

supabase -

목차

| | |
|-------------------------|----|
| >3/26일 | 2 |
| Supabase Storage란 무엇인가 | 2 |
| Supabase Storage 사용 흐름 | 2 |
| Supabase Storage 코드 예제 | 3 |
| Supabase Storage의 특징 요약 | 4 |
| >storage에서 버킷 만들기 | 5 |
| >삭제 추가 | 18 |
| >확장자에 따른 파일 이미지 추가가 | 26 |
| >다운로드 추가가 | 35 |
| >이미지도 다운로드 하기 | 51 |
| >복잡한 파일 이름 변경하기 | 67 |

Supabase Storage란 무엇인가

Supabase Storage는 클라우드 기반 파일 저장소 서비스로, 이미지, 문서, 비디오 등 다양한 파일을 손쉽게 업로드하고 관리할 수 있도록 지원한다.

AWS S3와 유사한 방식으로 동작하지만, Supabase 플랫폼 내에서 통합 관리되기 때문에 별도의 복잡한 설정 없이 쉽게 사용할 수 있다는 것이 큰 장점이다.

Storage에서는 파일을 버킷(bucket) 단위로 그룹화하여 저장한다.

버킷은 파일을 분류하고 권한을 관리하는 기본 단위로, 하나의 버킷에는 여러 파일과 폴더를 계층적으로 저장할 수 있다.

또한 Supabase Storage는 자체 데이터베이스(PostgreSQL)와 긴밀히 연동된다.

이를 통해 파일의 업로드, 다운로드, 삭제 이력을 데이터베이스처럼 관리할 수 있으며, 파일 메타데이터를 다른 데이터와 함께 질의할 수도 있다.

Supabase는 Storage에 접근할 수 있는 API를 제공한다. 개발자는 이 API를 이용하여 파일을 업로드하거나 다운로드하고, 파일의 URL을 가져오거나 삭제할 수 있다.

특히 별도의 서버 구축 없이도 프론트엔드 애플리케이션(예: Flutter, React 등)에서 직접 Storage를 사용할 수 있다는 점은 개발 생산성을 높이는 데 크게 기여한다.

Supabase Storage 사용 흐름

Supabase Storage를 사용하는 기본적인 과정은 다음과 같다.

1. 버킷 생성

파일을 저장할 버킷을 만든다. 버킷 이름은 고유해야 하며, 파일 접근 권한(공개/비공개)을 설정할 수 있다.

2. 파일 업로드

생성한 버킷 안에 파일을 업로드한다. 파일은 경로(path)를 지정하여 저장할 수 있으며, 폴더처럼 경로를 계층적으로 구성할 수도 있다.

3. 파일 URL 가져오기

업로드한 파일의 접근 경로(URL)를 얻어 클라이언트에서 사용할 수 있다. 파일이 공개 버킷에 있다면 누구나 접근할 수 있고, 비공개인 경우 인증된 사용자만 접근할 수 있다.

4. 권한 설정

파일이나 버킷 단위로 접근 권한을 설정할 수 있다. 기본적으로 비공개 버킷을 만들고, 필요한 파일만 공개 URL로 제공하는 방식이 권장된다.

Supabase Storage 코드 예제

아래는 JavaScript를 이용하여 Supabase Storage에 파일을 업로드하고, 해당 파일의 URL을 가져오는 예제이다.

javascript

복사편집

// 파일 업로드

```
const { data, error } = await supabase.storage
  .from('avatars')          // 버킷 이름
  .upload('user1234/profile.png', file); // 저장할 경로 및 파일 객체
```

업로드가 성공한 후, 해당 파일의 공개 URL을 가져오는 방법은 다음과 같다.

javascript

복사편집

// 파일 공개 URL 얻기

```
const { publicURL, error } = supabase
  .storage
```

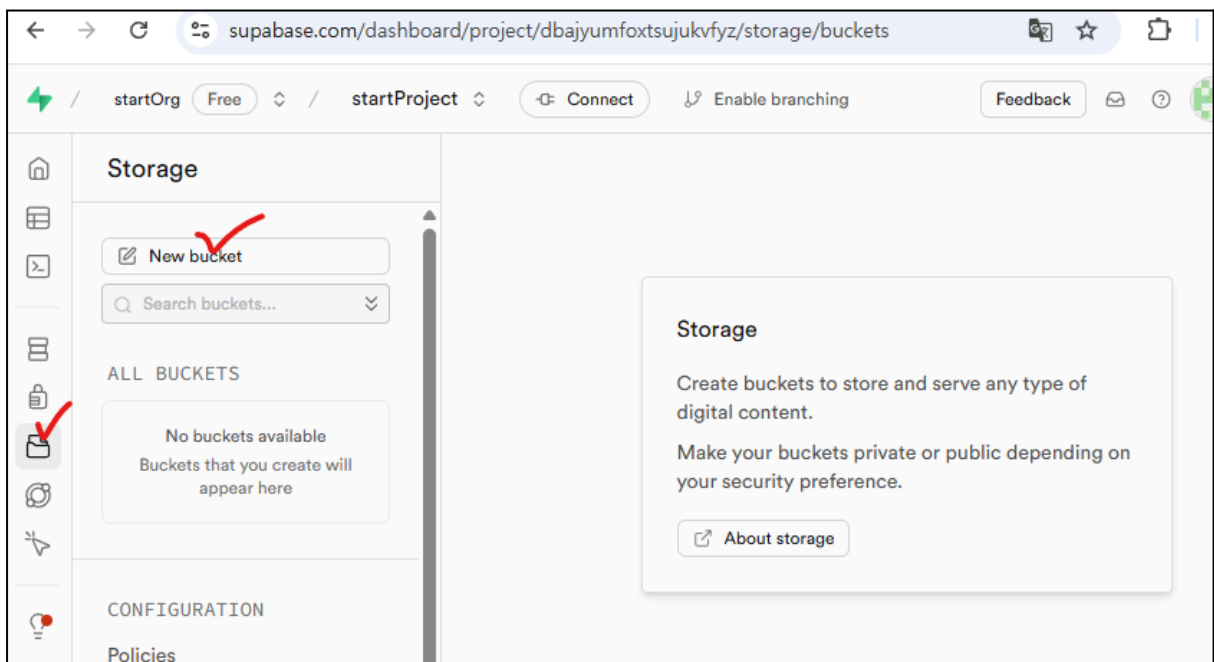
```
.from('avatars')  
.getPublicUrl('user1234/profile.png');
```

만약 버킷이나 파일이 비공개로 설정되어 있다면, 다운로드용 서명된(**signed**) URL을 생성하는 방법도 제공됩니다.

Supabase Storage의 특징 요약

- 파일을 버킷 단위로 관리할 수 있다.
- 파일 접근 권한을 세밀하게 설정할 수 있다.
- Storage와 PostgreSQL 데이터베이스를 함께 사용할 수 있다.
- 프론트엔드에서 직접 접근할 수 있는 API를 제공한다.
- 서버리스 환경에서도 빠르게 파일 업로드와 다운로드를 처리할 수 있다.

>storage에서 버킷 만들기



정책 추가 방법

Create storage bucket

Name of bucket

Buckets cannot be renamed once created.


fileupload

Only lowercase letters, numbers, dots, and hyphens

☒

Public bucket

Anyone can read any object without any authorization



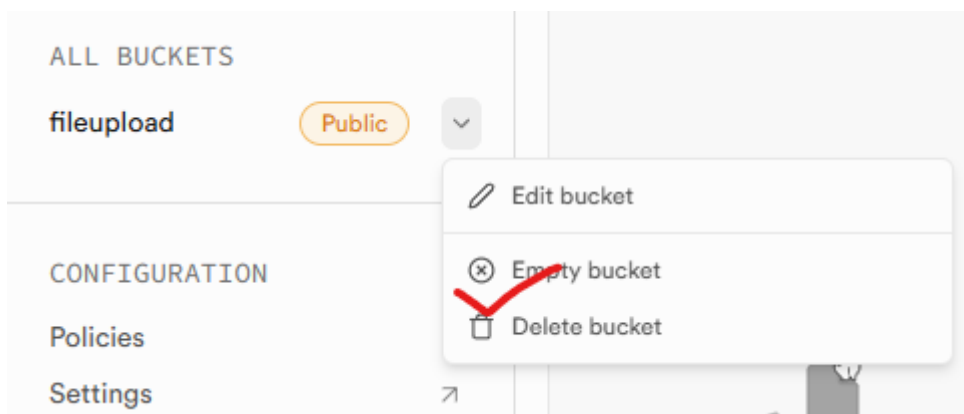
Public buckets are not protected
Users can read objects in public buckets without any authorization.
Row level security (RLS) policies are still required for other operations such as object uploads and deletes.

