Laravel E-Signature Project — Full Scaffold

Language: Laravel (PHP) Saurav Dahiya

Goal: Single Laravel application with two roles (Admin / User). Users can upload documents and add signatures (drag & drop or draw). Admin can approve/cancel documents. Dashboards for both roles with graphs and document lists.

Features (as requested)

- Roles: `admin`, `user` (role field on users table)
- Authentication: Laravel Breeze or Fortify (simple auth scaffolding)
- Admin sidebar: Dashboard (graphs), Categories: Documents → Pending / Approved / Cancelled, User Management (Create / List / Edit / Delete)
- User sidebar: Dashboard (report/graph), Upload Document, Document Approved, Document Pending, Add Document
- Header: Logout option
- Document flow:
- User uploads document (PDF / image / Word — convert to PDF via LibreOffice or accept PDF only)
- On document view, user can place signature(s) by: drag & drop a pre-saved signature image, or draw a signature using HTML5 canvas and drop it onto the document overlay
- Signatures are saved as coordinates + scale + rotation on the document (so they can be rendered on the PDF when viewing or printing)
- Admin can `approve`, `cancel` documents Saurav Dahiya
- Users can check status (Pending / Approved / Cancelled)

Tech stack & dependencies

- Laravel 10+
- PHP 8.1+
- MySQL / MariaDB
- Composer packages suggested:
- `laravel/breeze` or `laravel/ui`
- `barryvdh/laravel-dompdf` (generate PDFs with signatures applied)
- `spatie/laravel-permission` (optional, for roles/permissions)
- `ramsey/uuid` (optional for filenames)
- Frontend: Blade + Tailwind (you can use Bootstrap if preferred)
- JS: PDF.js (for rendering PDF pages inside the browser), Fabric.js (optional) or a simple overlay with absolute-positioned canvases for drag/drop

---

Database schema (important tables)

users

- id
- name
- email
- password
- role ENUM('admin','user') default 'user'
- timestamps

documents

- id
- user_id (uploader)
- title
- filename (stored path)
- original_filename
- mime
- status ENUM('pending','approved','cancelled') default 'pending'
- approved_by (nullable) user id
- approved_at (nullable) datetime
- notes (nullable)
- created_at, updated_at

signatures

- id
- user_id (who created the signature image or drawing)
- name
- image_path (signature png/svg)
- created_at

document_signatures (placements)

- id
- document_id
- signature_id (nullable if drawn ad-hoc stored separately)
- placed_by (user id)
- page_number (int)
- x (float) — percentage from left (0..100)
- y (float) — percentage from top (0..100)
- width_percent (float) — width relative to page width
- rotation (float)
- created_at

Migrations (example snippets)

```
// create_documents_table
Schema::create('documents', function (Blueprint $table) {
  $table->id();
  $table->foreignId('user_id')->constrained()->cascadeOnDelete();
  $table->string('title');
  $table->string('filename');
  $table->string('original_filename');
  $table->string('mime');
  $table->enum('status',['pending','approved','cancelled'])->default('pending');
  $table->foreignId('approved_by')->nullable()->constrained('users');
  $table->timestamp('approved_at')->nullable();
  $table->text('notes')->nullable();
  $table->timestamps();
});
```

```
// create_document_signatures_table
Schema::create('document_signatures', function (Blueprint $table) {
  $table->id();
  $table->foreignId('document_id')->constrained()->cascadeOnDelete();
  $table->foreignId('signature_id')->nullable()->constrained()->nullOnDelete();
  $table->foreignId('placed_by')->constrained('users')->cascadeOnDelete();
  $table->integer('page_number')->default(1);
  $table->float('x');
  $table->float('y');
  $table->float('width_percent');
  $table->float('rotation')->default(0);
  $table->timestamps();
});
```

Models

- `User` (with `isAdmin()` helper)
- `Document` (relationship: user, signatures)
- `Signature` (relationship: user)
- `DocumentSignature` (relationship: document, signature)

Add scopes: `scopePending`, `scopeApproved` on Document.

Routes (web.php) — core

```
Route::middleware(['auth'])->group(function(){
  // User routes
  Route::get('/dashboard','DashboardController@index')->name('dashboard');
  Route::get('/documents','DocumentController@index')->name('documents.index');
  Route::get('/documents/create','DocumentController@create')-
>name('documents.create');
  Route::post('/documents','DocumentController@store')->name('documents.store');
  Route::get('/documents/{document}','DocumentController@show')-
>name('documents.show');
  Route::post('/documents/{document}/place-
signature','DocumentController@placeSignature')-
>name('documents.placeSignature');

  // Admin-only
  Route::middleware('is_admin')->group(function(){
    Route::get('/admin','Admin\DashboardController@index');
    Route::resource('admin/users','Admin\UserController');
    Route::get('admin/documents/pending','Admin\DocumentController@pending');
    Route::post('admin/documents/{document}/
approve','Admin\DocumentController@approve');
    Route::post('admin/documents/{document}/
cancel','Admin\DocumentController@cancel');
  });
});
```

`is_admin` middleware checks `auth()->user()->role === 'admin'`.

Controllers — key logic

DocumentController@store

- Validate file (pdf, png, jpg)
- Store file in `storage/app/public/documents/{user_id}/...`
- Create Document record with `status = pending`

DocumentController@show

- Render view with PDF.js canvas rendering the PDF pages
- Overlay an absolutely-positioned HTML element (container) that represents page area
- Fetch existing `document_signatures` for that document and render each signature at calculated pixel positions using stored percentages
- Provide UI to drag & drop a signature image or draw one via canvas

DocumentController@placeSignature (AJAX)

- Accept `{ signature_id | dataURL, page_number, x_percent, y_percent, width_percent, rotation }`
- If dataURL (drawn), save PNG to disk and create a `Signature` record or create placement with `signature_id = null` but store `image_path`
- Create DocumentSignature row
- Return JSON success

Admin\DocumentController@approve

- `document->update(['status'=>'approved','approved_by'=>auth()->id(),'approved_at'=>now()])`
- Optionally render final PDF with signatures burned in using `barryvdh/laravel-dompdf` or external service

---

Frontend ideas (signature UI)

1. Use PDF.js to render PDF pages into `<canvas>` elements.
2. Place a transparent overlay div above each canvas of same size & position with `position:absolute; top:0; left:0;`.
3. When user drags a signature image from the signature-list, create a new absolutely-positioned `<img>` inside overlay and make it draggable/resizable (use interact.js or simple pointer events).
4. On drop, compute percentage coordinates:
5. `x_percent = (left_px / page_width_px) * 100`
6. `y_percent = (top_px / page_height_px) * 100`
7. `width_percent = (img_width_px / page_width_px) * 100`
8. POST to `documents/{id}/place-signature` with values.
9. For drawing: provide a simple canvas where user draws (signature). On save, convert to dataURL and upload similarly (store image and place it as above).

Small example JS pseudocode (drop handler):

```
// on drop
const rect = pageCanvas.getBoundingClientRect();
const left = droppedEl.offsetLeft;
const top = droppedEl.offsetTop;
const x_percent = (left/rect.width)*100;
const y_percent = (top/rect.height)*100;
const width_percent = (droppedEl.offsetWidth/rect.width)*100;
fetch(`/documents/${docId}/place-signature`, {
  method:'POST',
  headers:{ 'Content-Type':'application/json', 'X-CSRF-TOKEN': csrf },
  body: JSON.stringify({ page_number, x: x_percent, y: y_percent,
```

```
width_percent, signature_id })
})
```

Blade structure (suggested)

- `layouts/app.blade.php` — header (logout), sidebar (render based on `auth()->user()->role`), content
- `admin/*` views (user management, doc lists)
- `user/documents/*` (upload form, list, show)

Dashboard graphs

- Use Chart.js to show: total documents, pending, approved, cancelled; per-day uploads; top users by uploads
- Controller returns simple aggregated counts for the charts via JSON or blade-props

---

Seeders

- `UserSeeder` to create an admin account (email: admin@example.com / password: secret) and some sample users
- `DocumentSeeder` to create sample documents

Useful dev commands

```
composer install
cp .env.example .env
php artisan key:generate
# configure DB credentials in .env
php artisan migrate --seed
php artisan storage:link
npm install
npm run dev
php artisan serve
```

Security & production notes

- Validate uploads strictly and scan for harmful files.

- Serve documents from signed temporary URLs if private.
- When burning signatures to PDF ensure the process is tamper-evident (log who approved and when)

Deliverables included in this scaffold (what I prepared here)

1. Database schema & migration examples
2. Models & relationships description
3. All essential routes and controllers explained with example snippets
4. Frontend approach for PDF rendering and signature placement (drop/draw)
5. Admin/user UI structure and list of Blade views to implement
6. Seeders & installation steps

If you want, next steps I can do for you (pick one): - Generate the complete project file tree and provide a downloadable ZIP with working controllers, migrations and basic Blade views (so you can `composer install` and run). - Create a Git repository and push the scaffold (you can give a repo name). - Implement the full drag/drop signature UI using PDF.js + interact.js with working AJAX endpoints (this is more code but I can start and deliver a zip).

Notes: I placed detailed implementation guidance and code examples above so a developer can build this quickly. If you want the full ready-to-run Laravel repo ZIP, tell me and I will prepare it next.