# Fixed-Form Registration System – Full Technical Design

## 1. System Overview

This system is a **low–medium scale web application** that allows users to register, log in, and submit one of **12 fixed forms**. Users can view their own submissions and download PDF reports. Admin users can monitor all submissions and view dashboard statistics.

The system prioritizes: - Simplicity - Low operational cost - Clear relational data model - Predictable form & PDF behavior

---

## 2. Architecture Overview

**Architecture Style**: Monolithic Web Application

**High-level Components**: - Web Frontend (SSR or SPA) - Backend API - Relational Database - Background services (email, PDF)

```
Browser
  ↓
Web App (Laravel / Node)
  ↓
Database (PostgreSQL / MySQL)
```

---

## 3. Technology Stack (Recommended)

### Backend (Choose ONE)

**Preferred**: Laravel - PHP 8.x - Laravel Breeze / Fortify (auth) - Laravel Policies (RBAC) - DomPDF / Snappy (PDF)

**Alternative (Node)**: - Fastify or Express - Prisma ORM - Zod validation - JWT or session-based auth

---

### Frontend

Option A (Simplest): - Blade templates - Alpine.js (minimal interactivity)

Option B (SPA-like): - Vue 3 - Axios - Role-based routing

---

### Database

- PostgreSQL **or** MySQL
- Single database instance
- JSON avoided (except logs)

---

### Infrastructure

- VPS (1–2GB RAM)
- Ubuntu 22.04
- Nginx
- SSL (Let's Encrypt)

---

## 4. User Roles & Permissions

### Roles

- **Customer**
- **Admin**

### Permission Matrix

| Action | Customer | Admin |
|---|---|---|
| Register / Login | ✅ | ✅ |
| Submit forms | ✅ | ✅ |
| View own submissions | ✅ | ✅ |
| View all submissions | ❌ | ✅ |
| Download PDF | ✅(own) | ✅(all) |
| Dashboard metrics | ❌ | ✅ |

---

## 5. Database Design (Option A – Single Table)

### users

```
id (PK)
name
email (unique)
password
role (customer | admin)
created_at
updated_at
```

---

### submissions

```
id (PK)
user_id (FK -> users.id)
form_type (ENUM)

-- Common fields
applicant_name
applicant_ic
phone
address

-- Optional fields (nullable)
participant_name
participant_ic
package_type
animal_type
quantity
relationship
notes

created_at
updated_at
```

**Indexes**: - user_id - form_type - created_at

---

## 6. Form Types

Forms are **hard-coded** and validated server-side.

Example ENUM values:

```
aqiqah
qurban
umrah
haji
waqaf_quran
sabil_makanan
pelancongan
```

Each form maps to: - A frontend UI - A validation schema - A PDF template

---

# 7. Backend Modules

## Authentication

- Email + password
- Session-based auth (recommended)
- Password hashing (bcrypt)

## Authorization

- Role-based middleware
- Submission ownership checks

## Submission Handling

- One endpoint per form type
- Shared submission service

## PDF Service

- One template per form
- Generated on-demand
- Stored temporarily or streamed

## Notification Service

- Email on submission success
- Optional admin notification

---

## 8. API / Route Design

### Public

- GET /
- GET /forms

### Auth

- POST /login
- POST /register
- POST /logout

### Customer

- GET /my/submissions
- GET /my/submissions/{id}
- GET /my/submissions/{id}/pdf
- POST /submit/{form_type}

### Admin

- GET /admin/submissions
- GET /admin/submissions/{id}
- GET /admin/dashboard

---

## 9. Data Flow (Submission)

```
User selects form
  ↓
Frontend validation
  ↓
POST /submit/{form}
  ↓
Server validation
  ↓
Insert into submissions table
  ↓
Send email notification
  ↓
Redirect to detail page
```

---

## 10. PDF Generation Flow

```
Request PDF
  ↓
Authorize user/admin
  ↓
Load submission
  ↓
Render template
  ↓
Stream PDF response
```

PDF contents: - Header (Brand, Form Name) - Submission ID & timestamp - Key-value rows

---

## 11. Security Considerations

- HTTPS enforced
- CSRF protection
- Password hashing
- Role-based guards
- IDOR prevention (submission ownership)
- Rate-limit login

---

## 12. Deployment

### Server Setup

- Nginx
- PHP-FPM or Node runtime
- Supervisor (queue workers)

### Backups

- Daily DB dump
- Off-server storage

---

## 13. Cost Estimate (Monthly)

| Item | Cost |
|---|---|
| VPS | RM30–40 |

| Item | Cost |
|---|---|
| Backups | RM5 |
| Email (SES) | RM2–5 |
| Domain | RM4 |
| **Total** | **~RM40–55** |

---

## 14. Scalability Path (Future)

When traffic increases: 1. Move DB to managed service 2. Add Redis cache 3. Separate PDF worker 4. Horizontal scale app

---

## 15. Non-Goals (Intentionally Excluded)

- Dynamic form builder
- Microservices
- Event sourcing
- Kubernetes
- GraphQL

---

## 16. Summary

This design intentionally favors: - Predictability - Maintainability - Low cost - Fast development

It is suitable for **long-term production use** with minimal operational burden.