# Lecture 20: Cosine Similarity Implementation

## Definition

Cosine similarity measures how close two vectors are by comparing the cosine of the angle between them. It is useful for checking how semantically similar two words or texts are.

## Steps for Implementation

- Controller Setup:
  - Create a RestController with a POST API:

```
/api/similarity
```

  - Accept two inputs: text1 and text2 via @RequestParam.
- Generate Embeddings
  - Use EmbeddingModel.embed() to convert both texts into embeddings.

```
float[] embedding1 = embeddingModel.embed(text1);
float[] embedding2 = embeddingModel.embed(text2);
```

- Calculate Cosine Similarity steps in Code
  - Compute the dot product of the vectors.
  - Compute norms (magnitude) of both vectors.
  - Divide the dot product by the product of norms.
  - Multiply by 100 if you want percentage similarity.
- **Return Result**
  - Returns similarity score as double.
  - Example:
    - dog vs cat → ~0.6
    - computer vs laptop → higher similarity
    - computer vs banana → low similarity (~0.2)

## Code Implementation:

```java
@RestController
public class OpenAIController {

    private ChatClient chatClient;

    @Autowired
    private EmbeddingModel embeddingModel;

    public OpenAIController(OpenAiChatModel chatModel) {
        this.chatClient = ChatClient.create(chatModel);
    }

    @PostMapping("/api/similarity")
    public double getSimilarity(@RequestParam String text1, @RequestParam String text2) {
        float[] embedding1 = embeddingModel.embed(text1);
        float[] embedding2 = embeddingModel.embed(text2);

        double dotProduct = 0;
        double norm1 = 0;
        double norm2 = 0;

        for (int i = 0; i < embedding1.length; i++) {
            dotProduct += embedding1[i] * embedding2[i];
            norm1 += Math.pow(embedding1[i], 2);
            norm2 += Math.pow(embedding2[i], 2);
        }

        return dotProduct * 100 / (Math.sqrt(norm1) * Math.sqrt(norm2));
    }
}
```

## Usage

- Helps in **semantic search**.
- Finds related results even if exact keywords don’t match.
- **Example**: Searching for headphones may return AirPods or Bluetooth speakers.