Additional configuration

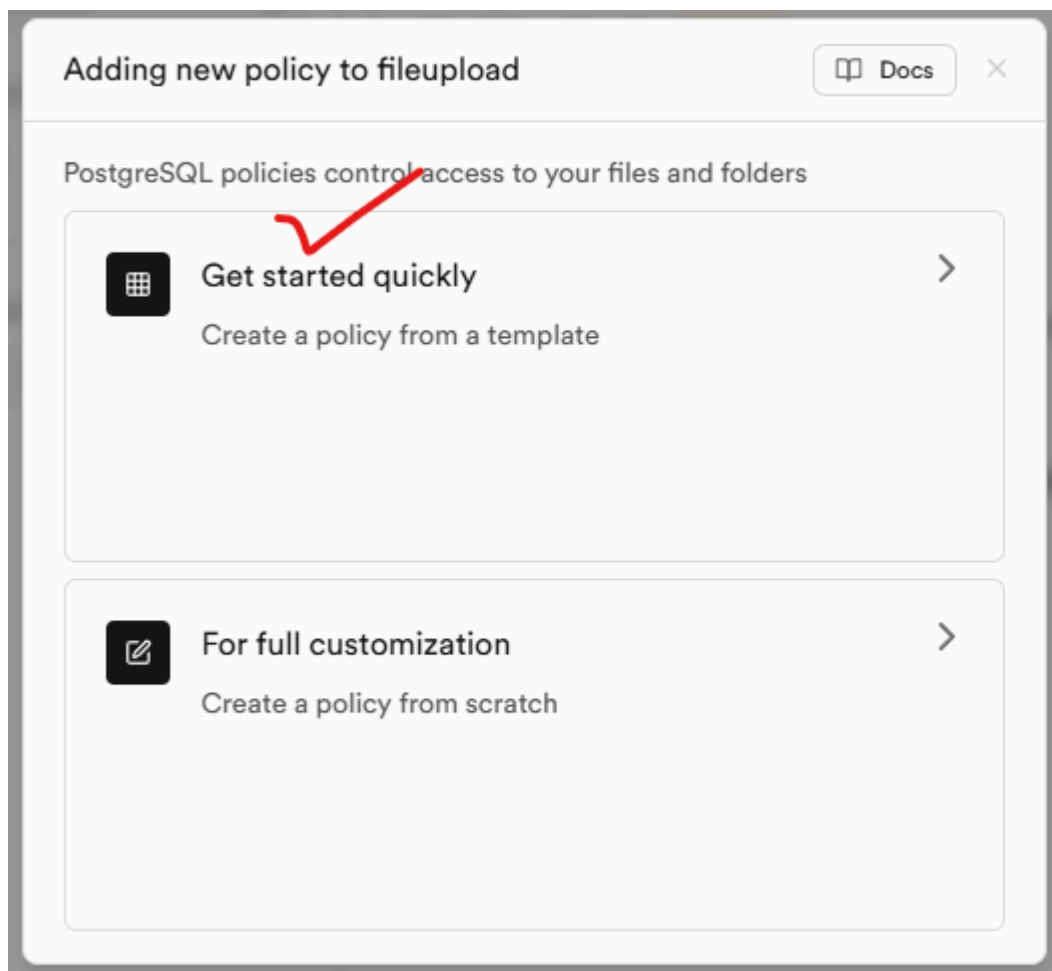
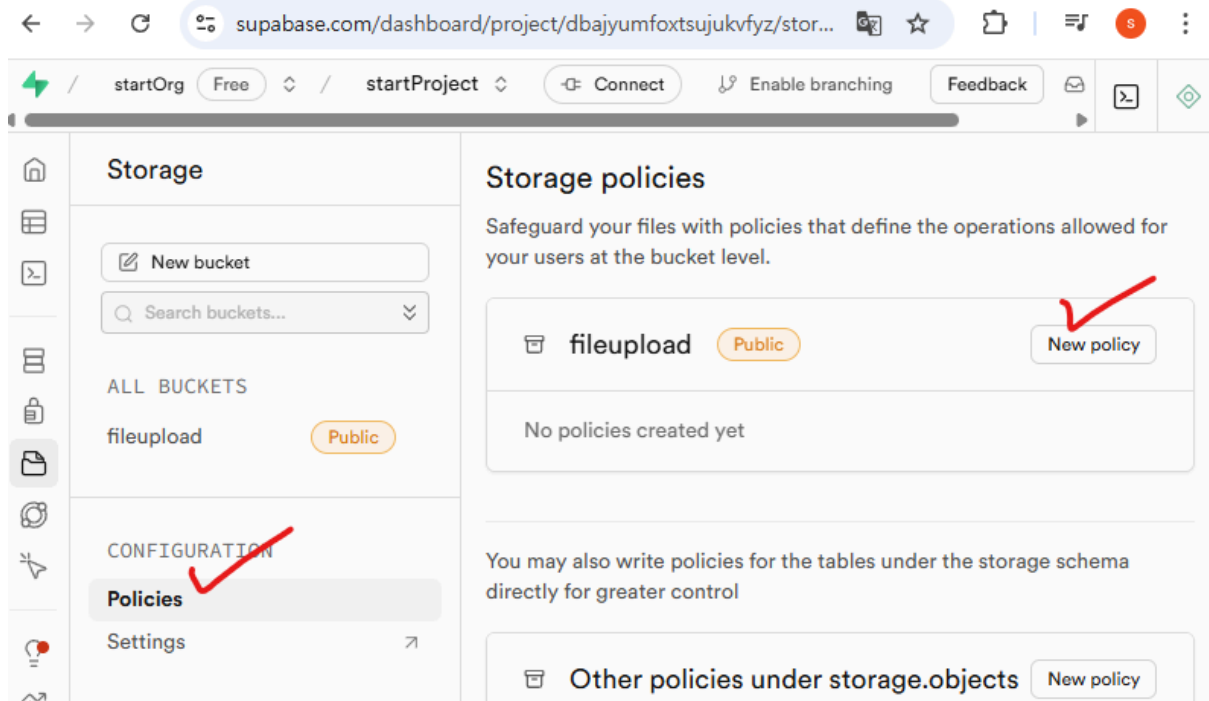
Cancel

Save

삭제 방법



새로운 정책 추가 방법



권한 설정

< Select a template to use for your new policy

Allow access to JPG images in a public folder to anonymous users

Give users access to only their own top level folder named as uid

Give users access to a folder only to authenticated users

Give access to a nested folder called admin/assets only to a specific user

Give access to a file to a user

Give users access to only their own top level folder named as uid

For example a user with id d7bed83c-44a0-4a4f-925f-efc384ea1e50 will be able to access anything under folder d7bed83c-44a0-4a4f-925f-efc384ea1e50/

Policy SQL template:

```
1 CREATE POLICY "policy_name"
2 ON storage.objects FOR {operation} {USING | WITH CHECK} (
3   -- restrict bucket
4   bucket_id = {bucket_name}
5   and (select auth.uid()::text) = (storage.foldername(name))[1]
6 );
```

This will override any existing code you've written

Use this template

다음은 상위 템플릿 관련 설명이야

Allow access to JPG images in a public folder to anonymous users 아무나 로그인 없이 **public** 폴더 안의 **.jpg** 파일만 접근할 수 있게 한다.

Give users access to only their own top level folder named as uid ****사용자 고유 ID(uid)****와 이름이 같은 폴더만 접근할 수 있게 한다. (예: 사용자 ID가 **abcd**면 **/abcd/** 폴더만 접근 가능)

Give users access to a folder only to authenticated users 로그인한 사용자만 특정 폴더에 접근할 수 있게 한다.

Give access to a nested folder called admin/assets only to a specific user **admin/assets** 라는 폴더에 특정 사용자 한 명만 접근할 수 있게 한다. (예: 관리자만 접근)

Give access to a file to a user 특정 파일 하나에 대해 특정 사용자만 접근할 수 있도록 한다.

Policy name

A descriptive name for your policy

Give users access to own folder 31/50

Allowed operation

Based on the operations you have selected, you can use the highlighted functions in the [client library](#).

☒ SELECT ☒ INSERT ☒ UPDATE ☒ DELETE

upload download list update move copy remove

createSignedUrl createSignedUrls getPublicUrl

Target roles

Apply policy to the selected roles

authenticated x

Policy definition

Provide a SQL conditional expression that returns a boolean.

```
1 bucket_id = 'fileupload' AND (select auth.uid()::text) = (storage.foldername(name))[1]
```

View templates Review

Policy name

정책 이름

정책에 대한 설명적인 이름을 입력하세요.

Give users access to own folder

사용자에게 자신의 폴더에 대한 접근 권한 부여

(※ '31/50'은 이름 글자 수 제한으로 31자 입력했고 최대 50자까지 가능하다는 의미로 보입니다.)

Allowed operation

허용된 작업

선택한 작업(operation)에 따라, 클라이언트 라이브러리에서 강조 표시된 기능들을 사용할 수 있습니다.

- SELECT (조회) INSERT (삽입) UPDATE (수정) DELETE (삭제)
- upload (업로드) download (다운로드)
- list (목록 조회) update (수정)
- move (이동) copy (복사) remove (삭제)
- createSignedUrl (서명된 URL 생성)

- createSignedUrls (여러 서명된 URL 생성)
- getPublicUrl (공개 URL 가져오기)

Target roles

대상 역할

선택한 역할에 이 정책을 적용합니다.

