**ABSTRACT**

Our Product AuthGen: Invisible Watermark is a full stack web application, with the main purpose of securing production lines by employing our services of invisible watermarking that comes with virtually zero downtime to any client and does not require additional hardware, which makes it highly cost effective.

The idea behind securing product lines lies with the product label. The most common form of counterfeiting is by duplicating a product label and pasting it on a counterfeit product. The existing solutions for counterfeiting like RAIL RFID (Implemented by NIKE), uniquely generated and two factor verification of invoice labels (Implemented by AMAZON), and many blockchain technologies. But the basic premise is a company has to decide between secure and cheap, with no options in between. These options are easy to get around by using RFID Duplicate Signal Generators and 4200+ dpi scanners.

By utilizing the option of hiding uniquely generated watermarks in a product label using Steganography, the watermark is hard to detect. Since it does not utilize any external hardware, it is immune to any duplicators. The way the image is encoded makes it immune to scanners, making this the perfect anti-counterfeiting system that is cheap, robust, quick and reliable.

1. **Introduction**
   1. **Overview**

AuthGen stands as a cutting-edge initiative in the realm of anti-counterfeiting, presenting a pioneering Invisible Watermark that amalgamates multiple technologies seamlessly. At its core, AuthGen aspires to introduce a revolutionary QR code-like print-scan authentication service. This innovative approach involves the strategic use of Steganography, a technique dedicated to concealing information within non-secret data, ensuring the existence of the embedded message remains undetected. In the digital landscape, this technique often finds application in embedding secret messages within images, audio files, or various other media types.

* 1. **Existing system**

Traditional anti-counterfeiting methods predominantly rely on visible features such as holograms or serial numbers. However, these conventional systems have demonstrated inherent limitations, rendering them susceptible to replication and counterfeit activities. AuthGen, in response to these challenges, strives to redefine the landscape of anti-counterfeiting by introducing an advanced, covert authentication system. This system is designed to operate beyond the visible spectrum, providing a layer of security that surpasses traditional methods.

* 1. **Motivation**

The driving force behind AuthGen's development lies in the pursuit of creating a robust and foolproof anti-counterfeiting solution. By introducing an Invisible Watermark, AuthGen aims to elevate the standards of security by incorporating features that are not only difficult to detect but also resilient against replication. The motivation stems from a keen awareness of the shortcomings inherent in traditional authentication methods, propelling AuthGen to pioneer advancements in security technology.

* 1. **Problem statement**

AuthGen identifies several challenges associated with conventional QR codes, including fault-tolerance, orientation, camera quality, and colour shifting. These challenges become particularly pronounced when attempting to implement a print-scan solution using Steganography. The system recognizes these issues as critical problems to solve in order to establish a reliable and efficient invisible watermarking system. The overarching goal is to create a solution that not only maintains functionality in the face of potential destruction but also adapts to varied orientations, diverse camera qualities, and alterations in RGB values resulting from environmental factors.

1. **Literature survey**
   1. **RAIL RFID**

The exploration into existing anti-counterfeiting solutions commences with a thorough examination of RAIL RFID technology. RAIL RFID, known for its Radio Frequency Identification capabilities, involves tagging products with unique identifiers that can be tracked throughout the supply chain. The survey delves into the strengths of RAIL RFID, such as real-time tracking, enhanced visibility, and streamlined logistics. However, it critically scrutinizes potential weaknesses, including susceptibility to cloning and the challenges associated with secure data transmission.

* 1. **Blockchain Encryption**

The study progresses to Blockchain encryption, a technology hailed for its decentralized and tamper-resistant nature. The survey evaluates the strengths of Blockchain in providing an immutable ledger for product authenticity, ensuring transparency and traceability. On the flip side, it delves into challenges such as scalability issues, energy consumption concerns in certain blockchain networks, and the need for widespread adoption to achieve optimal effectiveness.

* 1. **EM Tag Validation**

Electromagnetic (EM) Tag Validation emerges as another focal point in the literature survey. This technology involves the use of electromagnetic fields to validate product authenticity. The survey dissects the strengths, emphasizing the non-intrusive nature of EM validation and its applicability to a wide range of products. Simultaneously, it addresses weaknesses, including susceptibility to interference, potential for signal spoofing, and the need for specialized infrastructure.

* 1. **Uniquely Generated Invoices**

The exploration concludes with an analysis of anti-counterfeiting measures employing uniquely generated invoices. This approach relies on creating one-of-a-kind invoices tied to specific products, serving as a proof of authenticity. The survey scrutinizes the strengths, such as simplicity and cost-effectiveness, while acknowledging weaknesses like the potential for duplication and the challenge of scaling this approach for high-volume products.

**Understanding of the Current Landscape**

By meticulously exploring these technologies, the literature survey culminates in a nuanced understanding of the current anti-counterfeiting landscape. It becomes evident that each approach possesses its own set of merits and demerits. The strengths underscore the potential effectiveness of these technologies in safeguarding product authenticity, while the weaknesses shed light on areas that demand further innovation and refinement.

1. **System design**

The current testing system that validates AuthGen's anti-counterfeiting system operates on a formidable hardware configuration. It is powered by a Ryzen 5 processor boasting 8 cores, complemented by an Nvidia 1650ti GPU equipped with 8GB of RAM. This robust setup facilitates the concurrent operation of all four required servers, ensuring a seamless and responsive performance. Notably, the system achieves a remarkable maximum request turnaround time of 11 seconds, underscoring its efficiency in handling authentication requests with speed and precision.

1. **Implementation**
   1. **Explanation**
      1. **Hiding data inside image**
         1. **LSB Steganography**

LSB (Least Significant Bit) [2][7][8] steganography is a technique used to hide information within the least significant bits of digital data, such as images, audio, or video files. The basic idea behind LSB steganography is to replace the least significant bits of the data with the bits of the secret message without significantly altering the overall appearance of the data.

In binary representation, each byte (8 bits) can represent values from 0 to 255. The least significant bit is the rightmost bit, representing the smallest value (0 or 1). LSB steganography involves replacing these least significant bits with the bits of the secret message. Since the changes are small and affect only the least significant part of the data, the alterations are often imperceptible to the human eye or ear.

Here's a simplified example using an 8-bit binary representation:

Original file: 11111111 00000000 11111111 00000000

Secret message: 0011

Encoded byte: 1111111**0** 0000000**0** 1111111**1** 0000000**1**

In this example, the least significant bit of the original byte is replaced with the first bit of the secret message. The resulting byte looks almost the same as the original one, but it contains hidden information.

* + - 1. **DCT Steganography**

DCT (Discrete Cosine Transform) [3][7] steganography is a technique that hides information within the frequency domain of digital media, particularly in images that have been transformed using DCT. DCT is widely used in image compression algorithms, with JPEG being a notable example. In DCT steganography, data is embedded into the coefficients produced by the DCT transformation. DCT follows the below steps -

* Image Transformation: The image is divided into blocks, typically 8x8 pixels. Each block undergoes the DCT transformation, resulting in a set of frequency coefficients.
* Coefficient Modification: The steganographic process involves modifying specific DCT coefficients in each block to embed the secret message. The choice of coefficients for modification is crucial to minimize visual impact and avoid detection.
* Embedding Secret Message: The secret message is encoded and then spread across the selected DCT coefficients. LSBs of certain DCT coefficients may be used to hide the secret information.
* Reversing the Process: To extract the hidden message, the receiver applies an inverse DCT transformation on the image, revealing the modified coefficients. The LSBs of these coefficients are then extracted to reconstruct the embedded message.

DCT steganography provides a high capacity for hiding information because it operates in the frequency domain, where certain changes are less noticeable to the human eye.

* + - 1. **GAN Steganography**

GAN (Generative Adversarial Network) [4][5] steganography involves using GANs to embed and extract hidden information within digital media, such as images. GANs consist of a generator network and a discriminator network, which are trained simultaneously through adversarial training. In the context of steganography, GANs can be employed to create visually realistic modifications to digital content while hiding information within the generated data. GAN Steganography follows the below steps -

* Training the GAN: The GAN is trained on a dataset of normal, non-steganographic images. The generator network learns to generate realistic images, and the discriminator network learns to distinguish between real and generated images.
* Embedding Information: To hide information, modifications are introduced during the generation process. The generator is guided to produce images that not only look realistic but also carry hidden data.
* Detection Resistance: The adversarial nature of GANs can make the embedded information resistant to detection by traditional steganalysis techniques. The generator attempts to create modifications that are visually indistinguishable from non-steganographic images.
* Extraction of Hidden Information: The recipient, who has knowledge of the GAN and the embedding process, can use the generator to extract the hidden information from the modified images.

GAN steganography offers certain advantages, such as the potential for high-capacity information hiding and resistance to detection. However, it also presents challenges, including the need for robust training to ensure the generated content remains visually convincing and the potential for unintended information leakage. The model used in the project is trained different than the usual approach. A Steganalysis layer and Discriminator layer is defined. Steganalysis layer takes in 2 inputs, original image and encoded image. It uses PSNR ratio and statistical approach to differentiate between both images and returns prediction if the images are same or not. A certain percentage of digital random Perlin noise is added to both inputs to simulate real life noise. The Discriminator layer takes 2 inputs, a secret message which is 100 bits long and image to encode. Analysing the results of previous layers, it adds null characters in between to spread out the bits across the entire image, such that when the expanded secret is encoded in the image, the Steganalysis layer fails to differentiate between encoded and original image. Both layers are pitted against each other using Min-Max algorithm to maximize performance of our Discriminator layer.

* + - 1. **BCH Error Correcting Codes**

BCH (Bose-Chaudhuri-Hocquenghem) codes [1][6] are widely used for error correction in various communication systems and storage devices.

**1. BCH Encoding:**

**a. Message Representation:**

Suppose you have a message of length (k) bits that you want to encode using a BCH code.

**b. Constructing the Generator Polynomial:**

Choose parameters for the BCH code, including the code length (n), message length (k), and minimum distance (d). The generator polynomial is constructed based on these parameters. It is a polynomial of degree (n - k) with roots corresponding to the locations of the parity-check bits.

**c. Message Polynomial:**

Represent the message as a polynomial of degree (k - 1).

**d. Appending Parity Bits:**

Append (n - k) parity-check bits to the message polynomial by dividing the message polynomial by the generator polynomial using polynomial division.

**e. Generating Codeword:**

The resulting polynomial, which includes the original message polynomial and the added parity-check bits, represents the codeword.

**f. Binary Representation:**

The binary coefficients of the polynomial form the BCH codeword.

**2. BCH Decoding:**

**a. Received Codeword:**

The received codeword may be corrupted due to errors during transmission.

**b. Syndrome Calculation:**

Calculate the syndrome polynomial by dividing the received polynomial by the generator polynomial. The remainder is the syndrome.

**c. Error Detection:**

If the syndrome is zero, there are no errors. Otherwise, errors are present.

**d. Error Location Polynomial:**

Use the syndrome to construct an error location polynomial. The roots of this polynomial represent the positions of errors.

**e. Error Magnitude Polynomial:**

Compute an error magnitude polynomial to determine the magnitude of errors at each error location.

**f. Correcting Errors:**

Adjust the received codeword based on the error location and magnitude information to correct errors.

**3. Practical Considerations:**

**a. Parameter Selection:**

The choice of (n), (k), and (d) depends on the specific requirements of the application, considering factors such as the expected error patterns and the desired level of error correction.

**b. Decoding Algorithms:**

Various decoding algorithms, such as the Berlekamp-Massey algorithm or Reed-Solomon decoding, can be used for BCH codes.

**c. Code Rate:**

The code rate ((k/n)) represents the ratio of message length to code length. A higher code rate means less redundancy but may provide lower error correction capability.

* 1. **Code**
     1. **Backend Server**
        1. **Login Server**

from flask import Flask, request, jsonify

from flask\_mysqldb import MySQL

from flask\_cors import CORS

import random

import string

# import make\_image

app = Flask(\_\_name\_\_)

CORS(app)

app.config['MYSQL\_HOST'] = 'localhost'

app.config['MYSQL\_USER'] = 'root'

app.config['MYSQL\_PASSWORD'] = 'root'

app.config['MYSQL\_DB'] = 'mini\_project'

app.config['CORS\_HEADERS'] = 'Content-Type'

mysql = MySQL(app)

def generate\_random\_alphanumeric():

characters = string.digits + string.ascii\_uppercase + string.ascii\_lowercase

random\_sequence = ''.join(random.choice(characters) for \_ in range(7))

return random\_sequence

@app.route("/login", methods=['POST'])

async def login():

try:

data = request.get\_json()

# Access parameters from the JSON data

email = data.get('email')

password = data.get('password')

cur = mysql.connection.cursor()

users = cur.execute("SELECT \* FROM CLIENT\_CREDENTIALS WHERE email = %s", (email,))

if users < 1 :

return jsonify({"error" : True, "message" : "Invalid username"})

else :

users = cur.execute("SELECT \* FROM CLIENT\_CREDENTIALS WHERE email = %s AND password = %s", (email,password))

if users < 1:

return jsonify({"error" : True, "message" : "Invalid password"})

return jsonify({"error" : False, "message" : "Login Successful", "response" : cur.fetchall()})

except Exception as e:

# Handle exceptions, e.g., invalid JSON format

return jsonify({'error': str(e)})

@app.route("/signup", methods=['POST'])

async def signup():

try:

data = request.get\_json()

# Access parameters from the JSON data

name = data.get('name')

email = data.get('email')

password = data.get('password')

retype\_password = data.get('retype\_password')

company\_name = data.get('company\_name')

cur = mysql.connection.cursor()

if password == retype\_password:

users = cur.execute("SELECT \* FROM CLIENT\_CREDENTIALS WHERE email = %s", (email,))

if users >= 1 :

return jsonify({"error" : True, "message" : "Email exists"})

else :

cur.execute("SELECT COUNT(\*) FROM CLIENT\_CREDENTIALS")

count = cur.fetchone()[0] + 1

secret = generate\_random\_alphanumeric()

cur.execute("INSERT INTO CLIENT\_CREDENTIALS VALUES (%s,%s,%s,%s,%s);", (count,email,password,company\_name,secret))

mysql.connection.commit()

return jsonify({"error" : False, "message" : "Signin Successful", "secret" : secret})

return jsonify({"error" : True, "message" : "Passwords do not match"})

except Exception as e:

# Handle exceptions, e.g., invalid JSON format

return jsonify({'error': str(e)})

@app.route("/all", methods=['GET'])

async def all():

try:

cur = mysql.connection.cursor()

all\_companies = cur.execute("SELECT company\_name,uid FROM CLIENT\_CREDENTIALS")

return jsonify({"error" : False, "response" : cur.fetchall()})

except Exception as e:

# Handle exceptions, e.g., invalid JSON format

return jsonify({'error': str(e)})

if \_\_name\_\_ == "\_\_main\_\_":

app.run(debug=True)

* + - 1. **AI Encode Server**

import cv2

from flask import Flask, request, jsonify, send\_file

from flask\_cors import CORS

import numpy as np

import bchlib

from PIL import Image,ImageOps

import numpy as np

import tensorflow as tf

from tensorflow.python.saved\_model import tag\_constants

from tensorflow.python.saved\_model import signature\_constants

from io import BytesIO

app = Flask(\_\_name\_\_)

CORS(app)

# handle error

@app.errorhandler(400)

def not\_found\_error(error):

return jsonify({'error': 'Bad request'}), 400

@app.route('/make\_image', methods=['POST'])

async def make\_image():

if 'file' not in request.files:

return jsonify({'error': 'No file part'}), 400

file = request.files['file']

secret = request.form['secret']

if file.filename == '':

return jsonify({'error': 'No selected file'}), 400

if file:

image = Image.open(BytesIO(file.read()))

BCH\_POLYNOMIAL = 137 #137

BCH\_BITS = 5 #5

sess = tf.compat.v1.InteractiveSession(graph=tf.Graph())

model = tf.compat.v1.saved\_model.load(sess, [tag\_constants.SERVING], 'saved\_models/stegastamp\_pretrained')

input\_secret\_name = model.signature\_def[signature\_constants.DEFAULT\_SERVING\_SIGNATURE\_DEF\_KEY].inputs['secret'].name

input\_image\_name = model.signature\_def[signature\_constants.DEFAULT\_SERVING\_SIGNATURE\_DEF\_KEY].inputs['image'].name

input\_secret = tf.compat.v1.get\_default\_graph().get\_tensor\_by\_name(input\_secret\_name)

input\_image = tf.compat.v1.get\_default\_graph().get\_tensor\_by\_name(input\_image\_name)

output\_stegastamp\_name = model.signature\_def[signature\_constants.DEFAULT\_SERVING\_SIGNATURE\_DEF\_KEY].outputs['stegastamp'].name

output\_residual\_name = model.signature\_def[signature\_constants.DEFAULT\_SERVING\_SIGNATURE\_DEF\_KEY].outputs['residual'].name

output\_stegastamp = tf.compat.v1.get\_default\_graph().get\_tensor\_by\_name(output\_stegastamp\_name)

output\_residual = tf.compat.v1.get\_default\_graph().get\_tensor\_by\_name(output\_residual\_name)

width = 400

height = 400

bch = bchlib.BCH(prim\_poly = BCH\_POLYNOMIAL, t = BCH\_BITS)

data = bytearray(secret + ' '\*(7-len(secret)), 'utf-8')

ecc = bch.encode(data)

packet = data+ecc

packet\_binary = ''.join(format(x, '08b') for x in packet)

secret = [int(x) for x in packet\_binary]

secret.extend([0,0,0,0])

size = (width, height)

image = image.convert("RGB")

image = np.array(ImageOps.fit(image,size),dtype=np.float32)

image /= 255.

feed\_dict = {input\_secret:[secret],

input\_image:[image]}

hidden\_img, residual = sess.run([output\_stegastamp, output\_residual],feed\_dict=feed\_dict)

rescaled = (hidden\_img[0] \* 255).astype(np.uint8)

raw\_img = (image \* 255).astype(np.uint8)

im = Image.fromarray(np.array(rescaled))

file\_path = "out/out.jpg"

im.save(file\_path)

return send\_file(file\_path, as\_attachment=True)

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True, port=7500)

* + - 1. **AI Decode Server**

import cv2

from flask import Flask, request, jsonify, send\_file

from flask\_cors import CORS

import numpy as np

import bchlib

from PIL import Image,ImageOps

import numpy as np

import tensorflow as tf

from tensorflow.python.saved\_model import tag\_constants

from tensorflow.python.saved\_model import signature\_constants

from io import BytesIO

app = Flask(\_\_name\_\_)

CORS(app)

def get\_intersect(p1, p2, p3, p4):

s = np.vstack([p1,p2,p3,p4])

h = np.hstack((s, np.ones((4, 1))))

l1 = np.cross(h[0], h[1])

l2 = np.cross(h[2], h[3])

x, y, z = np.cross(l1, l2)

if z == 0:

return (0,0)

return (x/z, y/z)

def poly\_area(poly):

return 0.5\*np.abs(np.dot(poly[:,0],np.roll(poly[:,1],1))-np.dot(poly[:,1],np.roll(poly[:,0],1)))

def order\_points(pts):

rect = np.zeros((4, 2), dtype=np.float32)

s = pts.sum(axis=1)

rect[0] = pts[np.argmin(s)]

rect[2] = pts[np.argmax(s)]

diff = np.diff(pts, axis = 1)

rect[1] = pts[np.argmin(diff)]

rect[3] = pts[np.argmax(diff)]

return rect

# handle error

@app.errorhandler(400)

def not\_found\_error(error):

return jsonify({'error': 'Bad request'}), 400

@app.route('/validate\_image', methods=['POST'])

async def validate\_image():

if 'file' not in request.files:

return jsonify({'error': 'No file part'}), 400

file = request.files['file']

if file.filename == '':

return jsonify({'error': 'No selected file'}), 400

if file:

