

Retrieval Augmented Generation



Context Windows

- **AI models have context windows that limit how much text you can feed them**
 - **Ex) Gemini 3 Flash and Pro have 1 million token context window – about 1,500 pages of text**
- **Even if you can fit your documents in that window, the AI can get “lost in the middle” and have trouble answering your queries**
- **We need a way for the AI to intelligently select the relevant parts of your data in order to answer your question**

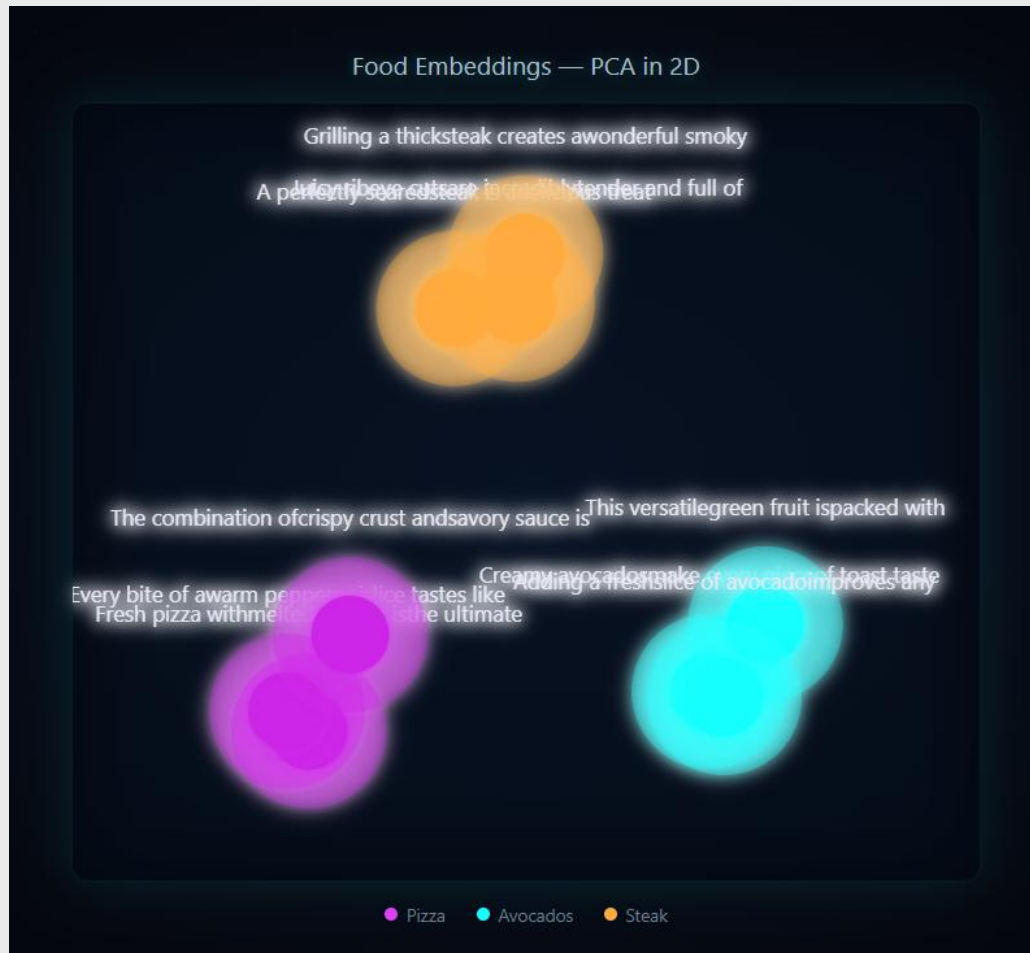
Retrieval Augmented Generation (RAG)

- **RAG allows an AI to selectively retrieve documents to answer your query**
- **All documents stored in a clever way in a special database**
- **This database uses clever techniques to find relevant documents for your query**

Text Embeddings

- **The key to RAG are AI powered text embeddings**
- **A text embedding maps text to a vector**
- **The vector location encodes the meaning of the text**

