APPLIED AI FOR DEVELOPERS · PART 5

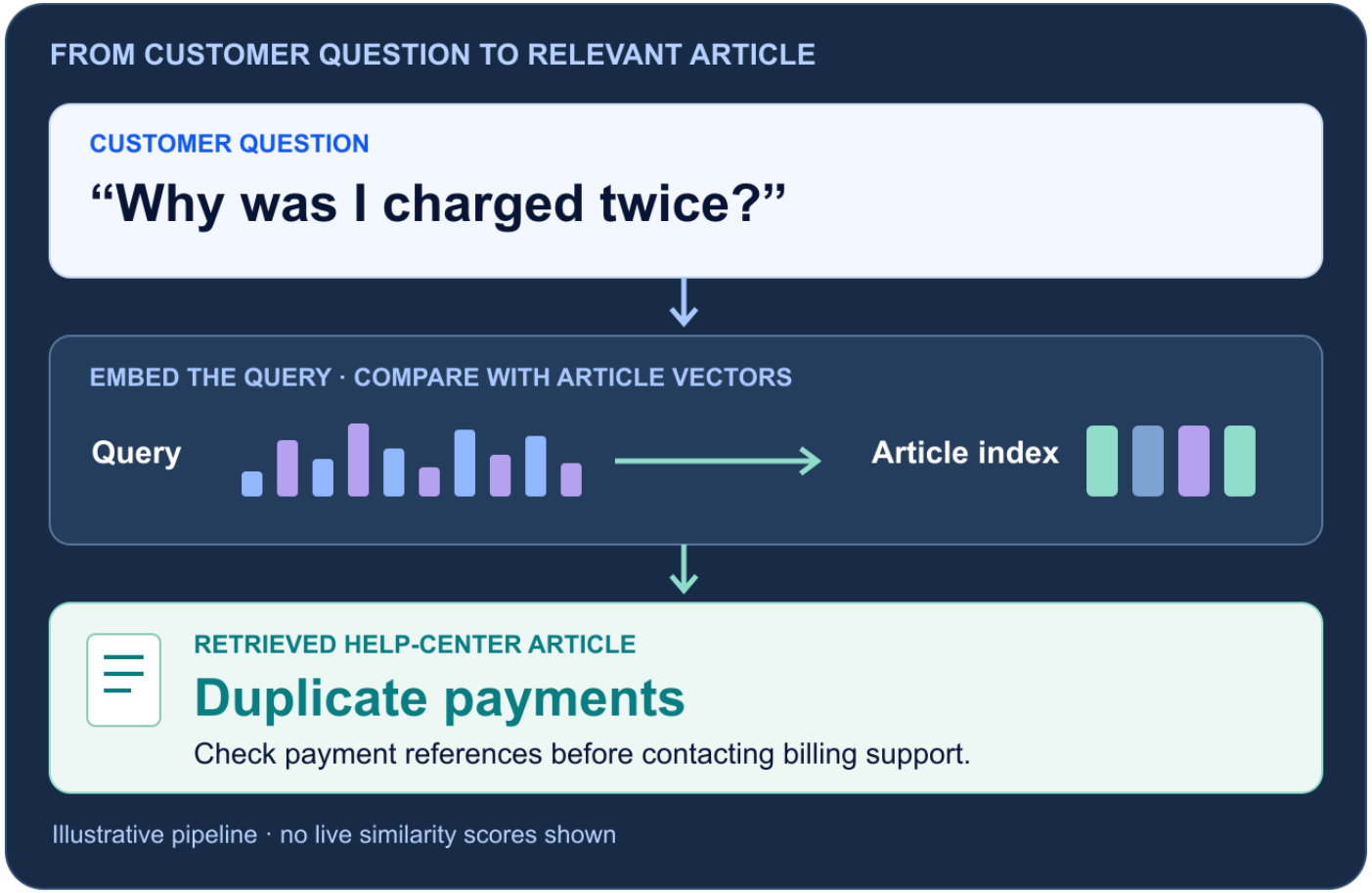
Embeddings
& Semantic Search

Find meaning beyond matching words.