BCH\_POLYNOMIAL = 137 #137

BCH\_BITS = 5 #5

config = tf.compat.v1.ConfigProto()

config.gpu\_options.allow\_growth = True

detector\_graph = tf.compat.v1.Graph()

decoder\_graph = tf.compat.v1.Graph()

with detector\_graph.as\_default():

detector\_sess = tf.compat.v1.Session()

detector\_model = tf.compat.v1.saved\_model.loader.load(detector\_sess, [tag\_constants.SERVING], 'detector\_models/stegastamp\_detector')

detector\_input\_name = detector\_model.signature\_def[signature\_constants.DEFAULT\_SERVING\_SIGNATURE\_DEF\_KEY].inputs['image'].name

detector\_input = detector\_graph.get\_tensor\_by\_name(detector\_input\_name)

detector\_output\_name = detector\_model.signature\_def[signature\_constants.DEFAULT\_SERVING\_SIGNATURE\_DEF\_KEY].outputs['detections'].name

detector\_output = detector\_graph.get\_tensor\_by\_name(detector\_output\_name)

with decoder\_graph.as\_default():

decoder\_sess = tf.compat.v1.Session()

decoder\_model = tf.compat.v1.saved\_model.loader.load(decoder\_sess, [tag\_constants.SERVING], 'saved\_models/stegastamp\_pretrained')

decoder\_input\_name = decoder\_model.signature\_def[signature\_constants.DEFAULT\_SERVING\_SIGNATURE\_DEF\_KEY].inputs['image'].name

decoder\_input = decoder\_graph.get\_tensor\_by\_name(decoder\_input\_name)

decoder\_output\_name = decoder\_model.signature\_def[signature\_constants.DEFAULT\_SERVING\_SIGNATURE\_DEF\_KEY].outputs['decoded'].name

decoder\_output = decoder\_graph.get\_tensor\_by\_name(decoder\_output\_name)

bch = bchlib.BCH(prim\_poly=BCH\_POLYNOMIAL, t=BCH\_BITS)

np\_array = np.frombuffer(file.read(), np.uint8)

frame = cv2.imdecode(np\_array, cv2.IMREAD\_COLOR)

f\_height, f\_width = frame.shape[0:2]

frame\_rgb = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)

detector\_image\_input = cv2.resize(frame\_rgb, (1024,1024))

detector\_image\_input = np.expand\_dims(np.float32(detector\_image\_input),axis=0)/255.0

output\_image = detector\_sess.run(detector\_output,feed\_dict={detector\_input:detector\_image\_input})

output\_image = np.array(output\_image[0,:,:,:])

output\_image = x = np.argmax(output\_image, axis = -1)

color\_codes = np.array([[255,255,255],[0,0,0]])

out\_vis\_image = color\_codes[output\_image.astype(int)]

mask\_im = cv2.resize(np.float32(out\_vis\_image), (f\_width,f\_height))

contours, \_ = cv2.findContours(cv2.cvtColor(mask\_im, cv2.COLOR\_BGR2GRAY).astype(np.uint8),1,2)

extrema = np.zeros((8,2))

corners = np.zeros((4,2))

for cnt in contours:

area = cv2.contourArea(cnt)

if area < 1000:

continue

hull = cv2.convexHull(cnt)

if len(hull) < 4:

continue

extrema[0,:] = hull[np.argmax(hull[:,0,0]),0,:]

extrema[1,:] = hull[np.argmax(hull[:,0,0]+hull[:,0,1]),0,:]

extrema[2,:] = hull[np.argmax(hull[:,0,1]),0,:]

extrema[3,:] = hull[np.argmax(-hull[:,0,0]+hull[:,0,1]),0,:]

extrema[4,:] = hull[np.argmax(-hull[:,0,0]),0,:]

extrema[5,:] = hull[np.argmax(-hull[:,0,0]-hull[:,0,1]),0,:]

extrema[6,:] = hull[np.argmax(-hull[:,0,1]),0,:]

extrema[7,:] = hull[np.argmax(hull[:,0,0]-hull[:,0,1]),0,:]

extrema\_lines = extrema - np.roll(extrema, shift=1, axis=0)

extrema\_len = extrema\_lines[:,0]\*\*2 + extrema\_lines[:,1]\*\*2

line\_idx = np.sort(extrema\_len.argsort()[-4:])

for c in range(4):

p1 = extrema[line\_idx[(c-1)%4],:]

p2 = extrema[(line\_idx[(c-1)%4]-1)%8,:]

p3 = extrema[line\_idx[c],:]

p4 = extrema[(line\_idx[c]-1)%8,:]

corners[c,:] = get\_intersect(p1, p2, p3, p4)

new\_area = poly\_area(corners)

if new\_area / area > 1.5:

continue

corners = order\_points(corners)

corners\_full\_res = corners

pts\_dst = np.array([[0,0],[399,0],[399,399],[0,399]])

h, status = cv2.findHomography(corners\_full\_res, pts\_dst)

try:

warped\_im = cv2.warpPerspective(frame\_rgb, h, (400,400))

w\_im = warped\_im.astype(np.float32)

w\_im /= 255.

except:

continue

for im\_rotation in range(4):

w\_rotated = np.rot90(w\_im, im\_rotation)

recovered\_secret = decoder\_sess.run([decoder\_output],feed\_dict={decoder\_input:[w\_rotated]})[0][0]

recovered\_secret = list(recovered\_secret)

recovered\_secret = [int(i) for i in recovered\_secret]

packet\_binary = "".join([str(bit) for bit in recovered\_secret[:96]])

footer = recovered\_secret[96:]

if np.sum(footer) > 0:

continue

packet = bytes(int(packet\_binary[i : i + 8], 2) for i in range(0, len(packet\_binary), 8))

packet = bytearray(packet)

data, ecc = packet[:-bch.ecc\_bytes], packet[-bch.ecc\_bytes:]

bitflips = bch.decode(data, ecc)

if bitflips != -1:

try:

code = data.decode("utf-8")

return jsonify({"success" : True, "code" : str(code)})

except:

continue

return jsonify({"success" : False, "message" : "No Code Detected"})

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True, port=8080)

* + 1. **Frontend**
       1. **./App.js**

import React from "react";

import Index from "pages/index";

import ClientLogin from "pages/clientLogin";

import ClientEncodeImage from "pages/clientEncodeImage";

import CustomerDecodeImage from "pages/customerDecodeImage";

import privacyPolicy from "pages/privacyPolicy";

import Page404 from "pages/page404";

import { BrowserRouter as Router, Switch, Route } from "react-router-dom";

import { createGlobalStyle } from "styled-components";

const GlobalStyles = createGlobalStyle`

body {

margin: 0;

padding: 0;

font-family: sans-serif;

}

`;

export default () => (

<Router>

<GlobalStyles />

<Switch>

<Route exact path='/' component={Index}/>

<Route exact path='/index' component={Index}/>

<Route exact path='/client-login' component={ClientLogin}/>

<Route exact path='/client-encode-image' component={ClientEncodeImage}/>

<Route exact path='/customer-decode-image' component={CustomerDecodeImage}/>

<Route exact path='/privacy-policy' component={privacyPolicy}/>

<Route component={Page404}/>

</Switch>

</Router>

);

* + - 1. **./index.js**

import React from "react";

import ReactDOM from "react-dom";

import App from "./App";

import QAPI from "qapi";

window.QAPI = QAPI;

ReactDOM.render(<App />, document.getElementById("root"));

* + - 1. **./pages/clientEncodeImage.js**

import React, {useState} from "react";

import theme from "theme";

import { Theme, Image, Text, LinkBox, Box, Section, Button } from "@quarkly/widgets";

import { GlobalQuarklyPageStyles } from "global-page-styles";

import { RawHtml, Override } from "@quarkly/components";

import \* as Components from "components";

import { FiMenu } from "react-icons/fi";

import { AiOutlineMenu } from "react-icons/ai";

import { FileUploader } from "react-drag-drop-files";

import { ColorRing } from 'react-loader-spinner';

export default (() => {

    const [isLoggedIn,set\_isLoggedIn] = useState(sessionStorage.getItem('isLoggedIn'));

    const fileTypes = ["JPEG", "PNG", "JPG"];

    const [File, setFile] = useState(null);

    const [Loading,setLoading] = useState(null);

    const handleChange = (file) => {

        setFile(file[0]);

    };

    const encode = async (event) => {

        event.preventDefault();

        const formData = new FormData();

        formData.append('file', File);

        formData.append('secret',sessionStorage.getItem('secret'));

        try {

            setLoading(true);

            const response = await fetch('http://localhost:7500/make\_image', {

              method: 'POST',

              body: formData,

            });

            if (response.ok) {

              const blob = await response.blob();

              const url = window.URL.createObjectURL(blob);

              const link = document.createElement('a');

              link.href = url;

              link.download = File.name.split('.')[0]+'\_encoded.jpg';

              document.body.appendChild(link);

              link.click();

              setLoading(false);

            } else {

              console.error('Failed to fetch image');

            }

            setLoading(false);

          } catch (error) {

            setLoading(false);

            console.error('Error:', error);

        }

    }

    if (isLoggedIn){

        return <Theme theme={theme}>

        <head>

            <title>Encrypt</title>

        </head>

        <GlobalQuarklyPageStyles pageUrl={"client-encode-image"} />

        <Section sm-padding="8px 0 8px 0" quarkly-title="Header-4" background="#EDF2F6">

            <Override slot="SectionContent" flex-direction="row" justify-content="space-between" />

            <Box

                display="flex"

                padding="12px 0"

                justify-content="space-between"

                align-items="flex-start"

                flex-direction="row"

                width="30%"

                sm-width="50%"

                sm-align-items="center"

                sm-flex-direction="row"

                sm-justify-content="flex-start"

                md-width="50%"

                lg-width="70%"

                md-justify-content="flex-start"

            >

                <LinkBox flex-direction="row" href="/index" display="flex" grid-gap="12px">

                    <Image src="https://uploads.quarkly.io/6457b971d28fe100213a0f35/images/flower-logo-template.svg?v=2023-08-06T19:34:51.878Z" display="block" width="36px" height="36px" />

                    <Text

                        margin="0"

                        md-margin="0px 0 0px 0"

                        text-align="left"

                        font="--headline3"

                        sm-margin="0px 0 0px 0"

                        display="block"

                    >

                        AuthGen

                    </Text>

                </LinkBox>

            </Box>

            <Components.QuarklycommunityKitMobileSidePanel

                menuPosition="full"

                breakpoint="lg"

                width="70%"

                sm-width="50%"

                md-width="50%"

                lg-width="30%"

            >

                <Override slot="Children" md-display="flex" />

                <Override

                    slot="Content"

                    padding="0px 0px 0px 0px"

                    background="rgba(255, 255, 255, 0)"

                    lg-background="#ffffff"

                    lg-margin="0px 0px 0px 0px"

                />

                <Override

                    slot="Button Text"

                    font="normal 600 16px/1.5 -apple-system, system-ui, 'Segoe UI', 'Roboto', 'Oxygen', 'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue', sans-serif"

                    text-transform="uppercase"

                    letter-spacing="1px"

                    sm-font="normal 600 14px/1.5 -apple-system, system-ui, 'Segoe UI', 'Roboto', 'Oxygen', 'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue', sans-serif"

                    sm-margin="0px 2px 0px 0px"

                    lg-margin="0px 0px 0px 0px"

                />

                <Override

                    slot="Button Icon :closed"

                    category="fi"

                    icon={FiMenu}

                    size="32px"

                    padding="5px 7px 5px 7px"

                    border-radius="50px"

                />

                <Override

                    slot="Button Icon"

                    width="28px"

                    height="28px"

                    category="ai"

                    icon={AiOutlineMenu}

                    color="--dark"

                    size="24px"

                    lg-width="32px"

                    lg-height="32px"

                />

                <Override

                    slot="Cross"

                    lg-width="32px"

                    lg-height="32px"

                    size="32px"

                    top="24px"

                    right="24px"

                />

                <Box

                    align-items="center"

                    lg-justify-content="center"

                    lg-align-items="flex-start"

                    justify-content="flex-end"

                    display="flex"

                    lg-flex-direction="column"

                    lg-margin="0px auto 0px auto"

                    lg-min-width="300px"

                    lg-max-width="1280px"

                    lg-width="90%"

                    lg-padding="24px 0px 48px 0px"

                >

                    <Box

                        display="none"

                        lg-width="100%"

                        lg-margin="0px 0px 24px 0px"

                        lg-display="flex"

                        lg-padding="12px 0px 12px 0px"

                    >

                        <LinkBox flex-direction="row" href="/index" display="flex" grid-gap="12px">

                            <Image src="https://uploads.quarkly.io/6457b971d28fe100213a0f35/images/flower-logo-template.svg?v=2023-08-06T19:34:51.878Z" display="block" width="36px" height="36px" />

                            <Text

                                margin="0"

                                md-margin="0px 0 0px 0"

                                text-align="left"

                                font="--lead"

                                sm-margin="0px 0 0px 0"

                                display="block"

                                lg-font="--headline3"

                            >

                                Company

                            </Text>

                        </LinkBox>

                    </Box>

                    <Components.QuarklycommunityKitMenu

                        display="flex"

                        filterMode="exclude"

                        filterPages="/index"

                        grid-gap="8px"

                        lg-flex-direction="column"

                        lg-padding="6px 0px 6px 0px"

                        lg-margin="0px 0px 24px 0px"

                        align-items="center"

                        flex-wrap="wrap"

                        overflow-x="visible"

                        overflow-y="visible"

                        lg-align-items="flex-start"

                    >

                        <Override

                            slot="link"

                            color="--darkL2"

                            hover-color="--primary"

                            font="--lead"

                            text-decoration-line="initial"

                            transition="color 0.1s ease 0s"

                            lg-font="--lead"

                        />

                        <Override

                            slot="item-active"

                            border-width="0 0 2px 0"

                            border-style="solid"

                            border-color="--color-darkL2"

                            lg-border-width="0 0 0 2px"

                        />

                        <Override slot="item" padding="6px 8px 6px 8px" lg-padding="6px 12px 6px 12px" />

                        <Override slot="link-active" cursor="default" color="--darkL2" hover-color="--darkL2" />

                        <Override slot="link-about" />

                    </Components.QuarklycommunityKitMenu>

                    <Box

                        width="25%"

                        display="none"

                        lg-width="100%"

                        lg-flex-direction="column"

                        lg-align-items="flex-start"

                        lg-display="flex"

                        justify-content="space-around"

                        align-items="center"

                        flex-wrap="wrap"

                        lg-margin="0px 0px 0px 0px"

                    >

                    </Box>

                </Box>

            </Components.QuarklycommunityKitMobileSidePanel>

        </Section>

        <Section sm-padding="8px 0 8px 0" quarkly-title="Header-5" background="#EDF2F6" min-height="85vh" align-items="center">

        <Box align-items="center" display="flex" justify-content="center" flex-direction="column">

        <FileUploader

            multiple={true}

            handleChange={handleChange}

            name="file"

            types={fileTypes}

        />

        <Text width="100%"

                    lg-width="100%"

                    font="--lead"

                    color="--grey"

                    margin="10px 0px 10px 0px"

                    text-align="center">{File ? `File name: ${File.name}` : "No files uploaded"}</Text>

        {File && <Button padding="11px 24px 11px 24px"

                                font="normal 400 20px/1.5 --fontFamily-sans"

                                sm-margin="10px 0px 0px 0px"

                                sm-width="100%"

                                focus-box-shadow="none"

                                color="--light"

                                background="#1A1B41"

                                border-radius="8px"

                                hover-background="--color-darkL1"

                                hover-color="--light"

                                hover-transition="background-color 0.5s ease 0s"

                                transition="background-color 0.1s ease 0s"

                                margin="10px 0px 15px 0px"

                                hover-border-width="0px"

                                hover-border-style="none"

                                className="signup"

                                id="signup-action"

                                onClick = {encode}>Encode</Button>}

            {Loading ? (<ColorRing

            visible={true}

            height="80"

            width="80"

            ariaLabel="blocks-loading"

            wrapperStyle={{}}

            wrapperClass="blocks-wrapper"

            colors={['#e15b64', '#f47e60', '#f8b26a', '#abbd81', '#849b87']}

        />) : (<></>)}

        </Box>

        </Section>

        <RawHtml>

            <style place={"endOfHead"} rawKey={"6575b98c1d712a00232c66dc"}>

                {":root {\n  box-sizing: border-box;\n}\n\n\* {\n  box-sizing: inherit;\n}"}

            </style>

        </RawHtml>

        </Theme>;

    }

    else{

        set\_isLoggedIn(sessionStorage.getItem('isLoggedIn'));

        window.location.href = '/client-login';

        return ;

    }

});

* + - 1. **./pages/clientLogin.js**

import React from "react";

import theme from "theme";

import { Theme, Link, Image, Text, LinkBox, Box, Section, Input, Button } from "@quarkly/widgets";

import { GlobalQuarklyPageStyles } from "global-page-styles";

import { RawHtml, Override } from "@quarkly/components";

import \* as Components from "components";

import { FiMenu } from "react-icons/fi";

import { AiOutlineMenu } from "react-icons/ai";

const login = async (event) => {

    event.preventDefault();

    var email = document.getElementById("login-email").value;

    var password = document.getElementById("login-password").value;

    var email\_box = document.getElementById("login-email");

    var password\_box = document.getElementById("login-password");

    var response = await fetch('http://localhost:5000/login', {

        method: 'POST',

        headers: {

            'Content-Type': 'application/json',

        },

        body: JSON.stringify({

            email: email,

            password: password,

        }),

    })

    response = await response.json();

    console.log(response);

    if (!response.error){

        sessionStorage.setItem('secret', response["response"][0][4]);

        sessionStorage.setItem('isLoggedIn', true);

        window.location.href = '/client-encode-image';

    }

    else{

        if(response.message === "Invalid username"){

            email\_box.style["border-color"] = "red";

            password\_box.style["border-color"] = "#E4E8EC";

        }

            if(response.message === "Invalid password"){

            email\_box.style["border-color"] = "#E4E8EC";

            password\_box.style["border-color"] = "red";

        }

    }

}

const signup = async (event) => {

    event.preventDefault();

    var email = document.getElementById("signup-email").value;

    var password = document.getElementById("signup-password").value;

    var retype\_password = document.getElementById("signup-retype-password").value;

    var company\_name = document.getElementById("signup-comapny-name").value;

    var email\_box = document.getElementById("signup-email");

    var password\_box = document.getElementById("signup-password");

    var retype\_password\_box = document.getElementById("signup-retype-password");

    var response = await fetch('http://localhost:5000/signup', {

        method: 'POST',

        headers: {

            'Content-Type': 'application/json',

        },

        body: JSON.stringify({

            email: email,

            password: password,

            retype\_password : retype\_password,

            company\_name : company\_name

        }),

    })

    response = await response.json();

    console.log(response);

    if (!response.error){

        sessionStorage.setItem('secret', response["secret"]);

        sessionStorage.setItem('isLoggedIn', true);

        window.location.href = '/client-encode-image';

