A MULTIMODAL APPROACH TO LOST & FOUND ITEM RETRIEVAL

Ashhad A. Kamran *– 2222511*
Qazi Maaz Pasha Syed *– 2109074*
Xintong Zhu *– 2220464*

Problem Statement

- Current systems are too inconsistent: Manual, inefficient, and low retrieval rates.
- Descriptions like “a bag” are too vague: They don’t capture visual details.

- **Objective: Build a smarter system that understands both images and text.**

Our Solution

A Visual Search Engine using pre-trained models

- Inspired by OpenAI's CLIP model, our system matches semantic meaning, not just keywords.
- Combines text + image inputs to find matching lost items.

Key Features:

- **Text → Image**: Enter "black Nike backpack", find matching photos
- **Image → Image**: Upload a photo, find similar items

Data Preparation

Dataset: Amazon Products Dataset 2023

Advantages:

1. Products (e.g., bags, electronics) align with common lost items.
2. Paired Data: Each entry includes product titles (text) + images
3. Large size (~1.4M entries) ensures robust training.

Cleaning Steps:

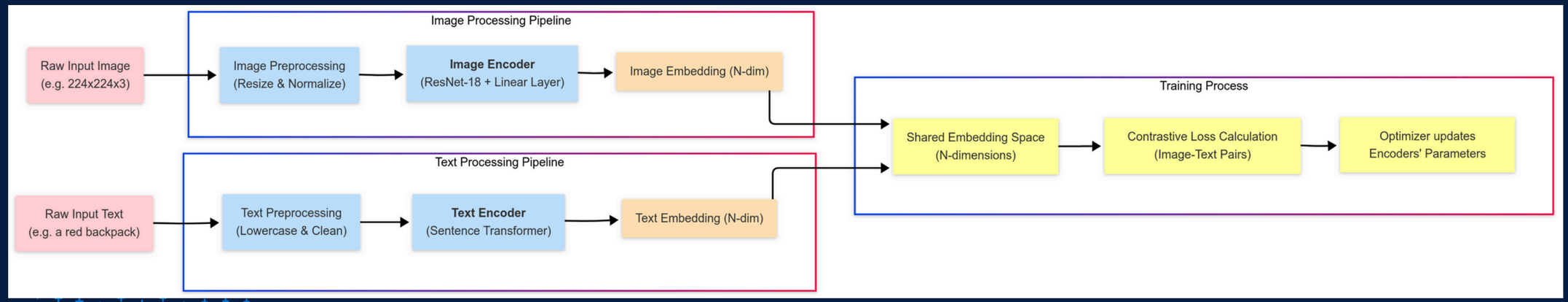
- Extract product titles (text) + image URLs.
- Remove noise from text (e.g., special characters)
- Resize/normalize images for model training

Model Architecture

Key challenge: How to compare a text description with a photo?

Two Encoders, One Smart Brain:

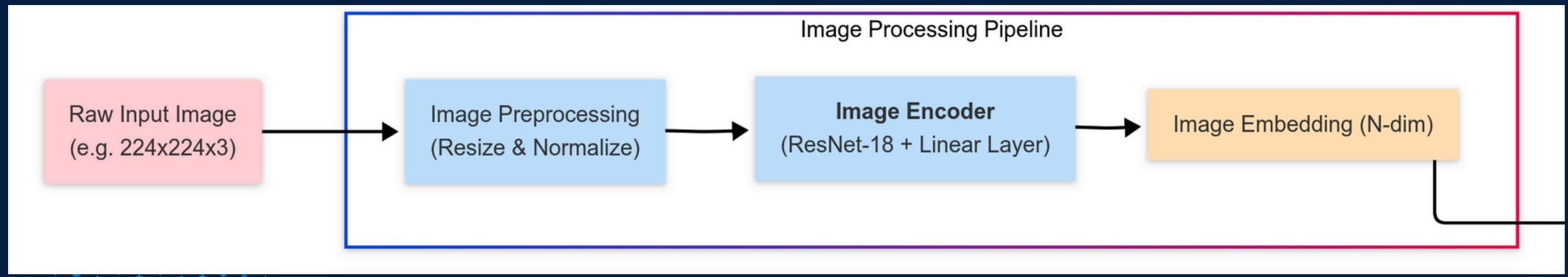
- **Image Encoder** (ResNet-18): Understands visual features
- **Text Encoder** (Sentence Transformer): Understands text meaning
- **Joint Embedding Space**: Maps similar items close together



