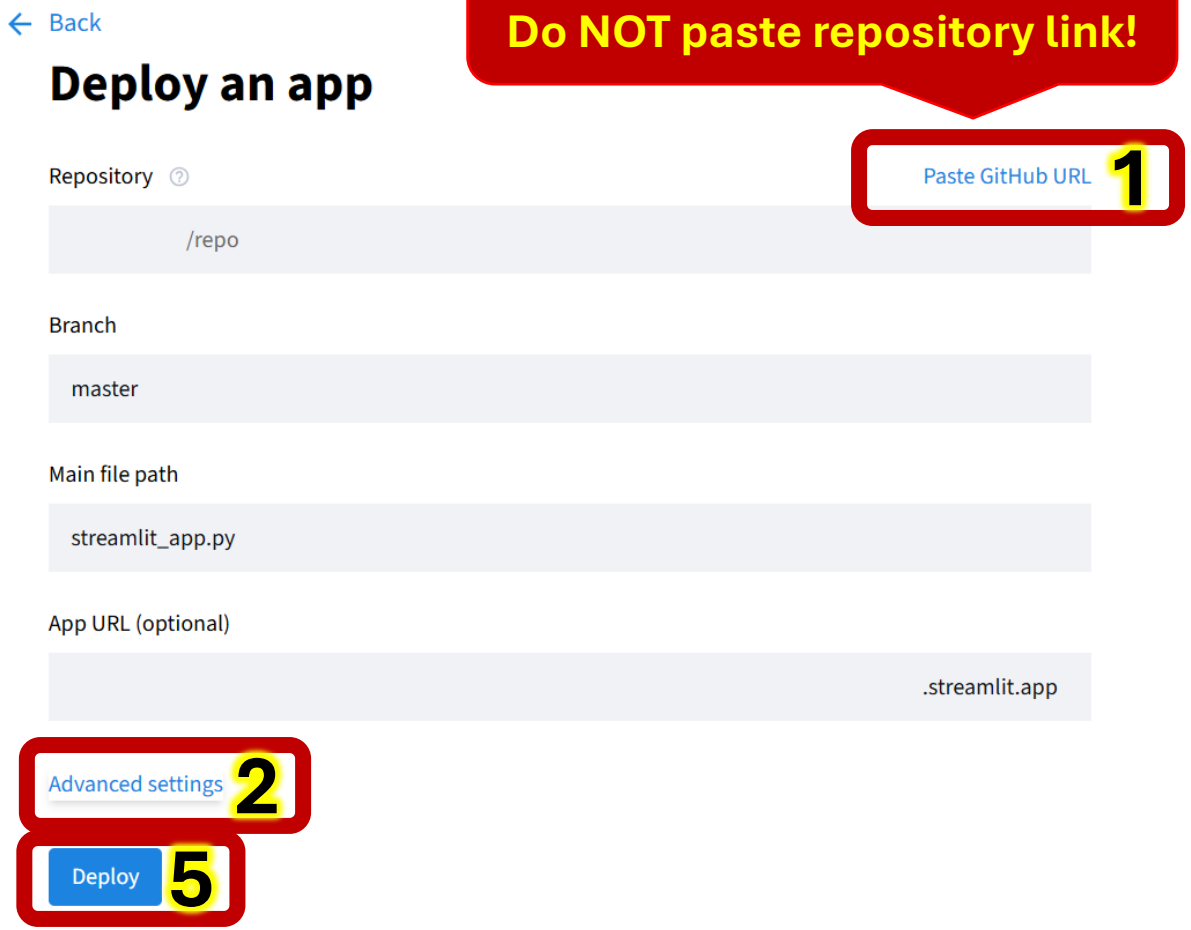
Creating Streamlit app

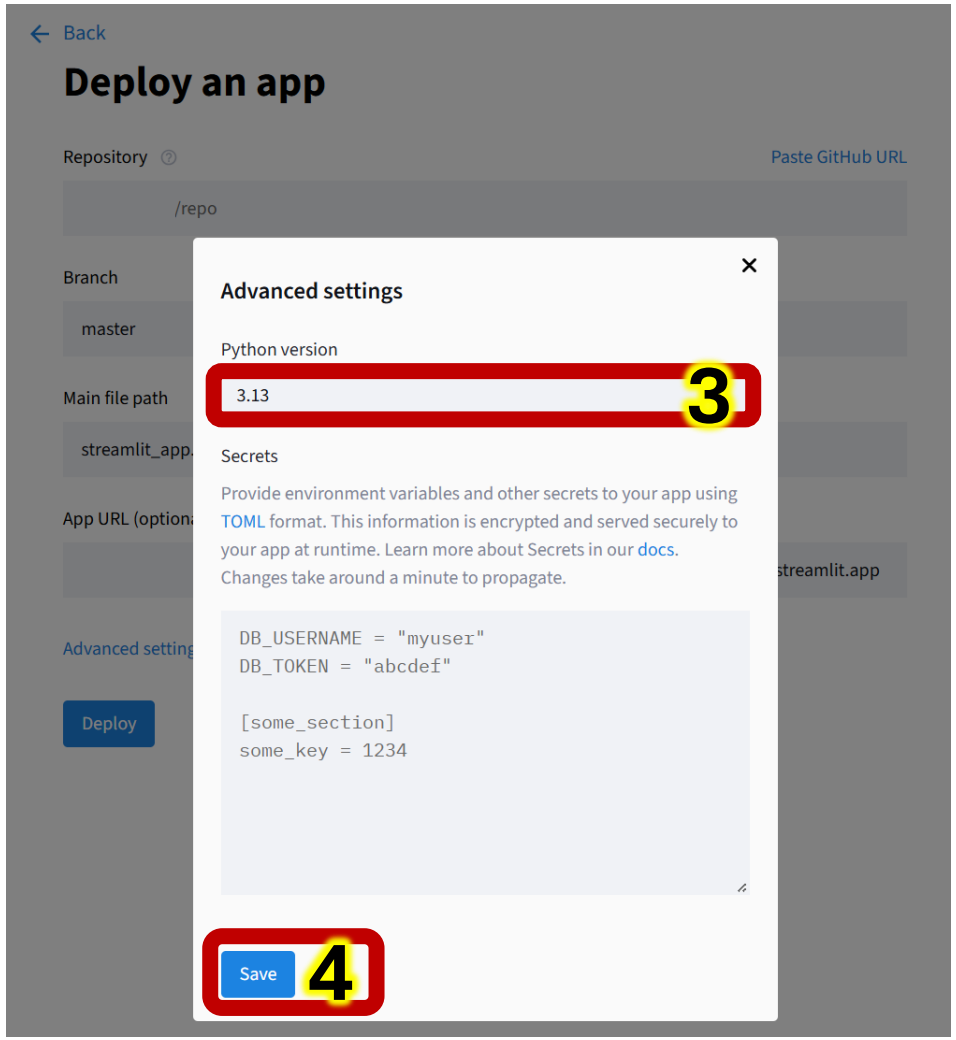
1. Create Streamlit account
2. Create app