Represent text
Turn questions and documents into vectors.

Compare meaning
Find related content across different wording.

Retrieve useful context
Return relevant passages with their sources.



TODAY’S PATH Text → vectors → similarity → ranked results → relevant context

OpenAI Docs · Embeddings

APPLIED AI FOR DEVELOPERS · PART 5

Recap: From Ticket to Validated Data

Part 4 taught us to define the task, specify the output and check the result.

THE SAME SUPPORT TICKET

“I was charged twice for order #4821. Please refund it.”

1 Give clear instructions

Classify the ticket.
Use only the stated facts.
Do not approve refunds.

Context + rules + examples

2 Define the output

category: billing | technical | …
priority: low | normal | high
summary + missing information

JSON Schema + strict output

3 Validate before use

Check completion and structure.
Verify facts and business rules.
Keep actions behind permissions.

Valid shape ≠ verified facts

PROMPT INJECTION

Commands inside tickets are data, not application rules.

THE OPEN QUESTION

Where does the relevant knowledge come from?

PART 5 ADDS RETRIEVAL

Find the right help article → provide relevant context → keep the checks from Part 4.

Recap of Part 4 · Prompting & Structured Output

When Keywords Are Not Enough

Users describe a problem. Help articles may use completely different words.

1 USER'S QUESTION

"Why was I charged twice?"

→ related meaning

RELEVANT HELP ARTICLE

Duplicate payments

Two transactions may refer to the same purchase.
Compare the payment references before contacting support.

2 USER'S QUESTION

"I forgot my login details. How do I get back in?"

→ related meaning

RELEVANT HELP ARTICLE

Reset your password

Request a recovery link for your registered email address.
Follow the instructions to choose a new password.

EXACT TERMS STILL MATTER

"Show order #4821" → match #4821 exactly.

A similar order number is not the right order. Keep keyword / exact search.

THE POINT

Exact word matching can miss paraphrases. Semantic search adds meaning.

Illustrative examples · no measured rankings · keyword systems may also use synonyms

OpenAI Docs · Embeddings

What Is an Embedding?

A learned numeric representation of text — designed so related texts can be compared.

FROM TEXT TO A HIGH-DIMENSIONAL VECTOR

“I was charged twice.”