Model Architecture

Image Encoder: based on ResNet-18

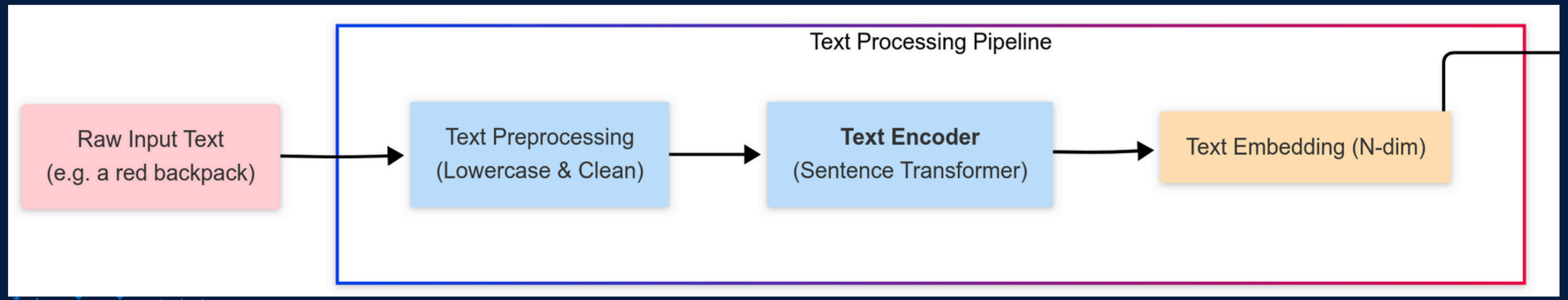
- Takes photos as input
- Extracts visual features (Replaces the last layer with a Linear layer to project image features.)
- L2 Normalization → Outputs 384-dimensional vector.



Model Architecture

Text Encoder (Sentence Transformer all-MiniLM-L6-v2):

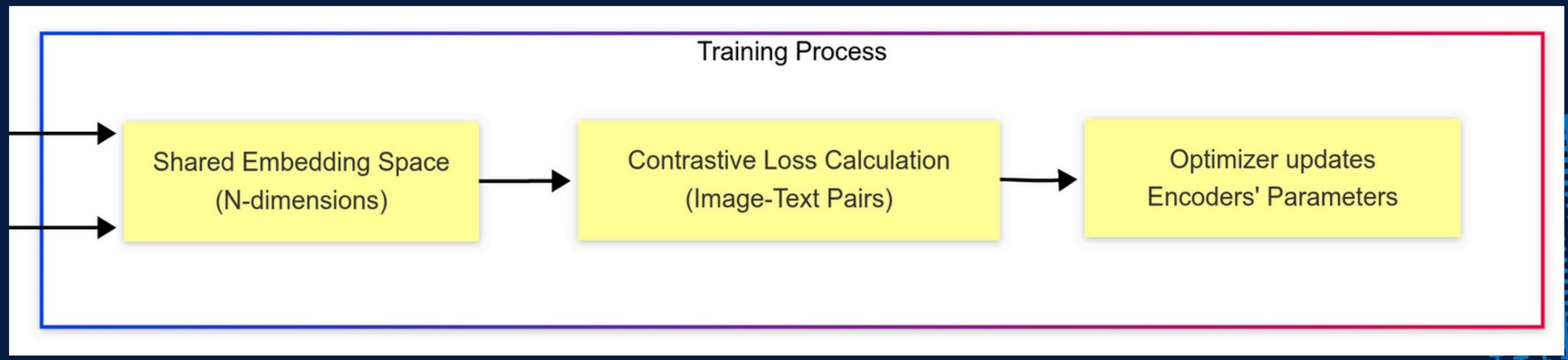
- Takes descriptions (the preprocessed titles from the dataset) as input
- Understands semantic meaning
- L2 Normalization → Outputs 384-dimensional vector