    }

    else{

        if(response.message === "Email exists"){

            email\_box.style["border-color"] = "red";

            password\_box.style["border-color"] = "#E4E8EC";

            retype\_password\_box.style["border-color"] = "#E4E8EC";

        }

        if(response.message === "Passwords do not match"){

            email\_box.style["border-color"] = "#E4E8EC";

            password\_box.style["border-color"] = "red";

            retype\_password\_box.style["border-color"] = "red";

        }

    }

}

export default (() => {

    return <Theme theme={theme}>

        <head>

            <title>Login</title>

        </head>

        <GlobalQuarklyPageStyles pageUrl={"client-login"} />

        <Section sm-padding="8px 0 8px 0" quarkly-title="Header-4" background="#EDF2F6">

            <Override slot="SectionContent" flex-direction="row" justify-content="space-between" />

            <Box

                display="flex"

                padding="12px 0"

                justify-content="space-between"

                align-items="flex-start"

                flex-direction="row"

                width="30%"

                sm-width="50%"

                sm-align-items="center"

                sm-flex-direction="row"

                sm-justify-content="flex-start"

                md-width="50%"

                lg-width="70%"

                md-justify-content="flex-start"

            >

                <LinkBox flex-direction="row" href="/index" display="flex" grid-gap="12px">

                    <Image src="https://uploads.quarkly.io/6457b971d28fe100213a0f35/images/flower-logo-template.svg?v=2023-08-06T19:34:51.878Z" display="block" width="36px" height="36px" />

                    <Text

                        margin="0"

                        md-margin="0px 0 0px 0"

                        text-align="left"

                        font="--headline3"

                        sm-margin="0px 0 0px 0"

                        display="block"

                    >

                        AuthGen

                    </Text>

                </LinkBox>

            </Box>

            <Components.QuarklycommunityKitMobileSidePanel

                menuPosition="full"

                breakpoint="lg"

                width="70%"

                sm-width="50%"

                md-width="50%"

                lg-width="30%"

            >

                <Override slot="Children" md-display="flex" />

                <Override

                    slot="Content"

                    padding="0px 0px 0px 0px"

                    background="rgba(255, 255, 255, 0)"

                    lg-background="#ffffff"

                    lg-margin="0px 0px 0px 0px"

                />

                <Override

                    slot="Button Text"

                    font="normal 600 16px/1.5 -apple-system, system-ui, 'Segoe UI', 'Roboto', 'Oxygen', 'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue', sans-serif"

                    text-transform="uppercase"

                    letter-spacing="1px"

                    sm-font="normal 600 14px/1.5 -apple-system, system-ui, 'Segoe UI', 'Roboto', 'Oxygen', 'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue', sans-serif"

                    sm-margin="0px 2px 0px 0px"

                    lg-margin="0px 0px 0px 0px"

                />

                <Override

                    slot="Button Icon :closed"

                    category="fi"

                    icon={FiMenu}

                    size="32px"

                    padding="5px 7px 5px 7px"

                    border-radius="50px"

                />

                <Override

                    slot="Button Icon"

                    width="28px"

                    height="28px"

                    category="ai"

                    icon={AiOutlineMenu}

                    color="--dark"

                    size="24px"

                    lg-width="32px"

                    lg-height="32px"

                />

                <Override

                    slot="Cross"

                    lg-width="32px"

                    lg-height="32px"

                    size="32px"

                    top="24px"

                    right="24px"

                />

                <Box

                    align-items="center"

                    lg-justify-content="center"

                    lg-align-items="flex-start"

                    justify-content="flex-end"

                    display="flex"

                    lg-flex-direction="column"

                    lg-margin="0px auto 0px auto"

                    lg-min-width="300px"

                    lg-max-width="1280px"

                    lg-width="90%"

                    lg-padding="24px 0px 48px 0px"

                >

                    <Components.QuarklycommunityKitMenu

                        display="flex"

                        filterMode="exclude"

                        filterPages="/index"

                        grid-gap="8px"

                        lg-flex-direction="column"

                        lg-padding="6px 0px 6px 0px"

                        lg-margin="0px 0px 24px 0px"

                        align-items="center"

                        flex-wrap="wrap"

                        overflow-x="visible"

                        overflow-y="visible"

                        lg-align-items="flex-start"

                    >

                        <Override

                            slot="link"

                            color="--darkL2"

                            hover-color="--primary"

                            font="--lead"

                            text-decoration-line="initial"

                            transition="color 0.1s ease 0s"

                            lg-font="--lead"

                        />

                        <Override

                            slot="item-active"

                            border-width="0 0 2px 0"

                            border-style="solid"

                            border-color="--color-darkL2"

                            lg-border-width="0 0 0 2px"

                        />

                        <Override slot="item" padding="6px 8px 6px 8px" lg-padding="6px 12px 6px 12px" />

                        <Override slot="link-active" cursor="default" color="--darkL2" hover-color="--darkL2" />

                        <Override slot="link-about" />

                    </Components.QuarklycommunityKitMenu>

                    <Box

                        width="25%"

                        display="none"

                        lg-width="100%"

                        lg-flex-direction="column"

                        lg-align-items="flex-start"

                        lg-display="flex"

                        justify-content="space-around"

                        align-items="center"

                        flex-wrap="wrap"

                        lg-margin="0px 0px 0px 0px"

                    >

                    </Box>

                </Box>

            </Components.QuarklycommunityKitMobileSidePanel>

        </Section>

        <Section padding="80px 0 80px 0" position="relative" quarkly-title="Form-3" background="#EDF2F6">

            <Override slot="SectionContent" max-width="1220px" />

            <Box

                z-index="1"

                left="0px"

                top="0px"

                right="auto"

                bottom="auto"

                position="static"

                grid-template-columns="repeat(2, 1fr)"

            >

                <Box min-width="100px" min-height="100px">

                    <Text

                        lg-padding="0px 0px 0px 0px"

                        sm-font="normal 700 36px/1.2 --fontFamily-sans"

                        margin="0px 0px 18px 0px"

                        font="normal 600 42px/1.2 --fontFamily-sans"

                        color="--darkL1"

                        text-align="left"

                        padding="0px 0 0px 0px"

                        lg-width="100%"

                        lg-margin="0px 0px 25px 0px"

                    >

                        Ready to redefine authenticity?

                    </Text>

                    <Text margin="0px 50% 50px 0px" font="normal 300 16px/1.5 --fontFamily-sansHelvetica" lg-margin="0px 0 50px 0px" />

                </Box>

                <Box

                    position="relative"

                    z-index="1"

                    display="flex"

                    lg-flex-direction="column"

                    flex-direction="row"

                >

                    <Components.QuarklycommunityKitNetlifyForm lg-margin="0px 0px 35px 0px" sm-margin="0px 0px 25px 0px">

                        <Box display="flex" margin="0px 0px 15px 0px" flex-direction="column" className="signup">

                            <Box

                                min-width="10px"

                                min-height="10px"

                                display="flex"

                                margin="0px 0px 20px 0px"

                                sm-flex-direction="column"

                                className="signup"

                            >

                                <Box

                                    min-width="10px"

                                    min-height="10px"

                                    margin="0px 15px 0px 0px"

                                    sm-margin="0px 0 15px 0px"

                                    lg-width="50%"

                                    sm-width="100%"

                                    className="signup"

                                >

                                    <Text margin="0px 0px 8px 0px" font="normal 400 16px/1.5 --fontFamily-sans">

                                        Name

                                    </Text>

                                    <Input

                                        margin="0px 10px 0px 0px"

                                        padding="12px 16px 12px 16px"

                                        width="100%"

                                        font="--lead"

                                        md-max-width="none"

                                        border-radius="8px"

                                        type="text"

                                        required

                                        md-margin="0px 0 16px 0px"

                                        background="rgba(255, 255, 255, 0.1)"

                                        border-color="--color-lightD2"

                                        className="signup"

                                        id="signup-name"

                                    />

                                </Box>

                                <Box

                                    min-width="10px"

                                    min-height="10px"

                                    lg-width="50%"

                                    sm-width="100%"

                                    className="signup"

                                >

                                    <Text margin="0px 0px 8px 0px" font="normal 400 16px/1.5 --fontFamily-sans">

                                        Email

                                    </Text>

                                    <Input

                                        margin="0px 0 0px 0px"

                                        padding="12px 16px 12px 16px"

                                        width="100%"

                                        font="--lead"

                                        md-max-width="none"

                                        border-radius="8px"

                                        type="text"

                                        required

                                        md-margin="0px 0 16px 0px"

                                        background="rgba(255, 255, 255, 0.1)"

                                        border-color="--color-lightD2"

                                        className="signup"

                                        id="signup-email"

                                    />

                                </Box>

                            </Box>

                            <Box

                                min-width="10px"

                                min-height="10px"

                                display="flex"

                                margin="0px 0px 20px 0px"

                                sm-flex-direction="column"

                                className="signup"

                            >

                                <Box

                                    min-width="10px"

                                    min-height="10px"

                                    margin="0px 15px 0px 0px"

                                    sm-margin="0px 0 15px 0px"

                                    lg-width="50%"

                                    sm-width="100%"

                                    className="signup"

                                >

                                    <Text margin="0px 0px 8px 0px" font="normal 400 16px/1.5 --fontFamily-sans">

                                        Password

                                    </Text>

                                    <Input

                                        margin="0px 10px 0px 0px"

                                        padding="12px 16px 12px 16px"

                                        width="100%"

                                        font="--lead"

                                        md-max-width="none"

                                        border-radius="8px"

                                        type="password"

                                        required

                                        md-margin="0px 0 16px 0px"

                                        background="rgba(255, 255, 255, 0.1)"

                                        border-color="--color-lightD2"

                                        className="signup"

                                        id="signup-password"

                                    />

                                </Box>

                                <Box

                                    min-width="10px"

                                    min-height="10px"

                                    lg-width="50%"

                                    sm-width="100%"

                                    className="signup"

                                >

                                    <Text margin="0px 0px 8px 0px" font="normal 400 16px/1.5 --fontFamily-sans">

                                        Re-type password

                                    </Text>

                                    <Input

                                        margin="0px 0 0px 0px"

                                        padding="12px 16px 12px 16px"

                                        width="100%"

                                        font="--lead"

                                        md-max-width="none"

                                        border-radius="8px"

                                        type="password"

                                        required

                                        md-margin="0px 0 16px 0px"

                                        background="rgba(255, 255, 255, 0.1)"

                                        border-color="--color-lightD2"

                                        className="signup"

                                        id="signup-retype-password"

                                    />

                                </Box>

                            </Box>

                            <Text margin="0px 0px 8px 0px" font="normal 400 16px/1.5 --fontFamily-sans">

                                Company Name

                            </Text>

                            <Input

                                margin="0px 0 12px 0px"

                                padding="12px 16px 12px 16px"

                                width="100%"

                                font="--lead"

                                md-max-width="none"

                                border-radius="8px"

                                type="text"

                                required

                                md-margin="0px 0 16px 0px"

                                background="rgba(255, 255, 255, 0.1)"

                                border-color="--color-lightD2"

                                className="signup"

                                id="signup-comapny-name"

                            />

                            <Button

                                padding="11px 24px 11px 24px"

                                font="normal 400 20px/1.5 --fontFamily-sans"

                                sm-margin="10px 0px 0px 0px"

                                sm-width="100%"

                                focus-box-shadow="none"

                                color="--light"

                                background="#1A1B41"

                                border-radius="8px"

                                hover-background="--color-darkL1"

                                hover-color="--light"

                                hover-transition="background-color 0.5s ease 0s"

                                transition="background-color 0.1s ease 0s"

                                margin="10px 0px 15px 0px"

                                hover-border-width="0px"

                                hover-border-style="none"

                                className="signup"

                                id="signup-action"

                                onClick = {signup}

                            >

                                Signup

                            </Button>

                            <Box min-width="10px" min-height="10px" display="flex" sm-flex-direction="column">

                                <Text margin="0px 5px 0px 0px" font="normal 400 18px/1.5 --fontFamily-sansHelvetica">

                                    We care about your data. Read our

                                </Text>

                                <Link href="/privacy-policy" color="#561dc6" font="normal 400 18px/1.5 --fontFamily-sansHelvetica" target="\_blank">

                                    privacy policy

                                </Link>

                            </Box>

                        </Box>

                    </Components.QuarklycommunityKitNetlifyForm>

                    <Components.QuarklycommunityKitNetlifyForm

                        lg-margin="0px 0px 35px 0px"

                        sm-margin="0px 0px 25px 0px"

                        position="relative"

                        left="px"

                        width="100% border-box"

                        order="0"

                        flex="0 1 auto"

                        display="block"

                        margin="0px 0px 0px 0px"

                        padding="0px 0px 0px px"

                        align-self="flex-start"

                        right={0}

                    >

                        <Override slot="Form" width="600px" />

                        <Box display="flex" margin="0px 0px 15px 20px" flex-direction="column" className="login">

                            <Box

                                min-width="10px"

                                min-height="10px"

                                display="flex"

                                margin="0px 0px 20px 0px"

                                sm-flex-direction="column"

                                width="100%"

                                padding="0px 0px 0px 75px"

                                className="login"

                            >

                                <Box

                                    min-width="10px"

                                    min-height="10px"

                                    margin="0px 0px 0px 0px"

                                    sm-margin="0px 0 15px 0px"

                                    lg-width="50%"

                                    sm-width="100%"

                                    width="600px"

                                    border-width={0}

                                    padding="0px 0px 0px 0px"

                                    className="login"

                                >

                                    <Text margin="0px 0px 8px 0px" font="normal 400 16px/1.5 --fontFamily-sans">

                                        Email

                                    </Text>

                                    <Input

                                        margin="0px 0 12px 0px"

                                        padding="12px 16px 12px 16px"

                                        width="75%"

                                        font="--lead"

                                        md-max-width="none"

                                        border-radius="8px"

                                        type="text"

                                        required

                                        md-margin="0px 0 16px 0px"

                                        background="rgba(255, 255, 255, 0.1)"

                                        border-color="--color-lightD2"

                                        className="login"

                                        id="login-email"

                                    />

                                </Box>

                            </Box>

                            <Box

                                min-width="10px"

                                min-height="10px"

                                display="flex"

                                margin="0px 0px 20px 0px"

                                sm-flex-direction="column"

                                width="100%"

                                padding="0px 0px 0px 75px"

                                className="login"

                            >

                                <Box

                                    min-width="10px"

                                    min-height="10px"

                                    margin="0px 0px 0px 0px"

                                    sm-margin="0px 0 15px 0px"

                                    lg-width="50%"

                                    sm-width="100%"

                                    border-width={0}

                                    padding="0px 0px 0px 0px"

                                    width="600px"

                                    className="login"

                                >

                                    <Text margin="0px 0px 8px 0px" font="normal 400 16px/1.5 --fontFamily-sans">

                                        Password

                                    </Text>

                                    <Input

                                        margin="0px 0 12px 0px"

                                        padding="12px 16px 12px 16px"

                                        width="75%"

                                        font="--lead"

                                        md-max-width="none"

                                        border-radius="8px"

                                        type="password"

                                        required

                                        md-margin="0px 0 16px 0px"

                                        background="rgba(255, 255, 255, 0.1)"

                                        border-color="--color-lightD2"

                                        className="login"

                                        id="login-password"

                                    />

                                </Box>

                            </Box>

                            <Button

                                padding="11px 24px 11px 24px"

                                font="normal 400 20px/1.5 --fontFamily-sans"

                                sm-margin="10px 0px 0px 0px"

                                sm-width="100%"

                                focus-box-shadow="none"

                                color="--light"

                                background="#1A1B41"

                                border-radius="8px"

                                hover-background="--color-darkL1"

                                hover-color="--light"

                                hover-transition="background-color 0.5s ease 0s"

                                transition="background-color 0.1s ease 0s"

                                margin="0px 0px 15px 75px"

                                hover-border-width="0px"

                                hover-border-style="none"

                                width="75%"

                                className="login"

                                id="login-action"

                                onClick = {login}

                            >

                                Login

                            </Button>

                        </Box>

                    </Components.QuarklycommunityKitNetlifyForm>

                </Box>

            </Box>

        </Section>

        <RawHtml>

            <style place={"endOfHead"} rawKey={"6575b98c1d712a00232c66dc"}>

                {":root {\n  box-sizing: border-box;\n}\n\n\* {\n  box-sizing: inherit;\n}"}

            </style>

        </RawHtml>

    </Theme>;

});

* + - 1. **./pages/customerDecodeImage.js**

import React, { useState } from 'react';

import theme from "theme";

import { Theme, Image, Text, LinkBox, Box, Section, Button } from "@quarkly/widgets";

import { GlobalQuarklyPageStyles } from "global-page-styles";

import { RawHtml, Override } from "@quarkly/components";

import \* as Components from "components";

import { FiMenu } from "react-icons/fi";

import { AiOutlineMenu } from "react-icons/ai";

import { FileUploader } from "react-drag-drop-files";

import stringSimilarity from 'string-similarity';

import { ColorRing } from 'react-loader-spinner';

export default (() => {

    const fileTypes = ["JPEG", "PNG", "JPG"];

    const [File, setFile] = useState(null);

    const handleChange = (file) => {

        setFile(file[0]);

    };

    const [Authenticity,setAuthenticity] = useState(null);

    const [Loading,setLoading] = useState(null);

    const findMostSimilarElement = (arrays, inputString) => {

        let maxSimilarity = -1;

        let mostSimilarElement = null;

        arrays.forEach(array => {

          if (array.length === 2) {

            const similarity = stringSimilarity.compareTwoStrings(array[1], inputString);

            if (similarity > maxSimilarity) {

              maxSimilarity = similarity;

              mostSimilarElement = array;

            }

          }

        });

        return mostSimilarElement;

    };

    const authenticate = async (event) => {

        event.preventDefault();

        const formData = new FormData();

        formData.append('file', File);

        try {

            setLoading(true);

            var response = await fetch('http://localhost:8080/validate\_image', {

              method: 'POST',

              body: formData,

            });

            if (response.ok) {

                response = await response.json();

                var success = response["success"]

                if(success){

                    var code = response["code"]

                    var all\_data = await fetch('http://localhost:5000/all',{

                    method : 'GET'

                    });

                    all\_data = await all\_data.json();

                    var result = findMostSimilarElement(all\_data["response"],code)[0];

                    const element = <Box

                        padding="55px 45px 55px 45px"

                        lg-padding="45px 30px"

                        md-padding="45px 45px"

                        background="#FFFFFF"

                        border-radius="24px"

                        justify-content="flex-start"

                        flex-direction="column"

                        display="flex"

                        max-width = "50vh"

                    >

                        <Text margin="0px 0px 18px 0px" color="#1A1B41" font="--headline3" lg-text-align="center">

                            Authentic product by {result}

                        </Text>

                        <Text

                            margin="0px 0px 0px 0px"

                            color="--greyD3"

                            font="--base"

                            lg-text-align="center"

                            flex="1 0 auto"

                        >

                            This product is an authentic product by {result}, a valued client of AuthGen.

                        </Text>

                    </Box>;

                    setLoading(false);

                    setAuthenticity(element);

                }

                else{

                    const element = <Box

                        padding="55px 45px 55px 45px"

                        lg-padding="45px 30px"

                        md-padding="45px 45px"

                        background="#FFFFFF"

                        border-radius="24px"

                        justify-content="flex-start"

                        flex-direction="column"

                        display="flex"

                        max-width = "50vh"

                    >

                        <Text margin="0px 0px 18px 0px" color="#1A1B41" font="--headline3" lg-text-align="center">

                            Possibly Counterfeit

                        </Text>

                        <Text

                            margin="0px 0px 0px 0px"

                            color="--greyD3"

                            font="--base"

                            lg-text-align="center"

                            flex="1 0 auto"

                        >

                            Our algorithm suggests there is a high possibility that this product is a counterfiet. We suggest you to contact the seller or re-take your image in better lighting.