- authenticated (인증된 사용자)

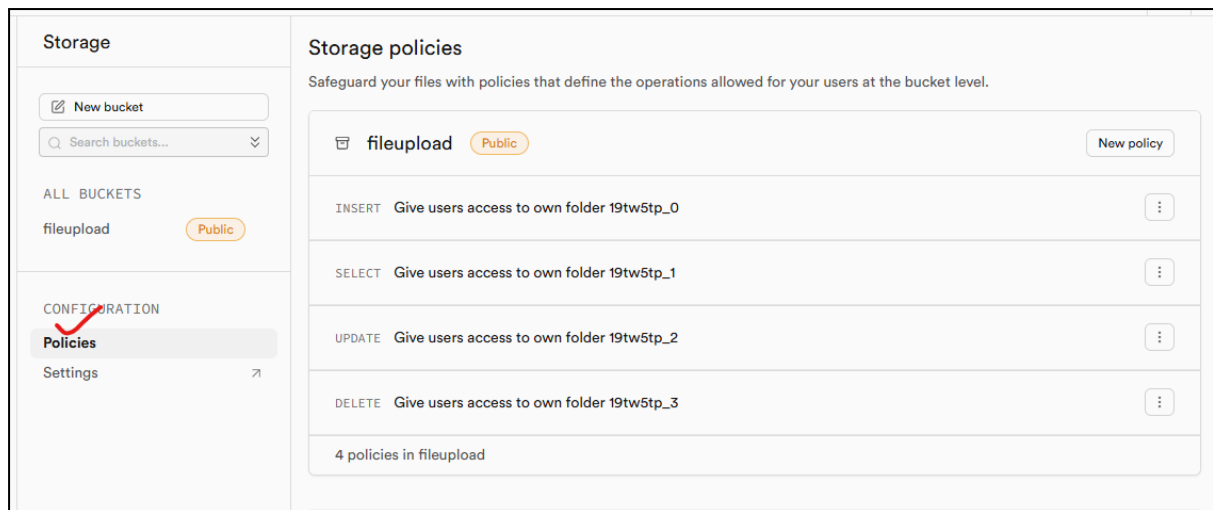
Policy definition

정책 정의

boolean 값을 반환하는 SQL 조건식을 입력하세요.

최종 선택후 확인을 거쳐 다음과 같이 등록된 내용을 확인할 수 있다.





로그인후 파일 업로드 하는 예제입니다.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Supabase File Upload App</title>
  <script src="https://cdn.jsdelivr.net/npm/@supabase/supabase-js@2"></script>
  <!-- UUID 라이브러리 최신 버전으로 변경 -->
  <script
src="https://cdn.jsdelivr.net/npm/uuid@8.3.2/dist/umd/uuid.min.js"></script>
  <style>
    body {
      font-family: Arial, sans-serif;
      max-width: 800px;
      margin: 0 auto;
      padding: 20px;
    }
    #auth-form, #user-area {
      border: 1px solid #ddd;
      padding: 20px;
      border-radius: 8px;
      margin-bottom: 20px;
    }
```

```
input[type="email"],
input[type="password"],
input[type="file"] {
    width: 100%;
    padding: 8px;
    margin: 10px 0;
    border: 1px solid #ccc;
    border-radius: 4px;
}
button {
    padding: 10px 15px;
    margin: 5px;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    background-color: #008CBA;
    color: white;
}
#logout-btn {
    background-color: #e74c3c;
}
.uploads img {
    width: 200px;
    margin: 10px;
    border: 1px solid #ddd;
    border-radius: 4px;
}
#status {
    margin: 10px 0;
    padding: 10px;
    border-radius: 4px;
}
.error {
    background-color: #ffebee;
    color: #c62828;
}
.success {
    background-color: #e8f5e9;
    color: #2e7d32;
}
</style>
</head>
```

```

<body>
  <div id="app">
    <div id="auth-form">
      <h2>Login / Sign Up</h2>
      <input type="email" id="email" placeholder="Email" required><br>
      <input type="password" id="password" placeholder="Password"
required><br>
      <button id="login-btn">Login</button>
      <button id="signup-btn">Sign Up</button>
      <div id="status"></div>
    </div>

    <div id="user-area" style="display: none;">
      <h2 id="welcome-msg"></h2>
      <input type="file" id="file-input" accept="image/*">
      <div class="uploads" id="uploads"></div>
      <button id="logout-btn">Logout</button>
    </div>
  </div>

  <script>
    // Initialize Supabase client first (before DOMContentLoaded)
    const supabaseUrl = 'https://dbajyumfoxtsujukvfyfz.supabase.co';
    const supabaseAnonKey =
'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSI6ImRiYWp5
dW1mb3h0c3VqdWt2Zn16Iiwicm9sZSI6ImFub24iLCJpYXQiOiJlE3NDUzODcyMjEsImV4cCI6MjA2MD
k2MzIyMX0.q1kH_jzgcsUGPgQ0Bx6k5u4276cm1pK-VtJi7AOTy4U';
    const supabase = window.supabase.createClient(supabaseUrl,
supabaseAnonKey);

    document.addEventListener('DOMContentLoaded', () => {
      let currentUser = null;
      const statusDiv = document.getElementById('status');

      function showStatus(message, isError = false) {
        statusDiv.textContent = message;
        statusDiv.className = isError ? 'error' : 'success';
      }

      async function getUser() {
        const { data: { session }, error } = await supabase.auth.getSession();
        if (error) {

```

```

        showStatus('Session Error: ' + error.message, true);
        return;
    }
    currentUser = session?.user || null;
    updateUI();
    if (currentUser) {
        showStatus(`Logged in as ${currentUser.email}`);
        getMedia();
    }
}

// 로그인 폼 처리
document.getElementById('login-btn').addEventListener('click', async (e)
=> {
    e.preventDefault();
    const email = document.getElementById('email').value.trim();
    const password = document.getElementById('password').value.trim();

    if (!email || !password) {
        showStatus('Please enter both email and password', true);
        return;
    }

    showStatus('Logging in...');

    const { data, error } = await supabase.auth.signInWithPassword({
        email,
        password,
        options: {
            redirectTo: window.location.href
        }
    });

    if (error) {
        showStatus('Login Error: ' + error.message, true);
    } else {
        currentUser = data.user;
        updateUI();
        getMedia();
        showStatus('Login successful!');
    }
});

```

```

// 회원가입 처리
document.getElementById('signup-btn').addEventListener('click', async
(e) => {
  e.preventDefault();
  const email = document.getElementById('email').value.trim();
  const password = document.getElementById('password').value.trim();

  if (!email || !password) {
    showStatus('Please enter both email and password', true);
    return;
  }

  showStatus('Signing up...');

  const { data, error } = await supabase.auth.signUp({
    email,
    password,
    options: {
      emailRedirectTo: window.location.href
    }
  });

  if (error) {
    showStatus('Signup Error: ' + error.message, true);
  } else {
    showStatus('Signup successful! Please check your email for
confirmation.');
```

```

// 로그아웃 처리
```

```

document.getElementById('logout-btn').addEventListener('click', async ()
=> {
  const { error } = await supabase.auth.signOut();
  if (error) {
    showStatus('Logout Error: ' + error.message, true);
  } else {
    currentUser = null;
    document.getElementById('uploads').innerHTML = '';
    updateUI();
    showStatus('Logged out successfully');
```

```

    }
  });

function updateUI() {
  if (currentUser) {
    document.getElementById('auth-form').style.display = 'none';
    document.getElementById('user-area').style.display = 'block';
    document.getElementById('welcome-msg').innerText = `Welcome
${currentUser.email}`;
  } else {
    document.getElementById('auth-form').style.display = 'block';
    document.getElementById('user-area').style.display = 'none';
  }
}

async function uploadImage(e) {
  const file = e.target.files[0];
  if (!file || !currentUser) return;

  showStatus('Uploading image...');

  // uuidv4() 대신 uuid.v4() 사용
  const filePath = `${currentUser.id}/${uuid.v4()}`;
  const { error } = await
supabase.storage.from('fileupload').upload(filePath, file);

  if (error) {
    showStatus('Upload Error: ' + error.message, true);
  } else {
    showStatus('Image uploaded successfully!');
    getMedia();
  }
}

async function getMedia() {
  if (!currentUser) return;

  showStatus('Loading images...');

  const { data, error } = await supabase.storage
    .from('test')
    .list(currentUser.id, {

```



```

        limit: 100,
        sortBy: { column: 'name', order: 'asc' }
    });

    const uploadsDiv = document.getElementById('uploads');
    uploadsDiv.innerHTML = '';

    if (error) {
        showStatus('Load Error: ' + error.message, true);
        return;
    }

    if (data.length === 0) {
        showStatus('No images found. Upload some!');
        return;
    }

    showStatus(`Found ${data.length} images`);

    for (const file of data) {
        const publicUrl =
`${supabaseUrl}/storage/v1/object/public/fileupload/${currentUser.id}/${file.name}`;

        const img = document.createElement('img');
        img.src = publicUrl;
        img.alt = file.name;
        uploadsDiv.appendChild(img);
    }
}

// 파일 입력 변경 이벤트 리스너
document.getElementById('file-input').addEventListener('change',
uploadImage);