3. Deploy a public app from GitHub



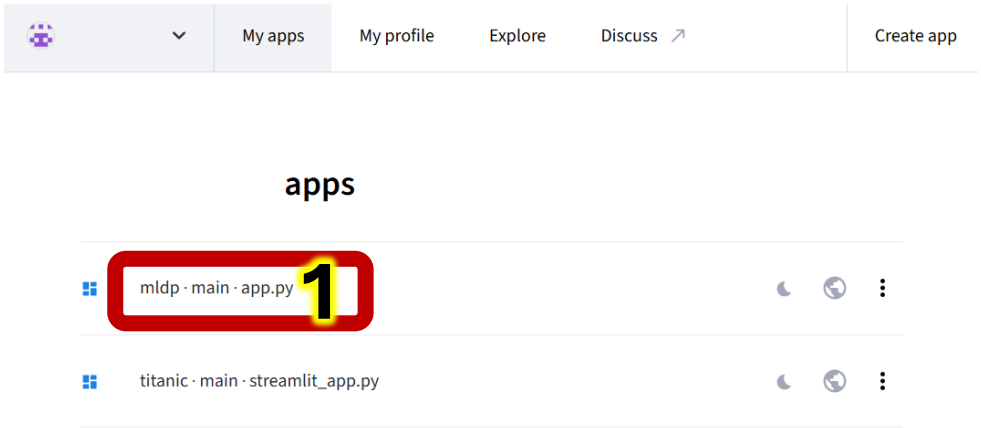
Deploying Streamlit app

Copy **app.py** link from GitHub
Do NOT paste repository link!