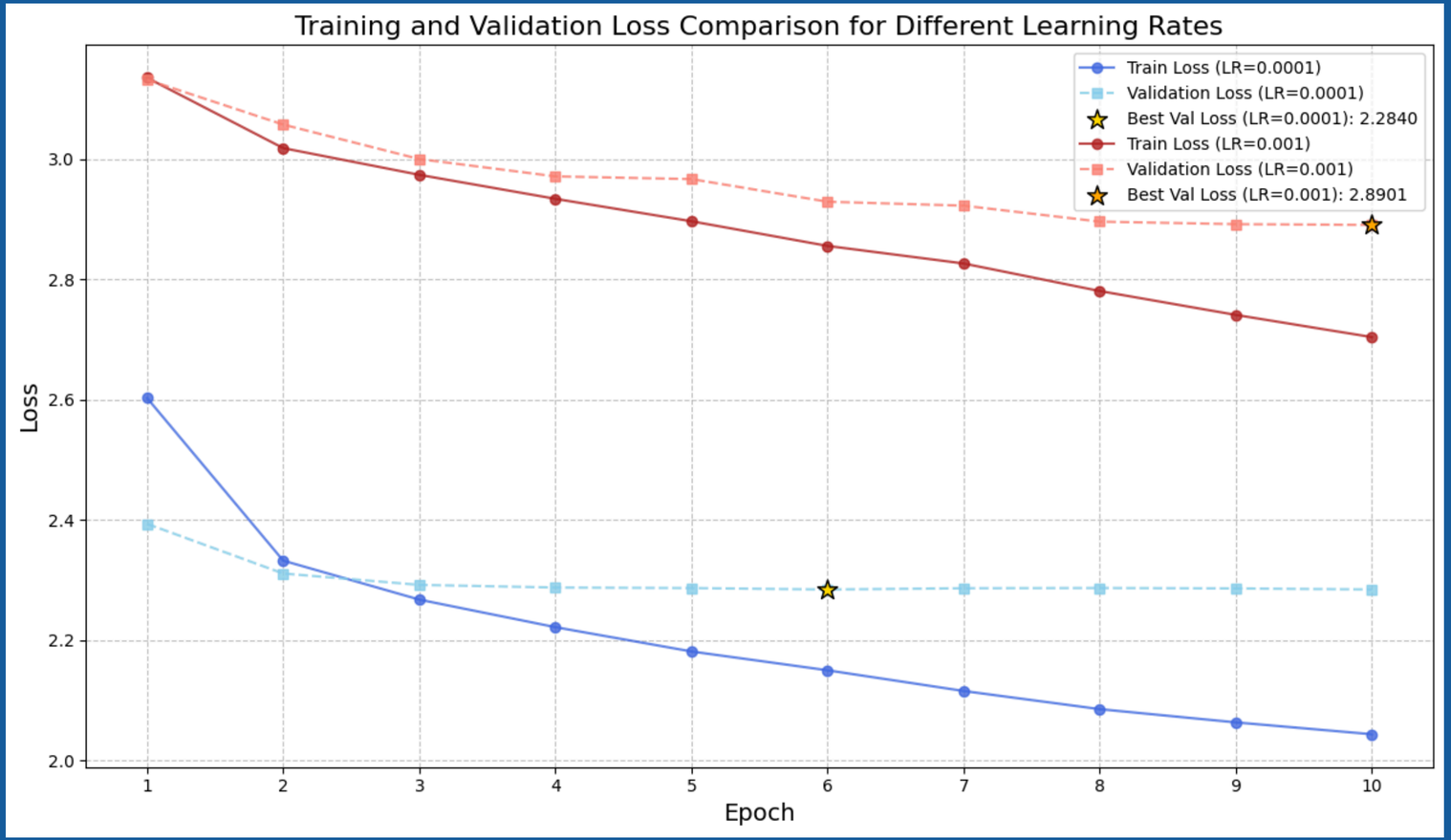
Model Architecture

Joint Embedding Space：

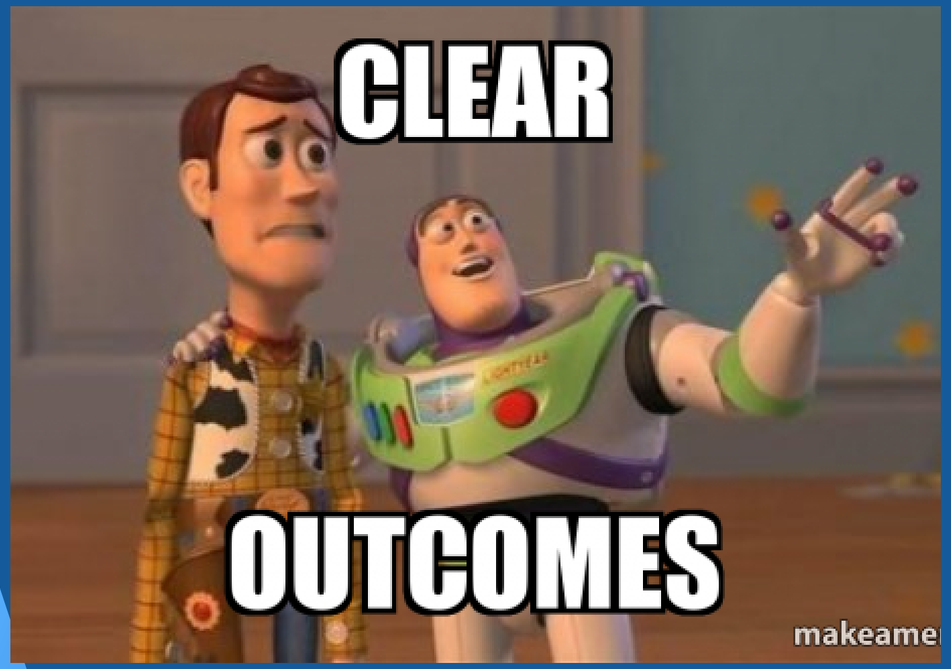
- Take Image Encoder + Text Encoder output vectors as input
- Computes cosine similarity between image-text pairs.
- Uses symmetric cross-entropy loss to align embeddings bidirectionally (both Image-to-Text and Text-to-Image)



