Build a Bulletin Board with PHP & SQLite

A Step-by-Step ”Codelab” for Beginners

What You Will Build

In this tutorial, you will create a functioning ”Bulletin Board” website. We will separate the logic into two pages:

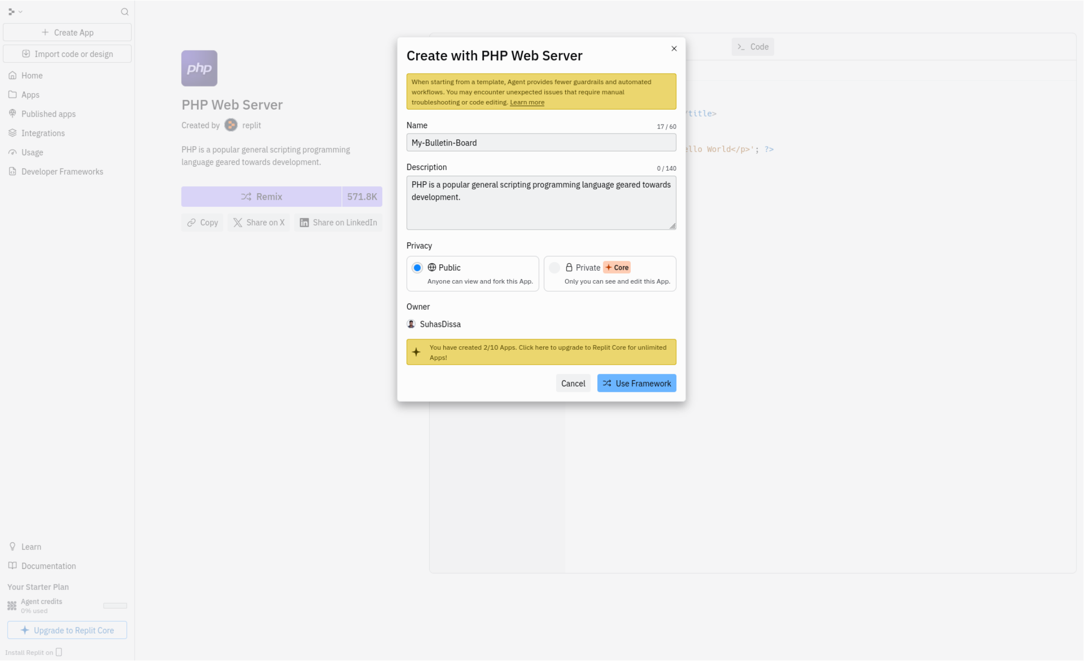
- **Index Page:** Shows the form and the list of messages.
- **Submit Page:** Receives the user’s message, saves it, and shows a success screen.

Contents

1 Setting Up Your Environment (Replit)

We will use **Replit**, a free online coding platform.

1. Go to replit.com and sign up.
2. Create a new Repl and select **PHP Web Server**.
3. Name it ”My-Bulletin-Board” and click **Create**.



PHP Web Server

Created by replit

PHP is a popular general scripting programming language geared towards development.

Create with PHP Web Server

When starting from a template, Agent provides fewer guardrails and automated workflows. You may encounter unexpected issues that require manual troubleshooting or code editing. Learn more

Name 17 / 60

My-Bulletin-Board

Description 0 / 140

PHP is a popular general scripting programming language geared towards development.

Privacy

● Public
Anyone can view and fork this App.

○ Private Core
Only you can see and edit this App.

Owner

SuhasDissa

You have created 2/10 Apps. Click here to upgrade to Replit Core for unlimited Apps!

2 Step 1: The HTML Skeleton (index.php)

First, let's create the main page structure. Delete existing code in `index.php` and add this:

File: index.php

```
<!DOCTYPE html>
<html>
<head>
<title>My Bulletin Board</title>
</head>
<body>
<h1>Welcome to My Bulletin Board</h1>

<hr>

<h2>Recent Posts</h2>
</body>
</html>
```

Action: Click **Run**. You should see your title headers in the Web View.

3 Step 2: Creating the Form

We need a form to collect data. Note the `action` attribute below. It tells the browser: ”When clicked, send this data to `submit.php`”.

