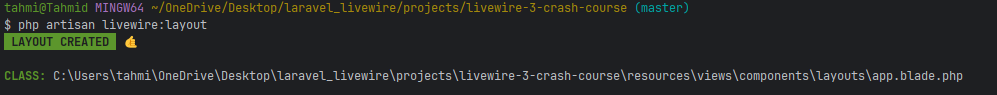
Step 1: create a livewire layout,

php artisan livewire:layout

This will create a livewire layout under components/layouts/app.blade.php



Step 2: add css to app.blade.php,

@vite('resources/css/app.css')

Full snippet,

<!DOCTYPE html>  
<html lang="{{ str\_replace('\_', '-', app()->getLocale()) }}">  
<head>  
 <meta charset="utf-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
  
 <title>{{ $title ?? 'Page Title' }}</title>  
 @vite('resources/css/app.css')  
</head>  
<body>  
<nav>  
 <div>  
 <h1>My Book Reviews</h1>  
 <a  
 @class(['active' => request()->is('/')])  
 href="/">  
 Book List  
 </a>  
 <a  
 @class(['active' => request()->is('create')])  
 href="/create">  
 Add a Book  
 </a>  
 </div>  
</nav>  
<main>  
 {{ $slot }}  
</main>  
</body>  
</html>

* Ekhane slot hocche child element gula ke render kore, and full **app.blade.php** hocche parent element er moto kaj kore.

Step 3: Routing er jonno PHP file ta ke target korte hobe, not the blade file

<?php  
  
use Illuminate\Support\Facades\Route;  
use App\Livewire\BookList;  
  
Route::get('/', BookList::class);

### **Changing the page title from BookList.php**,

#[Title('Book List - Home')]

Full code snippet,

<?php  
  
namespace App\Livewire;  
  
use App\Models\Book;  
use Livewire\Component;  
use Livewire\Attributes\Title;  
  
class BookList extends Component  
{  
 public $name = "Mario";  
  
 #[Title('Book List - Home')]  
  
 public function deleteBook(Book $book)  
 {  
 $book->delete();  
 }  
  
 public function render()  
 {  
 return view('livewire.book-list',  
 [  
 'books' => Book::*all*(),  
 ]);  
 }  
}

### **Using a different master layout,**

Add this to the **php** file, please note that, slot in master thik moto thakte hobe

#[Layout('components.layouts.second')]

Code snippet,

class CreateBook extends Component  
{  
 #[Layout('components.layouts.second')]  
 public function render()  
 {  
 return view('livewire.create-book');  
 }  
}

**### Nested components**

Create a new livewire component, for an example,

php artisan make:livewire page-header

Now use simply like components in livewire views, a view from book-list.blade.php,

<div>  
  
 {{-- livewire component --}}  
 <livewire:page-header></livewire:page-header>  
  
 <br>  
  
 <ul class="list">  
 @foreach($books as $book)  
 <li wire:key="{{ $book->id }}">  
 <h3>{{ $book->title }}</h3>  
 <h4>{{ $book->author }}</h4>  
 <p>Rating: {{ $book->rating }}/10</p>  
 <button wire:click="deleteBook({{ $book->id }})">Delete</button>  
 </li>  
 @endforeach  
 </ul>  
</div>

**### Passing data to components via props**

Suppose we need to pass props data to **page-header.blade.php** from **book-list.blade.php**.

Step 1: from book-list.blade.php,

<livewire:page-header name="Mario"></livewire:page-header>

Step 2: from PageHeader.php,

public $name;  
  
public function mount($name)  
{  
 $this->name = $name;  
}

Now the variable **$name** will be available in page-header.php like **props** data. {{ $name }}.