Training Results

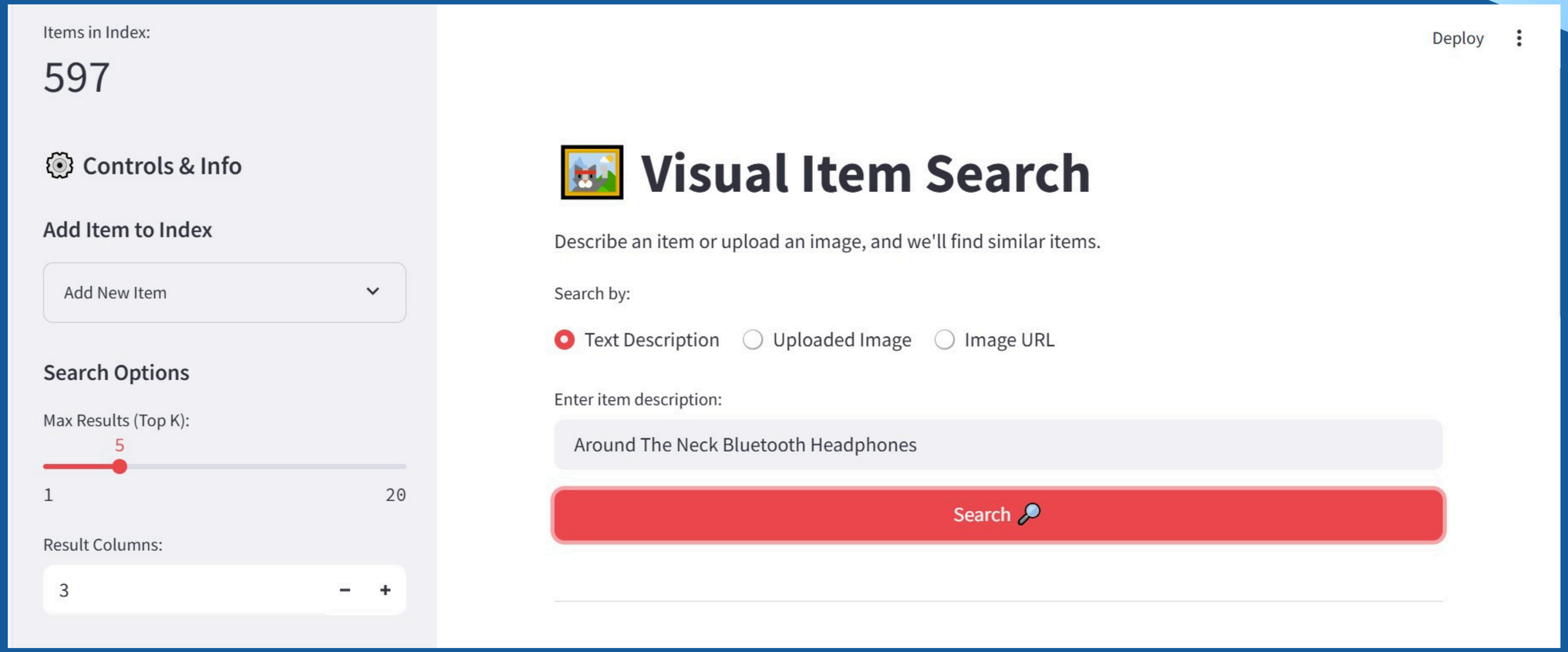


Training Results



- Optimizer: Adam
- **Best Learning Rate: 0.0001**
- **Validation loss:**
  improved from **2.8 to 1.58**
- The system returns high-similarity matches for both text-to-image and image-to-image queries.

