SECURITY & COMPLIANCE DOCUMENTATION

Security Architecture Overview

How the Miles Masterclass web platform and API are built, hosted, secured, and operated.

milesmasterclass.com · Web platform (Angular SSR) & REST API (Django)

CONFIDENTIAL — SHAREABLE UNDER NDA

DOCUMENT	Security Architecture Overview
VERSION	1.0
ISSUE DATE	22 June 2026
AUDIENCE	Prospective & existing customers, vendor security reviewers
SYSTEMS	miles-masterclass-v3 (web) · miles-masterclass-backend (API)
CLASSIFICATION	Confidential — Shareable under NDA

Prepared and issued by the Information Security Office, Miles Masterclass Inc.. This document describes the platform's security posture as of the issue date and is maintained on an ongoing basis.

Miles Masterclass

Miles Masterclass Inc.
SECURITY & COMPLIANCE

Contents

Companion documents: SOC 2 + GDPR/DPDP Compliance Mapping; Incident Response Plan (IRP); Written Information Security Program (WISP).

1. Purpose, Scope & Audience

This document describes the security architecture and controls in use across the Miles Masterclass platform. It is written for customer and vendor security reviewers who need to understand how the platform is designed, hosted, and protected before or during a commercial relationship.

Scope. Two production systems are in scope:

- **Web platform** (`miles-masterclass-v3`) — an Angular 21 application using Server-Side Rendering (SSR), delivered through the Vercel edge network. It also includes a staff-facing admin console backed by Supabase.
- **Backend API** (`miles-masterclass-backend`) — a Django REST Framework service providing course delivery, user accounts, webinars, certifications, notifications, orders, and payment processing.

Out of scope. Corporate IT, the separate Miles LMS/CPA products, and end-user devices are referenced only where they interface with the platform.

> **How to read this document**
>
> Statements describe the controls in place on the platform as of the issue date. The mapping of these controls to the SOC 2 Trust Services Criteria, the EU GDPR, and India's DPDP Act is provided in the companion SOC 2 + GDPR/ DPDP Compliance Mapping. This document, together with the Incident Response Plan and Written Information Security Program, forms our security documentation set for customer and vendor review.

2. Platform Overview

Miles Masterclass is an education (EdTech) platform delivering masterclasses, micro/nano-learning, webinars, learning pathways, and professional certifications, with e-commerce (subscriptions and one-time purchases) and multi-region, localized content (e.g. country/profession routing such as `/in/accounting`, `/us/cpa`).

The platform processes the following broad categories of data:

Data category	Examples	Sensitivity
Account & profile	Name, email, mobile, country, profession, employer/sector	Personal data (PII)
Authentication	Hashed passwords, JWT/session tokens, OTP verification state	Sensitive
Commerce	Orders, invoices, subscription state, Stripe customer/ transaction IDs	Personal + financial metadata
Payment instruments	Card data — NOT stored; handled by Stripe via tokenization	Out of scope for storage (PCI-delegated)
Learning activity	Progress, certifications, webinar attendance, engagement events	Personal data
Operational	Application logs, request metadata, device/push tokens	Internal

3. System Architecture

The platform follows a decoupled, multi-tier architecture. The web tier is server-rendered for performance and SEO and holds no business data of its own; all sensitive operations are performed by the API tier over authenticated HTTPS calls.

