EHS Management System — Executive Brief (1 Page)

Summary

A modern, modular Environmental, Health, and Safety (EHS) web platform that centralizes incident reporting, safety observations, trainings, permits-to-work (PTW), meetings/MoM, manpower/attendance, and project governance — with robust role-based access, approval workflows, and real-time notifications.

Business Outcomes

- Compliance acceleration (ISO 45001/14001, OSHA logging)
- Risk reduction via faster detection/closure, leading indicators
- Operational efficiency: standardized workflows, fewer spreadsheets
- Evidence-ready audits (signatures, approvals, immutable logs)
- Enterprise scale for multi-site rollouts

Key Differentiators

- Dual role model: usertype (org role) + django_user_type (approval class) for precise control without UX friction
- Centralized menu gating synced to approval state and role
- Realtime notifications integrated across approvals, meetings, actions
- Modular EHS domains; adopt incrementally

Core Capabilities

- Incidents: capture → classify → actions → verify → analytics
- Safety Observations: log → risk rank → actions → closeout → trends
- Training: induction/job; competency tracking
- PTW: request → hazard controls → authorize → monitor → close
- Meetings/MoM: schedule, live session, distribution, follow-ups
- Manpower: daily attendance, visualizations

Architecture Snapshot

- Frontend: React + TypeScript, Ant Design, React Router, Zustand
- API: Django REST Framework, JWT auth (access/refresh)
- Notifications: WebSockets
- Data: PostgreSQL, object storage for uploads
- CI/CD, Nginx reverse proxy, TLS

Security & Compliance

- JWT auth + refresh; CSRF header for non-GET
- Route protection by usertype; approval gating by django_user_type
- Upload validation; least privilege to storage/DB
- Audit logs; ready for ISO 45001/14001 alignment, OSHA recordkeeping

Deployment Options

- Containers behind Nginx; HTTPS; DB backups; object storage
- Staging + production; IaC recommended (Terraform/Ansible)

Implementation Timeline (Typical)

- Week 0–1: Discovery, tenant provisioning, SSO/MFA planning
- Week 2–3: Data model tailoring, role mapping, branding

- Week 4–6: Module rollout (Incidents, Observations, Training)
- Week 7–8: PTW & Meetings, reporting; user training

Pricing/Packaging (Illustrative)

- Core Platform (Incidents, Observations, Training)
- Advanced (PTW, MoM, Manpower) add-ons
- Enterprise: SSO/MFA, advanced analytics, priority support

KPI Impact (Example Year-1)

- 30–50% faster incident closeout
- 70% fewer missing training records
- 40% reduction in audit preparation time

Contact & Next Steps

- Request a tailored demo and data migration assessment
- Pilot with 1–2 sites, then scale to all facilities

EHS Management System — Technical & Functional Specification

Author: Augment Agent Version: 1.0

1. Overview

A modular EHS platform with role/approval-aware UX, real-time notifications, and comprehensive domain coverage (Incidents, Safety Observations, Training, PTW, Meetings/MoM, Manpower, Projects). This document is a definitive reference for engineering and enterprise stakeholders.

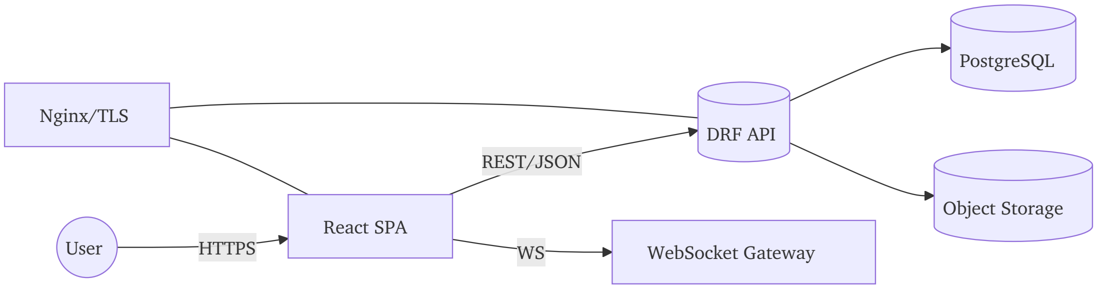
2. System Architecture

- SPA Frontend (React 18 + TypeScript, Vite)
- REST API (Django REST Framework), JWT auth (access/refresh)
- WebSockets for notifications
- PostgreSQL RDBMS; object storage for uploads
- Nginx reverse proxy; TLS; CI/CD

2.1 Mermaid — High-level System Diagram

```
flowchart LR
  User((User))-->|HTTPS|SPA[React SPA]
  SPA-->|REST/JSON|API[(DRF API)]
  SPA-->|WS|WS[WebSocket Gateway]
  API-->DB[(PostgreSQL)]
  API-->Store[(Object Storage)]
  Nginx[Nginx/TLS]---SPA
  Nginx---API
```