// 초기 사용자 상태 확인
getUser();
});
</script>
</body>
</html>

```

> 삭제 추가

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Supabase File Upload App</title>
  <script src="https://cdn.jsdelivr.net/npm/@supabase/supabase-js@2"></script>
  <script
src="https://cdn.jsdelivr.net/npm/uuid@8.3.2/dist/umd/uuid.min.js"></script>
  <style>
    body {
      font-family: Arial, sans-serif;
      max-width: 800px;
      margin: 0 auto;
      padding: 20px;
    }
    #auth-form, #user-area {
      border: 1px solid #ddd;
      padding: 20px;
      border-radius: 8px;
      margin-bottom: 20px;
    }
    input[type="email"],
    input[type="password"],
    input[type="file"] {
      width: 100%;
      padding: 8px;
      margin: 10px 0;
      border: 1px solid #ccc;
      border-radius: 4px;
    }
    button {
      padding: 10px 15px;
      margin: 5px;
      border: none;
      border-radius: 4px;
```

```
    cursor: pointer;
    background-color: #008CBA;
    color: white;
}
#logout-btn {
    background-color: #e74c3c;
}
#status {
    margin: 10px 0;
    padding: 10px;
    border-radius: 4px;
}
.error {
    background-color: #ffebee;
    color: #c62828;
}
.success {
    background-color: #e8f5e9;
    color: #2e7d32;
}
.image-container {
    display: flex;
    flex-direction: column;
    align-items: flex-start;
    margin: 15px 0;
    padding: 10px;
    border: 1px solid #eee;
    border-radius: 8px;
    background-color: #f9f9f9;
}
.image-container img {
    max-width: 100%;
    height: auto;
    margin-bottom: 10px;
    border-radius: 4px;
}
.delete-btn {
    background-color: #ff5252;
    margin-top: 5px;
    align-self: flex-end;
}
.file-info {
```

```

        margin-bottom: 10px;
        font-size: 14px;
        color: #666;
    }
</style>
</head>
<body>
    <div id="app">
        <div id="auth-form">
            <h2>Login / Sign Up</h2>
            <input type="email" id="email" placeholder="Email" required><br>
            <input type="password" id="password" placeholder="Password"
required><br>
            <button id="login-btn">Login</button>
            <button id="signup-btn">Sign Up</button>
            <div id="status"></div>
        </div>

        <div id="user-area" style="display: none;">
            <h2 id="welcome-msg"></h2>
            <input type="file" id="file-input" accept="image/*">
            <div class="uploads" id="uploads"></div>
            <button id="logout-btn">Logout</button>
        </div>
    </div>

    <script>
        const supabaseUrl = 'https://dbajyumfoxtsujukvfyfz.supabase.co';
        const supabaseAnonKey =
'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSI6ImRiYWp5
dW1mb3h0c3VqdWt2Zn16Iiwicm9sZSI6ImFub24iLCJpYXQiOiJlE3NDUzODcyMjEsImV4cCI6MjA2MD
k2MzIyMX0.q1kH_jzgcSUGPgQ0Bx6k5u4276cm1pK-VtJi7AOTy4U';
        const supabase = window.supabase.createClient(supabaseUrl,
supabaseAnonKey);

        document.addEventListener('DOMContentLoaded', () => {
            let currentUser = null;
            const statusDiv = document.getElementById('status');

            function showStatus(message, isError = false) {
                statusDiv.textContent = message;
                statusDiv.className = isError ? 'error' : 'success';
            }

```

```
}
```

```
async function getUser() {  
  const { data: { session }, error } = await supabase.auth.getSession();  
  if (error) {  
    showStatus('Session Error: ' + error.message, true);  
    return;  
  }  
  currentUser = session?.user || null;  
  updateUI();  
  if (currentUser) {  
    showStatus(`Logged in as ${currentUser.email}`);  
    getMedia();  
  }  
}
```

```
document.getElementById('login-btn').addEventListener('click', async (e)  
=> {
```

```
  e.preventDefault();  
  const email = document.getElementById('email').value.trim();  
  const password = document.getElementById('password').value.trim();  
  
  if (!email || !password) {  
    showStatus('Please enter both email and password', true);  
    return;  
  }
```

```
  showStatus('Logging in...');
```

```
  const { data, error } = await supabase.auth.signInWithPassword({  
    email,  
    password,  
    options: {  
      redirectTo: window.location.href  
    }  
  });
```

```
  if (error) {  
    showStatus('Login Error: ' + error.message, true);  
  } else {  
    currentUser = data.user;  
    updateUI();  
  }
```

```

        getMedia();
        showStatus('Login successful!');
    }
});

document.getElementById('signup-btn').addEventListener('click', async
(e) => {
    e.preventDefault();
    const email = document.getElementById('email').value.trim();
    const password = document.getElementById('password').value.trim();

    if (!email || !password) {
        showStatus('Please enter both email and password', true);
        return;
    }

    showStatus('Signing up...');

    const { data, error } = await supabase.auth.signUp({
        email,
        password,
        options: {
            emailRedirectTo: window.location.href
        }
    });

    if (error) {
        showStatus('Signup Error: ' + error.message, true);
    } else {
        showStatus('Signup successful! Please check your email for
confirmation.');
```

```

        updateUI();
        showStatus('Logged out successfully');
    }
});

function updateUI() {
    if (currentUser) {
        document.getElementById('auth-form').style.display = 'none';
        document.getElementById('user-area').style.display = 'block';
        document.getElementById('welcome-msg').innerText = `Welcome
${currentUser.email}`;
    } else {
        document.getElementById('auth-form').style.display = 'block';
        document.getElementById('user-area').style.display = 'none';
    }
}

async function uploadImage(e) {
    const file = e.target.files[0];
    if (!file || !currentUser) return;

    showStatus('Uploading image...');

    const filePath = `${currentUser.id}/${uuid.v4()}`;
    const { error } = await
supabase.storage.from('fileupload').upload(filePath, file);

    if (error) {
        showStatus('Upload Error: ' + error.message, true);
    } else {
        showStatus('Image uploaded successfully!');
        getMedia();
    }
}

async function getMedia() {
    if (!currentUser) return;

    showStatus('Loading images...');

    const { data, error } = await supabase.storage
        .from('fileupload')

```

```

        .list(currentUser.id, {
            limit: 100,
            sortBy: { column: 'name', order: 'asc' }
        });

const uploadsDiv = document.getElementById('uploads');
uploadsDiv.innerHTML = '';

if (error) {
    showStatus('Load Error: ' + error.message, true);
    return;
}

if (data.length === 0) {
    showStatus('No images found. Upload some!');
    return;
}

showStatus(`Found ${data.length} images`);

for (const file of data) {
    const publicUrl =
`${supabaseUrl}/storage/v1/object/public/fileupload/${currentUser.id}/${file.name}`;

    const container = document.createElement('div');
    container.className = 'image-container';

    const fileInfo = document.createElement('div');
    fileInfo.className = 'file-info';
    fileInfo.textContent = `File: ${file.name}`;

    const img = document.createElement('img');
    img.src = publicUrl;
    img.alt = file.name;

    const deleteBtn = document.createElement('button');
    deleteBtn.className = 'delete-btn';
    deleteBtn.textContent = '삭제';
    deleteBtn.onclick = () => deleteImage(file.name);

    container.appendChild(fileInfo);

```



```

        container.appendChild(img);
        container.appendChild(deleteBtn);

        uploadsDiv.appendChild(container);
    }
}

async function deleteImage(fileName) {
    if (!currentUser || !fileName) return;

    const confirmDelete = confirm('정말 이 파일을 삭제하시겠습니까?');
    if (!confirmDelete) return;

    showStatus('Deleting image...');

    const filePath = `${currentUser.id}/${fileName}`;
    const { error } = await
supabase.storage.from('fileupload').remove([filePath]);

    if (error) {
        showStatus('Delete Error: ' + error.message, true);
    } else {
        showStatus('Image deleted successfully!');
        getMedia();
    }
}

document.getElementById('file-input').addEventListener('change',
uploadImage);
    getUser();
});
</script>
</body>
</html>

```

>확장자에 따른 파일 이미지 추가가

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Supabase File Upload App</title>
  <script src="https://cdn.jsdelivr.net/npm/@supabase/supabase-js@2"></script>
  <script
src="https://cdn.jsdelivr.net/npm/uuid@8.3.2/dist/umd/uuid.min.js"></script>
  <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.cs
s">
  <style>
    body {
      font-family: Arial, sans-serif;
      max-width: 800px;
      margin: 0 auto;
      padding: 20px;
    }
    #auth-form, #user-area {
      border: 1px solid #ddd;
      padding: 20px;
      border-radius: 8px;
      margin-bottom: 20px;
    }
    input[type="email"],
    input[type="password"],
    input[type="file"] {
      width: 100%;
      padding: 8px;
      margin: 10px 0;
      border: 1px solid #ccc;
      border-radius: 4px;
    }
    button {
      padding: 10px 15px;
      margin: 5px;
```