Embedding Example



Embedding Models

- We can use the gemini-embedding-001 model to embed text
- Other AIs have similar embedding models
- Embedding is 768 dimensions
- Python code:

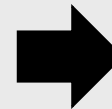
```
def get_embedding(text: str, client) -> list:
    """Generate 768-dim embedding (matches chat.py / rag_ingest.py style)."""
    result = client.models.embed_content(
        model="gemini-embedding-001",
        contents=text,
        config=types.EmbedContentConfig(
            task_type="RETRIEVAL_DOCUMENT",
            output_dimensionality=768,
        ),
    )
    return list[Any](result.embeddings[0].values)
```


Embedding Chunks

- We chop up the document in smaller chunks (maybe 1 or 2 pages)
 - Make chunks overlap a little bit so you don't cut important parts in the middle
- We embed each chunk one at a time
- Now we have a bunch of vectors, where do we store them?



```
def get_embedding(text: str, client) -> list:
    """Generate 768-dim embedding (matches chat.py / rag_ingest.py style)."""
    result = client.models.embed_content(
        model="gemini-embedding-001",
        contents=text,
        config=types.EmbedContentConfig(
            task_type="RETRIEVAL_DOCUMENT",
            output_dimensionality=768,
        ),
    )
    return list[Any](result.embeddings[0].values)
```



Vector Store

- **A vector store is a database where can store the text chunks and their vector embedding**
- **Vector store also has methods to let you rapidly search for vectors which are similar to an input vector**
 - Hierarchical navigable small world (HNSW)
 - Inverted File Index (IVF)
- **This lets us quickly find chunks similar to our query**
- **Many popular vector stores**
 - Pinecone
 - Chroma/Weaviate
 - Milvus
 - MongoDB 😊

RAG Workflow

- User query
- Embed query
- Search vector store for chunks similar to query

Whats so special about pizza?



$$[x_1 \ x_2 \ \dots \ x_m]$$

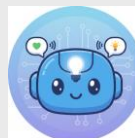


- Put the chunk text into the chat context

Whats so special about pizza?+ Pizza is delicious and satisfying + Nothing beats a hot slice of pizza + Pizza brings people together like nothing else



- AI replies to your query + relevant chunks



Pizza is tasty and brings us together

Embedding Cost

- Embeddings are very cheap

Google Gemini Pricing (February 2026)

Tier	Price per 1 Million Tokens	Rate Limits
Free Tier	\$0.00	Up to 1,500 requests per day (RPD).
Paid Tier	\$0.15	Up to 5,000,000 tokens per minute (TPM).
Batch API	\$0.075 (50% discount)	Optimized for massive, asynchronous datasets.

OpenAI Embedding Pricing (February 2026)

Tier	Price per 1 Million Tokens	Rate Limits (Tier 3 Example)
Standard (3-small)	\$0.02	Up to 5,000,000 tokens per minute (TPM).
Standard (3-large)	\$0.13	Up to 5,000,000 tokens per minute (TPM).
Batch API	\$0.01 – \$0.065 (50% discount)	Optimized for massive, non-urgent datasets.