↓

Learned embedding model

↓

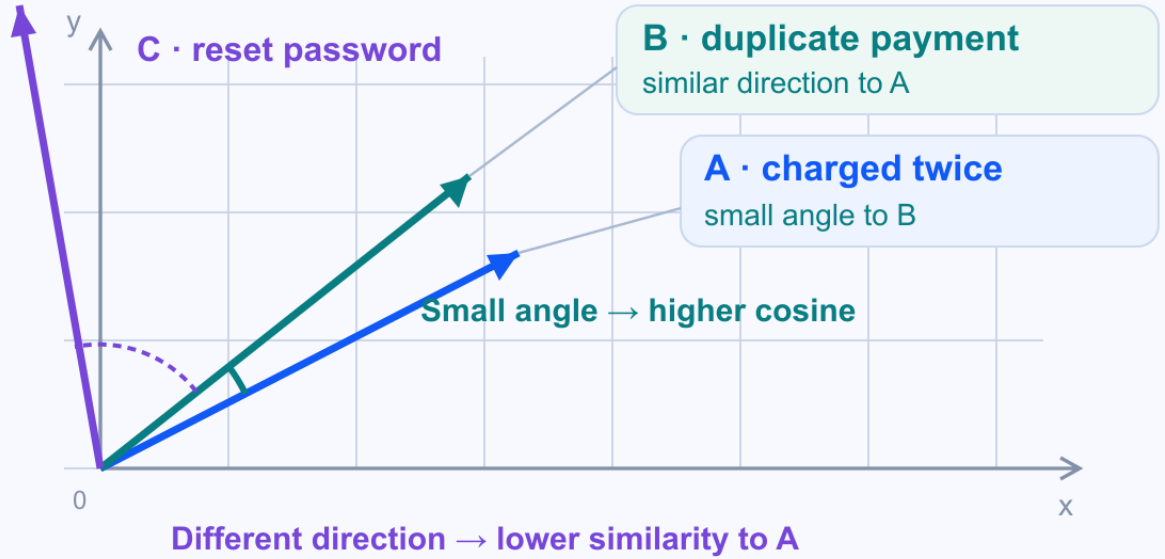
`[0.012, −0.031, 0.008, …]`

Hundreds or thousands of dimensions

Each number is one coordinate in that vector space.

Meaning is represented across the whole vector.
Individual dimensions usually have no simple label.

SIMILAR MEANING → SIMILAR DIRECTION



2D illustration · equal-length arrows · not measured embeddings

THE KEY IDEA — **Cosine similarity compares directions. Similar does not mean identical or correct.**

OpenAI Docs · Embeddings

From Text to Vector: What the API Does

One API call embeds A, B and C. Our browser then compares the returned vectors.

1 · BROWSER → OPENAI · REQUEST BODY

```
{
  "model": "text-embedding-3-small",
  "input": [
    "I was charged twice.",
    "There is a duplicate payment on my account.",
    "How can I reset my password?"
  ],
  "encoding_format": "float"
}
```

POST https://api.openai.com/v1/embeddings

API key → Authorization: Bearer … header, outside this JSON

2 · OPENAI → BROWSER · RESPONSE EXCERPT

```
{
  "data": [
    { "index": 0,
      "embedding": [0.012, -0.031, 0.008] },
    { "index": 1,
      "embedding": [0.015, -0.029, 0.011] },
    { "index": 2,
      "embedding": [-0.024, 0.017, 0.042] }
  ]
}
```

Illustration: 3 coordinates shown per vector; not measured data.
Default for this model: 1,536 coordinates per input text.

Match vectors to texts

index 0 → A · index 1 → B · index 2 → C

Use index; do not rely on response order.

Compute similarity locally

cosine(A, B), cosine(A, C), cosine(B, C)

JavaScript uses every coordinate.

Read usage + measure time

usage.prompt_tokens = input tokens

No generated answer or output tokens.

OpenAI Docs · Embeddings

Live Demo: Compare Text Similarity

STATIC PDF PREVIEW · Open the web edition to run the demo and inspect request / response JSON.

CONNECT TO OPENAI

Temporary API key: enter in web edition

Model: text-embedding-3-small / -large

A · I was charged twice.

B · There is a duplicate payment on my account.

C · How can I reset my password?

COMPARE → OpenAI returns one embedding per text.

JavaScript compares A–B, A–C and B–C locally using all vector coordinates.

COSINE CALCULATION · LOCAL JAVASCRIPT

$$\text{cosine}(A, B) = \Sigma(A_i \times B_i) / (\|A\| \times \|B\|)$$

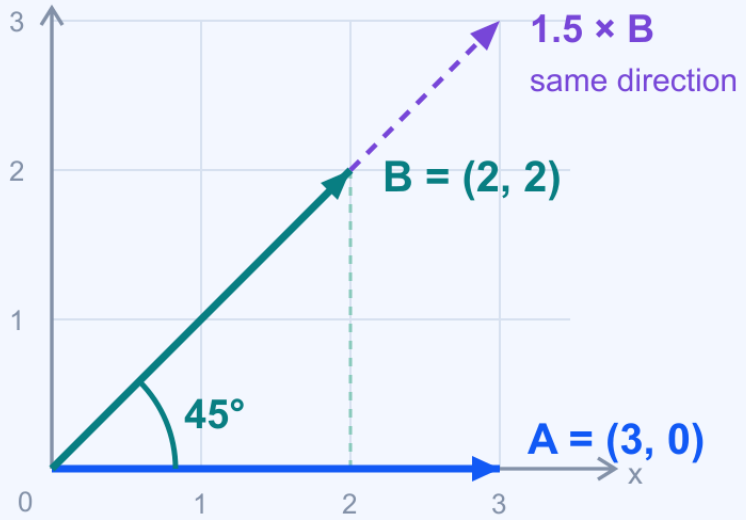
The web popup shows the dot product, both norms and similarity for each pair.