```
    border: none;
    border-radius: 4px;
    cursor: pointer;
    background-color: #008CBA;
    color: white;
}
#logout-btn {
    background-color: #e74c3c;
}
#status {
    margin: 10px 0;
    padding: 10px;
    border-radius: 4px;
}
.error {
    background-color: #ffebee;
    color: #c62828;
}
.success {
    background-color: #e8f5e9;
    color: #2e7d32;
}
.image-container {
    display: flex;
    flex-direction: column;
    align-items: flex-start;
    margin: 15px 0;
    padding: 10px;
    border: 1px solid #eee;
    border-radius: 8px;
    background-color: #f9f9f9;
}
.image-container img {
    max-width: 100%;
    height: auto;
    margin-bottom: 10px;
    border-radius: 4px;
}
.file-icon {
    font-size: 48px;
    margin-bottom: 10px;
    color: #555;
```

```

    }
    .delete-btn {
      background-color: #ff5252;
      margin-top: 5px;
      align-self: flex-end;
    }
    .file-info {
      margin-bottom: 10px;
      font-size: 14px;
      color: #666;
    }
    .file-preview {
      width: 100%;
      display: flex;
      flex-direction: column;
      align-items: center;
      margin-bottom: 10px;
    }
  </style>
</head>
<body>
  <div id="app">
    <div id="auth-form">
      <h2>Login / Sign Up</h2>
      <input type="email" id="email" placeholder="Email" required><br>
      <input type="password" id="password" placeholder="Password"
required><br>
      <button id="login-btn">Login</button>
      <button id="signup-btn">Sign Up</button>
      <div id="status"></div>
    </div>

    <div id="user-area" style="display: none;">
      <h2 id="welcome-msg"></h2>
      <input type="file" id="file-input">
      <div class="uploads" id="uploads"></div>
      <button id="logout-btn">Logout</button>
    </div>
  </div>

  <script>
    const supabaseUrl = 'https://dbajyumfoxtsujukvfyz.supabase.co';

```

```

    const supabaseAnonKey =
'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSI6ImRiYWp5dW1mb3h0c3VqdWt2Zn16Iiwicm9sZSI6ImFub24iLCJpYXQiOiE3NDUzODcyMjEsImV4cCI6MjA2MDk2MzIyMX0.q1kH_jzgcSUGPgQ0Bx6k5u4276cm1pK-VtJi7A0Ty4U';

    const supabase = window.supabase.createClient(supabaseUrl,
supabaseAnonKey);

    // 이미지 파일 확장자 확인 함수
    function isImageFile(filename) {
        const imageExtensions = ['.jpg', '.jpeg', '.png', '.gif', '.bmp',
'.webp'];
        const ext = filename.substring(filename.lastIndexOf('.')).toLowerCase();
        return imageExtensions.includes(ext);
    }

    // 파일 아이콘 결정 함수
    function getFileIcon(filename) {
        const ext = filename.substring(filename.lastIndexOf('.')).toLowerCase();

        const iconMap = {
            '.pdf': 'file-pdf',
            '.doc': 'file-word',
            '.docx': 'file-word',
            '.xls': 'file-excel',
            '.xlsx': 'file-excel',
            '.ppt': 'file-powerpoint',
            '.pptx': 'file-powerpoint',
            '.txt': 'file-alt',
            '.zip': 'file-archive',
            '.rar': 'file-archive',
            '.mp3': 'file-audio',
            '.wav': 'file-audio',
            '.mp4': 'file-video',
            '.mov': 'file-video'
        };

        return iconMap[ext] || 'file';
    }

    document.addEventListener('DOMContentLoaded', () => {
        let currentUser = null;
        const statusDiv = document.getElementById('status');

```

```

function showStatus(message, isError = false) {
  statusDiv.textContent = message;
  statusDiv.className = isError ? 'error' : 'success';
}

async function getUser() {
  const { data: { session }, error } = await supabase.auth.getSession();
  if (error) {
    showStatus('Session Error: ' + error.message, true);
    return;
  }
  currentUser = session?.user || null;
  updateUI();
  if (currentUser) {
    showStatus(`Logged in as ${currentUser.email}`);
    getMedia();
  }
}

```

```

document.getElementById('login-btn').addEventListener('click', async (e)
=> {
  e.preventDefault();
  const email = document.getElementById('email').value.trim();
  const password = document.getElementById('password').value.trim();

  if (!email || !password) {
    showStatus('Please enter both email and password', true);
    return;
  }

  showStatus('Logging in...');

  const { data, error } = await supabase.auth.signInWithPassword({
    email,
    password,
    options: {
      redirectTo: window.location.href
    }
  });

  if (error) {

```

```

        showStatus('Login Error: ' + error.message, true);
    } else {
        currentUser = data.user;
        updateUI();
        getMedia();
        showStatus('Login successful!');
    }
});

document.getElementById('signup-btn').addEventListener('click', async
(e) => {
    e.preventDefault();
    const email = document.getElementById('email').value.trim();
    const password = document.getElementById('password').value.trim();

    if (!email || !password) {
        showStatus('Please enter both email and password', true);
        return;
    }

    showStatus('Signing up...');

    const { data, error } = await supabase.auth.signUp({
        email,
        password,
        options: {
            emailRedirectTo: window.location.href
        }
    });

    if (error) {
        showStatus('Signup Error: ' + error.message, true);
    } else {
        showStatus('Signup successful! Please check your email for
confirmation.');
```

```

        showStatus('Logout Error: ' + error.message, true);
    } else {
        currentUser = null;
        document.getElementById('uploads').innerHTML = '';
        updateUI();
        showStatus('Logged out successfully');
    }
});

function updateUI() {
    if (currentUser) {
        document.getElementById('auth-form').style.display = 'none';
        document.getElementById('user-area').style.display = 'block';
        document.getElementById('welcome-msg').innerText = `Welcome
${currentUser.email}`;
    } else {
        document.getElementById('auth-form').style.display = 'block';
        document.getElementById('user-area').style.display = 'none';
    }
}

async function uploadImage(e) {
    const file = e.target.files[0];
    if (!file || !currentUser) return;

    showStatus('Uploading file...');

    const filePath =
`${currentUser.id}/${uuid.v4()}${file.name.substring(file.name.lastIndexOf('.')
)});`

    const { error } = await
supabase.storage.from('fileupload').upload(filePath, file);

    if (error) {
        showStatus('Upload Error: ' + error.message, true);
    } else {
        showStatus('File uploaded successfully!');
        getMedia();
    }
}

async function getMedia() {

```



```

if (!currentUser) return;

showStatus('Loading files...');

const { data, error } = await supabase.storage
  .from('fileupload')
  .list(currentUser.id, {
    limit: 100,
    sortBy: { column: 'name', order: 'asc' }
  });

const uploadsDiv = document.getElementById('uploads');
uploadsDiv.innerHTML = '';

if (error) {
  showStatus('Load Error: ' + error.message, true);
  return;
}

if (data.length === 0) {
  showStatus('No files found. Upload some!');
  return;
}

showStatus(`Found ${data.length} files`);

for (const file of data) {
  const publicUrl =
`${supabaseUrl}/storage/v1/object/public/fileupload/${currentUser.id}/${file.name}`;

  const container = document.createElement('div');
  container.className = 'image-container';

  const fileInfo = document.createElement('div');
  fileInfo.className = 'file-info';
  fileInfo.textContent = `File: ${file.name}`;

  const previewDiv = document.createElement('div');
  previewDiv.className = 'file-preview';

  if (isImageFile(file.name)) {

```

```

    const img = document.createElement('img');
    img.src = publicUrl;
    img.alt = file.name;
    previewDiv.appendChild(img);
  } else {
    const icon = document.createElement('i');
    icon.className = `fas fa-${getFileIcon(file.name)} file-icon`;
    previewDiv.appendChild(icon);
  }

  const deleteBtn = document.createElement('button');
  deleteBtn.className = 'delete-btn';
  deleteBtn.textContent = '삭제';
  deleteBtn.onclick = () => deleteImage(file.name);

  container.appendChild(fileInfo);
  container.appendChild(previewDiv);
  container.appendChild(deleteBtn);

  uploadsDiv.appendChild(container);
}
}

async function deleteImage(fileName) {
  if (!currentUser || !fileName) return;

  const confirmDelete = confirm('정말 이 파일을 삭제하시겠습니까?');
  if (!confirmDelete) return;

  showStatus('Deleting file...');

  const filePath = `${currentUser.id}/${fileName}`;
  const { error } = await
supabase.storage.from('fileupload').remove([filePath]);

  if (error) {
    showStatus('Delete Error: ' + error.message, true);
  } else {
    showStatus('File deleted successfully!');
    getMedia();
  }
}

```

