Rancangan Project — Aplikasi Organisasi

Ringkasan singkat Aplikasi web untuk organisasi/UKM yang menggabungkan: absensi anggota, jadwal pertemuan & event mendatang, serta fitur komunitas (post, like, komentar). Dirancang sebagai MVP yang dapat diperluas.

Isi dokumen

---

1. Tujuan & Scope

Tujuan: Membuat platform internal organisasi untuk memudahkan pengelolaan kehadiran, pengumuman/jadwal, dan interaksi anggota.

Scope MVP (versi awal): - Autentikasi (login, register untuk anggota) - Role: Admin / Pengurus / Anggota - Absensi (check-in per acara/pertemuan) dengan kode unik - Jadwal pertemuan mendatang (list & CRUD oleh pengurus) - Jadwal event mendatang (list & CRUD oleh pengurus) - Community posts (text) dengan like & komentar - Dashboard ringkas

Fitur yang *tidak* dimasukkan di MVP: upload video berat, pencarian full-text lanjutan, notifikasi push real-time, sistem patungan/dana.

---

2. Fitur MVP & fitur tambahan

Fitur Utama (MVP): - Auth: register/login, reset password - Role-based access control - Absensi: buat sesi absensi, anggota check-in (kode QR/unique code atau tombol) - Jadwal: CRUD pertemuan & event, tampilan kalender sederhana - Timeline: posting teks, like, komentar - Profil pengguna

Fitur Tambahan (fase 2+): - Upload foto pada post - Mention / tag user - Notifikasi (in-app / email) - Mod tools (hapus post, ban user) - Statistik kehadiran & ekspor CSV

---

3. Arsitektur & Tech Stack (rekomendasi)

Pilihan stack (direkomendasikan cepat & mudah): - Backend: **Laravel (PHP)** atau **Node.js + Express** - Frontend: **Blade + Bootstrap** (Laravel) atau **React** (jika ingin SPA) - DB: **MySQL / MariaDB** - Storage: filesystem atau S3 (untuk file di fase 2) - Autentikasi: JWT (SPA) atau session-based (monolith server-rendered)

Alasan: Laravel mempercepat development (auth scaffolding, migrations, policies). MySQL umum dan mudah deploy.

Arsitektur ramah deploy: monolithic (backend + DB) untuk MVP.

---

4. ERD & Skema Database

Tabel utama dan kolom penting (pakai tipe SQL umum):

users

```
id INT AUTO_INCREMENT PRIMARY KEY
name VARCHAR(100)
username VARCHAR(50) UNIQUE
email VARCHAR(150) UNIQUE
password VARCHAR(255)
role_id TINYINT -- FK ke roles
avatar VARCHAR(255) NULL
created_at TIMESTAMP
updated_at TIMESTAMP
```

roles

```
id TINYINT PRIMARY KEY
name VARCHAR(50) -- 'admin','pengurus','anggota'
```

absensi_sessions

```
id INT AUTO_INCREMENT PRIMARY KEY
title VARCHAR(150)
description TEXT NULL
```

```
starts_at DATETIME
ends_at DATETIME NULL
location VARCHAR(255) NULL
code VARCHAR(20) -- unique code untuk check-in
created_by INT -- FK users
created_at TIMESTAMP
```

attendances

```
id INT AUTO_INCREMENT PRIMARY KEY
session_id INT -- FK absensi_sessions
user_id INT -- FK users
status ENUM('present','absent','excused') DEFAULT 'present'
checked_at DATETIME
method ENUM('code','qr','manual')
notes TEXT NULL
```

schedules (gabungan meeting & event)

```
id INT AUTO_INCREMENT PRIMARY KEY
title VARCHAR(150)
description TEXT
start_at DATETIME
end_at DATETIME NULL
type ENUM('meeting','event')
location VARCHAR(255) NULL
created_by INT
```

posts

```
id INT AUTO_INCREMENT PRIMARY KEY
user_id INT
content TEXT
is_public BOOLEAN DEFAULT TRUE
created_at TIMESTAMP
updated_at TIMESTAMP
```

post_likes

```
id INT AUTO_INCREMENT PRIMARY KEY
post_id INT
user_id INT
```

```
created_at TIMESTAMP
UNIQUE(post_id,user_id)
```

post_comments

```
id INT AUTO_INCREMENT PRIMARY KEY
post_id INT
user_id INT
content TEXT
created_at TIMESTAMP
```

migrations & indexes

- index pada posts.created_at, schedules.start_at, absensi_sessions.starts_at
- FK constraints for data integrity

SQL create table (ringkas, contoh MySQL)

```
-- roles
CREATE TABLE roles (
  id TINYINT PRIMARY KEY,
  name VARCHAR(50) NOT NULL
);

-- users
CREATE TABLE users (
  id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(100) NOT NULL,
  username VARCHAR(50) UNIQUE,
  email VARCHAR(150) UNIQUE,
  password VARCHAR(255) NOT NULL,
  role_id TINYINT,
  avatar VARCHAR(255),
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  FOREIGN KEY (role_id) REFERENCES roles(id)
);

-- posts
CREATE TABLE posts (
  id INT AUTO_INCREMENT PRIMARY KEY,
  user_id INT NOT NULL,
  content TEXT NOT NULL,
  is_public TINYINT(1) DEFAULT 1,
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
```

```
  updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  FOREIGN KEY (user_id) REFERENCES users(id)
);
```

