Product Requirements Document (PRD): Admin Dashboard

Title:

Admin Dashboard Development for KYC and Compliance

Brief Summary

Business Goal: To create an intuitive and secure admin dashboard that facilitates efficient user management, compliance (KYC/AML), and analytics for cryptocurrency exchange operations.

Outcome of the Product:

- Streamlined KYC verification process.
- Enhanced oversight of user activities and compliance.
- Improved ticketing and support management for operations.
- Real-time analytics to monitor platform growth and ensure compliance with regulatory standards.

Problem Statement

Current Challenge to Solve: The existing system lacks a centralized admin interface to efficiently manage users, track compliance status, and handle escalations in a regulated and user-friendly manner. This creates inefficiencies in monitoring KYC/AML processes and responding to user queries.

Measurable Metrics (Estimated figures)

- Reduction in KYC processing time by 30%.
- 99.9% uptime for the admin dashboard.
- Ticket resolution time reduced by 40%.
- 90% adherence to SLA for flagged transaction reviews.
- Real-time analytics updating every 5 seconds.

Use Cases & User Stories

Use Case 1: KYC Management

User Story: As an admin Ram, wants to review and approve/reject user KYC submissions, so he can ensure compliance with regulations while onboarding users quickly.

Use Case 2: Analytics Monitoring

User Story: As a super admin Sita, wants to view platform metrics such as user geography, sign-ups, and transaction details, so shel can make data-driven decisions and prioritise key guests KYC process time duration.

Use Case 3: Ticketing System

User Story: As a support staff member, Sridhar, wants a ticketing system where he can handle user queries in a FIFO manner, so he can resolve issues systematically and efficiently.

Use Case 4: Role-Based Access

User Story: As a super admin, I want to configure permissions for other admin roles and staff, so I can maintain secure operations within the platform

---

Modules & Features

Module 1: Home Dashboard

- Overview of platform activities:
    - Total users
    - Active users
    - Pending KYC requests
    - Ticket status

Module 2: Analytics

- **Key Features:**
    - Geography-based user distribution.
    - User demographics (age, gender, etc.).
    - Transaction details (volume, value, currency,date).
- **Sub-Pages:**
    - User Growth Trends
    - KYC Approval Rates
    - Transaction Analytics

Module 3: KYC Status

- **Key Features:**
    - View pending, approved, and rejected KYC requests.
    - Filter requests by date, user ID, and status.
    - Bulk approvals/rejections with automated notifications.

Module 4: Support Center

- **Key Features:**
    - Ticketing system following FIFO (First In, First Out).
    - Priority tagging for escalated tickets.
    - Automated assignment of tickets to support staff.

Module 5: Compliance & Reports

- **Key Features:**
    - Adherence to Indian government and global KYC/AML standards.
    - Generate compliance audit reports for regulatory authorities.
    - Flagged transaction review and escalation.

---

2. Pitcher List

Below is the module-specific pitcher list**:**

A. Super Admin Dashboard

1. Home Page:
    - Overview of platform performance metrics (e.g., KYC status, transaction trends, active users).
2. Role Management:
    - Add, edit, or delete user roles and permissions.
3. KYC Management:
    - Access all KYC applications and status summaries.
4. Analytics Dashboard:
    - Access user demographic data, transaction volumes, and region-based trends.
5. Audit Logs:
    - Immutable logs of all platform activities. (**Timestamp, User** - e.g., username, IP address, system process, **Action:** e.g.,User logged in, Transaction initiated,KYC document uploaded, Role permission changed, **Target:** e.g., a specific user account, a transaction ID, a KYC document, **Details:** value of a transaction, the old and new values of a changed setting, or any error messages.)

B. Admin Dashboard

1. KYC Processing:
    - Detailed view of pending, approved, and rejected KYC applications.
    - Bulk processing options for flagged applications.
2. Transaction Monitoring:
    - Alerts for high-value or unusual transactions.
3. User Management:
    - Reset passwords, update user information, and assist with account recovery.
4. Analytics Sub-Pages:
    - Metrics specific to KYC (e.g., application processing times).

C. Supporting Staff Dashboard

1. Support Ticket Management:
    - FIFO (First In, First Out) queue for handling user queries.
2. User Profile View:
    - Read-only access to basic user details and transaction summaries.

D. Senior Staff Dashboard

1. Escalation Handling:
    - Priority ticket queue and flagged KYC cases.
2. Analytics Overview:
    - High-level insights into user activity and transaction trends.
3. Supervision Tools:
    - Review support staff performance and response times.