```
        document.getElementById('file-input').addEventListener('change',  
uploadImage);  
        getUser();  
    });  
</script>  
</body>  
</html>
```

>다운로드 추가

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <title>Supabase File Upload App</title>

  <script src="https://cdn.jsdelivr.net/npm/@supabase/supabase-js@2"></script>

  <script
src="https://cdn.jsdelivr.net/npm/uuid@8.3.2/dist/umd/uuid.min.js"></script>

  <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.cs
s">

  <style>

    body {

      font-family: Arial, sans-serif;

      max-width: 800px;

      margin: 0 auto;

      padding: 20px;

    }

    #auth-form, #user-area {

      border: 1px solid #ddd;

      padding: 20px;

      border-radius: 8px;

      margin-bottom: 20px;

    }

    input[type="email"],

    input[type="password"],

    input[type="file"] {
```

```
width: 100%;

padding: 8px;

margin: 10px 0;

border: 1px solid #ccc;

border-radius: 4px;

}

button {

padding: 10px 15px;

margin: 5px;

border: none;

border-radius: 4px;

cursor: pointer;

background-color: #008CBA;

color: white;

}

#logout-btn {

background-color: #e74c3c;

}

#status {

margin: 10px 0;

padding: 10px;

border-radius: 4px;

}

.error {

background-color: #ffebee;

color: #c62828;
```

```
}

.success {

    background-color: #e8f5e9;

    color: #2e7d32;

}

.image-container {

    display: flex;

    flex-direction: column;

    align-items: flex-start;

    margin: 15px 0;

    padding: 10px;

    border: 1px solid #eee;

    border-radius: 8px;

    background-color: #f9f9f9;

}

.image-container img {

    max-width: 100%;

    height: auto;

    margin-bottom: 10px;

    border-radius: 4px;

    cursor: pointer;

}

.file-icon {

    font-size: 48px;

    margin-bottom: 10px;

    color: #555;

}
```

```
    cursor: pointer;
}

.delete-btn {

    background-color: #ff5252;

    margin-top: 5px;

    align-self: flex-end;
}

.file-info {

    margin-bottom: 10px;

    font-size: 14px;

    color: #666;
}

.file-preview {

    width: 100%;

    display: flex;

    flex-direction: column;

    align-items: center;

    margin-bottom: 10px;
}

.downloadable {

    cursor: pointer;

    transition: transform 0.2s;
}

.downloadable:hover {

    transform: scale(1.05);
}
```

```
</style>

</head>

<body>

  <div id="app">

    <div id="auth-form">

      <h2>Login / Sign Up</h2>

      <input type="email" id="email" placeholder="Email" required><br>

      <input type="password" id="password" placeholder="Password"
required><br>

      <button id="login-btn">Login</button>

      <button id="signup-btn">Sign Up</button>

      <div id="status"></div>

    </div>

    <div id="user-area" style="display: none;">

      <h2 id="welcome-msg"></h2>

      <input type="file" id="file-input">

      <div class="uploads" id="uploads"></div>

      <button id="logout-btn">Logout</button>

    </div>

  </div>

  <script>

    const supabaseUrl = 'https://dbajyumfoxtsujukvfyz.supabase.co';

    const supabaseAnonKey =
'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSIsInJlZiI6ImRiYWp5
dW1mb3h0c3VqdWt2Zn16Iiwicm9sZSI6ImFub24iLCJpYXQiOi0jE3NDUzODcyMjEsImV4cCI6MjA2MD
k2MzIyMX0.q1kH_jzgcSUGPgQ0Bx6k5u4276cm1pK-VtJi7A0Ty4U';
```



```
const supabase = window.supabase.createClient(supabaseUrl,
supabaseAnonKey);
```

```
// 이미지 파일 확장자 확인 함수
```

```
function isImageFile(filename) {

    const imageExtensions = ['.jpg', '.jpeg', '.png', '.gif', '.bmp',
'.webp'];

    const ext = filename.substring(filename.lastIndexOf('.')).toLowerCase();

    return imageExtensions.includes(ext);

}
```

```
// 파일 아이콘 결정 함수
```

```
function getFileIcon(filename) {

    const ext = filename.substring(filename.lastIndexOf('.')).toLowerCase();

    const iconMap = {

        '.pdf': 'file-pdf',

        '.doc': 'file-word',

        '.docx': 'file-word',

        '.xls': 'file-excel',

        '.xlsx': 'file-excel',

        '.ppt': 'file-powerpoint',

        '.pptx': 'file-powerpoint',

        '.txt': 'file-alt',

        '.zip': 'file-archive',

        '.rar': 'file-archive',

        '.mp3': 'file-audio',
```

```

        '.wav': 'file-audio',

        '.mp4': 'file-video',

        '.mov': 'file-video'

    };

    return iconMap[ext] || 'file';
}

// 파일 다운로드 함수

async function downloadFile(userId, fileName) {

    const publicUrl =
`${supabaseUrl}/storage/v1/object/public/fileupload/${userId}/${fileName}`;

    // 파일 다운로드를 위한 임시 링크 생성

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

    a.href = publicUrl;

    a.download = fileName;

    document.body.appendChild(a);

    a.click();

    document.body.removeChild(a);
}

document.addEventListener('DOMContentLoaded', () => {

    let currentUser = null;

    const statusDiv = document.getElementById('status');

    function showStatus(message, isError = false) {

```

```

    statusDiv.textContent = message;

    statusDiv.className = isError ? 'error' : 'success';
}

```

```

async function getUser() {

    const { data: { session }, error } = await supabase.auth.getSession();

    if (error) {

        showStatus('Session Error: ' + error.message, true);

        return;

    }

    currentUser = session?.user || null;

    updateUI();

    if (currentUser) {

        showStatus(`Logged in as ${currentUser.email}`);

        getMedia();

    }

}

```

```

document.getElementById('login-btn').addEventListener('click', async (e)
=> {

    e.preventDefault();

    const email = document.getElementById('email').value.trim();

    const password = document.getElementById('password').value.trim();

    if (!email || !password) {

        showStatus('Please enter both email and password', true);

        return;

    }

```

```
}
```

```
showStatus('Logging in...');
```

```
const { data, error } = await supabase.auth.signInWithPassword({  
  email,  
  password,  
  options: {  
    redirectTo: window.location.href  
  }  
});
```

```
if (error) {  
  showStatus('Login Error: ' + error.message, true);  
} else {  
  currentUser = data.user;  
  updateUI();  
  getMedia();  
  showStatus('Login successful!');  
}  
});
```

```
document.getElementById('signup-btn').addEventListener('click', async  
(e) => {  
  e.preventDefault();  
  
  const email = document.getElementById('email').value.trim();  
  
  const password = document.getElementById('password').value.trim();
```

```

    if (!email || !password) {

        showStatus('Please enter both email and password', true);

        return;
    }

    showStatus('Signing up...');

    const { data, error } = await supabase.auth.signUp({

        email,

        password,

        options: {

            emailRedirectTo: window.location.href

        }

    });

    if (error) {

        showStatus('Signup Error: ' + error.message, true);

    } else {

        showStatus('Signup successful! Please check your email for confirmation.');
```

```

    }

});

document.getElementById('logout-btn').addEventListener('click', async ()
=> {
```

```

    const { error } = await supabase.auth.signOut();
```

```
if (error) {  
    showStatus('Logout Error: ' + error.message, true);  
} else {  
    currentUser = null;  
    document.getElementById('uploads').innerHTML = '';  
    updateUI();  
    showStatus('Logged out successfully');  
}  
});
```

```
function updateUI() {  
    if (currentUser) {  
        document.getElementById('auth-form').style.display = 'none';  
        document.getElementById('user-area').style.display = 'block';  
        document.getElementById('welcome-msg').innerText = `Welcome  
${currentUser.email}`;  
    } else {  
        document.getElementById('auth-form').style.display = 'block';  
        document.getElementById('user-area').style.display = 'none';  
    }  
}
```

```
async function uploadImage(e) {  
    const file = e.target.files[0];  
    if (!file || !currentUser) return;  
  
    showStatus('Uploading file...');
```

```
    const filePath =
`${currentUser.id}/${uuid.v4()}${file.name.substring(file.name.lastIndexOf('.')
))}`;
```

```
    const { error } = await
supabase.storage.from('fileupload').upload(filePath, file);
```

```
    if (error) {
        showStatus('Upload Error: ' + error.message, true);
    } else {
        showStatus('File uploaded successfully!');
        getMedia();
    }
}
```

```
async function getMedia() {
    if (!currentUser) return;
```

```
    showStatus('Loading files...');
```

```
    const { data, error } = await supabase.storage
        .from('fileupload')
        .list(currentUser.id, {
            limit: 100,
            sortBy: { column: 'name', order: 'asc' }
        });
```

```
const uploadsDiv = document.getElementById('uploads');

uploadsDiv.innerHTML = '';

if (error) {

    showStatus('Load Error: ' + error.message, true);

    return;

}

if (data.length === 0) {

    showStatus('No files found. Upload some!');

    return;

}

showStatus(`Found ${data.length} files`);

for (const file of data) {

    const publicUrl =
` ${supabaseUrl}/storage/v1/object/public/fileupload/${currentUser.id}/${file.name}`;

    const container = document.createElement('div');

    container.className = 'image-container';

    const fileInfo = document.createElement('div');

    fileInfo.className = 'file-info';

    fileInfo.textContent = `File: ${file.name}`;
```



```

const previewDiv = document.createElement('div');

previewDiv.className = 'file-preview';

if (isImageFile(file.name)) {

    const img = document.createElement('img');

    img.src = publicUrl;

    img.alt = file.name;

    img.className = 'downloadable';

    img.onclick = () => downloadFile(currentUser.id, file.name);

    previewDiv.appendChild(img);

} else {

    const icon = document.createElement('i');

    icon.className = `fas fa-${getFileIcon(file.name)} file-icon
downloadable`;

    icon.onclick = () => downloadFile(currentUser.id, file.name);

    previewDiv.appendChild(icon);

}

const deleteBtn = document.createElement('button');

deleteBtn.className = 'delete-btn';

deleteBtn.textContent = '삭제';

deleteBtn.onclick = (e) => {

    e.stopPropagation(); // 삭제 버튼 클릭 시 다운로드 방지

    deleteImage(file.name);

};

container.appendChild(fileInfo);