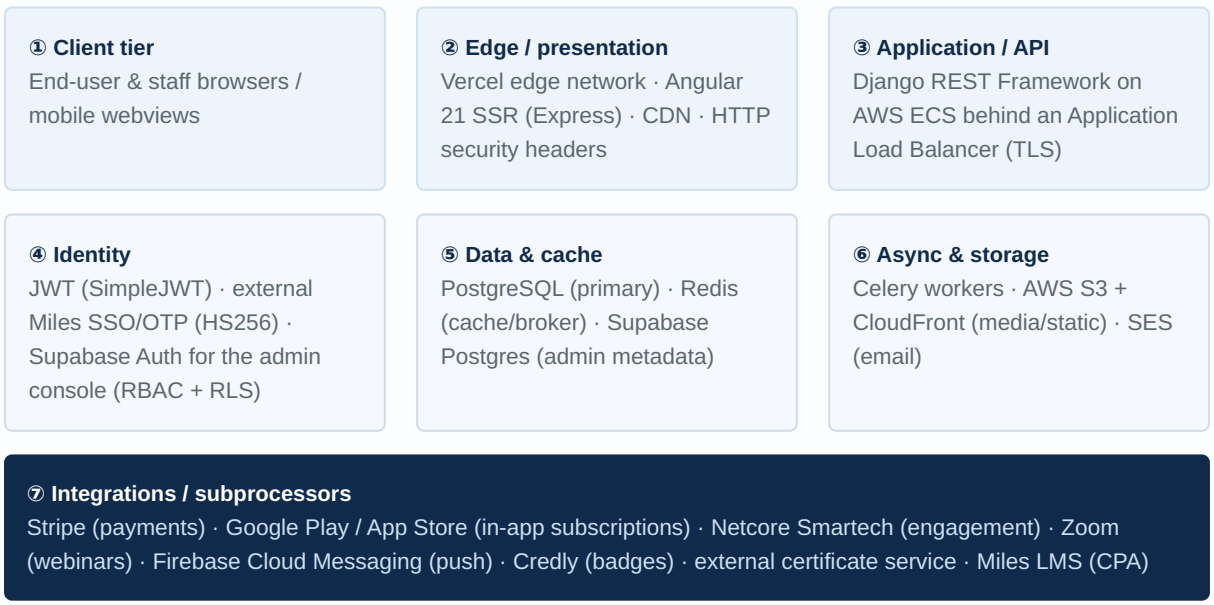

Figure 1 — Logical architecture: a server-rendered Angular web tier and a Django REST API tier, with managed identity, data, storage, and third-party services.

3.1 Web / presentation tier

Angular 21 with SSR runs on the Vercel edge platform. The SSR server (Express) restricts which hosts it will render for (an `allowedHosts` allow-list, mitigating SSRF), proxies only read-only requests to the public blog API, and holds no server-side application secrets. Static assets are served with long-lived, immutable cache headers; the service worker is served no-cache.

3.2 API / application tier

The Django REST Framework API runs in containers on AWS ECS behind an Application Load Balancer that terminates TLS. Background processing (emails, sync jobs, report generation, certificate issuance) runs on Celery workers using a Redis broker. The API is the single authority for authentication, authorization, and data access.

3.3 Admin console

A separate staff-facing admin console is part of the web application and authenticates against **Supabase Auth**. Authorization is enforced in the database itself through Row-Level Security (RLS) policies driven by a role/permission model (see §5.3), so the browser is never the security boundary.

4. Hosting, Network & Data Residency

Web tier	Vercel edge network (global CDN + SSR functions)
API tier	AWS ECS containers behind an Application Load Balancer (TLS termination), primary region **us-west-1**; UAT in **ap-south-1**
Object storage / CDN	AWS S3 with CloudFront distribution; S3 signature v4, object ACLs disabled (bucket-owner-enforced), querystring auth disabled
Databases	PostgreSQL (primary application data); Supabase-managed PostgreSQL (admin console metadata)
Cache & queue	Redis (cache + Celery broker/result backend; JSON-only serialization)
Email	AWS SES (ap-south-1)
Container build/registry	Docker images built in CI and pushed to AWS ECR

> **Multi-cloud footprint**
>
> The platform uses a deliberate best-of-breed design spanning Vercel, AWS (compute, storage, email), Supabase, and Google Cloud (Play Billing). Each provider is a reputable, independently audited platform (SOC 2 / ISO 27001). All subprocessors are enumerated in §10 and governed under the vendor-management process defined in our Written Information Security Program.

Data residency. Production primary storage and email operate in AWS `us-west-1` and `ap-south-1`. Because the platform serves users in India, the US, and the UAE, cross-border transfer mapping is addressed in the GDPR/India DPDP assessment.

5. Identity, Authentication & Access Control

5.1 End-user authentication

End users authenticate to the API using JSON Web Tokens issued by **SimpleJWT**. A passwordless OTP flow and single sign-on are provided through an external **Miles SSO** service; the API verifies SSO-issued tokens using an HMAC (HS256) shared secret and validates standard claims (expiry, issued-at). Passwords, where used, are stored using Django's salted PBKDF2 password hashing with the standard validator set (length, common-password, numeric, and user-attribute-similarity checks).

5.2 Token handling in the browser

The web client attaches the bearer token via a single HTTP interceptor and stores tokens in cookies (SameSite=Lax), which provides implicit CSRF protection for state-changing requests; only non-sensitive profile metadata is held in local storage. A queued refresh-token mechanism renews access transparently on expiry and avoids duplicate refresh calls.

5.3 Administrative access (RBAC + RLS)

The admin console uses a formal role/permission model in Supabase: roles (e.g. *super_admin, seo_manager, leads_manager, reports_viewer*), a permission catalogue (e.g. `seo:write`, `leads:read`, `admin:users:manage`), role→permission mappings, and per-user grant/deny overrides. Every table enforces Row-Level Security through a

central `has_admin_permission()` function, so authorization is decided in the database regardless of the client. No Supabase `service_role` (privileged) key is exposed to the browser.

5.4 Service-to-service & API throttling

Trusted integrations (lead ingestion, report pulls, LMS and partner webhooks) authenticate with scoped shared secrets or signed payloads transmitted over TLS; the public report endpoints use constant-time secret comparison and fail closed when unconfigured. The API applies DRF rate limiting by default — 100 requests/minute for anonymous callers and 300/minute per authenticated user.

6. Data Protection & Cryptography

6.1 Encryption in transit

All external traffic is HTTPS/TLS. The web tier sets **HTTP Strict Transport Security** (HSTS, 2-year max-age with `includeSubDomains; preload`). The API enforces SSL redirection, secure-only session and CSRF cookies, and honours the load balancer's forwarded-proto header in production.