MongoDB Vector Store

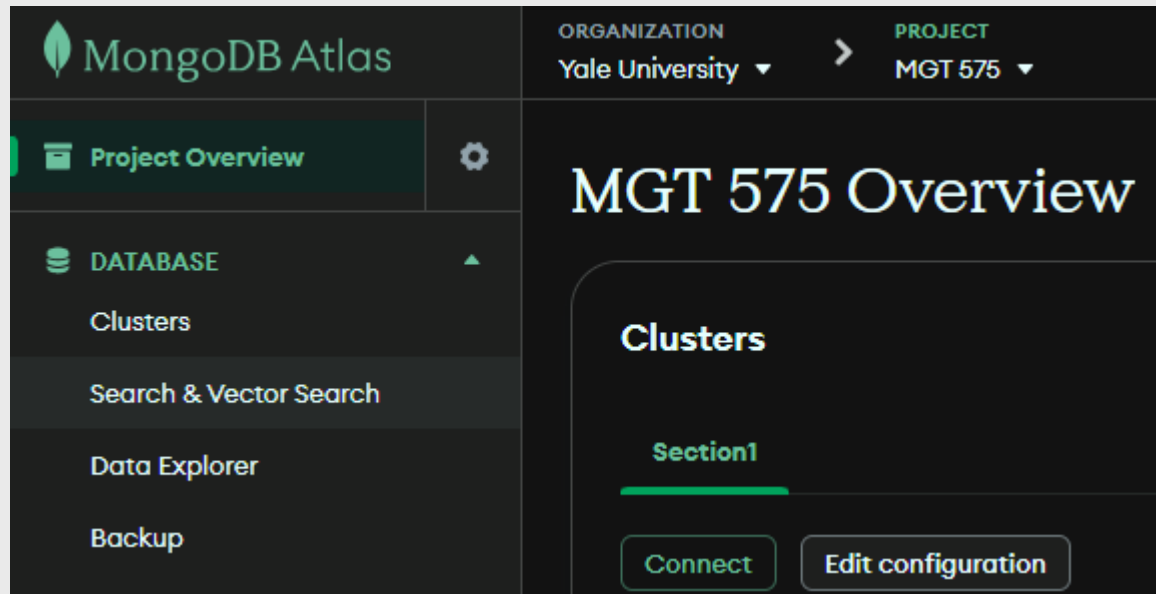
- After we embed the chunks with AI, we store them in a collection on MongoDB
 - Chunk text
 - Embedding vector
 - Useful metadata (can be AI generated, like summarize the chunk)

```
_id: ObjectId('699bc2d4145c585e9554853b')
caseName: "Nathan's Famous"
pageNumber: 1
content: "<p>Yale SCHOOL OF MANAGEMENT</p>
<p>YALE CASE 20-020 JULY 4, 2020</p>
_"
summary: "The case examines a dramatic shift on Nathan's Famous's balance sheet,_"
content_vector: Array (768)
  0: 0.006767863
  1: 0.003748851
  2: 0.01138478
  3: -0.061705638
  4: -0.0113745965
  5: 0.017322833
  6: -0.012120146
  7: -0.016870946
  8: -0.0032352665
  9: 0.0010216809
  10: 0.0051032566
  11: -0.011265037
  12: -0.0036041876
  13: 0.004373133
  14: 0.09579993
  15: 0.007351367
  16: 0.0039968383
  17: -0.0002347112
  18: -0.0022181787
  19: -0.01734156
  20: 0.0021275913
  21: 0.015317333
  22: 0.0048162555
  23: 0.0052397493
  24: 0.031737395
  ...
  743: 0.0000000000
  ...
  767: 0.0000000000
  768: 0.0000000000
  ...
  1535: 0.0000000000
  ...
  2303: 0.0000000000
  ...
  3071: 0.0000000000
  ...
  3839: 0.0000000000
  ...
  4607: 0.0000000000
  ...
  5375: 0.0000000000
  ...
  6143: 0.0000000000
  ...
  6911: 0.0000000000
  ...
  7679: 0.0000000000
  768: 0.0000000000
  ...
  1535: 0.0000000000
  ...
  2303: 0.0000000000
  ...
  3071: 0.0000000000
  ...
  3839: 0.0000000000
  ...
  4607: 0.0000000000
  ...
  5375: 0.0000000000
  ...
  6143: 0.0000000000
  ...
  6911: 0.0000000000
  ...
  7679: 0.0000000000
  ...
  8447: 0.0000000000
  ...
  9215: 0.0000000000
  ...
  9983: 0.0000000000
  ...
  10751: 0.0000000000
  ...
  11519: 0.0000000000
  ...
  12287: 0.0000000000
  ...
  13055: 0.0000000000
  ...
  13823: 0.0000000000
  ...
  14591: 0.0000000000
  ...
  15359: 0.0000000000
  ...
  16127: 0.0000000000
  ...
  16895: 0.0000000000
  ...
  17663: 0.0000000000
  ...
  18431: 0.0000000000
  ...
  19199: 0.0000000000
  ...
  19967: 0.0000000000
  ...
  20735: 0.0000000000
  ...
  21503: 0.0000000000
  ...
  22271: 0.0000000000
  ...
  23039: 0.0000000000
  ...
  23807: 0.0000000000
  ...
  24575: 0.0000000000
  ...
  25343: 0.0000000000
  ...
  26111: 0.0000000000
  ...
  26879: 0.0000000000
  ...
  27647: 0.0000000000
  ...
  28415: 0.0000000000
  ...
  29183: 0.0000000000
  ...
  29951: 0.0000000000
  ...
  30719: 0.0000000000
  ...
  31487: 0.0000000000
  ...
  32255: 0.0000000000
  ...
  33023: 0.0000000000
  ...
  33791: 0.0000000000
  ...
  34559: 0.0000000000
  ...
  35327: 0.0000000000
  ...
  36095: 0.0000000000
  ...
  36863: 0.0000000000
  ...
  37631: 0.0000000000
  ...
  38399: 0.0000000000
  ...
  39167: 0.0000000000
  ...
  39935: 0.0000000000
  ...
  40703: 0.0000000000
  ...
  41471: 0.0000000000
  ...