                        </Text>

                    </Box>;

                    setLoading(false);

                    setAuthenticity(element);

                }

            } else {

              console.error('Failed to fetch image');

            }

          } catch (error) {

            console.error('Error:', error);

        }

    }

    return <Theme theme={theme}>

        <head>

            <title>Authenticate</title>

        </head>

        <GlobalQuarklyPageStyles pageUrl={"customer-decode-image"} />

        <Section sm-padding="8px 0 8px 0" quarkly-title="Header-4" background="#EDF2F6">

            <Override slot="SectionContent" flex-direction="row" justify-content="space-between" />

            <Box

                display="flex"

                padding="12px 0"

                justify-content="space-between"

                align-items="flex-start"

                flex-direction="row"

                width="30%"

                sm-width="50%"

                sm-align-items="center"

                sm-flex-direction="row"

                sm-justify-content="flex-start"

                md-width="50%"

                lg-width="70%"

                md-justify-content="flex-start"

            >

                <LinkBox flex-direction="row" href="/index" display="flex" grid-gap="12px">

                    <Image src="https://uploads.quarkly.io/6457b971d28fe100213a0f35/images/flower-logo-template.svg?v=2023-08-06T19:34:51.878Z" display="block" width="36px" height="36px" />

                    <Text

                        margin="0"

                        md-margin="0px 0 0px 0"

                        text-align="left"

                        font="--headline3"

                        sm-margin="0px 0 0px 0"

                        display="block"

                    >

                        AuthGen

                    </Text>

                </LinkBox>

            </Box>

            <Components.QuarklycommunityKitMobileSidePanel

                menuPosition="full"

                breakpoint="lg"

                width="70%"

                sm-width="50%"

                md-width="50%"

                lg-width="30%"

            >

                <Override slot="Children" md-display="flex" />

                <Override

                    slot="Content"

                    padding="0px 0px 0px 0px"

                    background="rgba(255, 255, 255, 0)"

                    lg-background="#ffffff"

                    lg-margin="0px 0px 0px 0px"

                />

                <Override

                    slot="Button Text"

                    font="normal 600 16px/1.5 -apple-system, system-ui, 'Segoe UI', 'Roboto', 'Oxygen', 'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue', sans-serif"

                    text-transform="uppercase"

                    letter-spacing="1px"

                    sm-font="normal 600 14px/1.5 -apple-system, system-ui, 'Segoe UI', 'Roboto', 'Oxygen', 'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue', sans-serif"

                    sm-margin="0px 2px 0px 0px"

                    lg-margin="0px 0px 0px 0px"

                />

                <Override

                    slot="Button Icon :closed"

                    category="fi"

                    icon={FiMenu}

                    size="32px"

                    padding="5px 7px 5px 7px"

                    border-radius="50px"

                />

                <Override

                    slot="Button Icon"

                    width="28px"

                    height="28px"

                    category="ai"

                    icon={AiOutlineMenu}

                    color="--dark"

                    size="24px"

                    lg-width="32px"

                    lg-height="32px"

                />

                <Override

                    slot="Cross"

                    lg-width="32px"

                    lg-height="32px"

                    size="32px"

                    top="24px"

                    right="24px"

                />

                <Box

                    align-items="center"

                    lg-justify-content="center"

                    lg-align-items="flex-start"

                    justify-content="flex-end"

                    display="flex"

                    lg-flex-direction="column"

                    lg-margin="0px auto 0px auto"

                    lg-min-width="300px"

                    lg-max-width="1280px"

                    lg-width="90%"

                    lg-padding="24px 0px 48px 0px"

                >

                    <Box

                        display="none"

                        lg-width="100%"

                        lg-margin="0px 0px 24px 0px"

                        lg-display="flex"

                        lg-padding="12px 0px 12px 0px"

                    >

                        <LinkBox flex-direction="row" href="/index" display="flex" grid-gap="12px">

                            <Image src="https://uploads.quarkly.io/6457b971d28fe100213a0f35/images/flower-logo-template.svg?v=2023-08-06T19:34:51.878Z" display="block" width="36px" height="36px" />

                            <Text

                                margin="0"

                                md-margin="0px 0 0px 0"

                                text-align="left"

                                font="--lead"

                                sm-margin="0px 0 0px 0"

                                display="block"

                                lg-font="--headline3"

                            >

                                Company

                            </Text>

                        </LinkBox>

                    </Box>

                    <Components.QuarklycommunityKitMenu

                        display="flex"

                        filterMode="exclude"

                        filterPages="/index"

                        grid-gap="8px"

                        lg-flex-direction="column"

                        lg-padding="6px 0px 6px 0px"

                        lg-margin="0px 0px 24px 0px"

                        align-items="center"

                        flex-wrap="wrap"

                        overflow-x="visible"

                        overflow-y="visible"

                        lg-align-items="flex-start"

                    >

                        <Override

                            slot="link"

                            color="--darkL2"

                            hover-color="--primary"

                            font="--lead"

                            text-decoration-line="initial"

                            transition="color 0.1s ease 0s"

                            lg-font="--lead"

                        />

                        <Override

                            slot="item-active"

                            border-width="0 0 2px 0"

                            border-style="solid"

                            border-color="--color-darkL2"

                            lg-border-width="0 0 0 2px"

                        />

                        <Override slot="item" padding="6px 8px 6px 8px" lg-padding="6px 12px 6px 12px" />

                        <Override slot="link-active" cursor="default" color="--darkL2" hover-color="--darkL2" />

                        <Override slot="link-about" />

                    </Components.QuarklycommunityKitMenu>

                    <Box

                        width="25%"

                        display="none"

                        lg-width="100%"

                        lg-flex-direction="column"

                        lg-align-items="flex-start"

                        lg-display="flex"

                        justify-content="space-around"

                        align-items="center"

                        flex-wrap="wrap"

                        lg-margin="0px 0px 0px 0px"

                    >

                    </Box>

                </Box>

            </Components.QuarklycommunityKitMobileSidePanel>

        </Section>

        <Section sm-padding="8px 0 8px 0" quarkly-title="Header-5" background="#EDF2F6" min-height="85vh" align-items="center">

        <Box align-items="center" display="flex" justify-content="center" flex-direction="column">

        <FileUploader

            multiple={true}

            handleChange={handleChange}

            name="file"

            types={fileTypes}

        />

        <Text width="100%"

                    lg-width="100%"

                    font="--lead"

                    color="--grey"

                    margin="10px 0px 10px 0px"

                    text-align="center">{File ? `File name: ${File.name}` : "No files uploaded"}</Text>

        {File && <Button padding="11px 24px 11px 24px"

                                font="normal 400 20px/1.5 --fontFamily-sans"

                                sm-margin="10px 0px 0px 0px"

                                sm-width="100%"

                                focus-box-shadow="none"

                                color="--light"

                                background="#1A1B41"

                                border-radius="8px"

                                hover-background="--color-darkL1"

                                hover-color="--light"

                                hover-transition="background-color 0.5s ease 0s"

                                transition="background-color 0.1s ease 0s"

                                margin="10px 0px 15px 0px"

                                hover-border-width="0px"

                                hover-border-style="none"

                                className="signup"

                                id="signup-action"

                                onClick = {authenticate}>Authenticate</Button>}

        {Loading ? (<ColorRing

            visible={true}

            height="80"

            width="80"

            ariaLabel="blocks-loading"

            wrapperStyle={{}}

            wrapperClass="blocks-wrapper"

            colors={['#e15b64', '#f47e60', '#f8b26a', '#abbd81', '#849b87']}

        />) : (Authenticity)}

        </Box>

        </Section>

        <RawHtml>

            <style place={"endOfHead"} rawKey={"6575b98c1d712a00232c66dc"}>

                {":root {\n  box-sizing: border-box;\n}\n\n\* {\n  box-sizing: inherit;\n}"}

            </style>

        </RawHtml>

    </Theme>;

});

* + - 1. **./pages/index.js**

import React from "react";

import theme from "theme";

import { Theme, Link, Image, Text, LinkBox, Box, Section, Icon, Button } from "@quarkly/widgets";

import { GlobalQuarklyPageStyles } from "global-page-styles";

import { RawHtml, Override } from "@quarkly/components";

import \* as Components from "components";

import { FiMenu } from "react-icons/fi";

import { AiOutlineMenu } from "react-icons/ai";

import { FaGalacticRepublic, FaHornbill, FaReact } from "react-icons/fa";

export default (() => {

    return <Theme theme={theme}>

        <head>

            <title>AuthGen</title>

        </head>

        <GlobalQuarklyPageStyles pageUrl={"index"} />

        <Section sm-padding="8px 0 8px 0" quarkly-title="Header-6" padding="24px 0px 24px 0px" background="#EDF2F6">

            <Override

                slot="SectionContent"

                flex-direction="row"

                justify-content="space-between"

                max-width="100%"

                width="100%"

                padding="0px 24px 0px 24px"

            />

            <Components.QuarklycommunityKitMobileSidePanel

                menuPosition="full"

                breakpoint="lg"

                width="40%"

                sm-width="50%"

                md-width="50%"

                lg-width="40%"

                lg-justify-content="flex-start"

                md-justify-content="flex-end"

            >

                <Override slot="Children" md-display="flex" />

                <Override

                    slot="Content"

                    padding="0px 0px 0px 0px"

                    background="rgba(255, 255, 255, 0)"

                    lg-background="#ffffff"

                    lg-margin="0px 0px 0px 0px"

                />

                <Override

                    slot="Button Text"

                    font="normal 600 16px/1.5 -apple-system, system-ui, 'Segoe UI', 'Roboto', 'Oxygen', 'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue', sans-serif"

                    text-transform="uppercase"

                    letter-spacing="1px"

                    sm-font="normal 600 14px/1.5 -apple-system, system-ui, 'Segoe UI', 'Roboto', 'Oxygen', 'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue', sans-serif"

                    sm-margin="0px 2px 0px 0px"

                    lg-margin="0px 0px 0px 0px"

                />

                <Override

                    slot="Button Icon :closed"

                    category="fi"

                    icon={FiMenu}

                    size="32px"

                    padding="5px 7px 5px 7px"

                    border-radius="50px"

                />

                <Override

                    slot="Button Icon"

                    width="28px"

                    height="28px"

                    category="ai"

                    icon={AiOutlineMenu}

                    color="--dark"

                    size="24px"

                    lg-width="32px"

                    lg-height="32px"

                />

                <Override

                    slot="Cross"

                    lg-width="32px"

                    lg-height="32px"

                    size="32px"

                    top="24px"

                    right="24px"

                />

                <Box

                    align-items="center"

                    lg-justify-content="center"

                    lg-align-items="flex-start"

                    justify-content="flex-start"

                    display="flex"

                    lg-flex-direction="column"

                    lg-margin="0px auto 0px auto"

                    lg-min-width="300px"

                    lg-max-width="1280px"

                    lg-width="90%"

                    lg-padding="24px 0px 48px 0px"

                >

                    <Components.QuarklycommunityKitMenu

                        display="flex"

                        filterMode="exclude"

                        filterPages="/index"

                        grid-gap="4px"

                        lg-flex-direction="column"

                        lg-padding="6px 0px 6px 0px"

                        lg-margin="0px 0px 24px 0px"

                        align-items="center"

                        flex-wrap="wrap"

                        overflow-x="visible"

                        overflow-y="visible"

                        lg-align-items="flex-start"

                    >

                        <Override

                            slot="link"

                            color="--darkL2"

                            hover-color="--primary"

                            font="--base"

                            text-decoration-line="initial"

                            transition="color 0.1s ease 0s"

                            lg-font="--lead"

                        />

                        <Override

                            slot="item-active"

                            border-width="0 0 1px 0"

                            border-style="solid"

                            border-color="--color-darkL2"

                            lg-border-width="0 0 0 2px"

                        />

                        <Override slot="item" padding="6px 6px 6px 6px" lg-padding="6px 12px 6px 12px" />

                        <Override slot="link-active" cursor="default" color="--darkL2" hover-color="--darkL2" />

                        <Override slot="link-about" />

                    </Components.QuarklycommunityKitMenu>

                    <Box

                        width="25%"

                        display="none"

                        lg-width="100%"

                        lg-flex-direction="column"

                        lg-align-items="flex-start"

                        lg-display="flex"

                        justify-content="space-around"

                        align-items="center"

                        flex-wrap="wrap"

                        lg-margin="24px 0px 0px 0px"

                    >

                    </Box>

                </Box>

            </Components.QuarklycommunityKitMobileSidePanel>

            <Box

                display="flex"

                padding="12px 0"

                justify-content="center"

                align-items="flex-start"

                flex-direction="row"

                width="20%"

                sm-width="50%"

                sm-align-items="center"

                sm-flex-direction="row"

                sm-justify-content="flex-start"

                md-width="50%"

                lg-width="20%"

                md-justify-content="flex-start"

                md-order="-1"

            >

                <LinkBox flex-direction="row" href="/index" display="flex" grid-gap="12px">

                    <Image src="https://uploads.quarkly.io/6457b971d28fe100213a0f35/images/flower-logo-template.svg?v=2023-08-06T19:34:51.878Z" display="block" width="36px" height="36px" />

                    <Text

                        margin="0"

                        md-margin="0px 0 0px 0"

                        text-align="left"

                        font="--lead"

                        sm-margin="0px 0 0px 0"

                        display="block"

                        quarkly-title="Text"

                    >

                        AuthGen

                    </Text>

                </LinkBox>

            </Box>

            <Box

                display="flex"

                padding="12px 0"

                justify-content="flex-end"

                align-items="flex-start"

                flex-direction="row"

                width="40%"

                sm-width="50%"

                sm-align-items="center"

                sm-flex-direction="row"

                sm-justify-content="flex-start"

                md-width="50%"

                lg-width="40%"

                md-justify-content="flex-start"

                md-display="none"

            >

                <Link

                    border-radius="8px"

                    href="/client-login"

                    text-decoration-line="initial"

                    border-style="solid"

                    sm-padding="5px 10px 5px 10px"

                    md-padding="5px 10px 5px 10px"

                    sm-margin="10px 0px 10px 0px"

                    font="--base"

                    color="--darkL1"

                    margin="0px 0px 0px 34px"

                    md-margin="10px 0px 0px 0px"

                    border-width="2px"

                    padding="4px 12px 5px 12px"

                    border-color="#1A1B41"

                >

                    Sign Up

                </Link>

                <Link

                    font="--base"

                    text-decoration-line="initial"

                    margin="0px 0px 0px 16px"

                    background="#1A1B41"

                    border-radius="8px"

                    md-margin="0px 0px 0px 14px"

                    href="/client-login"

                    color="#ffffff"

                    padding="6px 12px 7px 12px"

                    md-order="1"

                >

                    Login

                </Link>

            </Box>

        </Section>

        <Section padding="80px 0 80px 0" background="#EDF2F6" sm-padding="60px 0 60px 0">

            <Box

                width="100%"

                display="flex"

                flex-direction="column"

                md-width="100%"

                md-align-items="center"

                md-justify-content="center"

                md-text-align="center"

                lg-width="100%"

                margin="0px 0px 56px 0px"

                align-items="center"

            >

                <Text

                    margin="0px 0px 16px 0px"

                    font="--headline2"

                    color="#1A1B41"

                    md-text-align="center"

                    sm-font="normal 700 32px/1.2 &quot;Source Sans Pro&quot;, sans-serif"

                    text-align="center"

                >

                    Take your first step to secure your production line

                </Text>

                <Text

                    margin="0px 0px 0px 0px"

                    font="--base"

                    color="--greyD3"

                    md-text-align="center"

                    text-align="center"

                    max-width="480px"

                >

                    Next-gen authentication technology in the palm of your hand

                </Text>

            </Box>

            <Box display="grid" grid-template-columns="repeat(3, 1fr)" grid-gap="32px 4%" md-grid-template-columns="1fr">

                <Box

                    padding="45px 45px"

                    lg-padding="45px 30px"

                    md-padding="45px 45px"

                    background="#FFFFFF"

                    border-radius="24px"

                    justify-content="flex-start"

                    flex-direction="column"

                    display="flex"

                >

                    <Icon

                        category="fa"

                        icon={FaGalacticRepublic}

                        margin="0px 0px 30px 0px"

                        color="--dark"

                        size="48px"

                    />

                    <Text margin="0px 0px 18px 0px" color="#1A1B41" font="--headline3" lg-text-align="left">

                        No delays

                    </Text>

                    <Text

                        margin="0px 0px 0px 0px"

                        color="--greyD3"

                        font="--base"

                        lg-text-align="left"

                        flex="1 0 auto"

                    >

                        Have virtually no delay in your production line using our 2 minutes one-time setup workflow.

                    </Text>

                </Box>

                <Box

                    padding="45px 45px"

                    lg-padding="45px 30px"

                    md-padding="45px 45px"

                    background="#FFFFFF"

                    border-radius="24px"

                    justify-content="flex-start"

                    flex-direction="column"

                    display="flex"

                >

                    <Icon

                        category="fa"

                        icon={FaHornbill}

                        margin="0px 0px 30px 0px"

                        color="--dark"

                        size="48px"

                    />

                    <Text margin="0px 0px 18px 0px" color="#1A1B41" font="--headline3" lg-text-align="left">

                        Pay-as-you-go

                    </Text>

                    <Text

                        margin="0px 0px 0px 0px"

                        color="--greyD3"

                        font="--base"

                        lg-text-align="left"

                        flex="1 0 auto"

                    >

                        With flexible payment options, you only pay for what you print. No hidden charges. Ever.

                    </Text>

                </Box>

                <Box

                    padding="45px 45px"

                    lg-padding="45px 30px"

                    md-padding="45px 45px"

                    background="#FFFFFF"

                    border-radius="24px"

                    justify-content="flex-start"

                    flex-direction="column"

                    display="flex"

                >

                    <Icon

                        category="fa"

                        icon={FaReact}

                        margin="0px 0px 30px 0px"

                        color="--dark"

                        size="48px"

                    />

                    <Text margin="0px 0px 18px 0px" color="#1A1B41" font="--headline3" lg-text-align="left">

                        Power to Customers

                    </Text>

                    <Text

                        margin="0px 0px 0px 0px"

                        color="--greyD3"

                        font="--base"

                        lg-text-align="left"

                        flex="1 0 auto"

                    >

                        AuthGen's Invisible QR code allows the end-users and customers the power to validate the products themselves.{" "}

                    </Text>

                </Box>

            </Box>

        </Section>

        <Section padding="80px 0 80px 0" sm-padding="60px 0 60px 0" background="#EDF2F6">

            <Override slot="SectionContent" flex-direction="row" flex-wrap="wrap" />

            <Box

                padding="0px 160px 0px 0px"

                border-radius="24px"

                margin="0px 0px 0px 0px"

                width="60%"

                lg-padding="0px 50px 0px 0px"

                lg-width="50%"

                md-width="100%"

                md-padding="0px 0px 0px 0px"

                display="flex"

                flex-direction="column"

                align-items="flex-start"

                md-margin="0px 0px 60px 0px"

                md-align-items="center"

            >

                <Text

                    margin="0px 0px 16px 0px"

                    font="--headline2"

                    color="#1A1B41"

                    md-text-align="center"

                    sm-font="normal 700 32px/1.2 &quot;Source Sans Pro&quot;, sans-serif"

                    border-color="#1A1B41"

                >

                    So, how do I start?

