

WEB アプリケーション説明書

21d2030006

カンシン

目次

WEB ページの紹介	3
作成手順	4
EC2 サーバーの作成	4
SSH 接続	8
Docker	9
Nginx	1 1
PHP	1 2
MYSQL	1 4

WEB ページ紹介

ページ説明

誰でも自由にテキストや画像を投稿できる掲示板の WEB サイト

構築要素

AWS EC2

Docker

Nginx

MY SQL

効能

- 投稿者が自由にテキストを投稿できること
 - 内容は MySQL に保存すること
 - XSS および SQL インジェクションの対策ができていていること
- 投稿それぞれに自動で投稿日時が付与されていること
- 投稿それぞれに自動で連番が付与されていること
- 投稿者が自由に画像を投稿できること
 - 5MB 以上の画像をアップロードできないように
 - 画像をブラウザ側(JavaScript で実装)で 5MB 以下に自動縮小しアップロードするように (これができたら+15 点)

作成手順

1. EC2 サーバー構築



インスタンスを起動を選択



好きな名前を入力する

aws

サービス

検索

[Alt+S]

バージニア北部

voclabs/user2002935=CHIEN_CHIN @ 0205-8794-0678

▼ インスタンスタイプ 情報

インスタンスタイプ

t2.micro

ファミリー: t2 1 vCPU 1 GiB メモリ 現行世代: true
オンデマンド Windows base 料金: 0.0162 USD 1 時間あたり
オンデマンド SUSE base 料金: 0.0116 USD 1 時間あたり
オンデマンド RHEL base 料金: 0.0716 USD 1 時間あたり
オンデマンド Linux base 料金: 0.0116 USD 1 時間あたり

無料利用枠の対象

すべての世代

インスタンスタイプを比較

Additional costs apply for AMIs with pre-installed software

▼ キーペア (ログイン) 情報

キーペアを使用してインスタンスに安全に接続できます。インスタンスを起動する前に、選択したキーペアにアクセスできることを確認してください。

キーペア名 - 必須

選択

新しいキーペアの作成

▼ ネットワーク設定 情報

編集

ネットワーク 情報

▼ 概要

インスタンス数 情報

1

ソフトウェアイメージ (AMI)

Amazon Linux 2023 AMI 2023.1.2...[続きを読む](#)
ami-051f7e7f6c2f40dc1

垂直サーバータイプ (インスタンスタイプ)

t2.micro

ファイアウォール (セキュリティグループ)

新しいセキュリティグループ

ストレージ (ボリューム)

1 ボリューム - 8 GiB

無料利用枠: 最初の 1 年には、1 か月あたりの無料利用枠による AMI での t2.micro (または t2.micro が利用できるリージョンでは t3.micro) インスタンスの 750 時間の使用、30 GiB の EBS ストレージ

×

キャンセル

インスタンスを起動

[コマンドを確認](#)

キーペアを作成



キーペア名

キーペアを使用すると、インスタンスに安全に接続できます。

キーペア名を入力

名前には最大 255 文字の ASCII 文字を使用できます。先頭または末尾にスペースを含めることはできません。

キーペアのタイプ



RSA

RSA で暗号化されたプライベートとパブリックのキーペア



ED25519

ED25519 で暗号化されたプライベートキーとパブリックキーのペア

プライベートキーファイル形式



.pem

OpenSSH で使用する場合



.ppk

PuTTY で使用する場合



プロンプトが表示されたら、コンピュータの安全でアクセス可能な場所にプライベートキーを保存してください。後でインスタンスに接続するときに必要な
なります。 [詳細はこちら](#)

キャンセル

キーペアを作成

キーペア名を入力して図のように選択する

セキュリティグループ

タイプ	プロトコル	ポート範囲
ssh	TCP	22

ソースタイプ	ソース	説明 - optional
任意の場所	CIDR、プレフィックスリスト、 0.0.0.0/0	例: 管理者のデスクトップの SSH

セキュリティグループ

タイプ	プロトコル	ポート範囲
HTTP	TCP	80

ソースタイプ	ソース	説明 - optional
任意の場所	CIDR、プレフィックスリスト、 0.0.0.0/0	例: 管理者のデスクトップの SSH

⚠ 送信元が 0.0.0.0/0 のルールを指定すると、すべての IP アドレスからインスタンスにアクセスすることが許可されます。セキュリティグループのルールを設定して、既知の IP アドレスからのみアクセスできるようにすることをお勧めします。

セキュリティグループを追加

概要

インスタンス数 1

ソフトウェアイメージ (AMI)