  42239: 0.0000000000
  ...
  43007: 0.0000000000
  ...
  43775: 0.0000000000
  ...
  44543: 0.0000000000
  ...
  45311: 0.0000000000
  ...
  46079: 0.0000000000
  ...
  46847: 0.0000000000
  ...
  47615: 0.0000000000
  ...
  48383: 0.0000000000
  ...
  49151: 0.0000000000
  ...
  49919: 0.0000000000
  ...
  50687: 0.0000000000
  ...
  51455: 0.0000000000
  ...
  52223: 0.0000000000
  ...
  52991: 0.0000000000
  ...
  53759: 0.0000000000
  ...
  54527: 0.0000000000
  ...
  55295: 0.0000000000
  ...
  56063: 0.0000000000
  ...
  56831: 0.0000000000
  ...
  57599: 0.0000000000
  ...
  58367: 0.0000000000
  ...
  59135: 0.0000000000
  ...
  59903: 0.0000000000
  ...
  60671: 0.0000000000
  ...
  61439: 0.0000000000
  ...
  62207: 0.0000000000
  ...
  62975: 0.0000000000
  ...
  63743: 0.0000000000
  ...
  64511: 0.0000000000
  ...
  65279: 0.0000000000
  ...
  66047: 0.0000000000
  ...
  66815: 0.0000000000
  ...
  67583: 0.0000000000
  ...
  68351: 0.0000000000
  ...
  69119: 0.0000000000
  ...
  69887: 0.0000000000
  ...
  70655: 0.0000000000
  ...
  71423: 0.0000000000
  ...
  72191: 0.0000000000
  ...
  72959: 0.0000000000
  ...
  73727: 0.0000000000
  ...
  74495: 0.0000000000
  ...
  75263: 0.0000000000
  ...
  76031: 0.0000000000
  ...
  76799: 0.0000000000
  ...
  77567: 0.0000000000
  ...
  78335: 0.0000000000
  ...
  79103: 0.0000000000
  ...
  79871: 0.0000000000
  ...
  80639: 0.0000000000
  ...
  81407: 0.0000000000
  ...
  82175: 0.0000000000
  ...
  82943: 0.0000000000
  ...
  83711: 0.0000000000
  ...
  84479: 0.0000000000
  ...
  85247: 0.0000000000
  ...
  86015: 0.0000000000
  ...
  86783: 0.0000000000
  ...
  87551: 0.0000000000
  ...
  88319: 0.0000000000
  ...
  89087: 0.0000000000
  ...
  89855: 0.0000000000
  ...
  90623: 0.0000000000
  ...
  91391: 0.0000000000
  ...
  92159: 0.0000000000
  ...
  92927: 0.0000000000
  ...
  93695: 0.0000000000
  ...
  94463: 0.0000000000
  ...
  95231: 0.0000000000
  ...
  95999: 0.0000000000
  ...
  96767: 0.0000000000
  ...
  97535: 0.0000000000
  ...
  98303: 0.0000000000
  ...
  99071: 0.0000000000
  ...
  99839: 0.0000000000
  ...
  100607: 0.0000000000
  ...
  101375: 0.0000000000
  ...
  102143: 0.0000000000
  ...
  102911: 0.0000000000
  ...
  103679: 0.0000000000
  ...
  104447: 0.0000000000
  ...
  105215: 0.0000000000
  ...
  105983: 0.0000000000
  ...
  106751: 0.0000000000
  ...
  107519: 0.0000000000
  ...
  108287: 0.0000000000
  ...
  109055: 0.0000000000
  ...
  109823: 0.0000000000
  ...
  110591: 0.0000000000
  ...
  111359: 0.0000000000
  ...
  112127: 0.0000000000
  ...
  112895: 0.0000000000
  ...
  113663: 0.0000000000
  ...
  114431: 0.0000000000
  ...
  115199: 0.0000000000
  ...
  115967: 0.0000000000
  ...
  116735: 0.0000000000
  ...
  117503: 0.0000000000
  ...
