Yes 👏 — that is **exactly the right and professional way** to structure your Laravel app.

Let’s break it down clearly so you can be confident you’re doing this in a clean, scalable way 👇

**🧱 Recommended Architecture**

**🔹 1. Portal (Frontend / Public Area)**

👉 For **Guests**, **Candidates**, and **Employers**  
This includes pages like:

* Home (/)
* Job listings
* Apply for a job
* Employer job posting
* Candidate resume creation
* Profile pages  
  …and so on.

📂 Example structure:

**resources/views/portal/**

**│**

**├── layouts/**

**│ ├── app.blade.php**

**│ ├── navbar.blade.php**

**│ └── footer.blade.php**

**│**

**├── home.blade.php**

**├── jobs/**

**│ ├── index.blade.php**

**│ ├── show.blade.php**

**│ └── create.blade.php**

**└── resumes/**

**├── index.blade.php**

**└── create.blade.php**

**🔹 2. Admin Dashboard (Backend Panel)**

👉 For **Admin only** — to manage everything:

* Users (candidates, employers)
* Jobs
* Applications
* Reports
* Settings

📂 Example structure:

**resources/views/admin/**

**│**

**├── layouts/**

**│ ├── app.blade.php**

**│ ├── sidebar.blade.php**

**│ └── header.blade.php**

**│**

**├── dashboard.blade.php**

**├── users/**

**│ ├── index.blade.php**

**│ └── edit.blade.php**

**└── jobs/**

**├── index.blade.php**

**└── edit.blade.php**

So basically:

|  |  |  |  |
| --- | --- | --- | --- |
| Role | Accesses | Example Routes | Interface Source |
| Guest | Public pages | /, /jobs, /contact | resources/views/portal |
| Candidate | Portal | /resume, /applications | resources/views/portal |
| Employer | Portal | /employer/dashboard, /jobs/create | resources/views/portal |
| Admin | Admin Dashboard | /admin/dashboard, /admin/users | resources/views/admin |

**🔒 3. Access Control (Laravel Middleware)**

Use middleware to make sure only the right users see the right areas.

**In app/Http/Middleware/CheckRole.php:**

**public function handle($request, Closure $next, ...$roles)**

**{**

**if (!in\_array(auth()->user()->role, $roles)) {**

**abort(403, 'Unauthorized.');**

**}**

**return $next($request);**

**}**

**Register it in Kernel.php:**

**'role' => \App\Http\Middleware\CheckRole::class,**

**Then protect your routes:**

**// Admin Routes**

**Route::middleware(['auth', 'role:admin'])->prefix('admin')->group(function () {**

**Route::get('/dashboard', [AdminController::class, 'dashboard'])->name('admin.dashboard');**

**});**

**// Employer Routes**

**Route::middleware(['auth', 'role:employer'])->group(function () {**

**Route::get('/jobs/create', [JobController::class, 'create'])->name('jobs.create');**

**});**

**// Candidate Routes**

**Route::middleware(['auth', 'role:candidate'])->group(function () {**

**Route::get('/resume', [ResumeController::class, 'index'])->name('resume.index');**

**});**

**🧩 4. Result**

✅ Clean separation of areas  
✅ Better security  
✅ Easier maintenance  
✅ Scalable — you can add new roles or features later without chaos

If you’d like, I can:

1. Generate a **ready Laravel route + middleware setup** for these roles, and
2. Build a **unified layout system** (portal layout + admin layout) with reusable components.

Would you like me to prepare that structure (middleware + route setup + layout folders)?