

A Web-Based Meditation Timer

Enhancing Mindfulness Through Technology

Date :- 13th May 2025

Team Name :- Code Warriors

Team Lead

Aathira V Sajive

Vrushali Mujumdar

Developers

Prasad Gujarathi

Vaishnavi Dhobale

Pooja Kothawade

Cloud Engineers

Abhishek Choudhary

Gaurav Chaudhari

Aadheesh Phade

Documentation

Pramod Patil (Developer)

Prajakta Wanjari (Developer)

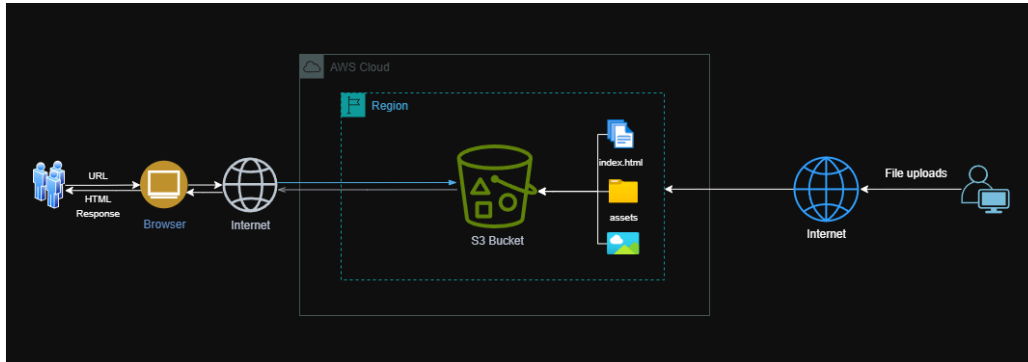
Parth Satish (Cloud Engineer)

Application Link: <http://codewarriorshackathon.s3-website.ap-south-1.amazonaws.com/>

GitHub Link: <https://github.com/VaishnaviDhobale/Code-Warriors>

Project Overview

The Mindfulness Timer is a web application designed to help users practice mindfulness and meditation through timed sessions. Users can choose predefined time durations and receive a gentle bell sound when the timer ends. This app aims to promote mental well-being and encourage brief, meaningful breaks.



Tech Stack

Frontend: HTML, CSS, JavaScript

Cloud Hosting: AWS S3 (Static Website Hosting)

Tools: Visual Studio Code, GitHub, AWS Management Console

Hosting & Deployment

The project is deployed using Amazon S3 static website hosting.

All static assets are served directly from S3.

Permissions and policies were configured to allow secure public access.

Implementation Details

- JavaScript's `setTimeout()` and `setInterval()` for timing logic
- `LocalStorage` for saving preference



Coding documentation

Index.html file

```
index.html x index.js style.css
index.html > html > body > div.container > header
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Mindfulness Timer</title>
6   <link rel="stylesheet" href="style.css">
7 </head>
8 <body>
9   <div class="container">
10     <header>
11       <h1>Mindfulness Timer</h1>
12     </header>
13     <main class="main-content">
14       <div id="timerName" class="timer-name">Timer Name (Topic)</div>
15       <div id="timerDisplay" class="timer-display">00:00:00</div>
16       <div class="timer-buttons">
17         <button id="stopBtn">Stop</button>
18         <button id="startBtn">Start</button>
19         <button id="pauseBtn">Pause</button>
20         <button id="resetBtn">Reset</button>
21       </div>
22     </main>
23     <aside class="controls">
24       <div class="input-group">
25         <label for="inputName">Set Timer Name</label>
26         <input type="text" id="inputName" placeholder="Enter Name">
27       </div>
28       <div class="input-group">
29         <label>Set Duration (HH:MM:SS)</label>
30         <input type="number" id="inputHours" placeholder="Hours">
31         <input type="number" id="inputMinutes" placeholder="Minutes">
32         <input type="number" id="inputSeconds" placeholder="Seconds">
33       </div>
34       <button id="setTimerBtn" class="create-btn">Create</button>
35     </aside>
36   </div>
37   <script src="index.js"></script>
38 </body>
39 </html>
```