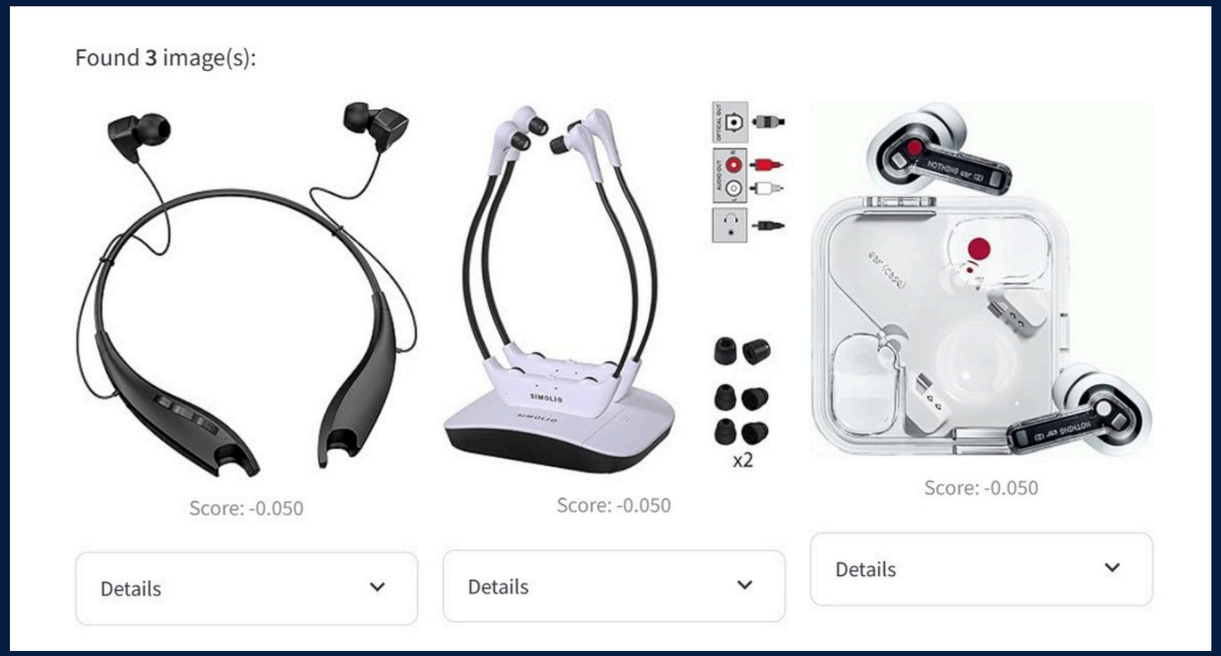
User Interface



Example 1 - Text Query

User enter “Around The Neck Bluetooth Headphones“

Top 3 items returned:



Example 2 - Image Query

Query Image (from URL):

Found 1 image(s):

Similarity: 0.9987

Details

Thank You