(Lanjutan untuk tabel lainnya pada implementasi)

---

5. API Endpoints (contoh RESTful)

Autentikasi: `/api/auth/register`, `/api/auth/login`, `/api/auth/logout`

Users - `GET /api/users/me` — profil - `PUT /api/users/me` — update profil

Absensi - `POST /api/attendance/sessions` — buat session (pengurus/admin) - `GET /api/attendance/sessions` — list session (filter upcoming) - `POST /api/attendance/sessions/:id/checkin` — check-in (body: code) - `GET /api/attendance/sessions/:id/attendees` — list kehadiran (pengurus)

Schedules - `GET /api/schedules?type=meeting` — list upcoming - `POST /api/schedules` — buat jadwal (pengurus) - `PUT /api/schedules/:id` — edit - `DELETE /api/schedules/:id`

Posts - `GET /api/posts` — timeline (pagination) - `POST /api/posts` — create post - `POST /api/posts/:id/like` — like/unlike - `POST /api/posts/:id/comment` — add comment - `GET /api/posts/:id/comments`

Search / filters - `GET /api/posts?user=12` — posts by user

Notes: Gunakan pagination pada timeline (`page`, `limit`).

---

6. Halaman UI + Wireframe (teks)

Berikut halaman inti dan elemen penting.

1) Login / Register

- field email/username, password, tombol login
- link lupa password

2) Dashboard (setelah login)

- ringkasan: next meeting, next event, recent posts
- shortcut: Absen sekarang, Buat jadwal, Buat post

3) Halaman Absensi

- list session mendatang & tombol "Check-in"
- halaman detail session: list attendees, tombol export CSV (pengurus)
- form buat session: title, waktu mulai, kode otomatis/atur manual

4) Halaman Jadwal (kalender list)

- tampilan list & kalender bulan (simple)
- tombol buat jadwal

5) Timeline / Community

- feed cards: user (avatar + name) — content — like, comment, timestamp
- modal komentar untuk membaca/menambah komentar

6) Profil pengguna

- foto, nama, bio, list posts user, statistik kehadiran

---

7. Struktur Folder (contoh Laravel)

```
/project-root
  /app
    /Http
      /Controllers
        AuthController.php
        PostController.php
        AttendanceController.php
        ScheduleController.php
  /database
    /migrations
    /seeders
  /resources
    /views
      /layouts
      dashboard.blade.php
      posts/index.blade.php
  /public
  /routes
    web.php
    api.php
```

Alternatif (React + Node): client/ server/ shared/

---

8. Milestone & Task Breakdown (per fitur)

Milestone 1 — Setup & Auth - init repo, setup Laravel - DB migration roles & users - Auth endpoints & UI

Milestone 2 — Absensi - CRUD session - Check-in flow (code) - Attendance listing & export

Milestone 3 — Jadwal - CRUD schedules - UI list & calendar

Milestone 4 — Community - Posts, likes, comments - Timeline UI

Milestone 5 — Polish - Access control - Input validation - Tests

---

9. Keamanan & Validasi

- Hash password (bcrypt)
- CSRF protection (web forms)
- Sanitasi content (escape output -> cegah XSS)
- Rate limit untuk endpoint posting/like
- Role checks (policy/gates di Laravel)
- Validasi file size & type jika menambah upload

---

10. Testing & Acceptance Criteria

Contoh acceptance criteria untuk Absensi: - Pengurus dapat membuat session -> muncul di list - Anggota bisa check-in dengan code -> tercatat di attendances - Admin dapat melihat semua records dan export CSV

Testing: unit test model, integration test API, manual UI tests.

---

11. Deployment & Operasional

- Deploy: shared hosting (PHP) atau VPS (Laravel + Nginx + MySQL)
- Backup DB rutin, schedule cron untuk maintenance
- SSL certificate (Let's Encrypt)

---

12. Next steps (apa yang bisa aku kerjakan sekarang)

- Aku bisa buatkan: ERD lengkap + SQL dump untuk import
- Atau scaffold project (Laravel) dengan migration + seeder awal
- Atau wireframe visual (Figma-like) dalam bentuk markdown + gambar

---

Lampiran: Contoh endpoint check-in (pseudo)

```
POST /api/attendance/sessions/:id/checkin
Body: { code: "ABCD123" }
Flow:
 - auth middleware
 - find session by id
 - if session.code == body.code && now between starts_at and ends_at -> create
attendance record
 - return success
```

Jika kamu mau, aku bisa langsung **membuat file migration SQL lengkap** dan/atau **scaffold project Laravel** (controller + model + migration + route) sekarang. Pilih salah satu: `SQL dump`, `Scaffold Laravel`, `Wireframe visual`, atau `Rancangan API lengkap`.

Dokumen ini dibuat sebagai rancangan awal. Kita bisa iterasi lagi sesuai preferensi kamu.