  118271: 0.0000000000
  ...
  119039: 0.0000000000
  ...
  119807: 0.0000000000
  ...
  120575: 0.0000000000
  ...
  121343: 0.0000000000
  ...
  122111: 0.0000000000
  ...
  122879: 0.0000000000
  ...
  123647: 0.0000000000
  ...
  124415: 0.0000000000
  ...
  125183: 0.0000000000
  ...
  125951: 0.0000000000
  ...
  126719: 0.0000000000
  ...
  127487: 0.0000000000
  ...
  128255: 0.0000000000
  ...
  129023: 0.0000000000
  ...
  129791: 0.0000000000
  ...
  130559: 0.0000000000
  ...
  131327: 0.0000000000
  ...
  132095: 0.0000000000
  ...
  132863: 0.0000000000
  ...
  133631: 0.0000000000
  ...
  134399: 0.0000000000
  ...
  135167: 0.0000000000
  ...
  135935: 0.0000000000
  ...
  136703: 0.0000000000
  ...
  137471: 0.0000000000
  ...
  138239: 0.0000000000
  ...
  139007: 0.0000000000
  ...
  139775: 0.0000000000
  ...
  140543: 0.0000000000
  ...
  141311: 0.0000000000
  ...
  142079: 0.0000000000
  ...
  142847: 0.0000000000
  ...
  143615: 0.0000000000
  ...
  144383: 0.0000000000
  ...
  145151: 0.0000000000
  ...
  145919: 0.0000000000
  ...
  146687: 0.0000000000
  ...
  147455: 0.0000000000
  ...
  148223: 0.0000000000
  ...
  148991: 0.0000000000
  ...
  149759: 0.0000000000
  ...
  150527: 0.0000000000
  ...
  151295: 0.0000000000
  ...
  152063: 0.0000000000
  ...
  152831: 0.0000000000
  ...
  153599: 0.0000000000
  ...
  154367: 0.0000000000
  ...
  155135: 0.0000000000
  ...
  155903: 0.0000000000
  ...
  156671: 0.0000000000
  ...
  157439: 0.0000000000
  ...
  158207: 0.0000000000
  ...
  158975: 0.0000000000
  ...
  159743: 0.0000000000
  ...
  160511: 0.0000000000
  ...
  161279: 0.0000000000
  ...
  162047: 0.0000000000
  ...
  162815: 0.0000000000
  ...
  163583: 0.0000000000
  ...
  164351: 0.0000000000
  ...
  165119: 0.0000000000
  ...
  165887: 0.0000000000
  ...
  166655: 0.0000000000
  ...
  167423: 0.0000000000
  ...
  168191: 0.0000000000
  ...
  168959: 0.0000000000
  ...
  169727: 0.0000000000
  ...
  170495: 0.0000000000
  ...
  171263: 0.0000000000
  ...
  172031: 0.0000000000
  ...
  172799: 0.0000000000
  ...
  173567: 0.0000000000
  ...
  174335: 0.0000000000
  ...
  175103: 0.0000000000
  ...
  175871: 0.0000000000
  ...
  176639: 0.0000000000
  ...
  177407: 0.0000000000
  ...
  178175: 0.0000000000
  ...
  178943: 0.0000000000
  ...
  179711: 0.0000000000
  ...
  180479: 0.0000000000
  ...
  181247: 0.0000000000
  ...
  182015: 0.0000000000
  ...
  182783: 0.0000000000
  ...
  183551: 0.0000000000
  ...
  184319: 0.0000000000
  ...
  185087: 0.0000000000
  ...
  185855: 0.0000000000
  ...
  186623: 0.0000000000
  ...
  187391: 0.0000000000
  ...
  188159: 0.0000000000
  ...
  188927: 0.0000000000
  ...
  189695: 0.0000000000
  ...
  190463: 0.0000000000
  ...
  191231: 0.0000000000
  ...
  191999: 0.0000000000