Starting Streamlit app



Project Specification Deployment Rubrics

Component	Evaluation Criteria	A (≥ 80%)	B (70 to <80%)	C (60 to <70%)	D (50 to <60%)	F (<50%)
Deployment of Model (10%)	**Correct prediction of output without error (4%)**	• Web app produces correct outputs for all tested cases • No errors or crashes • Include input validation and user-facing error message(s)	• Web app produces correct outputs in all tested cases • Minor warnings that do not affect outputs	• Web app produces correct outputs for most tested cases • Occasional errors or crashes	• Web app produces correct outputs for some tested cases • Frequent errors or crashes	• Web app fails to produce correct outputs • Persistent errors or crashes • Unusable for intended purpose

- Error: Prediction is counterintuitive with the inputs
  - e.g. Increase `sqft` but `houseprice` decreases

Project Specification Deployment Rubrics

Component	Evaluation Criteria	A (≥ 80%)	B (70 to <80%)	C (60 to <70%)	D (50 to <60%)	F (<50%)
Deployment of Model (10%)	**Interactive application (6%)**	• Highly interactive, responsive, and user-friendly app • All interactive elements (buttons, forms, feedback) work smoothly and enhance user experience • Design and visuals are appealing, and labels are suitable for target audience	• Interactive, responsive, and user-friendly app • Design and visuals are consistent (e.g., at most 2 minor issues), and labels are mostly clear (e.g., at most 2 vague terms) for target audience	• Interactive, responsive, and user-friendly app • Some elements may not respond as expected; user experience is acceptable but not engaging	• Largely static interface	• No interactivity • Present many broken elements that do not work or respond poorly; user experience is frustrating

- Appealing design and visuals
  - e.g., Blank vs appropriate background image

- Suitable for target audience
  - e.g., Labels easily understood?
    - Instead of typing in BMI, ask for height and weight instead.

E-Learning & References

- Choose a Streamlit app from the gallery
  - Comment on the aesthetics and user interface:
    - 2 areas that are done well
    - 1 area for improvement

Topic	Source
Streamlit	• https://streamlit.io/gallery • https://docs.streamlit.io/deploy/streamlit-community-cloud/get-started • https://www.datacamp.com/tutorial/streamlit • https://demo-ai-assistant.streamlit.app/?ref=streamlit-io-gallery-favorites
GitHub	• https://learn.microsoft.com/en-us/training/modules/introduction-to-github/ • https://learn.microsoft.com/en-us/training/modules/upload-project-github/

Learning Outcomes

1. ✅ [Machine Learning Fundamentals] Explain the fundamental concepts of machine learning;
    - 1.1 ✅ describe machine learning and its goals related to learning;
    - 1.2 ✅ identify different data types and characteristics or data;
    - 1.3 ▶️ explain the learning tasks and performance of machine learning algorithms.

2. ✅ [Machine Learning Development] Apply machine learning methods to solve problems;
    - 2.1 ✅ explain machine learning concepts;
    - 2.2 ✅ describe machine learning algorithms;
    - 2.3 ✅ perform data analysis and necessary data cleaning;
    - 2.4 ✅ train machine learning models;
    - 2.5 ✅ evaluate machine learning models.

Learning Outcomes

3. ✅ [Model Deployment] Deploy a machine learning model to an application;
    3.1 ✅ identify a platform and framework for the deployment of an application;
    3.2 ✅ develop an application on a platform;
    3.3 ✅ deploy a machine learning model on to an online platform.