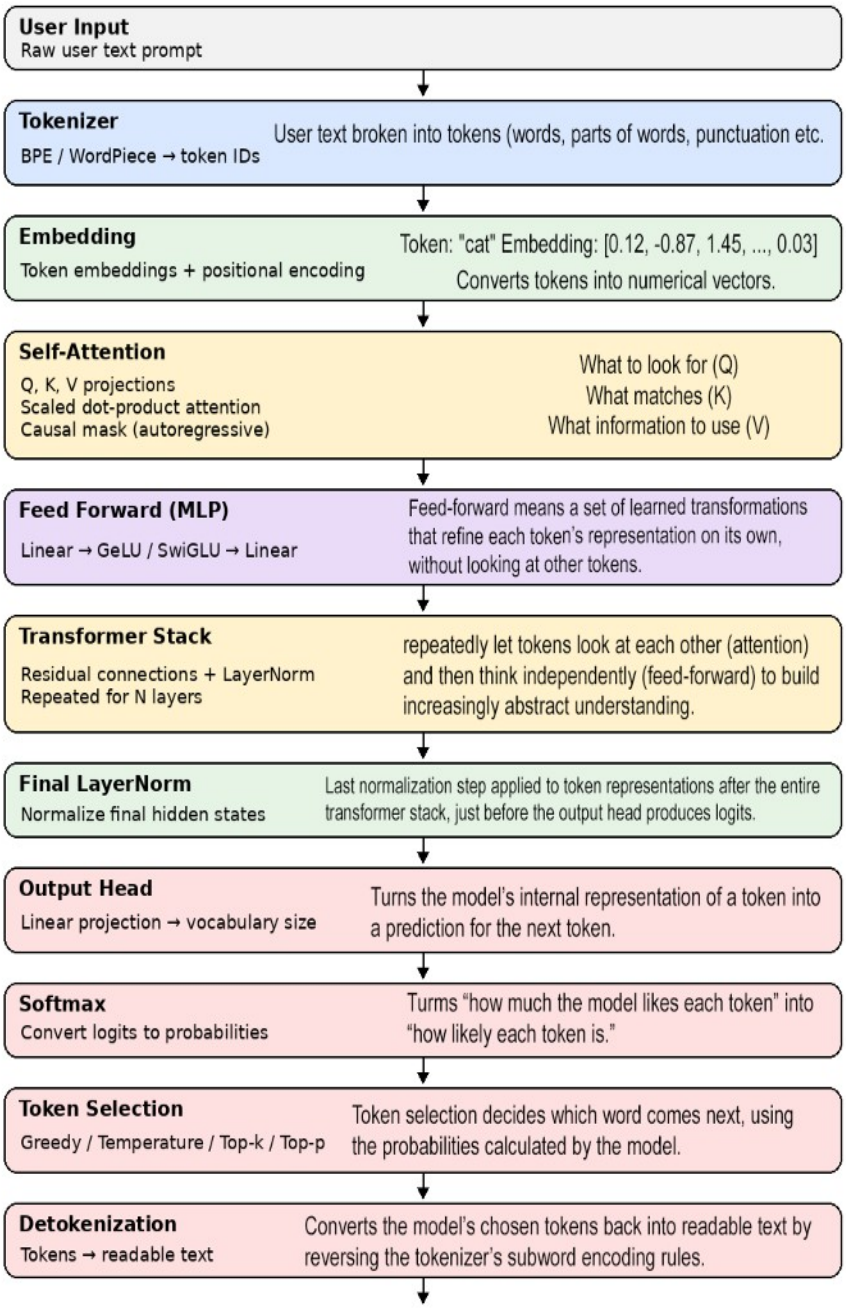
**Copilot Dialog:**

Understanding the process that AI LLMs from user input to the final response.

Prompt given to Microsoft Copilot:

***Produce a flowchart of the LLM process from user input all the way through output head and softmax.***

Below is a complete flowchart of the LLM (Transformer-based) inference process, starting from user input and ending at the output head and softmax. I'll first give you a clear visual flowchart, then a concise explanation of each block so it's useful for teaching, slides, or documentation.



**What Each Stage Is Doing (Brief but Precise)**

### 1. **Tokenizer**

* Converts raw text → token IDs
* Uses **subword units** (BPE, SentencePiece, WordPiece)
* Vocabulary size typically 30k–100k

***

### 2. **Token & Positional Embeddings**

* Token IDs → dense vectors (`d_model`, e.g. 4096)
* Positional information added:
  * Absolute embeddings
* Rotary (RoPE) or ALiBi (common in modern LLMs)

“An embedding matrix can be thought of as placing tokens into an abstract geometric space—often visualized as 3D for intuition, but actually thousands of dimensions—where distance and direction encode meaning.”

***

### 3. **Transformer Blocks (Core Computation)**

Each layer refines token representations with context.

**Self-Attention**

* Creates **Q, K, V**
* Computes attention weights:
  `softmax($QK^T$ / √d_k)`
* Causal mask prevents “looking ahead”

**MLP / Feed-Forward**

* Nonlinear transformation
* Major source of model capacity

**Residuals + LayerNorm**

* Stabilize training and inference
* Preserve information flow

***

### 4. **Final LayerNorm**

*   Normalizes final hidden states before prediction

***

### 5. **Output Head**
*   Linear projection:
    `hidden_state → vocab_size`
*   Produces **logits** (unnormalized scores)

***

### 6. **Softmax**

*   Converts logits into probabilities over the vocabulary

***

### 7. **Token Selection**

*   Strategies:
    *   Greedy (argmax)
    *   Temperature sampling
    *   Top-k / Top-p (nucleus)

***

### 8. **Detokenization**

*   Tokens → human-readable text
*   Loop repeats until stop token