  ...
  192767: 0.0000000000
  ...
  193535: 0.0000000000
  ...
  194303: 0.0000000000
  ...
  195071: 0.0000000000
  ...
  195839: 0.0000000000
  ...
  196607: 0.0000000000
  ...
  197375: 0.0000000000
  ...
  198143: 0.0000000000
  ...
  198911: 0.0000000000
  ...
  199679: 0.0000000000
  ...
  200447: 0.0000000000
  ...
  201215: 0.0000000000
  ...
  201983: 0.0000000000
  ...
  202751: 0.0000000000
  ...
  203519: 0.0000000000
  ...
  204287: 0.0000000000
  ...
  205055: 0.0000000000
  ...
  205823: 0.0000000000
  ...
  206591: 0.0000000000
  ...
  207359: 0.0000000000
  ...
  208127: 0.0000000000
  ...
  208895: 0.0000000000
  ...
  209663: 0.0000000000
  ...
  210431: 0.0000000000
  ...
  211199: 0.0000000000
  ...
  211967: 0.0000000000
  ...
  212735: 0.0000000000
  ...
  213503: 0.0000000000
  ...
  214271: 0.0000000000
  ...
  215039: 0.0000000000
  ...
  215807: 0.0000000000
  ...
  216575: 0.0000000000
  ...
  217343: 0.0000000000
  ...
  218111: 0.0000000000
  ...
  218879: 0.0000000000
  ...
  219647: 0.0000000000
  ...
  220415: 0.0000000000
  ...
  221183: 0.0000000000
  ...
  221951: 0.0000000000
  ...
  222719: 0.0000000000
  ...
  223487: 0.0000000000
  ...
  224255: 0.0000000000
  ...
  225023: 0.0000000000
  ...
  225791: 0.0000000000
  ...
  226559: 0.0000000000
  ...
  227327: 0.0000000000
  ...
  228095: 0.0000000000
  ...
  228863: 0.0000000000
  ...
  229631: 0.0000000000
  ...
  230399: 0.0000000000
  ...
  231167: 0.0000000000
  ...
  231935: 0.0000000000
  ...
  232703: 0.0000000000
  ...
  233471: 0.0000000000
  ...
  234239: 0.0000000000
  ...
  235007: 0.0000000000
  ...
  235775: 0.0000000000
  ...
  236543: 0.0000000000
  ...
  237311: 0.0000000000
  ...
  238079: 0.0000000000
  ...
  238847: 0.0000000000
  ...
  239615: 0.0000000000
  ...
  240383: 0.0000000000
  ...
  241151: 0.0000000000
  ...
  241919: 0.0000000000
  ...
  242687: 0.0000000000
  ...
  243455: 0.0000000000
  ...
  244223: 0.0000000000
  ...
  244991: 0.0000000000
  ...
  245759: 0.0000000000
  ...
  246527: 0.0000000000
  ...
  247295: 0.0000000000
  ...
  248063: 0.0000000000
  ...
  248831: 0.0000000000
  ...
  249599: 0.0000000000
  ...
  250367: 0.0000000000
  ...
  251135: 0.0000000000
  ...
  251903: 0.0000000000
  ...
  252671: 0.0000000000
  ...
  253439: 0.0000000000
  ...
  254207: 0.0000000000
  ...
  254975: 0.0000000000
  ...
  255743: 0.0000000000
  ...
  256511: 0.0000000000
  ...
  257279: 0.0000000000
  ...
  258047: 0.0000000000
  ...
  258815: 0.0000000000
  ...
  259583: 0.0000000000
  ...
  260351: 0.0000000000
  ...
  261119: 0.0000000000
  ...
  261887: 0.0000000000
  ...
  262655: 0.0000000000
  ...
  263423: 0.0000000000
  ...
  264191: 0.0000000000
  ...
  264959: 0.0000000000
  ...
  265727: 0.0000000000
  ...