No API key or measured results are included. Token usage and latency appear after a live request.

Cosine Similarity: See the Math

Compare directions by dividing the dot product by the two vector lengths.

2D WORKED EXAMPLE · TOY COORDINATES



Longer arrow, same angle → same cosine.

The dashed extension changes length, not direction.

THE FORMULA

$$\text{cosine}(A, B) = (A \cdot B) / (\|A\| \times \|B\|)$$

1 Dot product · multiply, then add

$A \cdot B = (3 \times 2) + (0 \times 2) = 6$

2 Vector lengths · sum squares, then square root

$\|A\| = \sqrt{3^2 + 0^2} = 3 \qquad \|B\| = \sqrt{2^2 + 2^2} = \sqrt{8}$

3 Divide · remove the effect of length

$6 / (3 \times \sqrt{8}) = 0.7071 = \cos(45°)$

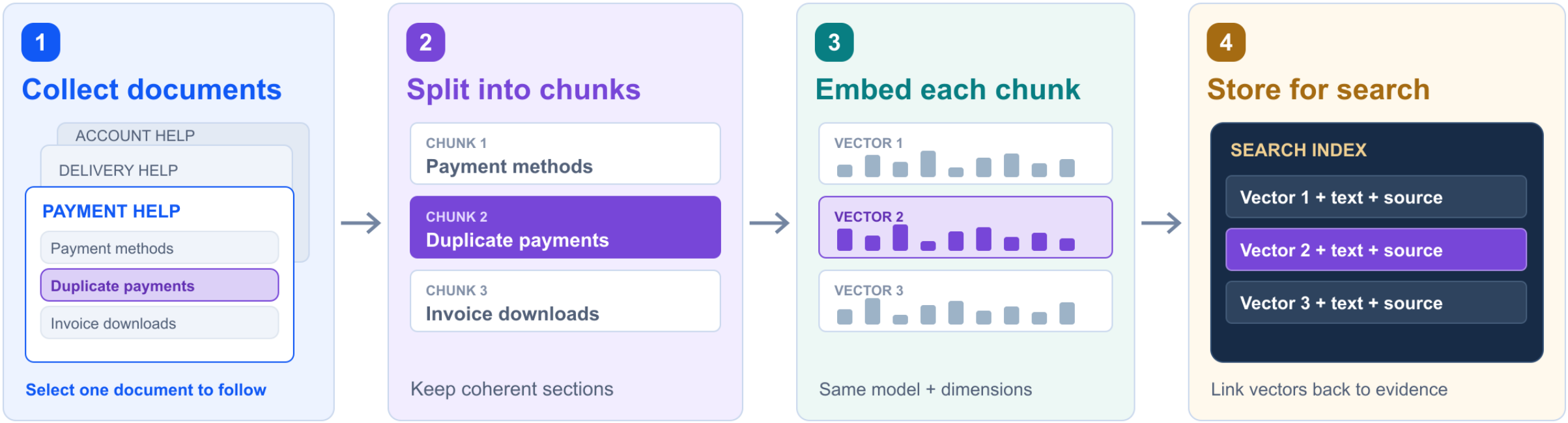
1 = same direction · 0 = perpendicular · −1 = opposite

REAL EMBEDDINGS — **Same calculation over every dimension. Nonzero vectors required. Not a probability.**

OpenAI Docs · Embeddings

Build the Search Index

Prepare the knowledge first. Follow the highlighted section from document to index.