```

```
        container.appendChild(previewDiv);

        container.appendChild(deleteBtn);

        uploadsDiv.appendChild(container);
    }
}
```

```
async function deleteImage(fileName) {

    if (!currentUser || !fileName) return;

    const confirmDelete = confirm('정말 이 파일을 삭제하시겠습니까?');

    if (!confirmDelete) return;

    showStatus('Deleting file...');

    const filePath = `${currentUser.id}/${fileName}`;

    const { error } = await
supabase.storage.from('fileupload').remove([filePath]);

    if (error) {

        showStatus('Delete Error: ' + error.message, true);

    } else {

        showStatus('File deleted successfully!');

        getMedia();

    }

}
```

```
        document.getElementById('file-input').addEventListener('change',
uploadImage);

        getUser();

    });

</script>

</body>

</html>
```

➤이미지도 다운로드 하기

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <title>Supabase File Upload App</title>

  <script src="https://cdn.jsdelivr.net/npm/@supabase/supabase-js@2"></script>

  <script
src="https://cdn.jsdelivr.net/npm/uuid@8.3.2/dist/umd/uuid.min.js"></script>

  <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.cs
s">

  <style>

    body {

      font-family: Arial, sans-serif;

      max-width: 800px;

      margin: 0 auto;

      padding: 20px;

    }

    #auth-form, #user-area {

      border: 1px solid #ddd;

      padding: 20px;

      border-radius: 8px;

      margin-bottom: 20px;

    }

    input[type="email"],

    input[type="password"],

    input[type="file"] {
```

```
width: 100%;

padding: 8px;

margin: 10px 0;

border: 1px solid #ccc;

border-radius: 4px;

}

button {

padding: 10px 15px;

margin: 5px;

border: none;

border-radius: 4px;

cursor: pointer;

background-color: #008CBA;

color: white;

}

#logout-btn {

background-color: #e74c3c;

}

#status {

margin: 10px 0;

padding: 10px;

border-radius: 4px;

}

.error {

background-color: #ffebee;

color: #c62828;
```

```
}

.success {

    background-color: #e8f5e9;

    color: #2e7d32;

}

.image-container {

    display: flex;

    flex-direction: column;

    align-items: flex-start;

    margin: 15px 0;

    padding: 10px;

    border: 1px solid #eee;

    border-radius: 8px;

    background-color: #f9f9f9;

}

.image-container img {

    max-width: 100%;

    height: auto;

    margin-bottom: 10px;

    border-radius: 4px;

    cursor: pointer;

    transition: transform 0.2s;

}

.image-container img:hover {

    transform: scale(1.02);

}
```

```
.file-icon {  
    font-size: 48px;  
    margin-bottom: 10px;  
    color: #555;  
    cursor: pointer;  
}  
  
.delete-btn {  
    background-color: #ff5252;  
    margin-top: 5px;  
    align-self: flex-end;  
}  
  
.file-info {  
    margin-bottom: 10px;  
    font-size: 14px;  
    color: #666;  
}  
  
.file-preview {  
    width: 100%;  
    display: flex;  
    flex-direction: column;  
    align-items: center;  
    margin-bottom: 10px;  
}  
  
</style>  
  
</head>  
  
<body>
```

```
<div id="app">

  <div id="auth-form">

    <h2>Login / Sign Up</h2>

    <input type="email" id="email" placeholder="Email" required><br>

    <input type="password" id="password" placeholder="Password"
required><br>

    <button id="login-btn">Login</button>

    <button id="signup-btn">Sign Up</button>

    <div id="status"></div>

  </div>

  <div id="user-area" style="display: none;">

    <h2 id="welcome-msg"></h2>

    <input type="file" id="file-input">

    <div class="uploads" id="uploads"></div>

    <button id="logout-btn">Logout</button>

  </div>

</div>

<script>

  const supabaseUrl = 'https://dbajyumfoxtsujukvfyz.supabase.co';

  const supabaseAnonKey =
'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSIsInJlZiI6ImRiYWp5
dW1mb3h0c3VqdWt2Zn16Iiwicm9sZSI6ImFub24iLCJpYXQiOiJlNDUzODcyMjEsImV4cCI6MjA2MD
k2MzIyMX0.q1kH_jzgcsUGPgQ0Bx6k5u4276cm1pK-VtJi7AOTy4U';

  const supabase = window.supabase.createClient(supabaseUrl,
supabaseAnonKey);
```



```
// 이미지 파일 확장자 확인 함수

function isImageFile(filename) {

    const imageExtensions = ['.jpg', '.jpeg', '.png', '.gif', '.bmp',
'.webp'];

    const ext = filename.substring(filename.lastIndexOf('.')).toLowerCase();

    return imageExtensions.includes(ext);

}
```

// 파일 아이콘 결정 함수

```
function getFileIcon(filename) {

    const ext = filename.substring(filename.lastIndexOf('.')).toLowerCase();

    const iconMap = {

        '.pdf': 'file-pdf',

        '.doc': 'file-word',

        '.docx': 'file-word',

        '.xls': 'file-excel',

        '.xlsx': 'file-excel',

        '.ppt': 'file-powerpoint',

        '.pptx': 'file-powerpoint',

        '.txt': 'file-alt',

        '.zip': 'file-archive',

        '.rar': 'file-archive',

        '.mp3': 'file-audio',

        '.wav': 'file-audio',

        '.mp4': 'file-video',

        '.mov': 'file-video'

    }
```

```
};
```

```
    return iconMap[ext] || 'file';  
  }  
}
```

```
// 파일 다운로드 함수 (React 코드 참고)
```

```
function getDownloadUrl(userId, fileName) {  
  return  
  `${supabaseUrl}/storage/v1/object/public/fileupload/${userId}/${fileName}?down  
load=${fileName}`;  
}
```

```
document.addEventListener('DOMContentLoaded', () => {
```

```
  let currentUser = null;
```

```
  const statusDiv = document.getElementById('status');
```

```
  function showStatus(message, isError = false) {
```

```
    statusDiv.textContent = message;
```

```
    statusDiv.className = isError ? 'error' : 'success';
```

```
  }
```

```
  async function getUser() {
```

```
    const { data: { session }, error } = await supabase.auth.getSession();
```

```
    if (error) {
```

```
      showStatus('Session Error: ' + error.message, true);
```

```
      return;
```

```
    }
```

```

    currentUser = session?.user || null;

    updateUI();

    if (currentUser) {

        showStatus(`Logged in as ${currentUser.email}`);

        getMedia();

    }

}

document.getElementById('login-btn').addEventListener('click', async (e)
=> {

    e.preventDefault();

    const email = document.getElementById('email').value.trim();

    const password = document.getElementById('password').value.trim();

    if (!email || !password) {

        showStatus('Please enter both email and password', true);

        return;

    }

    showStatus('Logging in...');

    const { data, error } = await supabase.auth.signInWithPassword({

        email,

        password,

        options: {

            redirectTo: window.location.href

        }

    });

```

```
});
```

```
if (error) {
```

```
  showStatus('Login Error: ' + error.message, true);
```

```
} else {
```

```
  currentUser = data.user;
```

```
  updateUI();
```

```
  getMedia();
```

```
  showStatus('Login successful!');
```

```
}
```

```
});
```

```
document.getElementById('signup-btn').addEventListener('click', async  
(e) => {
```

```
  e.preventDefault();
```

```
  const email = document.getElementById('email').value.trim();
```

```
  const password = document.getElementById('password').value.trim();
```

```
  if (!email || !password) {
```

```
    showStatus('Please enter both email and password', true);
```

```
    return;
```

```
}
```

```
showStatus('Signing up...');
```

```
const { data, error } = await supabase.auth.signUp({
```

```
  email,
```

```

        password,

        options: {

            emailRedirectTo: window.location.href

        }

    });

    if (error) {

        showStatus('Signup Error: ' + error.message, true);

    } else {

        showStatus('Signup successful! Please check your email for confirmation.');
```

```

    }

});

document.getElementById('logout-btn').addEventListener('click', async ()
=> {

    const { error } = await supabase.auth.signOut();

    if (error) {

        showStatus('Logout Error: ' + error.message, true);

    } else {

        currentUser = null;

        document.getElementById('uploads').innerHTML = '';

        updateUI();

        showStatus('Logged out successfully');