  266495: 0.0000000000
  ...
  267263: 0.0000000000
  ...
  268031: 0.0000000000
  ...
  268799: 0.0000000000
  ...
  269567: 0.0000000000
  ...
  270335: 0.0000000000
  ...
  271103: 0.0000000000
  ...
  271871: 0.0000000000
  ...
  272639: 0.0000000000
  ...
  273407: 0.0000000000
  ...
  274175: 0.0000000000
  ...
  274943: 0.0000000000
  ...
  275711: 0.0000000000
  ...
  276479: 0.0000000000
  ...
  277247: 0.0000000000
  ...
  278015: 0.0000000000
  ...
  278783: 0.0000000000
  ...
  279551: 0.0000000000
  ...
  280319: 0.0000000000
  ...
  281087: 0.0000000000
  ...
  281855: 0.0000000000
  ...
  282623: 0.0000000000
  ...
  283391: 0.0000000000
  ...
  284159: 0.0000000000
  ...
  284927: 0.0000000000
  ...
  285695: 0.0000000000
  ...
  286463: 0.0000000000
  ...
  287231: 0.0000000000
  ...
  287999: 0.0000000000
  ...
  288767: 0.0000000000
  ...
  289535: 0.0000000000
  ...
  290303: 0.0000000000
  ...
  291071: 0.0000000000
  ...
  291839: 0.0000000000
  ...
  292607: 0.0000000000
  ...
  293375: 0.0000000000
  ...
  294143: 0.0000000000
  ...
  294911: 0.0000000000
  ...
  295679: 0.0000000000
  ...
  296447: 0.0000000000
  ...
  297215: 0.0000000000
  ...
  297983: 0.0000000000
  ...
  298751: 0.0000000000
  ...
  299519: 0.0000000000
  ...
  300287: 0.0000000000
  ...
  301055: 0.0000000000
  ...
  301823: 0.0000000000
  ...
  302591: 0.0000000000
  ...
  303359: 0.0000000000
  ...
  304127: 0.0000000000
  ...
  304895: 0.0000000000
  ...
  305663: 0.0000000000
  ...
  306431: 0.0000000000
  ...
  307199: 0.0000000000
  ...
  307967: 0.0000000000
  ...
  308735: 0.0000000000
  ...
  309503: 0.0000000000
  ...
  310271: 0.0000000000
  ...
  311039: 0.0000000000
  ...
  311807: 0.0000000000
  ...
  312575: 0.0000000000
  ...
  313343: 0.0000000000
  ...
  314111: 0.0000000000
  ...
  314879: 0.0000000000
  ...
  315647: 0.0000000000
  ...
  316415: 0.0000000000
  ...
  317183: 0.0000000000
  ...
  317951: 0.0000000000
  ...
  318719: 0.0000000000
  ...
  319487: 0.0000000000
  ...
  320255: 0.0000000000
  ...
  321023: 0.0000000000
  ...
  321791: 0.0000000000
  ...
  322559: 0.0000000000
  ...
  323327: 0.0000000000
  ...
  324095: 0.0000000000
  ...
  324863: 0.0000000000
  ...
  325631: 0.0000000000
  ...
  326399: 0.0000000000
  ...
  327167: 0.0000000000
  ...
  327935: 0.0000000000
  ...
  328703: 0.0000000000
  ...
  329471: 0.0000000000
  ...
  330239: 0.0000000000
  ...
  331007: 0.0000000000
  ...
  331775: 0.0000000000
  ...
  332543: 0.0000000000
  ...
  333311: 0.0000000000
  ...
  334079: 0.0000000000
  ...
  334847: 0.0000000000
  ...
  335615: 0.0000000000
  ...
  336383: 0.0000000000
  ...
  337151: 0.0000000000
  ...
  337919: 0.0000000000
  ...
  338687: 0.0000000000
  ...
  339455: 0.0000000000
  ...
  340223: 0.0000000000

