PRITECH – Laravel Technical Task (Junior/Mid)

What you'll build

A Mini Issue Tracker where a small team can manage projects, issues, tags, and comments.

Core Entities & Relationships

Implement at least these entities and relationships:

- Project – Fields: name, description. Relationship: Project has many Issues (one-to-many).
- Issue – Fields: project_id, title, description, status (open|in_progress|closed), priority (low|medium|high), due_date (nullable). Relationships: Issue belongs to Project, Issue has many Comments (one-to-many), Issue belongs to many Tags (many-to-many via issue_tag).
- Tag – Fields: name (unique), color (nullable). Relationship: Tag belongs to many Issues (many-to-many).
- Comment – Fields: issue_id, author_name, body. Relationship: Comment belongs to Issue.
- *Add two new columns(start_date, deadline) with a new migration to the 'Project' table.*

Requirements

Functional

- Projects – List, create, edit, delete. Show a project with its issues.
- Issues – List (with filters by status, priority, and by tag). Create, edit, delete. Show issue detail (with tags and comments).
- Tags – Create new tags (unique name) and list them. Attach/detach tags to an issue via AJAX (no full page reload).
- Comments – On the issue detail page, load comments via AJAX (paginated). Add a new comment via AJAX (validate author_name and body).

UI/UX (Blade + AJAX)

- Use Blade templates (layouts/partials) and JavaScript for interactions.
- Use a modal or inline form to attach/detach tags on an issue without reloading.
- When adding a comment via AJAX, prepend it to the list and clear the form.
- Provide basic form validation errors on the page (no alert boxes only).

Technical

- Laravel: 10/11+ preferred.
- Routes/Controllers: Use Resource Controllers where appropriate.
- Validation: Use Form Request classes.
- Database: Use migrations, factories, and seeders for demo data.
- Eloquent: Proper relationships + eager loading to avoid N+1 queries.

Bonus (optional)

- Many-to-many with Users (assignment): Allow assigning multiple “members” (users) to an issue via a second pivot issue_user and show them on the issue page (attach/detach via AJAX).
- Authorization: Add simple Policies so only project owners can edit/delete a project (seed a couple users; Breeze or basic auth is fine).
- Search: Text search on issues (title/description) with debounce (AJAX).

Deliverables

- Git history with logical commits.
- Make sure the repo is public.