Add this code inside the `body` tag (under the h1):

File: index.php

```
<h3>Write a Message</h3>

<form action="submit.php" method="POST">
<label>Name:</label><br>
<input type="text" name="username" required><br><br>

<label>Message:</label><br>
<textarea name="message" required></textarea><br><br>

<button type="submit">Post Message</button>
</form>
```

4 Step 3: The Receiver Page (submit.php)

Now, create a **new file** in the file list called `submit.php`. This page's job is to catch the data sent by the form.

For now, let's just make sure we can receive the data. Add this to `submit.php`:

File: submit.php

```
<?php
// Get the data sent from the form
$incoming_name = $_POST['username'];
$incoming_msg = $_POST['message'];
?>
```

```
<h1>Success!</h1>
<p>Thanks for posting, <strong><?php echo $incoming_name; ?></strong>.</p>
<p>You said: <?php echo $incoming_msg; ?></p>

<a href="index.php">Go back home</a>
```

Action: Run the project. Fill out the form on the index page and click submit. You should be taken to the new page showing your message.

5 Step 4: Connecting the Database

Now that the pages are connected, let's save the data permanently using SQLite.

5.1 Setup the Database

We need to connect to the database on **both** pages.

Top of index.php: Add this at the very first line (Line 1):

Add to top of index.php

```
<?php
$db = new PDO('sqlite:database.db');
$db->exec("CREATE TABLE IF NOT EXISTS posts (
id INTEGER PRIMARY KEY,
username TEXT,
message TEXT
)");
?>
```

Top of submit.php: Add the connection line to Line 1 here too:

Add to top of submit.php

```
<?php
$db = new PDO('sqlite:database.db');
// ... rest of your code ...
?>
```

6 Step 5: Saving the Data

Modify `submit.php` to actually save the data instead of just printing it. Update your PHP block:

File: submit.php

```
<?php
$db = new PDO('sqlite:database.db');

$name = $_POST['username'];
$msg = $_POST['message'];

// Prepare an INSERT statement to save data safely
$stmt = $db->prepare("INSERT INTO posts (username, message) VALUES (?, ?)");
$stmt->execute([$name, $msg]);
?>

<h1>Message Saved!</h1>
<a href="index.php">Return to Board</a>
```

7 Step 6: Displaying Posts

Finally, go back to `index.php`. We need to fetch the saved messages and show them.

Add this PHP loop where you want the posts to appear (under `<h2>Recent Posts</h2>`):

File: index.php

```php
<?php
// Get all posts from the DB
$stmt = $db->query("SELECT * FROM posts ORDER BY id DESC");
$posts = $stmt->fetchAll();

foreach ($posts as $post) {
    echo "<div class='post'>";
    echo "<h3>" . htmlspecialchars($post['username']) . "</h3>";
    echo "<p>" . htmlspecialchars($post['message']) . "</p>";
    echo "</div>";
}
?>
```