```

MongoDB Vector Store

- Once the chunks are stored, we create an **index** on the collection
- Select “Search & Vector Search”



MongoDB Vector Store

- Choose “Vector Search” for search type

Back to search indexes

Create a Vector Search Index

1 Setup — 2 Configuration — 3 Review

Start Your Index Configuration

Search Type
Which Search type should I use?

Atlas Search
Full-text search for relevance-based app features.

Vector Search
For semantic search and AI applications.

MongoDB Vector Store

- Name your index and choose the database and collection where you stored your chunks

Index Name and Data Source

Search indexes are specific to a database and collection.

ⓘ At this time, search indexes cannot be created for time series collections. ✕

Index Name

Database and Collection

>

 case_blind_elephant

>

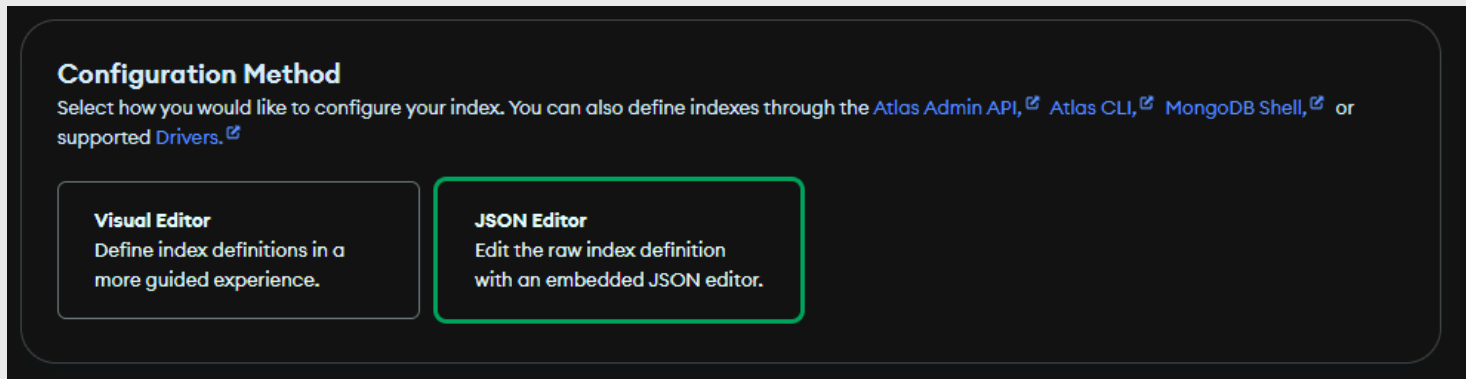
 case_nathans

>

 chatapp

MongoDB Vector Store

- Choose “JSON Editor” for configuration method



- Have the AI write the JSON index definition and paste it in the JSON Editor on MongoDB

```
{
  "fields": [
    {
      "type": "vector",
      "path": "content_vector",
      "numDimensions": 768,
      "similarity": "cosine"
    }
  ]
}
```


Coding Session

- **Put a large document into a vector store**
- **Clone a basic chat app**
- **Give that chat bot RAG functionality**