[Rendered diagram]



System Architecture

3. Frontend Architecture

- Libraries: React, React Router, Ant Design, Zustand, Axios, styled-components
- State: Auth store (token, roles, approval flags)
- Navigation: RoleBasedRoute + centralized menu config
- Approval banner + CTA routes (projectadmin → admindetail; adminuser → profile)

3.1 Key Code References

```
// Menu gating
export const getMenuItemsForUser = (usertype?, django_user_type?, isApproved=true,
hasSubmitted=true) => {
  if (usertype === 'master') return fullMasterMenu;
  if (!isApproved && (django_user_type === 'projectadmin' || django_user_type === 'adminuser')) {
    return getRestrictedMenuItems(django_user_type, hasSubmitted);
  }
  return fullMenuByRole(usertype, django_user_type);
};
```

```
// Approval status hook
if (usertype === 'master') { setAuthApprovalStatus(true, true); }
if (!['projectadmin','adminuser'].includes(django_user_type)) { /* skip */ }
const status = (await api.get('/authentication/approval/status/')).data;
```

4. Backend Architecture (Assumed Current)

- Django + DRF
- SimpleJWT (rotate+blacklist recommended)
- Channels/WebSocket notifications
- Endpoints: approval/status, admin/me, admin/detail/update, admin/detail/approve, userdetail/pending, admin/pending, companydetail, master-admin

5. Data Model (Logical)

- User(usertype, django_user_type, tokens)
- AdminDetail(company_name, phone, PAN/GST, logo, photo, approved)
- UserDetail(employee profile, attachments, approved)
- CompanyDetail(PAN/GST/logo)
- Incident, SafetyObservation, Training, PTW, MoM, Manpower, Project