E. Compliance Dashboard

1. KYC Review:
    - Automated alerts for potential issues in user documentation.
2. AML Monitoring:
    - Real-time transaction pattern analysis to detect fraud or laundering.
3. Compliance Reports:
    - Generate detailed reports for audits, adhering to Indian and global standards.

---

3. Integration for Each Role

- API Layer: APIs will enforce role-based restrictions, ensuring secure access.
- Frontend and Backend Sync: Each role's dashboard will fetch data relevant to their responsibilities via APIs.

Detailed Roles, Access, Restrictions, for the Admin Dashboard in a KYC Module

---

1. Roles, Access, and Restrictions

A. Super Admin

- Access:
    - Full access to all modules and system configurations.
    - Can create, edit, or remove roles and permissions for other users (admins, support staff, compliance team).
    - View and manage all KYC applications and transaction logs.
    - Access to audit logs, analytics, and reports (geography, user demographics, transaction details).
- Restrictions:
    - None. Super Admin has unrestricted access to ensure platform governance and oversight.
- Responsibilities:
    - Monitor platform performance, approve escalations, and oversee compliance adherence to regulatory standards.

B. Admin

- Access:
    - Manage KYC verification process and oversee transaction monitoring.
    - Access analytics dashboards with metrics such as the number of users, KYC status, and transaction details.
    - Approve/reject high-priority KYC applications flagged by the system.
    - Limited access to system configurations (e.g., updating transaction thresholds but not changing user roles).
- Restrictions:
    - Cannot modify or access Super Admin-level settings or permissions.
    - Cannot delete audit logs or override compliance decisions.
- Responsibilities:
    - Ensure smooth operations of KYC and AML processes, respond to escalations, and assist the compliance team.

C. Supporting Staff

- Access:
    - Access to user profiles for basic support tasks.
    - View-only access to KYC statuses and transaction details.
    - Can reset passwords and manage support tickets.
- Restrictions:
    - Cannot approve/reject KYC applications or access sensitive analytics data.
    - No access to compliance or audit logs.
- Responsibilities:
    - Address user queries, resolve basic issues, and escalate complex problems to senior staff.

D. Senior Staff

- Access:
    - Handle escalated support tickets and flagged KYC applications.
    - Oversee ticketing and compliance activities on a priority basis.

    - Access detailed analytics related to user and transaction trends.
- Restrictions:
    - Cannot make platform-wide changes or manage admin roles.
    - Limited access to audit logs (can view but not modify).
- Responsibilities:
    - Supervise supporting staff, resolve escalated issues, and coordinate between the admin and compliance teams.

E. Compliance Team

- Access:
    - Complete access to KYC and AML processes.
    - Can flag or approve/reject KYC applications based on documentation.
    - Access transaction logs for suspicious activity review.
- Restrictions:
    - Cannot modify user profiles or system configurations.
    - No access to analytics unrelated to compliance activities.
- Responsibilities:
    - Ensure regulatory compliance, monitor for fraud, and generate reports for audits.

Sprints Plan

Sprint 1: KYC Module Development

- Backend API for KYC status updates.
- Integration with the frontend for document uploads.
- Role-based access control implementation.
- Development time: 7 days. (Estimate)

Sprint 2: Analytics Module

- Real-time data aggregation.
- API endpoints for metrics (users, transactions).
- Sub-folder structures for detailed analytics.

- Development time: 7 days. (Estimate)

Sprint 3: Ticketing System

- Backend for ticket creation and assignment.
- FIFO logic implementation.
- Notification APIs for ticket status updates.
- Development time: 5 days. (Estimate)

Sprint 4: Compliance Module

- Backend for flagged transaction management.
- API integration for third-party tools (e.g., Chainalysis).
- Audit log generation.
- Development time: 7 days. (Estimate)

Integration for Super Admin and Other Admins

- Super Admin:
  - Full access to all modules.
  - Ability to assign permissions to other admins.
- Other Admins:
  - View and manage modules based on assigned permissions.
  - Restricted access to certain compliance settings.

Authentication

- OAuth 2.0-based authentication.
- Role-based access tokens for APIs.
- Multi-factor authentication for sensitive actions (e.g., approving flagged transactions).

Product Cycle

1. **Development Stage:**
   - Backend API creation.
   - Frontend UI/UX integration.
2. **QA Testing:**
   - Functional and security testing.
   - KYC flow validation with sample data.
3. **UAT (User Acceptance Testing):**
   - Conducted by client and internal teams.
   - Feedback incorporated.
4. **Production:**

    - Deployment to live servers.
    - Monitoring for initial user feedback.
5. **Growth Stage:**
    - Beta release.
    - Increase market share by 10% in the first quarter.
    - Gather feedback for future iterations.
6. **Maturity Stage:**
    - Regular updates for performance improvements.
    - Scale infrastructure to support user growth.
    - Revenue analysis and feature prioritization.