Javascript files

```
index.html x index.js x style.css
index.js > ...
1 const timerDisplay = document.getElementById("timerDisplay");
2 const inputName = document.getElementById("inputName");
3 const inputHours = document.getElementById("inputHours");
4 const inputMinutes = document.getElementById("inputMinutes");
5 const inputSeconds = document.getElementById("inputSeconds");
6 const timerNameDisplay = document.getElementById("timerName");
7
8 let timerName = '';
9 let duration = 0;
10 let remaining = 0;
11 let isPaused = false;
12 let interval = null;
13
14 function setTimer(){
15     timerName = inputName.value.trim();
16     const hours = parseInt(inputHours.value) || 0;
17     const minutes = parseInt(inputMinutes.value) || 0;
18     const seconds = parseInt(inputSeconds.value) || 0;
19
20     duration = (hours * 3600) + (minutes * 60) + seconds;
21     remaining = duration;
22
23     if (timerName === '' || duration <= 0) {
24         alert("Please enter a valid timer name and time.");
25         return;
26     }
27
28     timerNameDisplay.textContent = timerName;
29     updateDisplay();
30 }
31
32 function startTimer(){
33     if (remaining <= 0 || interval) return;
34
35     interval = setInterval(() => {
36         if (!isPaused) {
37             remaining--;
38             updateDisplay();
39         }
40     }, 1000);
41 }
```

Stylesheet(CSS) Files

```
index.html x index.js x style.css
style.css > .main-content
1 body {
2     margin: 0;
3     font-family: 'Comic Sans MS', cursive, sans-serif;
4     background-color: #111;
5     color: #c8a2ff;
6 }
7
8 .container {
9     display: grid;
10    grid-template-columns: 1fr 3fr 1fr;
11    grid-template-rows: auto 1fr;
12    gap: 20px;
13    padding: 20px;
14    height: 100vh;
15    box-sizing: border-box;
16    border: 2px solid #c8a2ff;
17    border-radius: 20px;
18 }
19
20 header {
21     grid-column: 1 / -1;
22     text-align: center;
23     padding: 10px 20px;
24     color: #ffb3b3;
25     font-size: 1.5em;
26 }
27
28 .main-content {
29     display: flex;
30     flex-direction: column;
31     align-items: center;
32     justify-content: center;
33     gap: 20px;
34 }
35
36 .timer-name {
37     border: 2px solid #c8a2ff;
38     border-radius: 15px;
39     padding: 10px 20px;
40     text-align: center;
41 }
```

Set Up AWS Environment

Step 1: Creating an S3 bucket in AWS Console.

- Go to Aws Console
- Search for S3
- Create New S3 Bucket
- Select the Bucket type As General purpose.
- Unmark the Block Public Access Settings for this Bucket so that Everyone can access from the URL.
- Now Click on Create Bucket.
- The new Bucket is Now Available.

The screenshot shows the 'Create bucket' page in the AWS S3 console. The breadcrumb navigation at the top reads 'Amazon S3 > Buckets > Create bucket'. The page title is 'Create bucket' with an 'info' icon. Below the title, it states 'Buckets are containers for data stored in S3.' The 'General configuration' section is active, showing the 'AWS Region' as 'Asia Pacific (Mumbai) ap-south-1'. Under 'Bucket type', the 'General purpose' option is selected with a radio button, while 'Directory' is unselected. The 'Bucket name' field contains 'codewarriorshackathon'. Below this, there is a 'Copy settings from existing bucket - optional' section with a 'Choose bucket' button. The 'Object Ownership' section shows 'ACLs disabled (recommended)' selected, with a note that all objects are owned by this account. The 'Object Ownership' summary at the bottom indicates 'Bucket owner enforced'.

Amazon S3 > Buckets > Create bucket

Create bucket info

Buckets are containers for data stored in S3.

General configuration

AWS Region
Asia Pacific (Mumbai) ap-south-1

Bucket type info

☒ **General purpose**
Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

☐ **Directory**
Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

Bucket name info

codewarriorshackathon

Bucket names must be 3 to 63 characters and unique within the global namespace. Bucket names must also begin and end with a letter or number. Valid characters are a-z, 0-9, periods (.), and hyphens (-). [Learn More](#)

Copy settings from existing bucket - optional
Only the bucket settings in the following configuration are copied.

[Choose bucket](#)

Format: s3://bucket/prefix

Object Ownership info

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

☒ **ACLs disabled (recommended)**
All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

☐ **ACLs enabled**
Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

Object Ownership
Bucket owner enforced

Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

☐ **Block all public access**
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

☐ **Block public access to buckets and objects granted through new access control lists (ACLs)**
S3 will block public access permissions applied to newly created buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.

☐ **Block public access to buckets and objects granted through any access control lists (ACLs)**
S3 will ignore all ACLs that grant public access to buckets and objects.

☐ **Block public access to buckets and objects granted through new public bucket or access point policies**
S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.