Amazon Linux 2023 AMI 2023.1.2...[続きを読む](#)

ami-051f7e76c2f40dc1

垂直サーバータイプ (インスタンスタイプ)

t2.micro

ファイアウォール (セキュリティグループ)

新しいセキュリティグループ

ストレージ (ボリューム)

1 ボリューム - 8 GiB

① 無料利用枠: 最初の 1 年には、1 か月あたりの無料利用枠による AMI での t2.micro (または t2.micro が利用できないリージョンでは t3.micro) インスタンスの 750 時間の使用。30 GiB の FRS ストレージ

キャンセル
インスタンスを起動

コマンドを確認

設定が完了したらインスタンスを起動する

2. SSH 接続

Powershell を開いて

SSH で接続するコマンドを打ちます

```
ssh ec2-user@{EC2 インスタンスのアドレス} -i {pem ファイルのパス}
```

{EC2 インスタンスのアドレス}は AWS の EC2 インスタンスで確認することができる

{pem ファイルのパス}は先ほど作ったキーペア

3. Docker

Docker とは

仮想的な環境を簡単に作るための技術です。仮想的な環境は簡単に作成と削除を行うことができ、面倒なミドルウェアのインストール・設定を毎回行う必要がなくなります。

仮想的な環境のことをコンテナと呼びます。一つのサーバー(コンピューター)内に沢山作ることができ、また同時に動かすことができます。

(ア) Docker インストール方法 & 自動起動化

コマンドを打ち込みます

```
sudo yum install -y docker
sudo systemctl start docker
sudo systemctl enable docker
sudo usermod -a -G docker ec2-user
```

(イ) Docker Compose インストール方法

複数の Docker のコンテナを簡単に扱うためのツール(Docker コマンドのサブコマンド)です。

コマンドを打ち込みます

```
sudo mkdir -p /usr/local/lib/docker/cli-plugins/
sudo curl -SL https://github.com/docker/compose/releases/download/v2.2.2/docker-
compose-linux-x86_64 -o /usr/local/lib/docker/cli-plugins/docker-compose
sudo chmod +x /usr/local/lib/docker/cli-plugins/docker-compose
```

(ウ) Docker Compose は、カレントディレクトリにある `compose.yml` という設定ファイルを参照します。(もちろんオプションで他の場所にある設定ファイルを参照することもできます。)

まずは作業用のディレクトリを作ってその中に移動しましょう。

```
mkdir dockertest
cd dockertest
```

そのため、まずは設定ファイルを書きましょう。

```
vim compose.yml
```

```

1  services:
2    web:
3      image: nginx:latest
4      ports:
5        - 80:80
6      volumes:
7        - ./nginx/conf.d:/etc/nginx/conf.d/
8        - ./public:/var/www/public/
9        - image:/var/www/upload/image/
10     depends_on:
11       - php
12   php:
13     container_name: php
14     build:
15       context: .
16       target: php
17     volumes:
18       - ./public:/var/www/public/
19       - image:/var/www/upload/image/
20   mysql:
21     container_name: mysql
22     image: mysql:8.0
23     environment:
24       MYSQL_DATABASE: techc
25       MYSQL_ALLOW_EMPTY_PASSWORD: 1
26       TZ: Asia/Tokyo
27     volumes:
28       - mysql:/var/lib/mysql
29     command: >
30     mysql:
31       --character-set-server=utf8mb4
32
33       --collation-server=utf8mb4_unicode_ci
34       --max_allowed_packet=4MB
35   volumes:
36     image:
37       driver: local
38   mysql:

```

(エ) PHP の拡張モジュール

PHP の拡張モジュールを利用するために
カレントディレクトリに Dockerfile を作成しましょう
コードは以下のようになります。

```

1  FROM php:8.1-fpm-alpine AS php
2
3  RUN docker-php-ext-install pdo_mysql
4
5  RUN install -o www-data -g www-data -d /var/www/upload/image/
6
7  RUN echo -e "post_max_size = 5M\nupload_max_filesize = 5M" >> ${PHP_INI_DIR}/php.ini

```

4. Nginx

自分の作ったファイルや、プログラムの出力結果も Web に配信することができる WEB サーバー
まず設定ファイル用のディレクトリを作ります。

```
mkdir nginx  
mkdir nginx/conf.d
```

設定ファイルを作成

```
vim nginx/conf.d/default.conf
```

ファイルのコードは以下のようになります

```
1      server {  
2          listen      0.0.0.0:80;  
3          server_name _;  
4          charset     utf-8;  
5          client_max_body_size 6M;  
6  
7          root /var/www/public;  
8  
9          location ~ /\.php$ {  
10             fastcgi_pass  php:9000;  
11             fastcgi_index index.php;  
12             fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;  
13             include       fastcgi_params;  
14         }  
15  
16         location /image/ {  
17             root /var/www/upload;  
18         }  
19     }
```