Compliance Considerations

- **Indian Standards:**
    - Adhere to Reserve Bank of India's KYC/AML guidelines.
- **Global Standards:**
    - FATF recommendations.
    - GDPR compliance for data privacy.

Report

- Weekly KYC approval/rejection rates.
- Monthly user growth trends.
- Quarterly compliance audits.

Related Frontend Work

- **UI/UX Design:**
    - Simple and intuitive navigation for admin roles.
    - Real-time data representation (charts, tables).
- **Integration Requirements:**
    - Consume APIs for all modules (e.g., KYC status, analytics).
    - Responsive design for cross-device compatibility.

How the Engineering Team Would Perform Tasks for Both Frontend and Backend Development

1. Backend Engineering Tasks

A. Architecture and Infrastructure Setup

1. **Database Design**
    - Define tables for users, roles, permissions, KYC statuses, transactions, tickets, and logs..
2. **Role-Based Access Control (RBAC)**
    - Implement connection to validate user roles and permissions for each API request.
    - Example: Only admins can approve KYC applications, while support staff have read-only access.

3. **API Development**

4. **KYC & AML Modules**
    - **KYC API:**
        - Receive user-submitted documents via the frontend.
        - Validate documents using third-party APIs like Onfido.
        - Flag suspicious applications for manual review.
    - **AML Monitoring API:**
        - Process transaction data to detect patterns of fraud or money laundering using rules and thresholds.
5. **Ticketing System**
    - Develop APIs to manage tickets (create, update, fetch, and escalate).
6. **Integration with Third-Party Tools**
    - Integrate compliance tools (e.g., Chainalysis for AML, Onfido for KYC).
    - Implement logging and monitoring tools like Splunk or Elasticsearch.
7. **Testing and Debugging**
    - Write unit and integration tests for all APIs.
    - Use automated tools to ensure performance under high loads.

---

2. Frontend Engineering Tasks

A. Design and Implementation

1. **UI/UX Design**
    - Collaborate with the design team to create wireframes and mockups for dashboards.
2. **Role-Specific Dashboards**
    - Build dynamic dashboards for different roles (e.g., Super Admin, Admin).
    - Components:
        - **Home Page:** Displays an overview of platform metrics.
        - **KYC Section:** Lists pending, approved, and rejected applications.
        - **Analytics:** Interactive charts showing user demographics and transaction trends.
        - **Support Center:** Ticket queue with filters and search capabilities.
3. **Frontend Framework**
    - Use modern frameworks like React, Angular, or Vue.js.
    - Component libraries: Material-UI, TailwindCSS, or Bootstrap.
4. **API Integration**
    - Fetch data from the backend APIs and display it on the dashboards.
    - Example workflows:
        - KYC Approval: Fetch pending applications using
        - Analytics: Use to populate graphs.
5. **Authentication and Authorization**
    - Implement login/logout functionality.
    - Role-based rendering of UI components (e.g., a Super Admin sees all analytics, while support staff only see tickets).
6. **Real-Time Features**
    - Provide live updates (e.g., new tickets in the queue, transaction alerts).
7. **Testing and QA**
    - Perform unit tests
    - Conduct end-to-end testing.

---

3. Collaboration Between Frontend and Backend Teams

A. Communication

- Regular stand-ups and sprint planning meetings to sync progress.
- Use collaboration tools

B. API Contracts

- Define clear API specifications early in the development process.

C. Staging Environment

- Set up a staging environment where both teams can test integrations.
- Backend serves mock data initially to unblock frontend development.

D. QA and Deployment

1. **Integration Testing**
    - Test frontend and backend modules together to ensure seamless data flow.
2. **UAT (User Acceptance Testing)**
    - Deploy to a UAT environment for client feedback before production.
3. **Release**
    - Roll out the platform in phases (e.g., beta testing for selected users).

Development Workflow

1. **Planning:** Define scope, roles, and timelines.
2. **Design:** Create backend architecture and frontend wireframes.
3. **Development:** Implement modules in sprints, prioritizing core features.
4. **Testing:** Ensure quality with unit, integration, and end-to-end tests.
5. **Release:** Launch the platform incrementally and monitor performance.
6. **Iteration:** Gather feedback, update features, and ensure scalability.