8 Step 7: Making it Pretty (CSS)

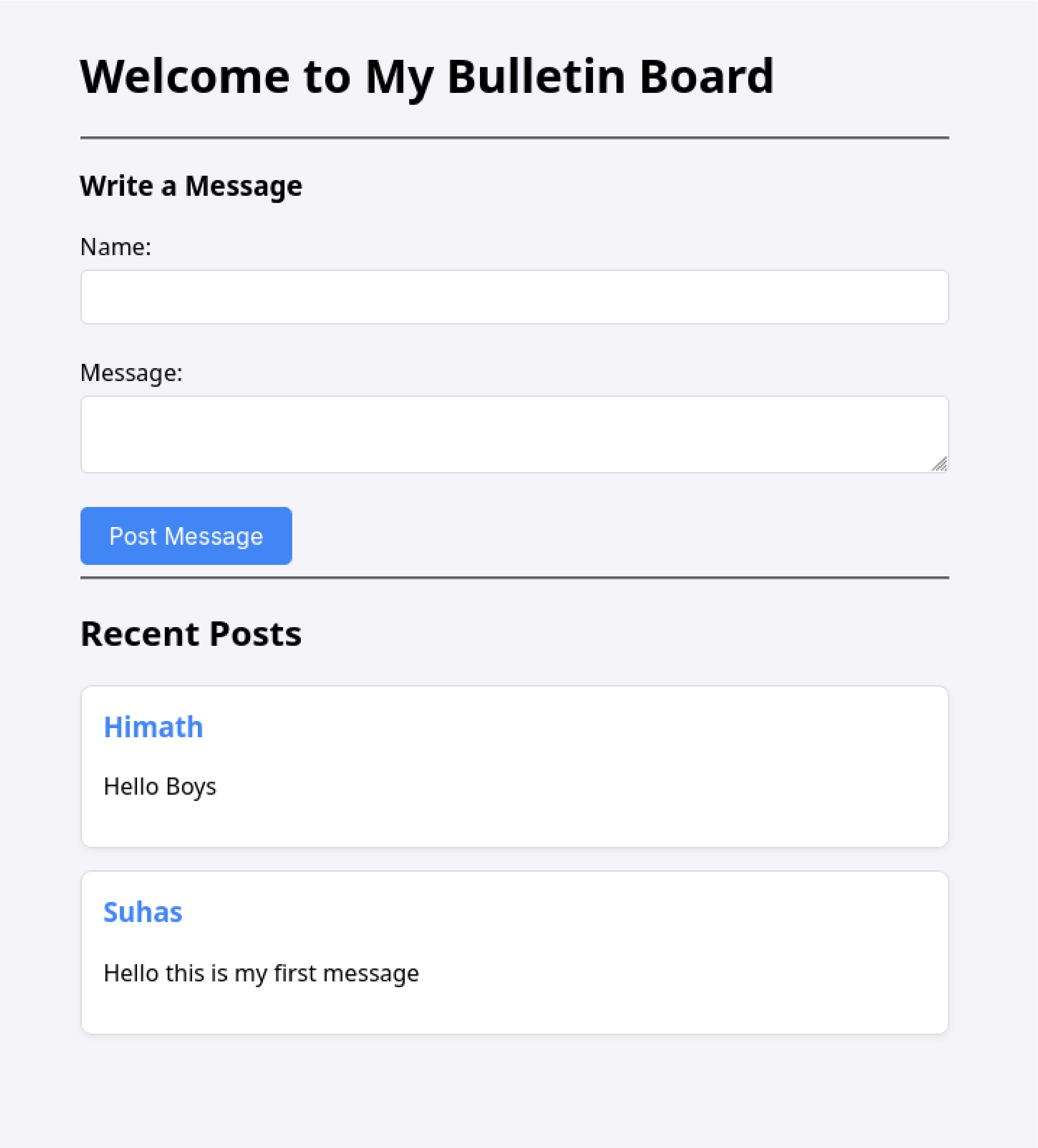
Right now the site looks very plain. Let's add some CSS to make it look professional.

Add this code block inside the `<head>` section of `index.php`:

File: index.php (inside ¡head¿)

```
<style>
body {
    font-family: sans-serif;
    max-width: 600px;
    margin: 2rem auto;
    padding: 0 1rem;
    background-color: #f4f4f9;
}

/* Form Styling */
input, textarea {
    width: 100%;
    padding: 10px;
    margin-top: 5px;
    border: 1px solid #ddd;
    border-radius: 5px;
    box-sizing: border-box;
}

button {
    background-color: #4285F4;
    color: white;
    padding: 10px 20px;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    font-size: 16px;
}

button:hover { background-color: #3367D6; }

/* Post Card Styling */
.post {
    background: white;
    border: 1px solid #ddd;
    padding: 15px;
    margin-bottom: 15px;
    border-radius: 8px;
    box-shadow: 0 2px 4px rgba(0,0,0,0.05);
}

.post h3 { margin-top: 0; color: #4285F4; }
</style>
```

Action: Refresh the page. Your bulletin board should now look professional and polished!



Welcome to My Bulletin Board

Write a Message

Name:

Message:

Recent Posts

Himath

Hello Boys

Suhas

Hello this is my first message

9 Conclusion

Congratulations! You have built a full-stack PHP application.

- You created a Form in HTML.
- You handled the request in a separate PHP file.
- You saved data to a SQLite database.
- You styled the result with CSS.