# 1. Architectural Overview

## 1.1 Layered Structure

* **Presentation Layer** (Frontend): A responsive web UI built with React or similar to render forms, dashboards, and status updates.
* **Application Layer** (Backend): Stateless RESTful services (e.g., Node.js/Express or Laravel) orchestrating business logic and enforcing workflows.
* **Data Layer**: A relational database (PostgreSQL/MySQL) for applicant records and a file store (AWS S3 or equivalent) for uploaded documents.
* **Integration Layer**: Connectors to external systems—payment gateways, email/SMS providers, and existing university systems via APIs.

## 1.2 Service‑Oriented Design

Each module is exposed as a lightweight service to allow independent scaling, replacement, or reuse in other contexts (e.g., mobile app or partner portals).

# 2. Functional Modules

## 2.1 User Management

* **Registration & Authentication**: Sign‑up, email verification, password reset using JWT tokens for session management.
* **Role-Based Access Control**: Define roles (applicant, admissions officer, admin) and guard routes accordingly.

## 2.2 Application Processing

* **Dynamic Forms**: Multi‑step forms capturing personal details, academic history, and program choices, with client‑ and server‑side validation.
* **Workflow Engine**: Tracks each application through states (draft → submitted → under review → decision).

## 2.3 Document Management

* **Secure Upload**: Applicants upload transcripts, IDs, and other documents; files are virus‑scanned and stored with unique identifiers.
* **Versioning & Retrieval**: Maintain history of uploads and allow admissions officers to view/download.

## 2.4 Status Tracking & Notifications

* **Applicant Dashboard**: Real‑time status updates, missing‑document alerts, and submission timestamps.
* **Notification Service**: Automated emails/SMS for key events (submission received, decision posted) integrated via SendGrid/Twilio.

# 3. Data Flow & Integration

1. **Applicant Submits Form** → Frontend calls POST /applications → Backend validates and stores data.
2. **File Upload** → Frontend streams file to /documents → Service saves to S3 and records metadata in DB.
3. **Status Change** → Admissions officer updates via /applications/{id} → Triggers notification job.
4. **Payment Processing** → Applicant pays fee via integrated gateway → Payment webhook updates application status.

# 4. UI/UX & Information Architecture

* **Clear Navigation**: Prominent “Apply Now” call‑to‑action, segmented by program type (undergrad, postgrad, ODL).
* **Progress Indicators**: Show steps completed vs. remaining to reduce user friction.
* **Mobile‑First Design**: Ensure usability on smartphones, critical for rural applicants with limited desktop access.
* **Accessibility Compliance**: WCAG 2.1 AA standards—semantic HTML, ARIA labels, keyboard navigation.

# 5. Security & Compliance

* **Transport Security**: HTTPS/TLS mandatory for all endpoints.
* **Encryption at Rest**: AES‑256 for sensitive data, encrypted fields for PII.
* **Input Validation & Sanitization**: Prevent SQL injection, XSS, and file‑based attacks.
* **Audit Logging**: Track critical operations (logins, data changes) for compliance under Malawi’s Data Protection Act.

# 6. Scalability, Monitoring & Maintenance

* **Containerization**: Dockerize services for consistent environments; orchestrate with Kubernetes or AWS ECS.
* **CI/CD Pipeline**: Automated tests and deployments via GitHub Actions or Jenkins.
* **Monitoring & Alerts**: Prometheus + Grafana for metrics; PagerDuty for incident response.
* **Automated Backups**: Scheduled dumps of database and file snapshots stored offsite.

**Next Steps**

1. **Validate Requirements** with stakeholders and refine workflows.
2. **Prototype UI** using Figma or similar to gather user feedback.
3. **Set Up Dev Environment**, define coding standards and branch policies.
4. **Implement Core Modules** in iterative sprints:
   * Sprint 1: Auth & User Management
   * Sprint 2: Application Forms & Workflow
   * Sprint 3: Document Handling & Storage
   * Sprint 4: Notifications & Dashboard
5. **Conduct Testing** (unit, integration, E2E) and **Security Audits**.
6. **Pilot Launch** with a small cohort before full rollout.

This high‑level design provides a clear roadmap for your development team to build a robust, secure, and scalable online admissions platform for MUST—streamlining the application process for all prospective students.