                </Text>

                <Text

                    margin="0px 0px 40px 0px"

                    font="--base"

                    color="#1A1B41"

                    md-text-align="center"

                    md-margin="0px 0px 20px 0px"

                >

                    Our setup takes less than 2 minutes.{" "}

                </Text>

                <Link

                    href="/client-login"

                    text-decoration-line="initial"

                    color="--light"

                    font="--lead"

                    hover-opacity="0.9"

                    background="#1A1B41"

                    border-radius="8px"

                    padding="11px 24px 12px 24px"

                >

                    Signup

                </Link>

            </Box>

            <Box

                display="flex"

                lg-flex-wrap="wrap"

                width="40%"

                flex-direction="column"

                lg-width="50%"

                md-width="100%"

            >

                <Box

                    width="100%"

                    display="flex"

                    flex-direction="row"

                    margin="0px 0px 40px 0px"

                    lg-margin="0px 4% 15px 0px"

                    sm-width="100%"

                    sm-margin="0px 0px 30px 0px"

                >

                    <Text

                        padding="7px 24px 8px 24px"

                        margin="0px 30px 0px 0px"

                        font="--headline3"

                        border-radius="50px"

                        align-items="center"

                        display="flex"

                        justify-content="center"

                        color="--darkL2"

                        width="49px"

                        height="49px"

                        background="--color-lightD1"

                        sm-margin="0px 14px 0px 0px"

                        border-color="#1A1B41"

                    >

                        1

                    </Text>

                    <Box>

                        <Text margin="0px 0px 18px 0px" color="#1A1B41" font="--headline3" lg-text-align="left">

                            Upload your product label

                        </Text>

                        <Text margin="0px 0px 0px 0px" color="--greyD3" font="--base" lg-text-align="left">

                            After Login, all you have to do is to upload the product label, which will then be processed in our servers.

                        </Text>

                    </Box>

                </Box>

                <Box

                    width="100%"

                    display="flex"

                    flex-direction="row"

                    margin="0px 0px 40px 0px"

                    lg-margin="0px 4% 15px 0px"

                    sm-width="100%"

                    sm-margin="0px 0px 30px 0px"

                >

                    <Text

                        padding="7px 24px 8px 24px"

                        margin="0px 30px 0px 0px"

                        font="--headline3"

                        border-radius="50px"

                        align-items="center"

                        display="flex"

                        justify-content="center"

                        color="#1A1B41"

                        width="49px"

                        height="49px"

                        background="--color-lightD1"

                        sm-margin="0px 14px 0px 0px"

                    >

                        2

                    </Text>

                    <Box>

                        <Text margin="0px 0px 18px 0px" color="#1A1B41" font="--headline3" lg-text-align="left">

                            Download. Print. Ship.

                        </Text>

                        <Text margin="0px 0px 0px 0px" color="--greyD3" font="--base" lg-text-align="left">

                            Download the encoded product label, print it and use it as usual.{" "}

                        </Text>

                    </Box>

                </Box>

                <Box

                    width="100%"

                    display="flex"

                    flex-direction="row"

                    margin="0px 0px 40px 0px"

                    lg-margin="0px 4% 15px 0px"

                    sm-width="100%"

                    sm-margin="0px 0px 30px 0px"

                >

                    <Text

                        padding="7px 24px 8px 24px"

                        margin="0px 30px 0px 0px"

                        font="--headline3"

                        border-radius="50px"

                        align-items="center"

                        display="flex"

                        justify-content="center"

                        color="#1A1B41"

                        width="49px"

                        height="49px"

                        background="--color-lightD1"

                        sm-margin="0px 14px 0px 0px"

                    >

                        3

                    </Text>

                    <Box>

                        <Text

                            margin="0px 0px 18px 0px"

                            color="--darkL2"

                            font="--headline3"

                            lg-text-align="left"

                            border-color="#1A1B41"

                        >

                            Scan the product label

                        </Text>

                        <Text margin="0px 0px 0px 0px" color="--greyD3" font="--base" lg-text-align="left">

                            Take a clear picture of your product label and scan it using our decoder. Our job is to tell you whether it is authentic or not.

                        </Text>

                    </Box>

                </Box>

            </Box>

        </Section>

        <Section padding="80px 0 80px 0" quarkly-title="Content-10" background="#EDF2F6">

            <Override

                slot="SectionContent"

                padding="0px 0 0px 0px"

                lg-padding="0px 0 0px 0px"

                display="flex"

                grid-template-columns="repeat(2, 1fr)"

                lg-grid-template-columns="1fr"

                align-items="start"

            />

            <Box min-width="100px" min-height="100px" margin="0px 0px 16px 0px">

                <Text margin="0px 0px 10px 0px" font="normal 400 18px/1.5 --fontFamily-sans" color="#5d20a9">

                    Introducing

                </Text>

                <Text margin="0px 0px 30px 0px" font="normal 600 42px/1.2 --fontFamily-sans" border-color="#1A1B41" color="#1A1B41">

                    AuthGen's Invisible QR code - the Technology behind it

                </Text>

                <Box

                    min-width="100px"

                    min-height="100px"

                    display="grid"

                    grid-template-columns="repeat(2, 1fr)"

                    grid-gap="16px 54px"

                    margin="0px 0px 25px 0px"

                    md-grid-template-columns="1fr"

                >

                    <Box min-width="100px" min-height="100px">

                        <Text margin="0px 0px 25px 0px" font="normal 300 16px/1.5 --fontFamily-sans" color="#505257">

                            Steganography, the art of hiding information within seemingly innocuous data, has evolved significantly with technological advancements. We developed a unique approach here at AuthGen that generates invisible QR codes. This innovative process combines BCH codes for error correction, Generative Adversarial Networks (GANs), and Discrete Cosine Transform (DCT) steganography to create visually imperceptible yet robust hidden messages.

                        </Text>

                        <Text margin="0px 0px 25px 0px" font="normal 300 18px/1.5 --fontFamily-sans" color="#505257">

                            Steganography involves concealing information within non-secret data to ensure that the existence of the message remains undetected. In the digital realm, this often involves embedding information within images, audio files, or other media types. In our scenario, the goal is to create invisible QR codes, embedding secret messages within images.

                        </Text>

                        <Text margin="0px 0px 0 0px" font="normal 300 18px/1.5 --fontFamily-sans" color="#505257">

                            GANs are a class of machine learning models consisting of a generator and a discriminator. The generator attempts to create realistic data, while the discriminator aims to distinguish between real and generated data. Through adversarial training, GANs learn to produce content that is increasingly indistinguishable from real data. In our application, the GAN is employed to generate a minimal residue image.

                        </Text>

                    </Box>

                    <Box min-width="100px" min-height="100px">

                        <Text margin="0px 0px 25px 0px" font="normal 300 18px/1.5 --fontFamily-sans" color="#505257">

                            BCH codes, an error-correcting code, play a crucial role in ensuring the integrity of the hidden message. By encoding the secret message using BCH codes, the system becomes more resilient to data corruption or loss during the steganographic process. This adds a layer of robustness to the overall system, enabling accurate extraction of the hidden message even in the presence of errors.

                        </Text>

                        <Text margin="0px 0px 25px 0px" font="normal 300 18px/1.5 --fontFamily-sans" color="#505257">

                            The GAN, having been trained on the foreground image and BCH-encoded message, generates a minimal residue image. This residue image captures the difference between the foreground image and the encoded message, aiming to minimize visual artifacts. This step is crucial in achieving a seamless integration of the hidden message into the foreground image without raising suspicions.

                        </Text>

                        <Text margin="0px 0px 25px 0px" font="normal 300 18px/1.5 --fontFamily-sans" color="#505257">

                            DCT steganography involves manipulating the frequency domain of an image using the DCT. By carefully combining the minimal residue image with the foreground image in this domain, the system achieves a balance where the hidden message is imperceptible to the human eye. DCT steganography ensures that the alterations made during the embedding process are visually minimal, preserving the integrity of the foreground image.

                        </Text>

                    </Box>

                </Box>

            </Box>

                <Button

                        href="https://papers.nips.cc/paper\_files/paper/2017/file/838e8afb1ca34354ac209f53d90c3a43-Paper.pdf"

                        text-decoration-line="initial"

                        color="--light"

                        font="--lead"

                        hover-opacity="0.9"

                        background="#1A1B41"

                        border-radius="8px"

                        padding="11px 24px 12px 24px"

                    >

                    Read Research Paper for reference

                </Button>

            <Box

                min-width="100px"

                min-height="100px"

                display="flex"

                margin="0px 0 0px 0px"

                lg-margin="10px 10px 0px 10px"

                lg-padding="10px 0px 0px 0"

            >

                <Image

                    src="https://i.imgur.com/HzUdpiv.png"

                    display="block"

                    border-radius="25px"

                    width="100%"

                    sizes="(max-width: 576px) 100vw,(max-width: 768px) 100vw,(max-width: 992px) 100vw,100vw"

                />

            </Box>

        </Section>

        <Section padding="80px 0 80px 0" background="#EDF2F6">

            <Box

                display="flex"

                align-items="center"

                justify-content="center"

                flex-direction="column"

                margin="0px 0px 32px 0px"

                width="100%"

            >

                <Text margin="0px 0px 0px 0px" font="--headline1" color="#1A1B41" text-align="center">

                    The Team

                </Text>

            </Box>

            <Box

                display="grid"

                lg-flex-wrap="wrap"

                align-items="stretch"

                grid-template-columns="repeat(4, 1fr)"

                grid-gap="16px"

                lg-grid-template-columns="repeat(2, 1fr)"

                sm-grid-template-columns="1fr"

                width="100%"

            >

                <Box

                    position="relative"

                    display="flex"

                    flex-direction="column"

                    align-items="center"

                    justify-content="flex-start"

                    padding="24px 24px 0px 24px"

                    align-content="center"

                    left="300px"

                >

                    <Box

                        width="100%"

                        height="auto"

                        overflow-x="hidden"

                        overflow-y="hidden"

                        position="relative"

                        padding="100% 0px 0px 0px"

                    >

                        <Image

                            border-radius="50%"

                            src="https://i.imgur.com/JgHJEs6.jpg"

                            object-fit="cover"

                            position="absolute"

                            top={0}

                            left={0}

                            bottom={0}

                            right={0}

                            display="block"

                            width="100%"

                            max-height="100%"

                            sizes="(max-width: 576px) 100vw,(max-width: 768px) 100vw,(max-width: 992px) 100vw,100vw"

                        />

                    </Box>

                    <Box padding="0px 20px 0px 20px" margin="0px 0px 0px 0px">

                        <Text

                            margin="21px 0px 0px 0px"

                            font="--headline3"

                            display="block"

                            text-align="center"

                            color="#1A1B41"

                        >

                            Sohan Gundoju

                        </Text>

                        <Text

                            margin="16px 0px 26px 0px"

                            font="--base"

                            display="block"

                            text-align="center"

                            color="--greyD2"

                        >

                            1602-21-748-051

                        </Text>

                    </Box>

                </Box>

                <Box

                    position="relative"

                    display="flex"

                    flex-direction="column"

                    align-items="center"

                    justify-content="flex-start"

                    padding="24px 24px 0px 24px"

                    left="300px"

                >

                    <Box

                        width="100%"

                        height="auto"

                        overflow-x="hidden"

                        overflow-y="hidden"

                        position="relative"

                        padding="100% 0px 0px 0px"

                    >

                        <Image

                            border-radius="50%"

                            src="https://i.imgur.com/CTcu8o3.jpg"

                            object-fit="cover"

                            position="absolute"

                            top={0}

                            left={0}

                            bottom={0}

                            right={0}

                            display="block"

                            width="100%"

                            max-height="100%"

                            sizes="(max-width: 576px) 100vw,(max-width: 768px) 100vw,(max-width: 992px) 100vw,100vw"

                        />

                    </Box>

                    <Box padding="0px 20px 0px 20px" margin="0px 0px 0px 0px">

                        <Text

                            margin="21px 0px 0px 0px"

                            font="--headline3"

                            display="block"

                            text-align="center"

                            color="#1A1B41"

                        >

                            Praneeth Thanneru

                        </Text>

                        <Text

                            margin="16px 0px 26px 0px"

                            font="--base"

                            display="block"

                            text-align="center"

                            color="--greyD2"

                        >

                            1602-21-748-033

                        </Text>

                    </Box>

                </Box>

            </Box>

        </Section>

        <RawHtml>

            <style place={"endOfHead"} rawKey={"6575b98c1d712a00232c66dc"}>

                {":root {\n  box-sizing: border-box;\n}\n\n\* {\n  box-sizing: inherit;\n}"}

            </style>

        </RawHtml>

    </Theme>;

});

* + - 1. **./pages/page404.js**

import React from "react";

import theme from "theme";

import { Theme, Link, Image, Text, Box, Section } from "@quarkly/widgets";

import { GlobalQuarklyPageStyles } from "global-page-styles";

import { RawHtml, Override } from "@quarkly/components";

export default (() => {

    return <Theme theme={theme}>

        <head>

            <title>404 Error!</title>

        </head>

        <GlobalQuarklyPageStyles pageUrl={"404"} />

        <Section padding="88px 0 88px 0" min-height="100vh" quarkly-title="404-6">

            <Override slot="SectionContent" max-width="1220px" justify-content="center" />

            <Box align-items="center" display="flex" justify-content="center" flex-direction="column">

                <Image margin="0px 0px 7px 0px" src="https://uploads.quarkly.io/612695d67f2b1f001fa06c1f/images/warning-error-delete%201.png?v=2021-08-31T00:16:56.957Z" sizes="(max-width: 576px) 100vw,(max-width: 768px) 100vw,(max-width: 992px) 100vw,100vw" />

                <Text

                    color="--dark"

                    margin="0px 0px 24px 0px"

                    sm-font="normal 700 42px/1.2 &quot;Source Sans Pro&quot;, sans-serif"

                    font="--headline1"

                    border-color="#1A1B41"

                >

                    404 Error!

                </Text>

                <Text

                    width="40%"

                    lg-width="80%"

                    font="--lead"

                    color="--grey"

                    margin="0px 0px 40px 0px"

                    text-align="center"

                >

                    The page you are looking for is not available or doesn’t belong to this website!

                </Text>

                <Link

                    font="--lead"

                    padding="12px 24px 12px 24px"

                    background="#1A1B41"

                    border-radius="8px"

                    href="/index"

                    text-decoration-line="initial"

                    color="--light"

                >

                    Go Back to Home

                </Link>

            </Box>

        </Section>

        <RawHtml>

            <style place={"endOfHead"} rawKey={"6575b98c1d712a00232c66dc"}>

                {":root {\n  box-sizing: border-box;\n}\n\n\* {\n  box-sizing: inherit;\n}"}

            </style>

        </RawHtml>

    </Theme>;

});

* + - 1. **./pages/privacyPolicy.js**

import React from "react";

import theme from "theme";

import { Theme, Link, Image, Text, Box, Section } from "@quarkly/widgets";

import { GlobalQuarklyPageStyles } from "global-page-styles";

import { RawHtml, Override } from "@quarkly/components";

export default (() => {

    return <Theme theme={theme}>

        <head>

            <title>Privacy Policy</title>

        </head>

        <GlobalQuarklyPageStyles pageUrl={"privacy-policy"} />

        <Section padding="88px 0 88px 0" min-height="100vh" quarkly-title="404-6">

            <Override slot="SectionContent" max-width="1220px" justify-content="center" />

            <Box align-items="center" display="flex" justify-content="center" flex-direction="column">

                <Image margin="0px 0px 7px 0px" src="https://i.imgur.com/9iBn6C4.png" sizes="(max-width: 576px) 100vw, (max-width: 768px) 100vw, (max-width: 992px) 100vw, 100vw" style={{ maxWidth: '128px', maxHeight: '128px' }} />

                <Text

                    color="--dark"

                    margin="0px 0px 24px 0px"

                    sm-font="normal 700 42px/1.2 &quot;Source Sans Pro&quot;, sans-serif"

                    font="--headline1"

                    border-color="#1A1B41"

                >

                    Privacy Policy

                </Text>

                <Text

                    width="80%"

                    lg-width="80%"

                    font="--lead"

                    color="--black"

                    margin="0px 0px 40px 0px"

                    text-align="justify"

                >

                    <Text color="--black"><b>Last Updated: 14th December, 2023</b></Text>

                    <Text color="--grey">

                        Welcome to AuthGen, a platform committed to safeguarding your privacy and ensuring the security of your data. This Privacy Policy outlines our practices regarding the collection, use, and protection of your personal information. By using AuthGen, you agree to the terms and conditions of this policy.

                    </Text>

                    <Text color="--black"><b>1. Information Collection and Usage</b></Text>

                    <Text color="--grey">

                        <b>a. User Data:</b> AuthGen operates on a "Zero Data Storage" principle. We do not store any user data, including personally identifiable information or uploaded images, on our servers.

                    </Text>

                    <Text color="--grey">

                        <b>b. Anonymous Usage Data:</b> To enhance our services, AuthGen may collect anonymous usage data, such as website traffic statistics and user interactions. This data is used solely for improving user experience and does not include any personally identifiable information.

                    </Text>

                    <Text color="--black"><b>2. Data Security</b></Text>

                    <Text color="--grey">

                        <b>a. Secure Transmission:</b> All data exchanged between your device and AuthGen is transmitted over secure channels using industry-standard encryption protocols.

                    </Text>

                    <Text color="--grey">

                        <b>b. No Server Storage:</b> As part of our commitment to user privacy, AuthGen does not retain any user data, including uploaded images, on our servers. Once data is processed, it is immediately discarded.

                    </Text>

                    <Text color="--grey">

                        <b>c. Protection Measures:</b> We employ state-of-the-art security measures to safeguard against unauthorized access, disclosure, alteration, or destruction of your information. Our security protocols are regularly updated to meet or exceed industry standards.

                    </Text>

                    <Text color="--black"><b>3. Information Sharing and Third Parties</b></Text>

                    <Text color="--grey">

                        <b>a. Non-Disclosure:</b> AuthGen does not share, sell, or disclose any user information to third parties unless required by law or in compliance with a legal process.

                    </Text>

                    <Text color="--grey">

                        <b>b. Third-Party Services:</b> We may use third-party services for analytics, but these services are selected carefully, and they adhere to stringent privacy and security standards.

                    </Text>

                    <Text color="--black"><b>4. Cookies and Tracking Technologies</b></Text>

                    <Text color="--grey">

                        <b>a. Cookie Usage:</b> AuthGen may use cookies to enhance user experience. These cookies are used for session management and do not store personally identifiable information.

                    </Text>

                    <Text color="--grey">

                        <b>b. Opt-out Options:</b> Users have the option to disable cookies through their browser settings. However, this may affect certain functionalities of AuthGen.

                    </Text>

                    <Text color="--black"><b>5. Policy Updates</b></Text>

                    <Text color="--grey">

                        <b>a. Notification of Changes:</b> In the event of any updates to this Privacy Policy, AuthGen will notify users through the platform or other designated communication channels.

                    </Text>

                    <Text color="--grey">

                        <b>b. Review Periodically:</b> Users are encouraged to review this Privacy Policy periodically to stay informed about how AuthGen is protecting their data.

