6. BERT Embeddings with Transformers

In [ ]:
```
# Import the BERT tokenizer and model from Hugging Face Transformers
from transformers import BertTokenizer, BertModel
```

In [2]:
```
# Load the pre-trained BERT tokenizer and model (uncased, i.e., lowercase)
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
model = BertModel.from_pretrained('bert-base-uncased')
```

```
Xet Storage is enabled for this repo, but the 'hf_xet' package is not installed. Falling back to regular HTTP download. For better performance, install the package with: `pip install huggingface_hub[hf_xet]` or `pip install hf_xet`
model.safetensors:   0%|          | 0.00/440M [00:00<?, ?B/s]
d:\Academics\Aissms\B.Tech.Docs\Sem-7\Practical\BE-Practical-Practice\GenAi\.venv\lib\site-packages\huggingface_hub\file_download.py:143: UserWarning: `huggingface_hub` cache-system uses symlinks by default to efficiently store duplicated files but your machine does not support them in C:\Users\ASUS\.cache\huggingface\hub\models--bert-base-uncased. Caching files will still work but in a degraded version that might require more space on your disk. This warning can be disabled by setting the `HF_HUB_DISABLE_SYMLINKS_WARNING` environment variable. For more details, see https://huggingface.co/docs/huggingface_hub/how-to-cache#limitations.
To support symlinks on Windows, you either need to activate Developer Mode or to run Python as an administrator. In order to activate developer mode, see this article: https://docs.microsoft.com/en-us/windows/apps/get-started/enable-your-device-for-development
  warnings.warn(message)
```

In [ ]:
```
# Tokenize the input sentence and prepare it for the BERT model
inputs = tokenizer("Generative AI creates realistic images", return_tensors="pt")
# Pass the tokenized input through the BERT model to get embeddings
outputs = model(**inputs)
```

In [4]:
```
# Print the shape of the output tensor (tokens × embedding size)
print("BERT Output Shape:", outputs.last_hidden_state.shape)

# Print the first 5 values of the embedding for the first token
print("First token embedding:", outputs.last_hidden_state[0][0][:5])
```

```
BERT Output Shape: torch.Size([1, 8, 768])
First token embedding: tensor([-0.2924,  0.0249, -0.1646,  0.2106, -0.0188], grad_fn=<SliceBackward0>)
```

In [ ]:
```

```