**User Input**

The script asks the user for the medicine name via:

medicine_name = input("Enter the medicine name: ")

This is required to:

1. Filter the dataset,
2. Load the correct LSTM model file.

---

**Inference Workflow (What Happens Internally)**

**Step 1: Load Dataset & Model**

- Reads Excel file from:

C:\Users\Strix\Desktop\Boehm Tech\demand forecasting\data\demand_prediction_weekly1.xlsx

- Loads the medicine-specific trained LSTM model:

C:\Users\Strix\Desktop\Boehm Tech\demand forecasting\saved models\lstm_<medicine_name>.keras

---

**Step 2: Filter Weekly Data for the Selected Medicine**

The dataset is filtered by Product_Name and grouped by Week to obtain:

| Week | Total_Quantity |
|---|---|
| 2024-W40 | 510 |
| 2024-W41 | 480 |
| ... | ... |

This ensures **one weekly quantity per week**.

---

**Step 3: Scale Data (Normalization)**

LSTM models require scaled inputs.

scaler = MinMaxScaler()

scaled_data = scaler.fit_transform(product_df[['Total_Quantity']])

Data is scaled between **0 and 1**.

---

**Step 4: Prepare Input Sequence for LSTM Prediction**

The model was trained using the last **4 weeks** as a sequence (time_steps = 4).

So, the input for inference is:

[last 4 weeks demand] → LSTM → next-week prediction

This sequence is reshaped into:

(1, 4, 1) → [batch, time_steps, features]

---

**Step 5: Iterative Forecasting Procedure**

The model predicts one week at a time.

For each prediction:

1. LSTM outputs the next week's demand in **scaled** format.
2. Prediction is appended to future_predictions.
3. The sliding window moves forward:
    - Remove the oldest value in the sequence
    - Append the new predicted value

next_pred = model.predict(last_sequence)

last_sequence = [... updated with next prediction ...]

4. After all predictions are generated, values are **inverse-transformed** back to original scale:

future_predictions = scaler.inverse_transform(...)