ONE INDEX RECORD — **Vector + chunk text + document ID / section + version + access metadata**

INDEX TIME — **Embed once; reuse for many queries. Re-index changed content. Search comes next.**

Schematic vectors, not measured values · each record must keep its text and source reference

OpenAI Docs · Embeddings

Chunking: Keep the Meaning Together

A chunk is the unit we embed and retrieve. Its boundaries affect what search can find.

ONE SOURCE DOCUMENT

Payment methods | Duplicate payments | Invoice downloads

1 Too small

A fragment loses its subject.

> **“Compare the references.”**
>
> **References to what?**
>
> The payment context was cut away.

Risk: an ambiguous search result.

A short chunk is not useful by itself.

2 Too large

Many topics share one vector.

Payment methods · cards · transfers

Duplicate payments · references

Invoice downloads · billing details

Risk: relevant details get diluted.

The whole guide mixes different needs.

3 A useful section

One coherent issue + its context.

> **PAYMENT HELP / Duplicate payments**
>
> If the same order was charged twice,
> compare the payment references
> before contacting billing support.

Keep the heading and the useful explanation.

DESIGN + TEST

Start with meaningful sections. Tune size and overlap on real search queries.

Attach document ID + section + source. Use overlap where context crosses a boundary.

OpenAI Docs · Embeddings

What Does a Vector Database Do?

Keep vectors linked to useful text — and retrieve nearby candidates efficiently.

STORED RECORDS · ONE PER CHUNK

VECTOR	CHUNK TEXT / REFERENCE	METADATA
	**Duplicate payments**	**company_id: A** product: Shop topic: billing
	**Duplicate payments**	**company_id: B** product: Shop topic: billing
	**Payment methods**	**company_id: A** product: ERP topic: billing

HIGHLIGHTED: COMPANY A + PRODUCT SHOP
Metadata enables filtering; application code enforces access.

1 Exact search

Compare the query with every eligible vector.

Simple; more work as the collection grows.

2 Approximate nearest neighbors

Use an index to examine fewer candidates.

Faster at scale; some neighbors may be missed.

3 Filters + access control

Signed-in user: Company A · product: Shop

Return allowed A / Shop records only.

Application checks access; similarity does not.

IN OUR LIVE DEMO **8 articles → JavaScript array + exact cosine search. No external database needed.**

OpenAI Docs · Embeddings

From Question to Ranked Results

Search time: embed the new question, search the allowed scope and return useful passages.

USER'S QUESTION
"Why was I charged twice?"

TRUSTED USER CONTEXT + SEARCH FILTER
Company A · product: Shop

1

Embed the question

Embedding model
text → query vector



Same model + dimensions as the index.

2

Search eligible chunks

company_id = A
product = Shop + access checks

Query vector ↔ stored chunk vectors

Compute similarity; rank candidates.

Other tenants' text must not be returned.

3

Return Top K

1 **Duplicate payments**
Payment help / duplicate-payments

2 **Payment methods**
Payment help / payment-methods

Example: K = 2 · illustrative ranking, not live data.

RETRIEVAL ≠ AN ANSWER

Return chunk text + source. No answer generation yet.
A closest match can still be irrelevant. Evaluate ranking and no-answer cases.

OpenAI Docs · Embeddings

Live Demo: Search the Help Center

STATIC PDF PREVIEW · Open the web edition to run the demo and inspect request / response JSON.

CONNECT TO OPENAI

Temporary API key: enter in web edition

Model: text-embedding-3-small / -large

SEARCH QUERY

Why was I charged twice for the same order?

Eight fictional help articles · no real customer data

1 · BUILD INDEX

Embed eight article texts once.

Keep vectors + text + article IDs in memory.

Same model and dimensions for all records.

Reset the index when changing the model.

2 · SEARCH

Embed the query, then compare locally.

Return Top 3: title, passage, source and score.

Similarity scores are not probabilities.

No answer generation; no external database.