                    </Text>

                    <Text color="--black"><b>6. Contact Information</b></Text>

                    <Text color="--grey">

                        If you have any questions, concerns, or requests regarding your privacy or this Privacy Policy, please contact us at: <i><b>customer@authgen.com</b></i>

                    </Text>

                    <Text color="--black"><b>Conclusion</b></Text>

                    <Text color="--grey">

                        Your privacy is of utmost importance to us at AuthGen. By using our platform, you entrust us with your information, and we take that responsibility seriously. This Privacy Policy is designed to be transparent about our practices, ensuring that you can use AuthGen with confidence, knowing that your data is secure and your privacy is respected. Thank you for choosing AuthGen.

                    </Text>

                </Text>

                <Link

                    font="--lead"

                    padding="12px 24px 12px 24px"

                    background="#1A1B41"

                    border-radius="8px"

                    href="/client-login"

                    text-decoration-line="initial"

                    color="--light"

                >

                    Go Back

                </Link>

            </Box>

        </Section>

        <RawHtml>

            <style place={"endOfHead"} rawKey={"6575b98c1d712a00232c66dc"}>

                {":root {\n  box-sizing: border-box;\n}\n\n\* {\n  box-sizing: inherit;\n}"}

            </style>

        </RawHtml>

    </Theme>;

});

* + - 1. **./theme/index.js**

export default {

    "defaultWidth": 1280,

    "breakpoints": {

        "sm": [

            {

                "type": "max-width",

                "value": 576

            }

        ],

        "md": [

            {

                "type": "max-width",

                "value": 768

            }

        ],

        "lg": [

            {

                "type": "max-width",

                "value": 992

            }

        ]

    },

    "color": {

        "dark": "#04080C",

        "darkL1": "#0E1317",

        "darkL2": "#191E22",

        "grey": "#6F839F",

        "greyD1": "#637897",

        "greyD2": "#586D8E",

        "light": "#F7FBFF",

        "lightD1": "#EDF2F6",

        "lightD2": "#E4E8EC",

        "green": "#00875A",

        "primary": "#0077CC",

        "secondary": "#FFC400",

        "orange": "#FF7C22",

        "red": "#DE350B",

        "purple": "#FD6DF9",

        "indigo": "#9B6CFC"

    },

    "fontFamily": {

        "sans": "-apple-system, system-ui, 'Segoe UI', 'Roboto', 'Oxygen', 'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue', sans-serif",

        "sansHeavy": "Impact, Haettenschweiler, 'Franklin Gothic Bold', Charcoal, 'Helvetica Inserat', 'Bitstream Vera Sans Bold', 'Arial Black', sans-serif",

        "sansHelvetica": "Frutiger, 'Frutiger Linotype', Univers, Calibri, 'Gill Sans', 'Gill Sans MT', 'Myriad Pro', Myriad, 'DejaVu Sans Condensed', 'Liberation Sans', 'Nimbus Sans L', Tahoma, Geneva, 'Helvetica Neue', Helvetica, Arial, sans-serif",

        "sansVerdana": "Corbel, 'Lucida Grande', 'Lucida Sans Unicode', 'DejaVu Sans', 'Bitstream Vera Sans', 'Liberation Sans', Verdana, 'Verdana Ref', sans-serif",

        "sansTrebuchet": "'Segoe UI', Candara, 'Bitstream Vera Sans', 'DejaVu Sans', 'Trebuchet MS', Verdana, 'Verdana Ref', sans-serif",

        "mono": "Consolas, 'Andale Mono WT', 'Andale Mono', 'Lucida Console', 'Lucida Sans Typewriter', 'DejaVu Sans Mono', 'Bitstream Vera Sans Mono', 'Liberation Mono', 'Nimbus Mono L', Monaco, 'Courier New', Courier, monospace",

        "serifTimes": "Cambria, 'Hoefler Text', Utopia, 'Liberation Serif', 'Nimbus Roman No9 L Regular', Times, 'Times New Roman', serif",

        "serifGeorgia": "Constantia, 'Lucida Bright', Lucidabright, 'Lucida Serif', Lucida, 'DejaVu Serif', 'Bitstream Vera Serif', 'Liberation Serif', Georgia, serif",

        "serifGaramond": "'Palatino Linotype', Palatino, Palladio, 'URW Palladio L', 'Book Antiqua', Baskerville, 'Bookman Old Style', 'Bitstream Charter', 'Nimbus Roman No9 L', Garamond, 'Apple Garamond', 'ITC Garamond Narrow', 'New Century Schoolbook', 'Century Schoolbook', 'Century Schoolbook L', Georgia, serif"

    },

    "font": {

        "headline1": "normal 900 72px/1.2 --fontFamily-sans",

        "headline2": "normal 900 42px/1.2 --fontFamily-sans",

        "headline3": "normal 500 28px/1.2 --fontFamily-sans",

        "base": "normal 300 16px/1.5 --fontFamily-sans",

        "lead": "normal 300 20px/1.5 --fontFamily-sans"

    },

    "boxShadow": {

        "m": "0 4px 5px -1px rgba(0, 0, 0, 0.1)",

        "l": "0 10px 15px -3px rgba(0, 0, 0, 0.1)",

        "xl": "0 20px 25px -5px rgba(0, 0, 0, 0.1)",

        "xxl": "0 25px 50px -12px rgba(0, 0, 0, 0.25)"

    },

    "background": {

        "primaryGradient": "linear-gradient(180deg, --color-primary, transparent) no-repeat 0 0",

        "secondaryGradient": "linear-gradient(180deg, --color-secondary, transparent) no-repeat 0 0"

    },

    "transform": {

        "rotate90": "rotate(90deg)",

        "rotate180": "rotate(180deg)",

        "flipX": "scaleX(-1)",

        "flipY": "scaleY(-1)"

    },

    "transition": {

        "opacityOut": "opacity --transitionDuration-normal --transitionTimingFunction-easeOut",

        "transformOut": "transform --transitionDuration-normal --transitionTimingFunction-easeOut",

        "transformInOut": "transform --transitionDuration-normal --transitionTimingFunction-easeInOut"

    },

    "transitionTimingFunction": {

        "easeIn": "cubic-bezier(0.4, 0, 1, 1)",

        "easeOut": "cubic-bezier(0, 0, 0.2, 1)",

        "easeInOut": "cubic-bezier(0.4, 0, 0.2, 1)",

        "sharp": "cubic-bezier(0.4, 0, 0.6, 1)"

    },

    "transitionDuration": {

        "fastest": "0.1s",

        "fast": "0.2s",

        "normal": "0.3s"

    },

    "filter": {

        "grayscale": "grayscale(100%)",

        "invert": "invert(100%)",

        "blur": "blur(10px)"

    },

    "animation": {},

    "keyframes": {

        "fadeIn": [

            {

                "key": "from",

                "props": [

                    {

                        "opacity": 0

                    }

                ]

            },

            {

                "key": "to",

                "props": [

                    {

                        "opacity": 1

                    }

                ]

            }

        ],

        "fadeOut": [

            {

                "key": "from",

                "props": [

                    {

                        "opacity": 1

                    }

                ]

            },

            {

                "key": "to",

                "props": [

                    {

                        "opacity": 0

                    }

                ]

            }

        ]

    },

    "components": {

        "section": {

            "maxWidth": {

                "default": "1280px"

            },

            "minWidth": {

                "default": "300px"

            },

            "width": {

                "default": "90%"

            }

        },

        "stack": {

            "gap": {

                "default": "32px",

                "small": "16px"

            }

        }

    },

    "version": 1

}

* + - 1. **./global-pages-styles.js**

import React from "react";

import { transformVar } from "@quarkly/atomize";

import { createGlobalStyle } from "styled-components";

const pageStyles = {

    "404": {},

    "index": {},

    "client-login": {},

    "client-encode-image": {},

    "customer-decode-image": {},

    "privacy-policy" : {}

};

const PageStyles = createGlobalStyle`

    body {

        ${({ styles }) =>

            Object.entries(styles || {}).map(

                ([prop, value]) => `${prop}: ${transformVar(prop, value)};`

            )}

    }

`;

export const GlobalQuarklyPageStyles = ({ pageUrl }) => <PageStyles styles={pageStyles[pageUrl]} />

* + - 1. **./qapi.js**

export default {

    "pages": {

        "root": {

            "id": "root",

            "pageUrl": "root",

            "name": "root",

            "children": [

                "1",

                "2",

                "3",

                "4",

                "5"

            ]

        },

        "1": {

            "id": "1",

            "name": "404",

            "pageUrl": "404"

        },

        "2": {

            "id": "2",

            "name": "index",

            "pageUrl": "index"

        },

        "3": {

            "id": "3",

            "pageUrl": "client-login",

            "name": "Login"

        },

        "4": {

            "id": "4",

            "pageUrl": "client-encode-image",

            "name": "Encrypt"

        },

        "5": {

            "id": "5",

            "pageUrl": "customer-decode-image",

            "name": "Authenticate"

        }

    },

    "mode": "production",

    "projectType": "create-react-app",

    "site": {

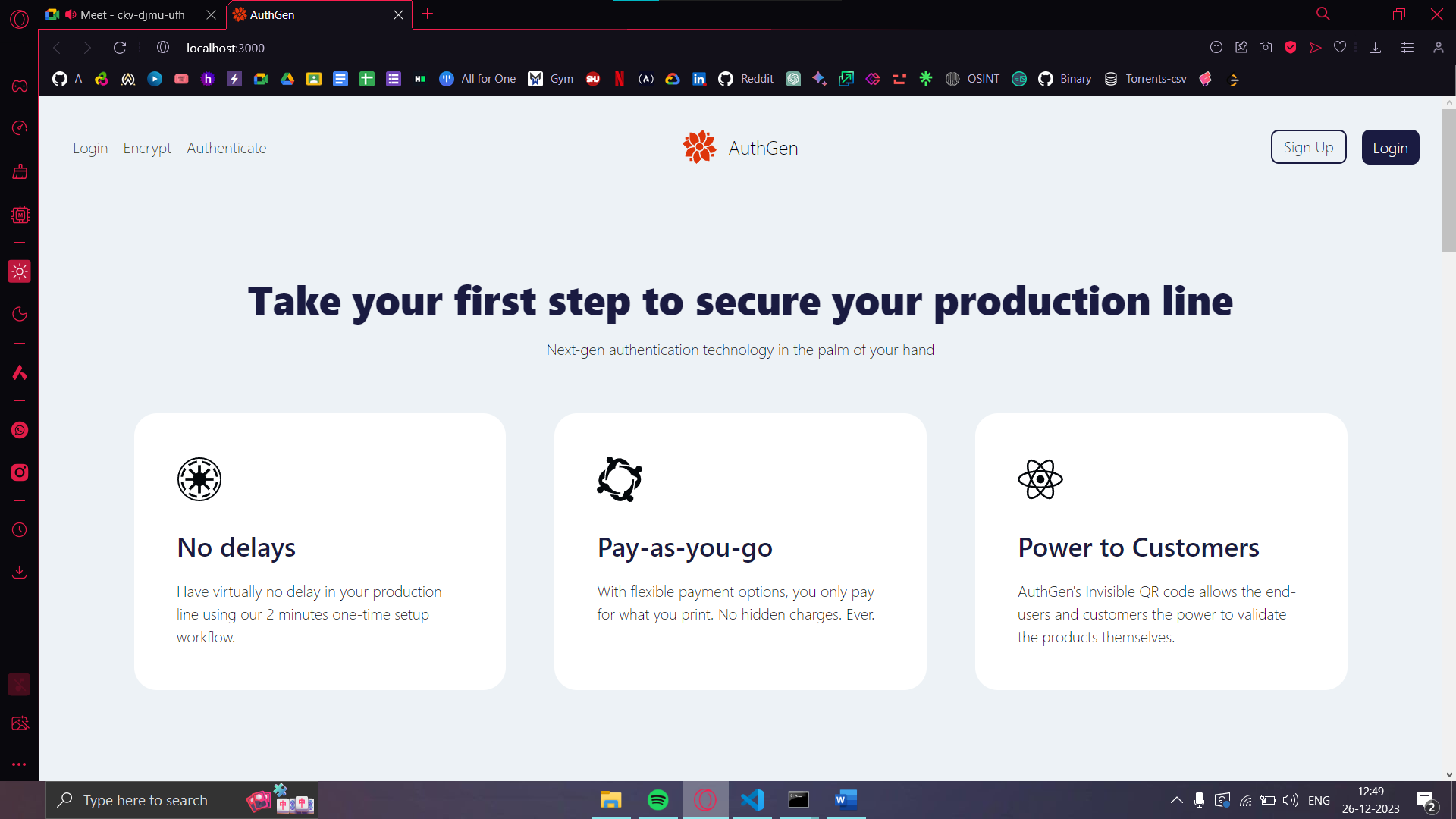
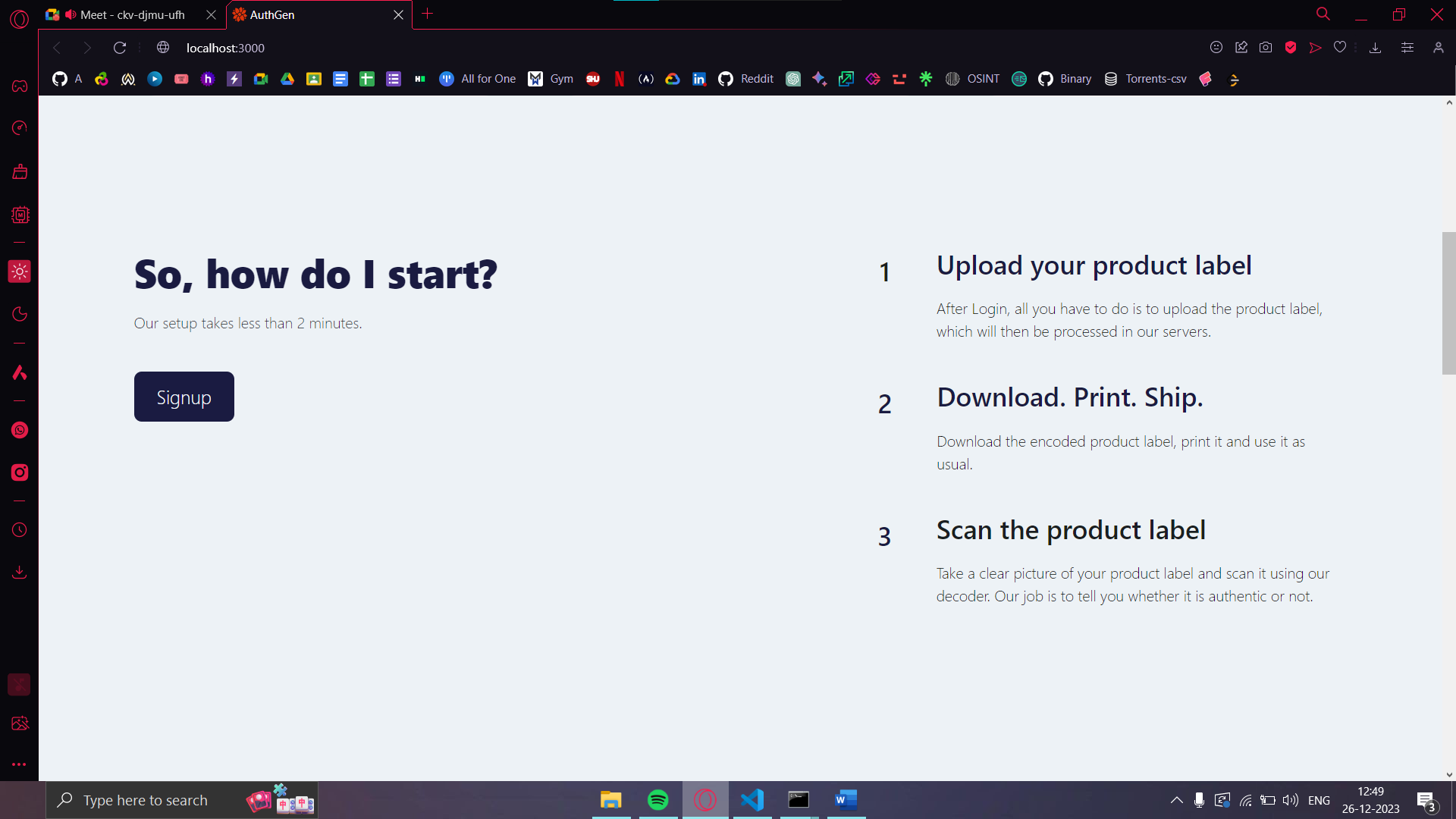
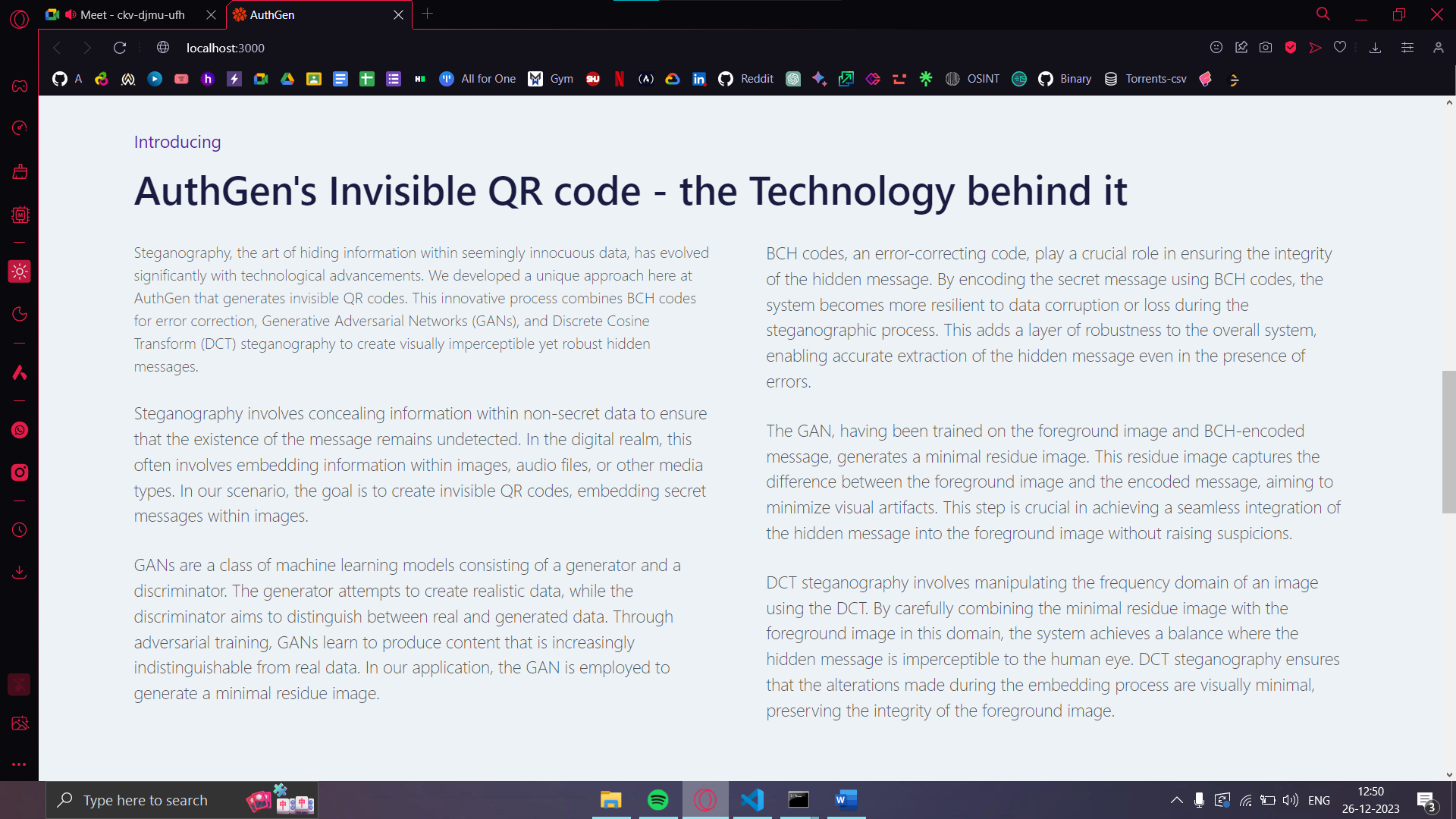
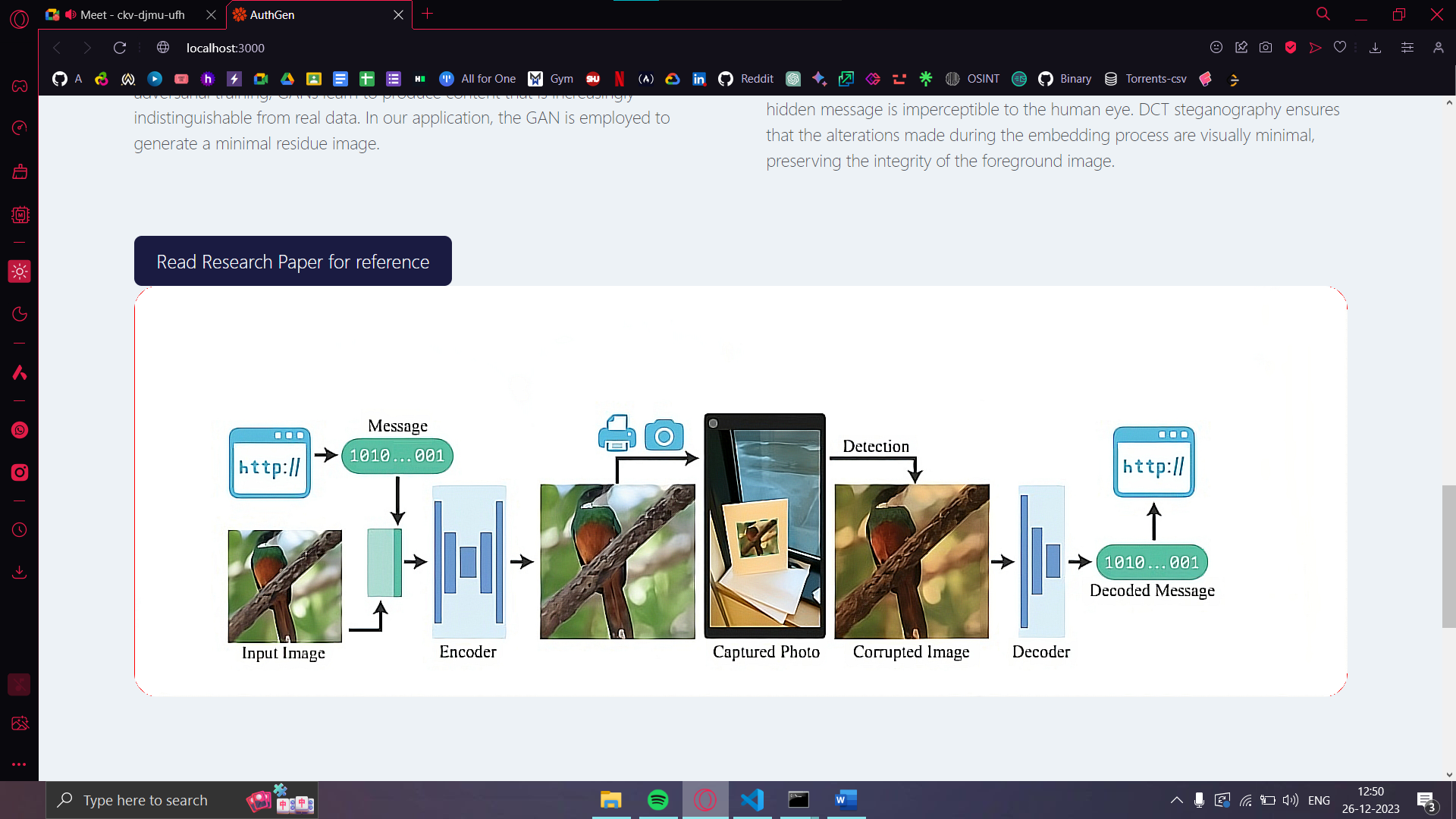
        "styles": {},

