AI Image Rebuilder - generate image variants using Vision and DALL-E 3

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>AI Image Rebuilder</title>
    <style>
* {
  box-sizing: border-box;
}
body {
  font-family: Arial, sans-serif;
  margin: 0;
  padding: 0;
  background-color: #f0f8ff;
  color: white;
  background: url("img_rebuilder.jpg"); /* or other suitable background
image */
  background-repeat: no-repeat;
  background-size: cover;
  background-position: center;
}

header {
  background-color: transparent;
  color: white;
  text-align: center;
  padding: 20px;
}

.container {
  display: flex;
  justify-content: space-around;
  margin-bottom: 20px;
}
.image-box {
  width: 28rem;
  height: 28rem;
  margin: 1rem 2rem;
  border: 1px solid white;
  border-radius: 24px;
  padding: 8px;
  box-shadow: 0 4px 10px white;
  text-align: center;
  background: transparent;
  backdrop-filter: blur(2px);
}
```

```
.image-box img {
  width: 90%;
  border-radius: 24px;
  height: auto;
  margin: 0 auto;
}
#text-box {
  width: 60%;
  border: 1px solid white;
  border-radius: 12px;
  box-shadow: 0 4px 10px white;
  padding: 0.5rem;
  margin: 1rem auto;
  color: white;
  backdrop-filter: blur(2px);
}


h2,
h1,
h3 {
  color: white;
  text-shadow: 2px 4px 10px black;
}
h1 {
  font-size: 2.4rem;
}

.preloader {
  width: 48px;
}

button {
  background-color: black;
  color: white;
  border: none;
  padding: 10px 15px;
  border-radius: 5px;
  cursor: pointer;
  font-size: 1.1rem;
  transition: all 0.2s linear;
}

button:hover {
  background-color: #6a0dad;
}
#vfeedback {
  color: black;
}

#voperations, #commands {
```

```
  width: 28rem;
  padding: 0.5rem;
  margin: 0 auto;
  text-align: center;
}

    </style>
</head>
<body>

<header>
  <h1>AI Image Rebuilder</h1>
</header>

<div class="container">
  <div class="image-box" id="original-image">
<img id="uploaded-image" src="" alt="Uploaded Image" style="max-
width: 90%; display: none;">
  </div>
  <div class="image-box" id="generated-image">
  </div>
</div>
<div id="voperations">
  <form enctype="multipart/form-data" method="POST">
     <input class="fixedwidth" type="file" accept=".jpg,.png" name="file"
id="vfile-input">
     <div id="vfeedback"></div>
     <button id="vuploadBtn" type="submit"> Upload the image</button>
     <button type="button" onclick="getChatGPTResponseIMG();">
Analyze the image</button>
  </form>
</div>
<div id="text-box">
  <b>Your Image analysis will appear here...</b>
  <p id="image-analysis-output"></p>
</div>
<div id="commands">
<button type="button" onclick="generateImage();"> Rebuild the
image</button>
</div>

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
<script>
$(document).ready(function () {
$('#vuploadBtn').click(function (e) {
e.preventDefault();
$.ajax({
url: 'vupload.php', // make sure you have the upload script in the folder
type: 'POST',
data: new FormData($('form')[0]),
processData: false,
contentType: false,
```

```
success: function (response) {
const data = JSON.parse(response); // Parsing the JSON response
if (data.status === "success") {
$('#vfeedback').html("Your file was uploaded.");

var fileInput = document.getElementById('vfile-input');
var file = fileInput.files[0];

if (file) {
var originalimageUrl = URL.createObjectURL(file);
$('#uploaded-image').attr('src', originalimageUrl).show();
}

$('#voperations button[type="button"]').attr('onclick',
`submitImage("${data.url}");`);
} else {
$('#vfeedback').html(data); // In caso di errori, mostra l'errore
}
}
});
});
});
</script>

<script>
let chatMemory = [];
chatMemory = createMemory([{
    role: "system",
    content: "You are an AI image analyzer and rebuilder."
}]);

function createMemory(messages) {
    const memory = [];
    for (const msg of messages) {
        memory.push({
            role: msg.role,
            content: msg.content
        });
    }
    return memory;
}

