That's the perfect next question. Using your saved model to make predictions on new data is called **inference**. It's a straightforward process.

Here’s exactly how you can use your saved .pkl file to predict the severity for a new patient based on their vitals.

## The Prediction Process

The process involves three main steps:

1. **Load** your saved xgboost_baseline_71bal_acc.pkl model.
2. **Format** the new patient's vitals into the exact same 20-feature structure the model was trained on. This is the most critical step.
3. **Use** the loaded model to make a prediction.

## Prediction Script (predict_severity.py)

Here is a complete, runnable Python script that shows you how to do this. You can save this code in a new file named predict_severity.py in your project folder.

Python

```python
import joblib
import pandas as pd
import numpy as np

# --- 1. LOAD THE SAVED MODEL ---
# Make sure you have renamed your best model file to this name
model_filename = 'xgboost_baseline_71bal_acc.pkl'
print(f"Loading model: {model_filename}...")
model = joblib.load(model_filename)
print("✅ Model loaded successfully.")
```

```
# --- This is the same function from your training script ---
# The new data MUST go through the same feature engineering steps.
def add_engineered_features(df):
    df = df.copy()
    if {"heartrate", "sbp"}.issubset(df.columns):
        sbp_safe = df["sbp"].replace(0, np.nan)
        df["shock_index"] = df["heartrate"] / sbp_safe
    if {"sbp", "dbp"}.issubset(df.columns):
        df["pulse_pressure"] = df["sbp"] - df["dbp"]
    if "o2sat" in df.columns:
        df["hypoxia_flag"] = (df["o2sat"] < 90).astype(int)
    if "temperature" in df.columns:
        df["fever_flag"] = (df["temperature"] >= 38).astype(int)
    return df


# --- 2. PREPARE THE NEW PATIENT'S DATA ---
# This is an example of vitals you might get from sensors for one patient.
# IMPORTANT: You must provide a value for all 20 original features.
# If a value is missing, you can use np.nan, and we will fill it.
new_patient_data = {
    'temperature': 39.1,
    'heartrate': 125,
    'resprate': 28,
    'o2sat': 88,
    'sbp': 95,
    'dbp': 60,
    'heartrate_median': 125,
    'heartrate_min': 120,
    'heartrate_max': 130,
    'resprate_median': 28,
    'o2sat_median': 89,
    'o2sat_min': 87,
    'temperature_median': 39.1,
    'sbp_median': 95,
    'dbp_median': 60,
    'pain_median': 8,
    # The following engineered features will be calculated by our function
    'shock_index': np.nan,
    'pulse_pressure': np.nan,
    'hypoxia_flag': np.nan,
    'fever_flag': np.nan
}
```

```
# Convert the dictionary to a pandas DataFrame
input_df = pd.DataFrame([new_patient_data])

# Run the feature engineering function
input_df_engineered = add_engineered_features(input_df)

# Ensure the column order is exactly the same as during training
# We can create a list of the original 20 feature names
original_features = [
    'temperature', 'heartrate', 'resprate', 'o2sat', 'sbp', 'dbp',
    'heartrate_median', 'heartrate_min', 'heartrate_max', 'resprate_median',
    'o2sat_median', 'o2sat_min', 'temperature_median', 'sbp_median',
    'dbp_median', 'pain_median', 'shock_index', 'pulse_pressure',
    'hypoxia_flag', 'fever_flag'
]
input_df_final = input_df_engineered[original_features]

print("\nPrepared input data for the model:")
print(input_df_final)


# --- 3. MAKE THE PREDICTION ---
print("\nPredicting severity...")

# Get the predicted class (0, 1, or 2)
prediction_index = model.predict(input_df_final)[0]

# Get the probabilities for each class
prediction_probabilities = model.predict_proba(input_df_final)[0]

# Map the index to a human-readable label
severity_map = {0: 'Critical', 1: 'Moderate', 2: 'Low Urgency'}
predicted_severity = severity_map[prediction_index]


# --- 4. DISPLAY THE RESULT ---
print("\n--- PREDICTION RESULT ---")
print(f"Predicted Severity: {predicted_severity}")
print("Confidence Scores:")
print(f"  - Critical:    {prediction_probabilities[0]:.2%}")
print(f"  - Moderate:    {prediction_probabilities[1]:.2%}")
print(f"  - Low Urgency: {prediction_probabilities[2]:.2%}")
print("-------------------------\n")
```

## How to Run It

1. **Save** the code above as predict_severity.py in your project folder.
2. Make sure your xgboost_baseline_71bal_acc.pkl file is in the same folder.
3. Run the script from your Anaconda Prompt:
   Bash
   ```
   python predict_severity.py
   ```

You can change the values in the new_patient_data dictionary to see how the model's prediction changes for different patient conditions.