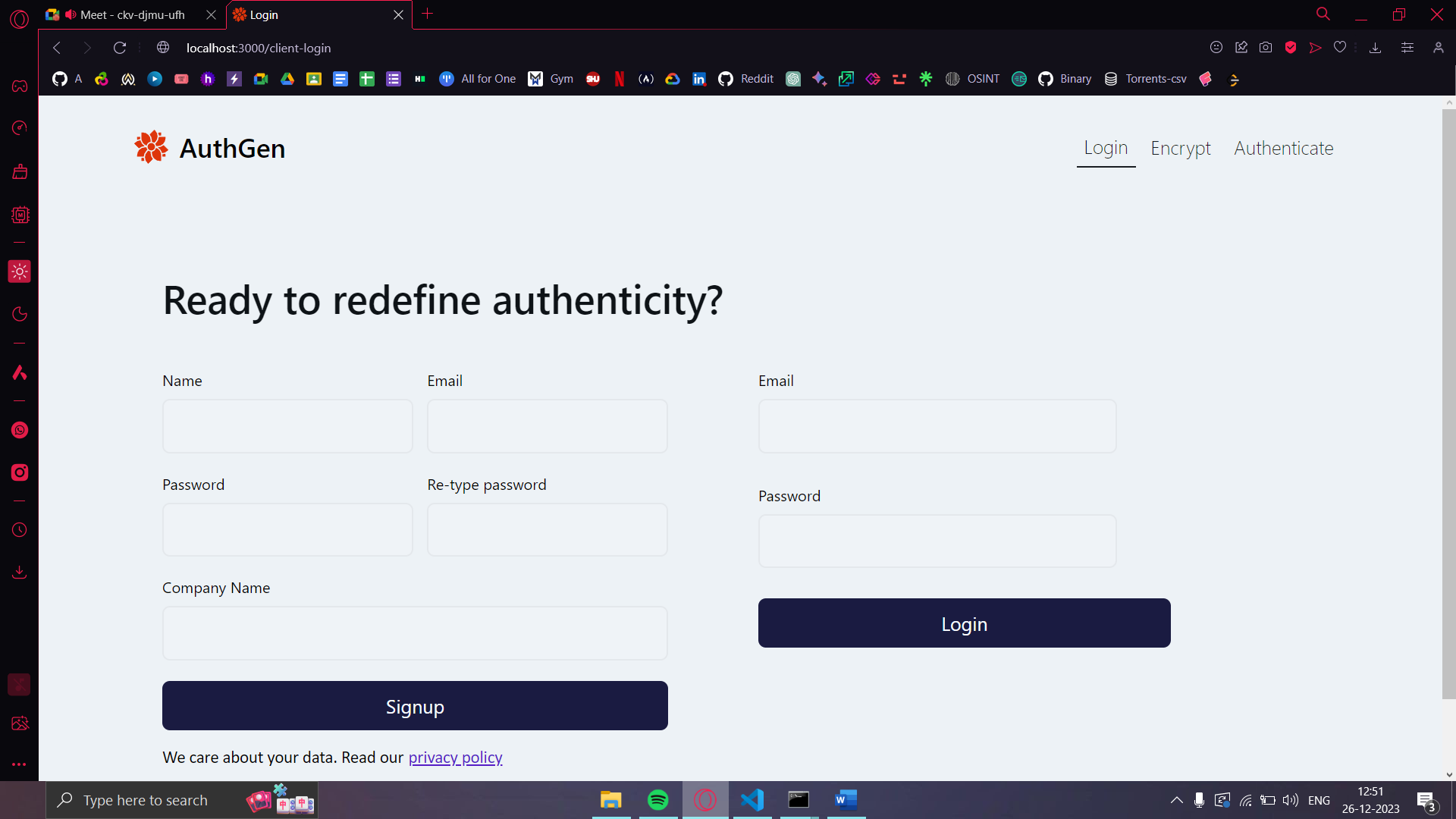
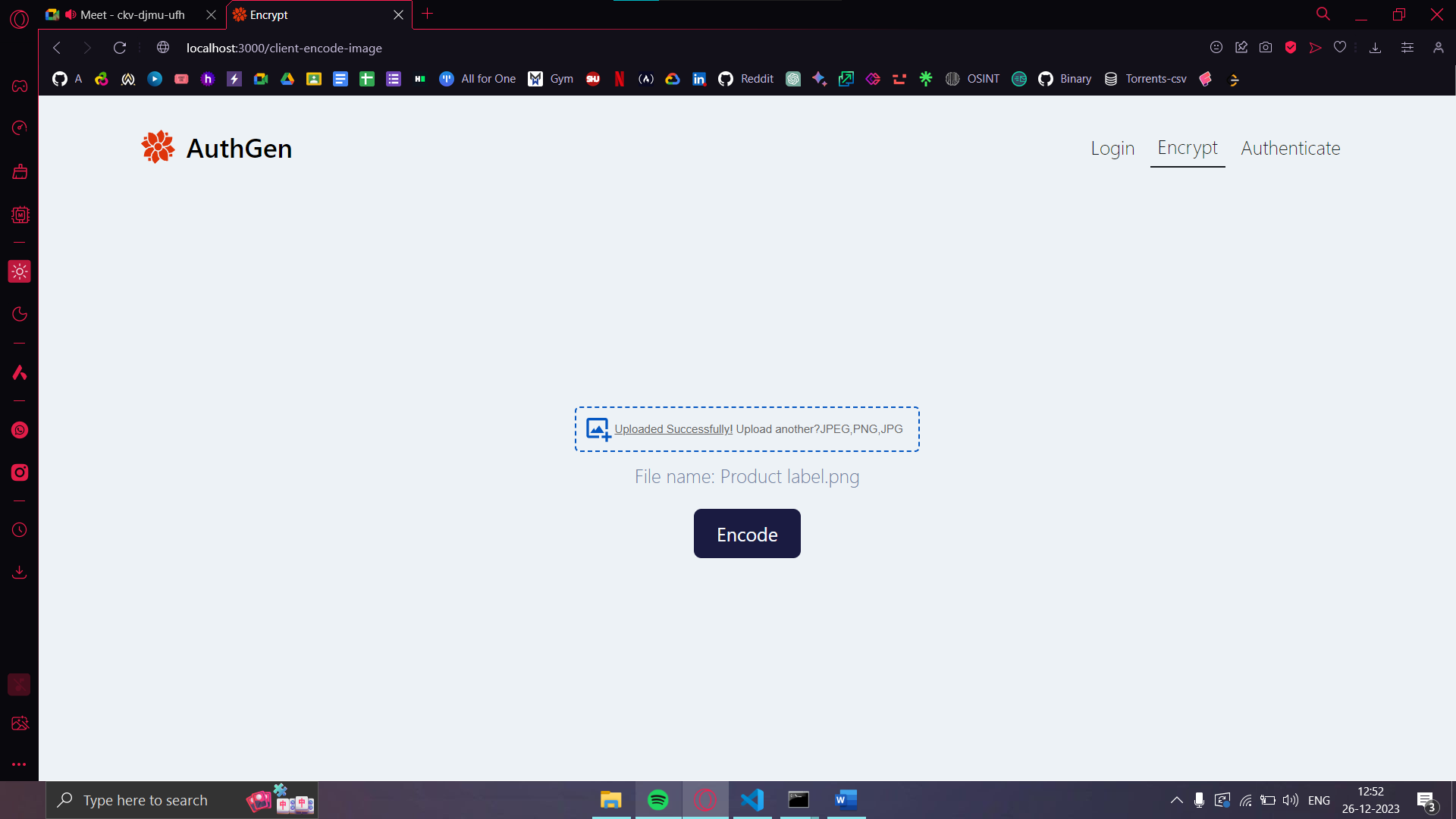
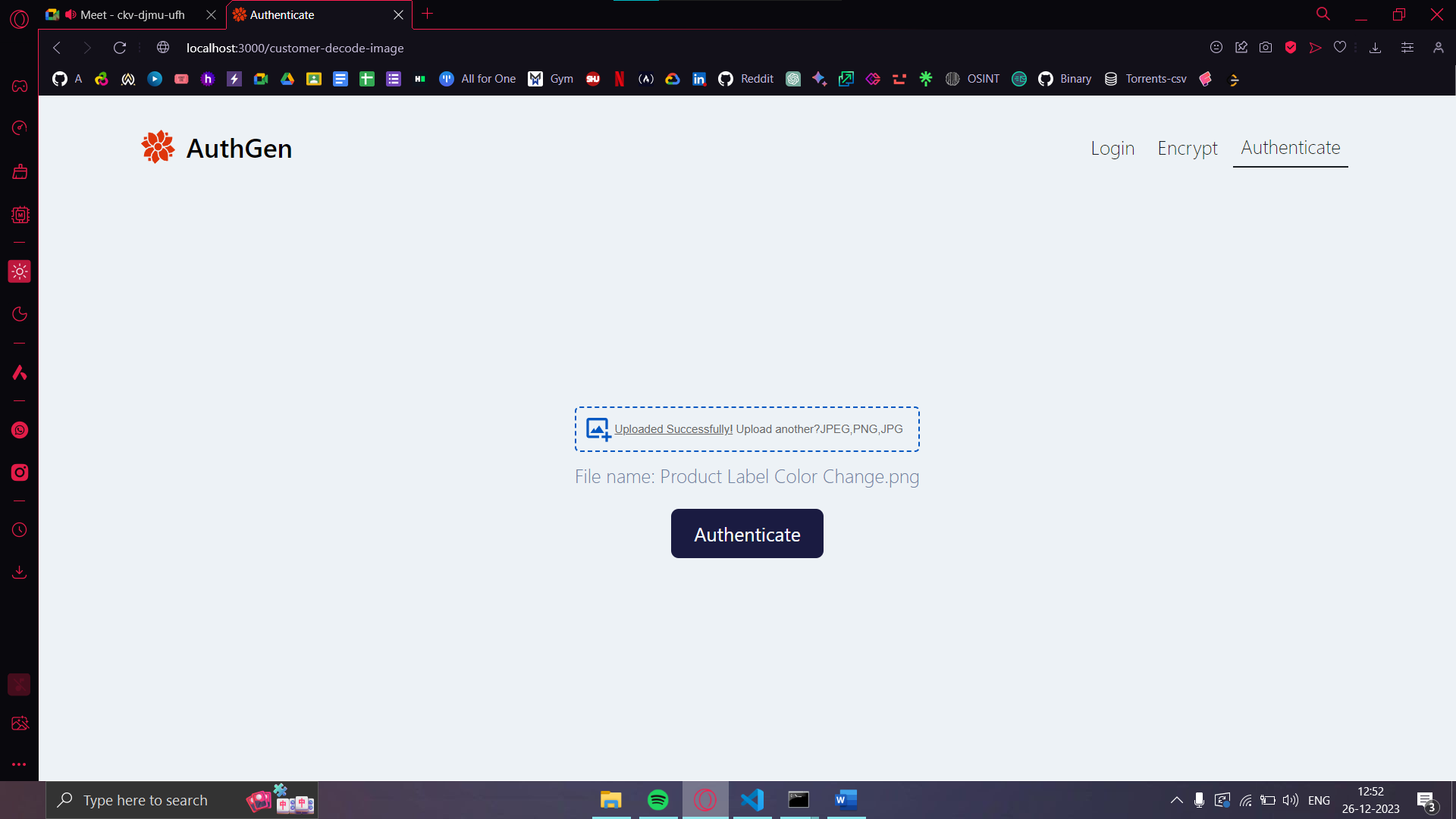
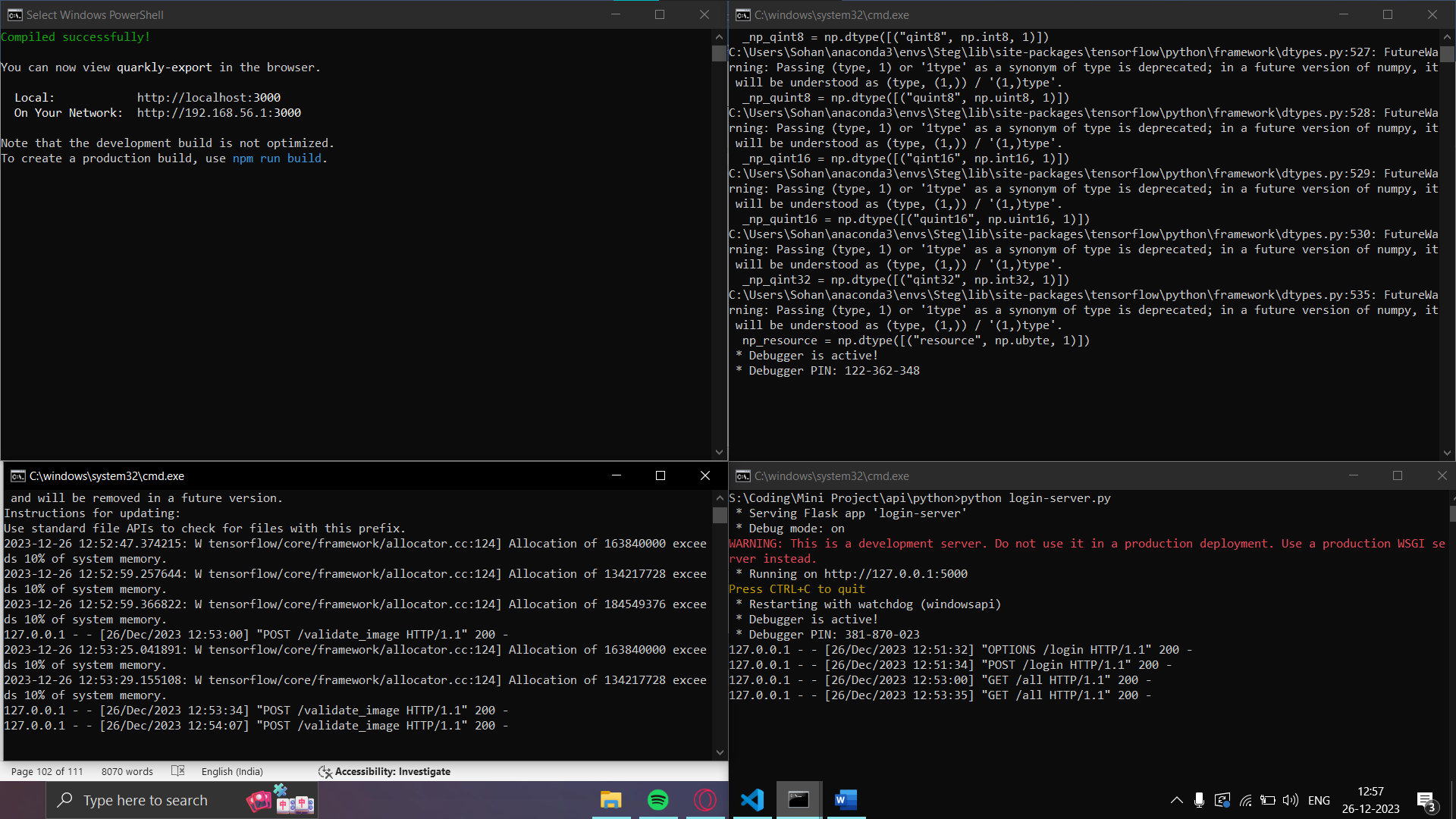
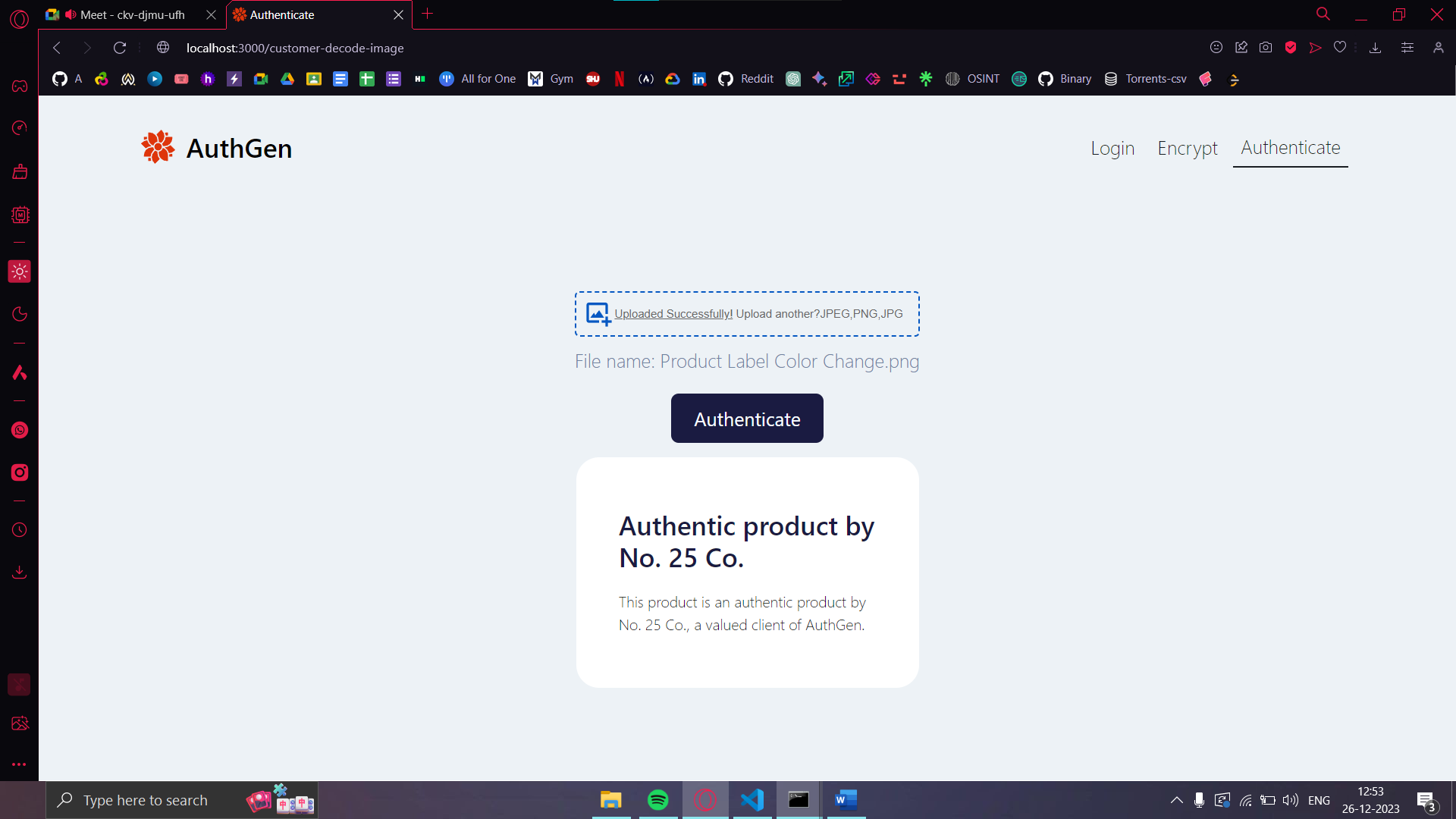
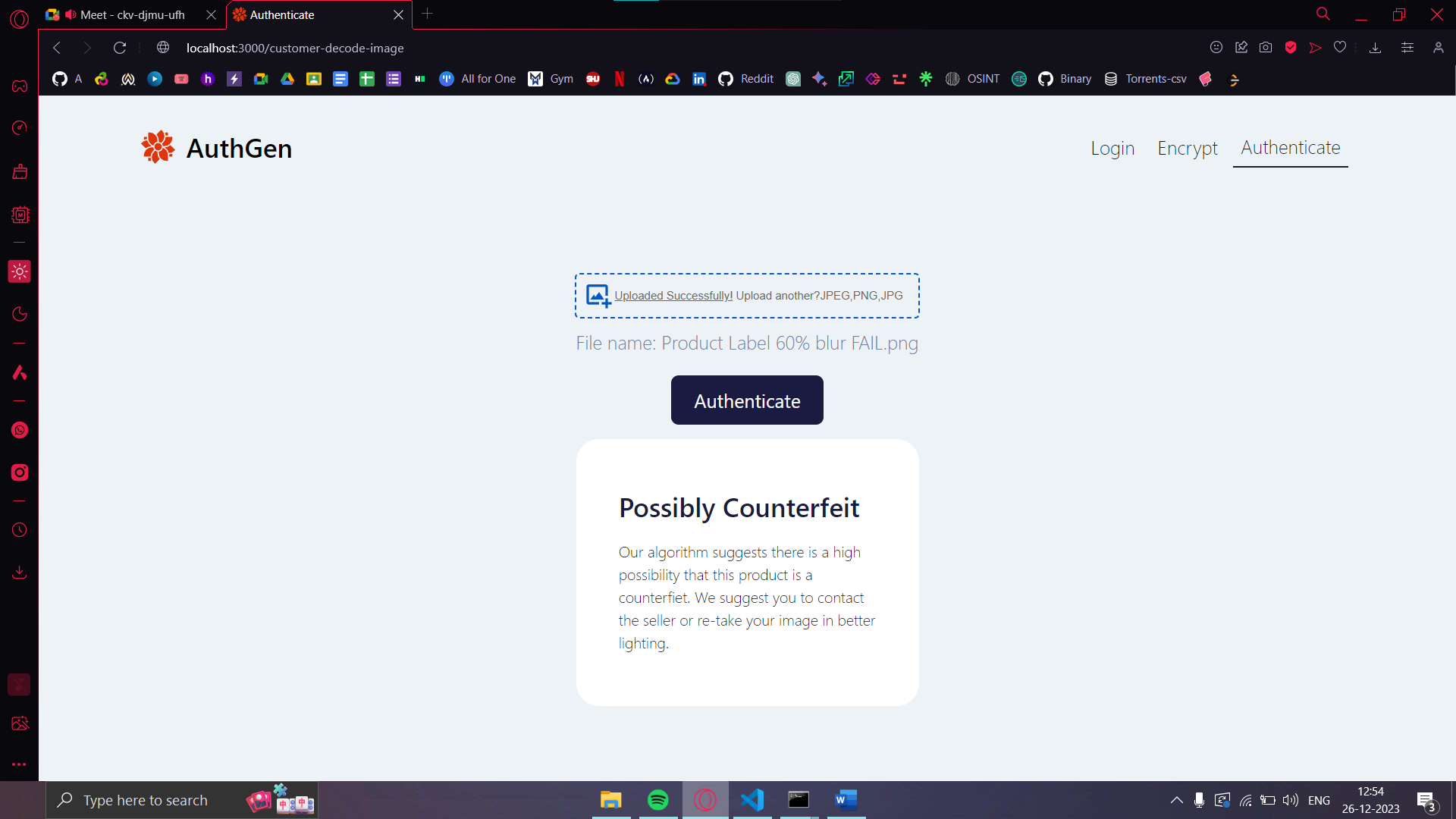
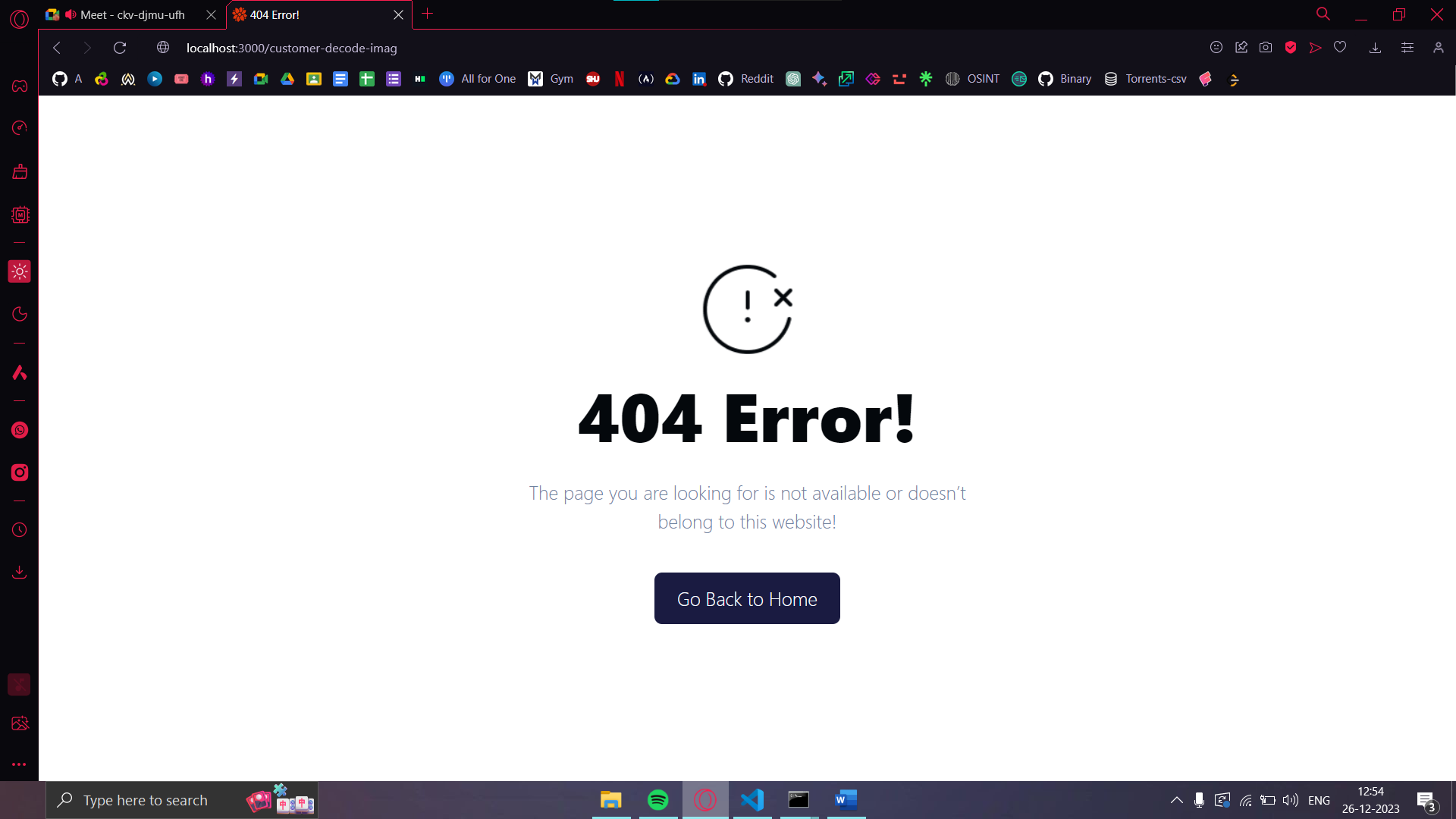
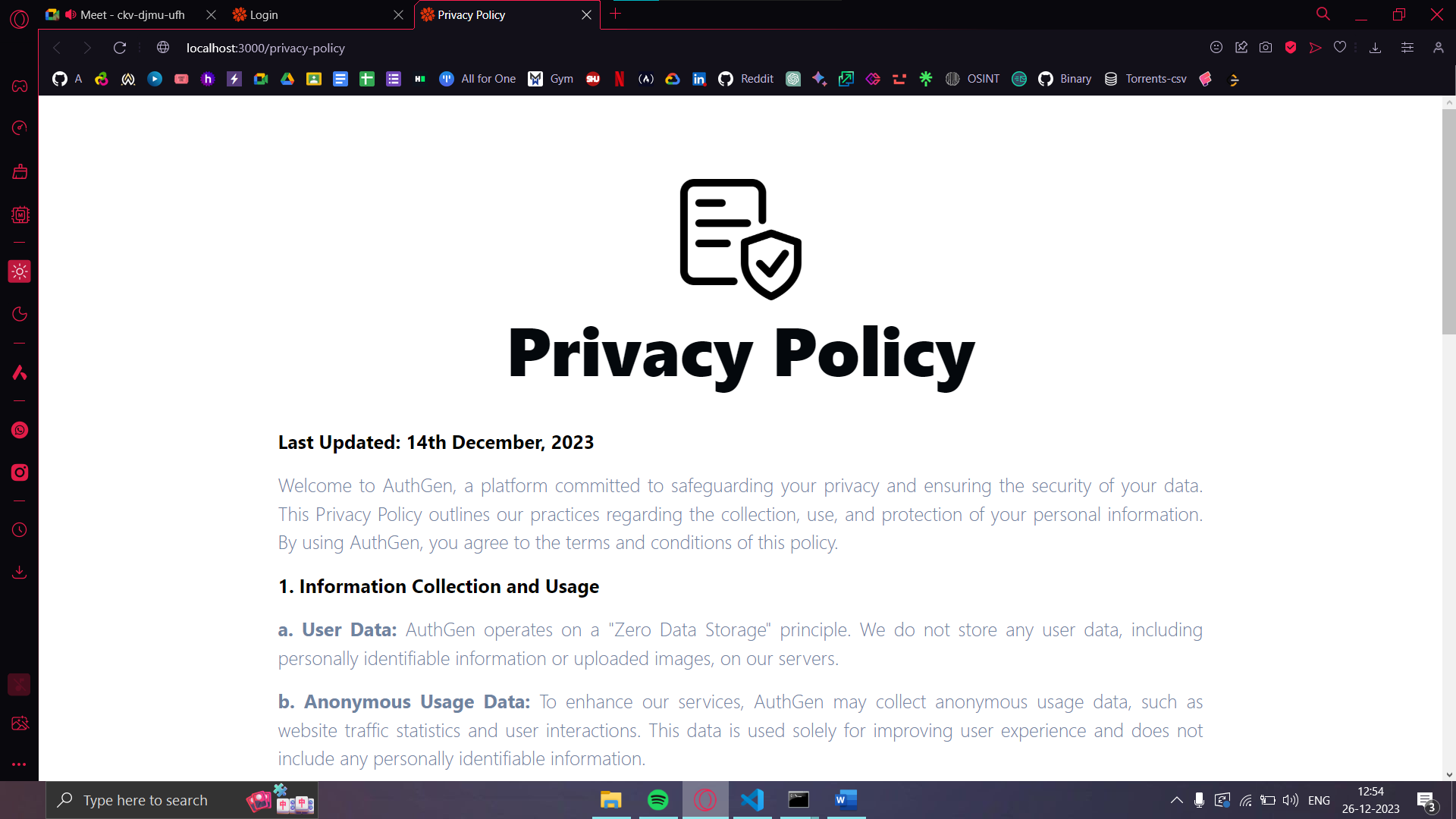
        "seo": {}

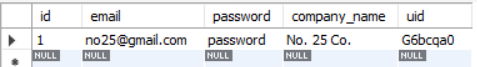
    }

}

**6. RESULTS / SCREENSHOTS:**

*****Fig 6.1* Home Page**** *Fig 6.2* “How do I start?” section **** *Fig 6.3* “Technology behind it” section **** *Fig 6.4* “Process Flowchart” section

**** *Fig 6.6* Login/Signup page**** *Fig 6.7* Encrypt/Encode image page **** *Fig 6.8* Authenticate Page **** *Fig 6.9* Parallel server setup **** *Fig 6.10* Authentication Confirmation alert**** *Fig 6.11* Authentication Failed alert **** *Fig 6.12* Error 404 Page **** *Fig 6.13* Privacy Policy Page

****

*Fig 6.14* Error 404 Page



*Fig 6.15* Original Product Image



*Fig 6.15* Encoded Product Label



*Fig 6.16* Side by Side Comparison



*Fig 6.17* Encoded Image : 5% Blur PASS



*Fig 6.18* Encoded Image : 15% Blur PASS



*Fig 6.19* Encoded Image : 45% Blur PASS



*Fig 6.20* Encoded Image : 15% Blur 5% Duotone PASS



*Fig 6.21* Encoded Image : 30% Blur 15% Duotone PASS



*Fig 6.22* Encoded Image : 30% Blur 30% Duotone PASS



*Fig 6.23* Encoded Image : 30% Blur Irregular Focus PASS



*Fig 6.24* Encoded Image : Complete Colour Shift PASS



*Fig 6.25* Encoded Image : 60% Blur FAIL

**7. CONCLUSION & FUTURE SCOPE:**

This is a working proof of concepts of the power of print-scan solutions using GAN Steganography and BCH in succession to solve a real-life problem of counterfeiting.

The current GAN Model is trained to process only 400x400 px images. To increase robustness, the model should be trained on larger images, that might take 28 hours and more. The current implementation can easily be modified to replace dynamic QR codes and can be further improved to optimize time to encode and decode to become more than a proof of concept.

**8. REFERENCES:**

[1] M. Walters and S. S. Roy, "Constant-time BCH Error-Correcting Code," School of Computer Science, University of Birmingham, United Kingdom.

[2] D. Neeta, K. Snehal, and D. Jacobs, "Implementation of LSB Steganography and Its Evaluation for Various Bits," in 2006 1st International Conference on Digital Information Management, Bangalore, India, 2007, pp. 173-178, DOI: 10.1109/ICDIM.2007.369349.

[3] A. A. M. Khalaf, O. Fouad, A. Hussein, H. Hamed, H. Kelash, and H. Ali, "Hiding data in images using DCT steganography techniques with compression algorithms," Telkomnika, vol. 17, no. 3, 2019, DOI: 10.12928/telkomnika.v17i3.

[4] K. A. Zhang, A. Cuesta-Infante, L. Xu, and K. Veeramachaneni, "SteganoGAN: High Capacity Image Steganography with GANs," arXiv:1901.03892 [cs.CV], Jan. 2019. DOI: 10.48550/arXiv.1901.03892.

[5] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Networks," arXiv:1406.2661 [stat.ML], Jun. 2014. DOI: 10.48550/arXiv.1406.2661.

[6] C. Ding, X. Du, and Z. Zhou, "The Bose and Minimum Distance of a Class of BCH Codes," IEEE Transactions on Information Theory, vol. 61, no. 5, pp. 2351, May 2015.

[7] E. Walia, P. Jain, and Naveed, "An analysis of LSB & DCT based steganography," Global Journal of Computer Science and Technology, 2010.

[8] J. Fridrich, M. Goljan, and R. Du, "Detecting LSB steganography in color and gray-scale images," IEEE Multimedia, vol. 8, pp. 22-28, 2001. DOI: 10.1109/93.959097.