# **LZW Compression & Decompression Task**

## **📝 Task Description**

LZW (Lempel-Ziv-Welch) is a lossless compression algorithm used in file compression and image formats like GIF and TIFF. Your task is to implement LZW compression and decompression in JavaScript and build a React UI to demonstrate its working.

## **📌 Requirements**

1. Implement LZW Compression: Convert a string into a compressed numeric sequence.
2. Implement LZW Decompression: Convert the compressed numeric sequence back into the original string.
3. Create a simple React UI where users can:
   * Enter text
   * Click a button to compress it
   * View the compressed output
   * Click a button to decompress it
   * View the decompressed output

## **🛠 Steps to Follow**

### **1️⃣ Understand LZW Compression**

* Start with a dictionary of ASCII characters (0-255).
* Read input character by character, forming new sequences.
* If the sequence exists in the dictionary, continue expanding it.
* If it does not exist, add it to the dictionary and output the code for the previous sequence.
* Repeat until the full input is processed.

### **2️⃣ Implement Compression Algorithm**

* Create a dictionary initialized with ASCII characters.
* Process the input string and replace repeating patterns with numeric codes.
* Return an array of compressed numeric values.

### **3️⃣ Implement Decompression Algorithm**

* Reconstruct the dictionary from compressed numeric codes.
* Use the same logic to expand the compressed sequence back to the original string.

### **4️⃣ Build a React UI**

* A text area for user input.
* A "Compress" button to trigger compression.
* A "Decompress" button to trigger decompression.
* Display compressed and decompressed results.

## **📌 Example Walkthrough**

### **Compression Example**

#### **Input String:** "ABABABA"

#### **Compression Steps:**

| Step | Read Input | Dictionary Entry (Code) | Output Code |
| --- | --- | --- | --- |
| 1 | A | Already Exists | 65 |
| 2 | B | Already Exists | 66 |
| 3 | AB | Added as 256 | 65,66 |
| 4 | BA | Added as 257 | 256 |
| 5 | AB | Exists (256) | 257 |
| 6 | ABA | Added as 258 | 256,258 |

🔹 Compressed Output: [65, 66, 256, 257, 258]

### **Decompression Example**

#### **Compressed Input:** [65, 66, 256, 257, 258]

#### **Decompression Steps:**

| Step | Code | Dictionary Entry | Output |
| --- | --- | --- | --- |
| 1 | 65 | A | A |
| 2 | 66 | B | AB |
| 3 | 256 | AB | ABA |
| 4 | 257 | BA | ABAB |
| 5 | 258 | ABA | ABABABA |

🔹 Decompressed Output: "ABABABA" (Matches original input)