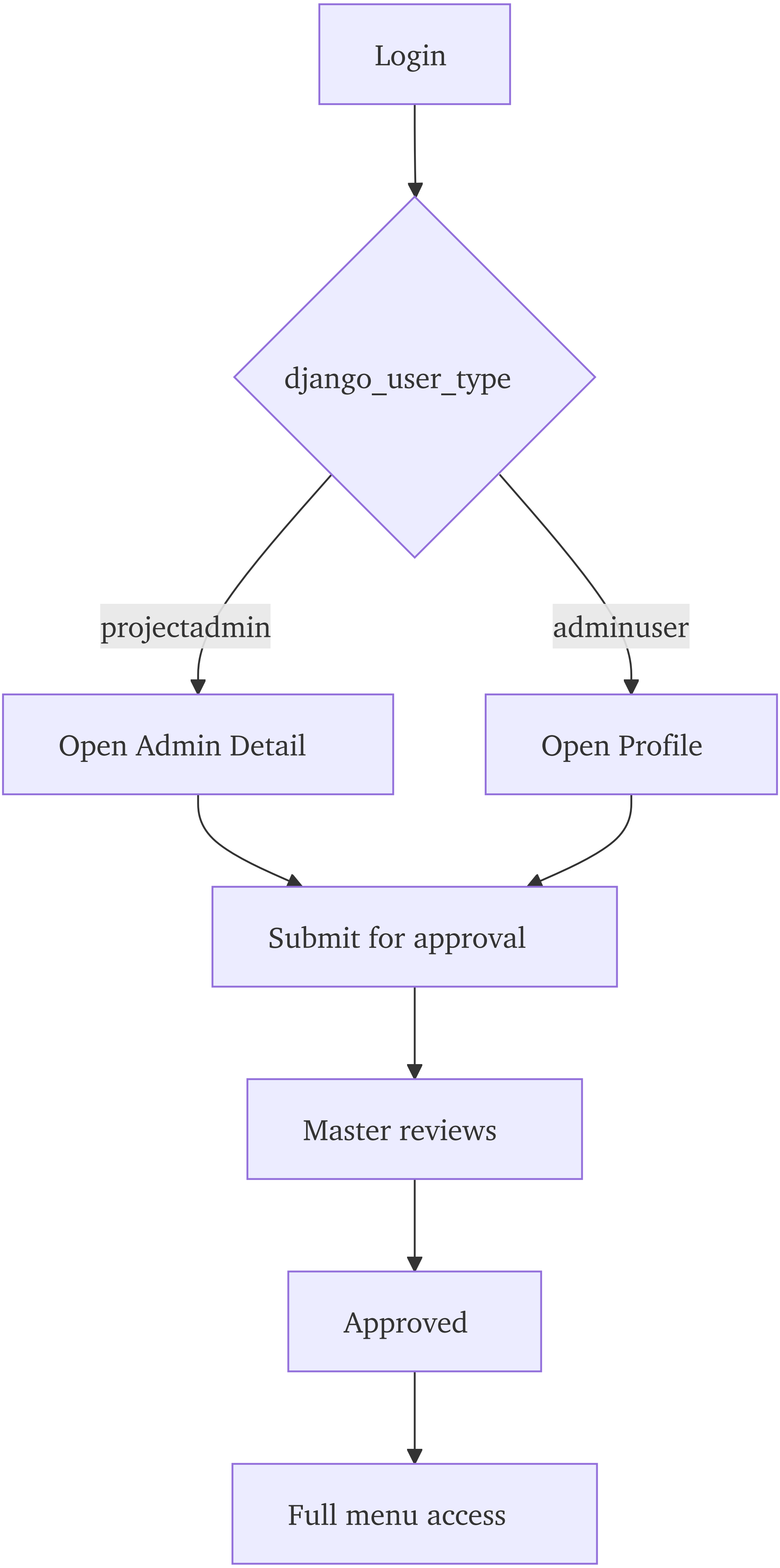
6. Workflows

- Auth: JWT access/refresh, axios interceptors, forced logout on blacklist
- Approval: role-aware CTAs, restricted menus, master approval path
- EHS modules: incident/observation/training/PTW/MoM/manpower lifecycles

6.1 Mermaid — Approval Flow

```
flowchart TD
  A[Login]-->B{django_user_type}
  B--projectadmin-->C[Open Admin Detail]
  B--adminuser-->D[Open Profile]
  C-->E[Submit for approval]
  D-->E
  E-->F[Master reviews]
  F-->G[Approved]
  G-->H[Full menu access]
```

[Rendered diagram]


Login
django_user_type
projectadmin
adminuser
Open Admin Detail
Open Profile
Submit for approval
Master reviews
Approved
Full menu access

Approval Flow

7. Security

- JWT auth; refresh with rotation; blacklist invalid tokens
- CSRF header for non-GET (defense-in-depth)
- Route protection by usertype; approval gating by django_user_type
- Upload validation; least-privilege storage/DB
- Suggested: MFA, rate limiting, audit logs, AV scanning

8. Performance & Scalability

- Vite code-splitting; lazy routes for heavy modules
- WebSockets for push notifications
- DB indexing; pagination; background jobs for reports
- Horizontal scaling: API, WS, SPA via CDN

9. Compliance

- ISO 45001/14001 alignment, OSHA 1904 recordkeeping
- Access controls, evidence trails, immutable approvals/signatures

10. Deployment

- Containers (API + SPA) behind Nginx; TLS
- Env: API base URL, secrets, DB, storage, WS endpoints
- CI/CD with migrations, rollbacks, backups

11. Operations

- Logging/metrics/APM; alerting on error rates
- Backup/restore drills; DB maintenance
- Security reviews; dependency updates

12. Roadmap

- MFA + SSO
- Advanced analytics (risk heatmaps, predictive)
- Offline/mobile-first capture
- IoT sensors; GIS/geofencing for PTW
- DMS with versioning; automated regulatory reports

13. Appendix — Example Snippets

```
// Role route guard
const RoleBasedRoute: React.FC<{ allowedRoles: string[]; children: React.ReactElement; }> = ({
allowedRoles, children }) => {
  const { usertype, token, isAuthenticated } = useAuthStore();
  if (!token || !isAuthenticated()) return <Navigate to="/login" replace />;
  const ok = !!usertype && (allowedRoles.includes(usertype) || (usertype.startsWith('contractor')
&& allowedRoles.some(r=>r.startsWith('contractor'))));
  return ok ? children : <Navigate to="/dashboard" replace />;
};
```

EHS Management System — Workflow Guide

This guide details end-to-end workflows for core EHS modules, including data responsibilities, states, roles, and approvals.

1. Role & Approval Model

- usertype (route access): master, client, epc, contractor, clientuser, epcuser, contractoruser
- django_user_type (approval class): projectadmin (completes Admin Detail), adminuser (completes Profile)
- Master is exempt from approval gating.

