In [1]:
```
# Step 1: Install Gradio (only once in Colab)
!pip install gradio --quiet

# Step 2: Imports
import pandas as pd
import numpy as np
import gradio as gr
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline

# Step 3: Load Dataset
df = pd.read_csv("/content/real_estate_rent_data.csv")  # replace with your pa
df = df.drop(columns=["Property_ID"])  # remove ID column

# Step 4: Prepare data
X = df.drop(columns=["Monthly_Rent"])
y = df["Monthly_Rent"]

# Simulate semi-supervised setup
X_labeled, X_unlabeled, y_labeled, _ = train_test_split(X, y, train_size=0.2,
y_unlabeled = pd.Series([np.nan] * len(X_unlabeled), index=X_unlabeled.index)

X_combined = pd.concat([X_labeled, X_unlabeled])
y_combined = pd.concat([y_labeled, y_unlabeled])

# Step 5: Preprocessing
preprocessor = ColumnTransformer([
    ("cat", OneHotEncoder(handle_unknown="ignore"), ["Location"]),
    ("num", "passthrough", ["Size_sqft", "Num_Bedrooms", "Num_Bathrooms"])
])

# Step 6: Model pipeline
model = Pipeline([
    ("prep", preprocessor),
    ("lr", LinearRegression())
])

# Step 7: Self-training loop
for i in range(3):  # 3 iterations
    known = ~y_combined.isna()
    unknown = y_combined.isna()

    model.fit(X_combined[known], y_combined[known])

    if unknown.sum() == 0:
        break

    preds = model.predict(X_combined[unknown])

    # Filter confident predictions (IQR)
```

```
    q1, q3 = np.percentile(preds, [25, 75])
    iqr = q3 - q1
    lb, ub = q1 - 1.5 * iqr, q3 + 1.5 * iqr
    confident = (preds > lb) & (preds < ub)

    pseudo_indices = X_combined[unknown].iloc[confident].index
    y_combined.loc[pseudo_indices] = preds[confident]

# Step 8: Final model trained on all known + confident pseudo-labels
model.fit(X_combined[~y_combined.isna()], y_combined[~y_combined.isna()])

# Step 9: Gradio prediction function
def predict_rent(size, beds, baths, location):
    input_df = pd.DataFrame([[size, beds, baths, location]],
                            columns=["Size_sqft", "Num_Bedrooms", "Num_Bathroo
    rent = model.predict(input_df)[0]
    return f"🏠 Estimated Monthly Rent: ₹{int(rent):,}"

# Step 10: Launch Gradio UI
locations = sorted(df["Location"].unique().tolist())

gr.Interface(
    fn=predict_rent,
    inputs=[
        gr.Number(label="Size (sqft)", value=1000),
        gr.Number(label="Number of Bedrooms", value=2),
        gr.Number(label="Number of Bathrooms", value=1),
        gr.Dropdown(choices=locations, label="Location")
    ],
    outputs="text",
    title="🏠 Real Estate Rent Estimator",
    description="Predict monthly rent using a self-learning regression model t
).launch()
```

```
It looks like you are running Gradio on a hosted a Jupyter notebook. For the Gr
adio app to work, sharing must be enabled. Automatically setting `share=True`
(you can turn this off by setting `share=False` in `launch()` explicitly).

Colab notebook detected. To show errors in colab notebook, set debug=True in la
unch()
* Running on public URL: https://2a7f31529df265002d.gradio.live

This share link expires in 1 week. For free permanent hosting and GPU upgrades,
run `gradio deploy` from the terminal in the working directory to deploy to Hug
ging Face Spaces (https://huggingface.co/spaces)
```

Out[1]: