Project Roles & Governance Overview

Core User Roles

- Admin: full-control superuser; can read/manage every resource (clients, brands, tasks, users), seed new managers, view complete dashboards, and access the back-office UI (Clients list, Users, Roles, Settings). Admin-authenticated API routes are wrapped in Spatie permission middleware (RoleMiddleware::class . ':Admin').
- Manager: intended “brand owner.” Managers can create new brands (unlimited now), edit their brands, manage tasks related to their brands, and consume a limited dashboard scoped to the brands they manage. Managers are automatically provisioned whenever a client is created with a contact email; they are assigned the default password password@123 on first creation.
- Team Member / Task Assignee (implicit): standard users (usually assigned to tasks) with constrained task visibility. They do not have direct create/edit access to clients or brands; task view logic checks assignment.

Automatic User Provisioning Workflow

1. Admin creates a new Client via /clients/create.
2. ClientController@store validates the payload; if contact_email is provided, it calls User::firstOrCreate to either provision or reuse a user with that email.
3. The new/existing user receives the Manager role (Spatie) and an auto-generated password password@123 (auto-verified).
4. Managers can immediately log into the platform to create brands; the client show screen exposes “Add Brand” which passes client_id into the brand create route so the brand is linked automatically.

Client/Brand Hierarchy

- A Client owns many Brands; this relationship is enforced by brands.client_id (nullable for legacy). The client detail page lists linked brands. Brand creation/edit forms require the client selector.
- Brands track optional metadata: description, target audience, started date, additional files (logo/guidelines). Managers and admins can edit these attributes (team members cannot).
- The rest of the workflow (tasks, updates, statistics) reference brands through brand_id.

Brand Permissions & Forms

- Manager assignment dropdown is intentionally hidden from the Brand create/edit forms; managers are tied implicitly via provisioning or edits from admin UI.
- Form submission strips manager_id from payloads (server ignores the field now).
- Create/edit uses Inertia forms that drop file fields from the request unless a new file is uploaded, preventing validation errors when editing.

Dashboard Metrics Logic

- Dashboard metrics use DashboardMetricsService, which scopes data to:

- Recent activity merges task changes and brand updates; now uses a raw DB::table('brands') query to avoid caching Eloquent models (prevents “getKey on array”).

Route & Policy Controls

- Spatie roles and policies guard controller methods:

- Inertia layout (AuthenticatedLayout) gates Clients menu to admins only; managers see brand/task options but not client management.

Data Entry Requirements

- Clients require name, status, and optionally website/contact info. Validation ensures contact email is valid before provisioning the manager account.
- Brands require name, status, and linked client ID. Additional fields (audience, details, logo, guidelines) are optional but stored when present.

Default Credentials & Environment Notes

- Auto-provisioned managers get password@123; encourage immediate password change.
- .env / env files should set up mail, queue, and storage for file uploads and notifications (not covered here but ensure storage symlink exists for logos/guidelines).

Safety & Password Handling

- Passwords use Laravel’s Hash::make; ensure password@123 is only a provisioning default; communicate to new managers to update via profile settings.
- Audit logs (recent activity) track actor names for accountability; relies on created_by fields being populated (brand/task creation should set the current user ID).