☐ **Block public and cross-account access to buckets and objects through any public bucket or access point policies**
S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Turning off block all public access might result in this bucket and the objects within becoming public
AWS recommends that you turn on block all public access, unless public access is required for specific and verified use cases such as static website hosting.

☐ I acknowledge that the current settings might result in this bucket and the objects within becoming public.

Bucket Versioning

Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. [Learn more](#)

Bucket Versioning

☒ Disable

☐ Enable

Successfully created bucket "codewarriorshackathon"

To upload files and folders, or to configure additional bucket settings, choose [View details](#).

Account snapshot - updated every 24 hours [All AWS Regions](#)

[View Storage Lens dashboard](#)

Storage lens provides visibility into storage usage and activity trends. Metrics don't include directory buckets. [Learn more](#)

[General purpose buckets](#) | [Directory buckets](#)

General purpose buckets (1/1) [info](#) [All AWS Regions](#)

Buckets are containers for data stored in S3.

[Copy ARN](#) [Empty](#) [Delete](#) [Create bucket](#)

Name	AWS Region	IAM Access Analyzer	Creation date
codewarriorshackathon	Asia Pacific (Mumbai) ap-south-1	View analyzer for ap-south-1	May 18, 2025, 11:29:58 (UTC+05:30)

codewarriorshackathon [info](#)

[Objects](#) | [Properties](#) | [Permissions](#) | [Metrics](#) | [Management](#) | [Access Points](#)

Objects (0) [Copy S3 URI](#) [Copy URL](#) [Download](#) [Open](#) [Delete](#) [Actions](#) [Create folder](#) [Upload](#)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Name	Type	Last modified	Size	Storage class
No objects				
You don't have any objects in this bucket.				

[Upload](#)

Step 2: Add Policy

An S3 bucket policy in AWS is a resource-based policy that grants access permissions to your S3 bucket and the Object within it is the JSON document that defines permission for accessing a specific bucket.

- Click on Edit Bucket Policy
- Click on Policy Generator.
- Select the Type as S3 Bucket Policy.
- In Add Statement Allow the Effect and Enter * in principal
- Enter the ARN which is available in bucket details.

The policy will be generated. Copy the policy and paste it.

Step 3: Public Block Access disable

If the Block Public Access bucket setting is ON By default, then turn it to OFF , so that it will Not Block Any Access Request

Edit bucket policy [info](#)

Bucket policy

The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. [Learn more](#)

Bucket ARN
arn:aws:s3:::codewarriorshackathon

Policy

1

Edit statement

Select a statement

Select an existing statement in the policy or add a new statement.

[+ Add new statement](#)

The AWS Policy Generator is a tool that enables you to create policies that control access to Amazon Web Services (AWS) products and resources. For more information about creating policies, see [key concepts in Using AWS Identity and Access Management](#). Here are [sample policies](#).

Step 1: Select Policy Type

A policy is a container for permissions. The different types of policies you can create are an IAM Policy, an S3 Bucket Policy, an SNS Topic Policy, a VPC Endpoint Policy, and an SQS Queue Policy.

Select Type of Policy S3 Bucket Policy

Step 2: Add Statement(s)

A statement is the formal description of a single permission. See a [description of elements](#) that you can use in statements.

Effect ☒ Allow ☐ Deny

Principal *

Use a comma to separate multiple values.

AWS Service Amazon S3 ☐ All Services (**)

Use multiple statements to add permissions for more than one service.

Actions 1 Action(s) Selected ☐ All Actions (**)

Amazon Resource Name (ARN) arn:aws:s3:::codewarriorshackathon

ARN should follow the following format: arn:aws:s3:::[bucketname]/[keyName].
Use a comma to separate multiple values.

[Add Conditions \(Optional\)](#)

[Add Statement](#)

You added the following statements. Click the button below to Generate a policy.

Principal(s)	Effect	Action	Resource	Conditions
* *	Allow	* s3:GetObject	arn:aws:s3:::codewarriorshackathon/*	None

Step 3: Generate Policy

A policy is a document (written in the [Access Policy Language](#)) that acts as a container for one or more statements.

[Generate Policy](#) [Start Over](#)

For Improvement

While the some functionality was successfully implemented, the following areas are identified for future improvement:

1. Add Persistent Data Storage

Currently, no user data is saved (e.g., timer history or preferences).
Future versions can integrate a database (like AWS DynamoDB).

2. Save custom timers

Track daily meditation/study logs
Personalize user experience

3. Deployment Upgrades

Add CDN (via CloudFront) for faster global access
Enable HTTPS using AWS Certificate Manager and custom domain via Route 53

Note: Delete the resources once you are done with the project.

THANK YOU