5. PHP

配信するページの実装

配信するページ用のディレクトリを作ります。

```
mkdir public
```

配信ページを作成

```
vim public/bbsimagetest.php
```

中身のコードは以下のようになります。

```
1  <?php
2  $dbh = new PDO('mysql:host=mysql;dbname=techc', 'root', '');
3
4  if (isset($_POST['body'])) {
5      // POSTで送られてくるフォームパラメータ body がある場合
6
7      $image_filename = null;
8      if (isset($_FILES['image']) && !empty($_FILES['image']['tmp_name'])) {
9          // アップロードされた画像がある場合
10         if (preg_match('/^image\/', mime_content_type($_FILES['image']['tmp_name'])) != 1) {
11             // アップロードされたものが画像ではなかった場合
12             header("HTTP/1.1 302 Found");
13             header("Location: ../bbsimagetest.php");
14         }
15
16         // 元のファイル名から拡張子を取得
17         $pathinfo = pathinfo($_FILES['image']['name']);
18         $extension = $pathinfo['extension'];
19         // 新しいファイル名を決める。他の投稿の画像ファイルと重複しないように時間+乱数で決める。
20         $image_filename = strval(time()) . bin2hex(random_bytes(25)) . '.' . $extension;
21         $filepath = '/var/www/upload/image/' . $image_filename;
22         move_uploaded_file($_FILES['image']['tmp_name'], $filepath);
23     }
24
25     // insertする
26     $insert_sth = $dbh->prepare("INSERT INTO bbs_entries (body, image_filename) VALUES (:body, :image_filename)");
27     $insert_sth->execute([
28         ':body' => $_POST['body'],
29         ':image_filename' => $image_filename,
30     ]);
```

```

31
32 // 処理が終わったらリダイレクトする
33 // リダイレクトしないと、リロード時にまた同じ内容でPOSTすることになる
34 header("HTTP/1.1 302 Found");
35 header("Location: ../bbsimagetest.php");
36 return;
37 }
38
39 // いままで保存してきたものを取得
40 $select_sth = $dbh->prepare('SELECT * FROM bbs_entries ORDER BY created_at DESC');
41 $select_sth->execute();
42 ?>
43 <head>
44 <title>画像投稿できる掲示板</title>
45 </head>
46
47 <!-- フォームのPOST先はこのファイル自身にする -->
48 <form method="POST" action=" ../bbsimagetest.php" enctype="multipart/form-data">
49 <textarea name="body"></textarea>
50 <div style="margin: 1em 0;">
51 <input type="file" accept="image/*" name="image" id="imageInput">
52 </div>
53 <button type="submit">送信</button>
54 </form>
55
56 <hr>
57
58 <?php foreach($select_sth as $entry): ?>
59 <dl style="margin-bottom: 1em; padding-bottom: 1em; border-bottom: 1px solid #ccc;">
60 <dt>ID</dt>
61 <dd><?=$entry['id'] ?></dd>
62 <dt>日時</dt>
63 <dd><?=$entry['created_at'] ?></dd>
64
65 <dt>内容</dt>
66 <dd>
67 <?= nl2br(htmlspecialchars($entry['body'])) // 必ず htmlspecialchars() すること ?>
68 <?php if(!empty($entry['image_filename'])): // 画像がある場合は img 要素を使って表示 ?>
69 <div>
70 
71 </div>
72 <?php endif; ?>
73 </dd>
74 </dl>
75 <?php endforeach ?>
76
77 <script>
78 document.addEventListener("DOMContentLoaded", () => {
79 const imageInput = document.getElementById("imageInput");
80 imageInput.addEventListener("change", () => {
81 if (imageInput.files.length < 1) {
82 // 未選択の場合
83 return;
84 }
85 if (imageInput.files[0].size > 5 * 1024 * 1024) {
86 // ファイルが5MBより多い場合
87 alert("5MB以下のファイルを選択してください。");
88 imageInput.value = "";
89 }
90 });
91 </script>

```

6. MY SQL

WEB ページで使うデータベースを構築する

文字や画像をデータベースに登録して読み込ませる

コードは以下のようになります

```
CREATE TABLE `bbs_entries` (  
  `id` INT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,  
  `body` TEXT NOT NULL,  
  `created_at` DATETIME DEFAULT CURRENT_TIMESTAMP  
);  
  
ALTER TABLE `bbs_entries` ADD COLUMN image_filename TEXT DEFAULT NULL;
```