2. Authentication & Session Flow

1. User logs in; JWT access + refresh issued
2. Axios attaches Authorization Bearer token; CSRF header on non-GET
3. 401 triggers refresh; invalid/blacklisted → forced logout
4. RoleBasedRoute guards by usertype; approval gating checks django_user_type

3. Profile Completion & Approval

projectadmin (Admin Detail)

- Trigger: Banner/CTA → /dashboard/admindetail
- First-time: admin/me 404/403 → show blank form, prefill username
- Submit → pending approval by master
- Master reviews & approves → full menu access enabled

adminuser (Profile)

- Trigger: Banner/CTA → /dashboard/profile
- Submit → pending approval (as configured)
- Approval → full access for user-level features

4. Incident Management Workflow

1. Report Incident
   - Inputs: time, location, type, description, attachments
   - Optional: immediate actions
2. Triage & Classification
   - Severity, category, root cause hypothesis
3. Corrective/Preventive Actions (CAPA)
   - Assign action owners, due dates
4. Verification & Closure
   - Evidence upload; approvals
5. Analytics
   - KPIs, trends, recurrence analysis

Roles: - Reporter (any authenticated user allowed by policy) - Incident Manager (client/epc/contractor admin roles) - Approver (master or designated)

5. Safety Observation Workflow

1. Log observation (unsafe act/condition)
2. Risk ranking; assign actions
3. Follow-up; corrective steps
4. Closeout and review
5. Dashboard trends by site/department

Roles: - All staff (capture) - Supervisors (review/assign) - HSE team (analysis)

6. Training (Induction & Job)

1. Enrollment (new joiners / role change)
2. Session scheduling and delivery
3. Attendance capture; assessment
4. Certification / competency update
5. Expiry reminders & refreshers

Data: - Trainee, trainer, syllabus, assessment, certificate

7. Permit to Work (PTW)

1. Permit Request
   - Job details, location, duration
2. Hazard Identification & Controls
   - JSA, PPE, isolation, monitoring
3. Authorization
   - Permit issuer → accept/deny
4. Active Monitoring
   - Spot checks, gas tests, shift handovers
5. Closure & Post-Review
   - Sign-off, lessons learned

Roles: - Requester (contractoruser/adminuser) - Issuer/Approver (client/epc) - HSE oversight (projectadmin/master)

8. Meetings/MoM

1. Schedule meeting; send invites (notifications)
2. Live session notes; decisions and action items
3. Distribution; action follow-up
4. Closeout and archival

9. Manpower & Attendance

1. Daily attendance input (per contractor/site)
2. Visualization by project/date/contractor
3. Export for payroll/planning

10. Company Details & Auto-Fill

- EPC: auto-fill PAN, GST, and logo for Admin Detail when available
- Sync company logo to UI headers where applicable

11. Notifications & Tasks

- WebSocket-driven alerts for approvals, meetings, and critical changes
- Inbox UI and toast notifications

12. Data Lifecycle & Retention

- Uploads: photos, signatures, attachments (object storage)
- Suggested retention policies by module (per compliance)
- Export: CSV/PDF for audits

13. Audit Readiness

- Immutable approvals (timestamps, user IDs)
- Digital signatures (admin/user templates)
- Traceability: who, when, what changed

14. Exceptions & Error Handling

- 404/403 on first-time admin detail → render blank form
- 401 with invalid token → forced logout with message
- Network fallbacks and user feedback to retry

15. KPIs & Reports (Examples)

- Incident closeout time; CAPA SLA adherence
- Observation count by department; risk distribution
- Training completion rate; certification expiries

- PTW durations; non-compliance events
- Attendance trends by contractor

16. Implementation Checklist

- Configure roles/usertypes and django_user_types
- Company details (logo, PAN/GST)
- Enable modules and routing
- Notification endpoints and policy
- Backup and monitoring

17. Handover & Admin SOP

- Approver matrix maintenance
- Periodic data quality checks
- Quarterly compliance review
- Access reviews and deprovisioning

Slide 1 — Title EHS Management System — Enterprise Overview
Slide 2 — Problem & Opportunity - Fragmented EHS data and manual workflows - Regulatory exposure and audit effo

effort - Opportunity: unified, modular EHS platform with real-time visibility

Slide 3 — Solution Snapshot

- Incidents, Observations, Training, PTW, MoM, Manpower
- Role/approval-aware UX, notifications, analytics
- Secure, scalable, integrable

Slide 4 — Architecture - React + TypeScript SPA; DRF API; WebSockets; PostgreSQL; Nginx/TLS - JWT auth, approval workflows

---

Slide 5 — Key Differentiators - Dual

Dual role model (user type + django_user_type) - Centralized menu gating bound to approval state - Real-time notifications

Slide 6 — Capabilities

- Incident lifecycle, safety observation, training, PTW, MoM, manpower

Slide 7 — Security & Compliance - JWT, CSRF header, least privilege, audit logs - ISO 45001/14001 alignment,

OSHA recordkeeping
Slide 8 — Deployment & Ops - Containers, Nginx, TLS, backups, monitoring

Slide 9 — ROI & KPIs

- Faster closeouts, fewer missing records, reduced audit prep time

Slide 10 — Roadmap - MFA/SSO, analytics, offline, IoT, DMS

Slide 11 — Next Steps

- Tailored demo, pilot, enterprise rollout plan