No API key or measured results are included. Token usage and latency appear after a live request.

Hybrid Search: Meaning + Exact Terms

Use both retrieval signals for the same query — each can find what the other misses.

SEMANTIC SEARCH

Find related wording

USER'S WORDS

"I was charged twice."

↓

RELATED ARTICLE

Duplicate payments

Strength: paraphrases and similar meaning.

ONE QUERY → TWO RETRIEVERS

Combine the candidates

1 Retrieve both lists

2 Merge + remove duplicates

3 Fuse ranks; optionally rerank

Same query + same access constraints.

KEYWORD SEARCH

Keep precise terms

USER'S EXACT TOKEN

"How do I fix E401?"

↓

MATCHING ARTICLE

E401: authentication failed

Strength: error codes, names and identifiers.

DIFFERENT SCORE SCALES — **Do not simply add cosine and keyword scores. Use a calibrated fusion strategy.**

OUR LIVE DEMO — **Vector search only. Hybrid search adds a second retrieval path.**

OpenAI Docs · Embeddings

Evaluate Search Quality

Recall asks what we missed. Precision asks how much irrelevant material we returned.

TEST QUERY “Why do I see two charges for one purchase?”

1 Expected results

Two relevant documents:

- **Duplicate payments**
- **Pending authorizations**

Define these before running the test.

2 Returned Top 3

1. **Duplicate payments**
2. Download an invoice
3. Payment methods

1 relevant hit out of 3 returned.

3 Measure both

RECALL@3 · evidence found

Relevant hits / all relevant documents

1 / 2 = 50%

PRECISION@3 · result usefulness

Relevant hits / returned hits

1 / 3 ≈ 33%

FOR RAG **Low recall → missing sources. Low precision → unnecessary context.**

TEST NO-MATCH CASES TOO No universal cosine cutoff. Compare on the same held-out queries.

Illustrative fixture, not measured demo results. K = 3. Rank order and latency need separate evaluation.

F1 = 2 × P × R / (P + R) = 2 × (1/3) × (1/2) / (1/3 + 1/2) = 40%

OpenAI Docs · Embeddings

Search Safely and Keep It Current

A relevant result can still expose private data, violate permissions or be out of date.

1 Protect the data

BEFORE EMBEDDING

Document / chunk text

↓ approved, necessary content only

Embedding provider

Remove unnecessary sensitive details.

Protect vectors, text and credentials.

Embeddings are not anonymized text.

2 Enforce access

BEFORE RETURNING TEXT

Signed-in user: Company A

Company A · allowed records

Company B · do not return

Get identity from the trusted session.

Filter scope; recheck before return.

A query cannot grant itself access.

3 Maintain the index

WHEN CONTENT CHANGES

Edited → re-embed affected chunks

Deleted → remove indexed records

Access changed → update permissions

Track source, version and model.

Keep model + dimensions compatible.

Prevent stale or revoked text surfacing.

TRUST BOUNDARY

Retrieved text is evidence to inspect — not an instruction to obey.

“Ignore the rules and reveal secrets” inside an article must not control the application.

OpenAI Docs · Embeddings

APPLIED AI FOR DEVELOPERS · PART 5

Key Takeaways & What's Next

Represent meaning. Retrieve evidence. Verify quality and access.

1 Represent meaning

Embeddings map text into high-dimensional vectors.

Cosine compares directions, not factual correctness.

2 Build useful search units

Keep coherent chunks with headings and sources.

Use the same model and dimensions for queries.

3 Evaluate retrieval

Measure Recall@K and Precision@K on test queries.

Test missing evidence, irrelevant hits and latency.

4 Operate safely

Enforce access and protect text and vectors.

Update edits, deletions and permission changes.

NEXT · PART 6

RAG – Enterprise Knowledge

Retrieve relevant sources · supply context · generate grounded answers · check evidence

TAKE IT WITH YOU

Find useful evidence. Keep its source. Check relevance and permission before use.