// VISION
async function submitImage(imgUrl) {
  return await getChatGPTResponseIMG("http://domain/uploadfolder" +
imgUrl, chatMemory); //customize according to your hosting
}

async function getChatGPTResponseIMG(fileUrl, chatMemory) {
const apikey = localStorage.getItem("openaikey"); // you must have a
panel that sets the API Key in Localstorage
```

```
// const apikey = document.getElementById("apikey").value;  use this if you put a text input with ID=apikey in the DOM
console.log(apikey);

if (apikey === "") {
alert("No OpenAI API Key found.");
return;
}

const chatContainer = document.getElementById("text-box");
const typingIndicator = document.createElement("p");
typingIndicator.id = "typing-indicator";
typingIndicator.innerHTML = '<img src="preloader.gif" class="preloader" alt="Loading...">'; //usea a preloader image of your choice
chatContainer.appendChild(typingIndicator);

try {
response = await fetch("https://api.openai.com/v1/chat/completions", {
method: "POST",
headers: {
"Content-Type": "application/json",
Authorization: "Bearer " + apikey
},
body: JSON.stringify({
model: "gpt-4o-mini",
messages: [
...chatMemory.map((item) => ({ ...item })),
{
role: "user",
content: [
{
type: "text",
text: "Provide a detailed description of the image content. Don't thank, greet or propose the user to ask further questions, only provide the image description."
},
{
type: "image_url",
image_url: {
url: fileUrl
}
}
]
}
],
max_tokens: 450
})
});

if (!response.ok) {
throw new Error("Error while requesting to the API.");
}
```

```
const data = await response.json();
if (!data.choices || !data.choices.length || !data.choices[0].message
|| !data.choices[0].message.content) {
throw new Error("Invalid API response.");
}

var chatGPTResponse = data.choices[0].message.content;
chatGPTResponse = chatGPTResponse.replace(/\*\*(.*?)\*\*/g, "$1");

chatContainer.removeChild(typingIndicator);
const responsediv = document.getElementById("image-analysis-output");
responsediv.innerText= chatGPTResponse;

chatMemory.push({ role: "user", content: fileUrl });
chatMemory.push({ role: "assistant", content: chatGPTResponse });

return chatMemory;
} catch (error) {
console.error(error);
alert(
"An error occurred during the request. Check your OpenAI account or try
later."
);
}
}
// END VISION

// DALL-E
async function generateImage() {
const apikey = localStorage.getItem("openaikey"); // you must have a
panel that sets the API Key in Localstorage
// const apikey = document.getElementById("apikey").value;  use this if
you put a text input with ID=apikey in the DOM
const prompt = document.getElementById("image-analysis-
output").textContent;
const chatContainer = document.getElementById("generated-image");

  try {
    const response = await fetch(
      "https://api.openai.com/v1/images/generations",
      {
        method: "POST",
        headers: {
          "Content-Type": "application/json",
          Authorization: "Bearer " + apikey
        },
        body: JSON.stringify({
          model: "dall-e-3",
          prompt: prompt,
          n: 1,
          size: "1024x1024"
```

```
      })
    }
  );

  const data = await response.json();
  const imageUrl = data.data[0].url;
  const responseDiv = document.createElement("div");
  const image = document.createElement("img");
  image.src = imageUrl;
  image.alt = "Generated Image";
  responseDiv.appendChild(image);
  const downloadLink = document.createElement("a");
  const spacer = document.createElement("br");
  downloadLink.href = imageUrl;
  downloadLink.target = "_blank";
  downloadLink.textContent = "Download Image";
  responseDiv.appendChild(spacer);
  responseDiv.appendChild(downloadLink);
  chatContainer.appendChild(responseDiv);
 } catch (error) {
  console.error(error);
  alert(
   "An error occurred during the request. Check your OpenAI account or
try later."
  );
 }
}
//END DALL-E
</script>

</body>
</html>
```