# **Documentation for Image-Sharing App**

## **Overview**

The image-sharing app "Y" allows users to upload images and share temporary links. Each link has an expiration time, ensuring that shared images are only accessible for a limited period. The application is designed with TypeScript using React for the frontend and Node.js (Express) for the backend, ensuring scalability and production-level quality.

## **Backend**

### **Endpoints**

#### **1. POST /v1/images**

Handles image uploads and returns a shareable URL.

* **Request**:
  + Form-data containing:
    - image: File input.
    - expiration: Time in seconds after which the image expires.
* **Response**:
  + 200 OK: Returns the shareable URL in JSON format.

Example:  
 {

"url": "http://localhost:3000/v1/images/:imageID"

}

#### **2. GET /v1/images/:imageID**

Serves an image if it exists and has not expired.

* **Request**:
  + :imageID: The unique identifier of the image.
* **Response**:
  + 200 OK: Returns the image file.
  + 404 Not Found: If the image does not exist or has expired.

### **Image Storage**

* Uploaded images are stored in a directory (uploads/).
* Each image is associated with metadata, including its expiration timestamp.

### **Expiration Handling**

* **Automatic Deletion**:
  + A background process (scheduleImageExpiration) deletes expired images using setTimeout.
  + Edge cases (e.g., server restarts) are handled by checking all images’ metadata on startup.

## **Frontend**

### **Features**

#### **1. Image Upload UI**

* **File Upload**:
  + A drag-and-drop or standard file input to upload images.
* **Expiration Time**:
  + Input field or dropdown to specify expiration time.

#### **2. Shareable Link**

* **Notification/Modal**:
  + Displays the shareable link with a "Copy to Clipboard" button.
  + The user receives a list of all images that were loaded

## **Testing**

### **Unit Tests**

#### **Image Expiration Logic**

Tests ensure expired images are deleted correctly. Example test with Jest:

## **Assumptions and Notes**

1. The user application is allowing any user to view any non expired image.
2. The server port 3003, accepts origin only for localhost:3001 - the user port.
3. The UI user will view all images that were loaded with link and copy button as a list - implemented with state management.
4. A refresh will remove the links - assuming it can be another user, and not required to view images from server call.