    }

});
```

```

function updateUI() {
  if (currentUser) {
    document.getElementById('auth-form').style.display = 'none';
    document.getElementById('user-area').style.display = 'block';
    document.getElementById('welcome-msg').innerText = `Welcome
${currentUser.email}`;
  } else {
    document.getElementById('auth-form').style.display = 'block';
    document.getElementById('user-area').style.display = 'none';
  }
}

async function uploadImage(e) {
  const file = e.target.files[0];
  if (!file || !currentUser) return;

  showStatus('Uploading file...');

  const filePath =
`${currentUser.id}/${uuid.v4()}${file.name.substring(file.name.lastIndexOf('.')
))}`;

  const { error } = await
supabase.storage.from('fileupload').upload(filePath, file);

  if (error) {
    showStatus('Upload Error: ' + error.message, true);
  } else {
    showStatus('File uploaded successfully!');
  }
}

```

```
    getMedia();  
  }  
}
```

```
async function getMedia() {  
  if (!currentUser) return;  
  
  showStatus('Loading files...');  
  
  const { data, error } = await supabase.storage  
    .from('fileupload')  
    .list(currentUser.id, {  
      limit: 100,  
      sortBy: { column: 'name', order: 'asc' }  
    });  
  
  const uploadsDiv = document.getElementById('uploads');  
  uploadsDiv.innerHTML = '';  
  
  if (error) {  
    showStatus('Load Error: ' + error.message, true);  
    return;  
  }  
  
  if (data.length === 0) {  
    showStatus('No files found. Upload some!');
```

```
        return;
    }

    showStatus(`Found ${data.length} files`);

    for (const file of data) {

        const publicUrl =
`${supabaseUrl}/storage/v1/object/public/fileupload/${currentUser.id}/${file.name}`;

        const downloadUrl = getDownloadUrl(currentUser.id, file.name);

        const container = document.createElement('div');
        container.className = 'image-container';

        const fileInfo = document.createElement('div');
        fileInfo.className = 'file-info';
        fileInfo.textContent = `File: ${file.name}`;

        const previewDiv = document.createElement('div');
        previewDiv.className = 'file-preview';

        if (isImageFile(file.name)) {

            // 이미지인 경우 다운로드 링크로 감싸기

            const imgLink = document.createElement('a');
            imgLink.href = downloadUrl;
            imgLink.target = '_blank';
```



```
const img = document.createElement('img');

img.src = publicUrl;

img.alt = file.name;

imgLink.appendChild(img);

previewDiv.appendChild(imgLink);
} else {

    // 이미지가 아닌 경우 아이콘 표시

    const icon = document.createElement('i');

    icon.className = `fas fa-${getFileIcon(file.name)} file-icon`;

    icon.onclick = () => window.open(downloadUrl, '_blank');

    previewDiv.appendChild(icon);
}

const deleteBtn = document.createElement('button');

deleteBtn.className = 'delete-btn';

deleteBtn.textContent = '삭제';

deleteBtn.onclick = (e) => {

    e.stopPropagation();

    deleteImage(file.name);

};

container.appendChild(fileInfo);

container.appendChild(previewDiv);

container.appendChild(deleteBtn);
```

```
        uploadsDiv.appendChild(container);
    }
}
```

```
async function deleteImage(fileName) {
    if (!currentUser || !fileName) return;

    const confirmDelete = confirm('정말 이 파일을 삭제하시겠습니까?');
    if (!confirmDelete) return;

    showStatus('Deleting file...');

    const filePath = `${currentUser.id}/${fileName}`;

    const { error } = await
supabase.storage.from('fileupload').remove([filePath]);

    if (error) {
        showStatus('Delete Error: ' + error.message, true);
    } else {
        showStatus('File deleted successfully!');
        getMedia();
    }
}
```

```
document.getElementById('file-input').addEventListener('change',
uploadImage);

getUser();
```

```
});  
  
</script>  
  
</body>  
  
</html>
```

>복잡한 파일 이름 변경하기

```
// 파일 다운로드 함수 (React 코드 참고)  
function getDownloadUrl(userId, fileName) {  
    return  
`${supabaseUrl}/storage/v1/object/public/fileupload/${userId}/${fileName}?down  
load=요기변경${fileName}`;  
}
```

결론부터 말하면:

- **Supabase Storage** 자체는 "원래 파일 이름" 같은 메타데이터를 저장해주지 않아.
- 그래서 업로드할 때 파일명을 따로 **DB** 테이블에 저장해둬야 해.
- 그리고 "원래 이름으로 다운로드" 하려면 **DB**를 조회해서 파일명을 알아내고, **Storage**에서 파일을 가져오는 식으로 처리해야 해.
- **Edge Function** 없이도 이거 가능해. 그냥 클라이언트 코드(ex: Flutter, Next.js)나 서버 코드(ex: Node.js)에서 구현하면 돼!

요약 흐름

1. 사용자가 파일을 업로드한다.
2. 업로드하면서
 - **Storage**에는 시스템 이름(ex: **UUID**로 바꾼 이름)으로 저장하고
 - **DB** 테이블에는 → **original_filename**, **stored_filename** 같이 둘 다 저장.
3. 다운로드할 때
 - **DB**에서 **original_filename** 찾고
 - **stored_filename**으로 **Storage**에서 파일을 불러온 다음
 - 다운로드 응답에 **Content-Disposition** 헤더를 설정해서 원래 파일명으로 다운로드.

Supabase만 사용할 때 방법

1. 테이블 만들기

(예: **files** 테이블)

| 컬럼 이름 | 타입 | 설명 |
|----------------------|------------------|---------------------------------|
| id | uuid | 기본키 |
| original_name | text | 사용자가 업로드한 원래 파일명 |
| stored_path | text | Storage 에 저장된 경로(파일명 포함) |
| created_at | timestamp | 업로드 시간 |

2. 업로드할 때

- **Storage**에 저장하고
- 테이블에 기록까지 같이 남긴다.

javascript

복사편집

```
const storedPath = `uploads/${crypto.randomUUID()}.png`;
```

```
const { data: uploadData, error: uploadError } = await supabase
  .storage
  .from('your-bucket')
  .upload(storedPath, file);
```

```
if (!uploadError) {
  await supabase
    .from('files')
    .insert([
      { original_name: file.name, stored_path: storedPath }
    ]);
}
```

3. 다운로드할 때

- DB에서 원래 이름을 가져오고
- Storage에서 파일 다운로드할 때 이름 설정해줄 수 있어야 하는데,
- 일반 URL 다운로드 는 원래 이름 지정이 안 되니까 →
API 서버 하나 두고 거기서 파일 읽고, Content-Disposition 헤더 붙여서 보내야 진짜 "다운로드 시 원래 파일명"이 가능해.

=> 이 부분은 서버(API)를 하나 두거나, Edge Function 쓰면 깔끔해진다!

정리하면

| 방법 | 가능 여부 | 설명 |
|---------------|-------|--------|
| DB에 파일명 따로 저장 | ✓ | 꼭 해야 함 |

Edge Function 없이
처리



가능 (조금 번거로움)

진짜 "다운로드 시 원래
파일명"



브라우저 기본 download 링크로는 힘들.
서버(API)나 Edge Function 필요

추가로

Edge Function 쓰면 Storage에서 파일 읽고,

`Content-Disposition: attachment; filename="원래파일명.png"`

이런 식으로 응답 보내서 깔끔하게 해결할 수 있어.

필요하면,

Edge Function 예제나

서버(API) 코드 예제

둘 중 원하는 방향으로 바로 짜줄게.

👉 "서버 코드로 할까?" 아니면 "Edge Function 써볼까?" 어떤 걸 하고 싶어? 🙄
(또는 "Flutter 쪽에서 다 해결하고 싶어요" 이렇게 말해도 됨)

Supabase의 **bucket(Storage)**에 저장된 파일은 쿼리로 직접 **CRUD** 작업하는 건 불가능해.

bucket은 데이터베이스 테이블이 아니라, 파일 스토리지야.

대신 Supabase는 Storage용 API를 따로 제공해!

즉, 파일 업로드/다운로드/삭제/목록 조회 같은 건 **Storage API**로 해야 해.

정리하면:

작업

방법

파일 업로드

`supabase.storage.from('bucket명').upload('경로/파일명', 파일)`

| | |
|------------------|--|
| 파일 다운로드 URL 가져오기 | <code>supabase.storage.from('bucket명').getPublicUrl('경로/파일명')</code> |
| 파일 삭제 | <code>supabase.storage.from('bucket명').remove(['경로/파일명'])</code> |
| 파일 목록 조회 | <code>supabase.storage.from('bucket명').list('폴더 경로')</code> |

예시 코드 (JavaScript 기준으로):

javascript

복사편집

// 업로드

```
const { data, error } = await supabase
  .storage
  .from('your-bucket')
  .upload('folder/filename.png', file);
```

// 삭제

```
const { data, error } = await supabase
  .storage
  .from('your-bucket')
  .remove(['folder/filename.png']);
```

// 파일 목록 조회

```
const { data, error } = await supabase
  .storage
  .from('your-bucket')
  .list('folder', { limit: 10 });
```

주의할 점

- **bucket**에 접근하려면 퍼블릭 권한 설정하거나,
JWT 인증된 유저로 접근해야 해. (안 그러면 **401 Unauthorized** 에러 나올 수 있어)
- **Storage**는 **SQL** 쿼리처럼 **SELECT**, **INSERT** 이런 거 안 되고,
대신 **Storage API** 통해 작업해야 해.