6.2 Encryption at rest

Data at rest is encrypted using the storage-layer encryption of the underlying cloud providers (AWS S3 and managed PostgreSQL, Supabase). Application-layer field encryption is additionally applied to selected sensitive fields. Card data is never stored — payment instruments are tokenized by Stripe — which keeps the platform out of most PCI-DSS storage obligations.

6.3 Key & secret management

Production and UAT secrets are loaded at runtime from **AWS Secrets Manager** (the API fetches them on boot rather than baking them into images); CI injects deployment secrets from Secrets Manager into the ECS task definition rather than into the container image. Local development uses a git-ignored `.env` file.

> **Payments & PCI scope**
>
> Because card data is captured and processed entirely by Stripe (tokenization; Stripe Elements/Checkout) and only Stripe customer/transaction identifiers are retained, the platform's PCI-DSS scope is limited to a SAQ-A style posture. Stripe webhooks are verified using the Stripe SDK signature check with a dedicated signing secret.

7. Application Security Controls

Control	Implementation
Transport security	HTTPS everywhere; HSTS (2y, preload) at the edge; SSL redirect + secure cookies on the API
HTTP security headers	CSP, X-Frame-Options: SAMEORIGIN, X-Content-Type-Options: nosniff, Referrer-Policy, Permissions-Policy (set at the edge)
Clickjacking	Frame-ancestors 'self' (CSP) and X-Frame-Options on both tiers
CORS	Explicit origin allow-list on the API (not wildcard in production); scoped methods and headers
CSRF	Django CSRF middleware + trusted-origins list; SameSite cookies on the web client
Rate limiting	DRF throttling — 100/min anonymous, 300/min per authenticated user
Input / upload limits	Request body and upload size caps; bounded max number of fields per request
AuthN/AuthZ	JWT + external SSO/OTP for users; Supabase RBAC enforced via database RLS for admins
Password policy	Django PBKDF2 hashing with length / common-password / numeric / similarity validators
Error handling	Custom 403/404/500 handlers and a centralized exception handler to avoid leaking internals
Serialization safety	Celery restricted to JSON content (no pickle), removing a deserialization RCE vector

Defense in depth at the edge

A comprehensive Content-Security-Policy is enforced at the edge alongside HSTS, anti-clickjacking, MIME-sniffing protection, and a restrictive Permissions-Policy. Combined with server-side input validation, output encoding, and rate limiting, this provides layered protection against common web attacks (XSS, clickjacking, injection).

8. Secure Development & Infrastructure

8.1 Build & deployment pipeline

Both repositories use Git with GitHub Actions. The API pipeline builds a Docker image, pushes it to AWS ECR (tagged by commit SHA), pulls secrets from AWS Secrets Manager, registers a new ECS task definition, and performs a rolling update; GitHub Actions runs with least-privilege (`contents: read`) permissions. The web app is built and deployed via Vercel. Pre-commit hooks run linting and formatting (ESLint/Prettier) on the web codebase.

8.2 Containerization

The API ships as a slim Python 3.12 image. A single entrypoint supports multiple roles (web via Gunicorn, Celery worker, beat scheduler, monitoring) with auto-tuned worker counts and preloading. Migrations and static collection run at start-up.

8.3 Dependency posture

Core dependencies are maintained at current versions — Django 5.2, Django REST Framework 3.16, SimpleJWT 5.5, `cryptography` 45, Stripe SDK 11, Gunicorn 23 (API); Angular 21 and Supabase JS 2.x (web). Automated dependency (SCA), static-analysis (SAST), and secret scanning run in the CI pipeline to catch vulnerable packages and inadvertent secret exposure before release.

9. Logging, Monitoring & Operations

The API performs structured request, response, and error logging with rotation and retention, plus user/session tracking. Sensitive fields — authorization headers, OTP codes, and personal data — are redacted before logs are written. Logs are centrally aggregated with alerting on security-relevant events, and uptime and performance are monitored with alerts routed to on-call. Redis-backed caching uses fail-open behaviour with short timeouts so a cache outage does not take down the API, and database connections use health checks and connection reuse.

> **Resilience by design**
>
> The platform combines autoscaling compute, fail-open caching, and health-checked databases so that the failure of any single component degrades gracefully rather than causing an outage. Monitoring and alerting provide early warning of anomalies.

10. Third-Party Subprocessors

The platform relies on the following major third parties. Each should be covered by a data-processing agreement and the vendor-management process defined in the WISP.

Subprocessor	Function	Data shared / role
Vercel	Web hosting / edge / SSR	Request data, in transit; no application DB
Amazon Web Services	Compute (ECS), storage (S3), CDN (CloudFront), email (SES), secrets	Hosts application, media, and PII at rest
Supabase	Admin console identity + metadata DB	Admin user accounts, roles, SEO/leads metadata
Stripe	Payment processing	

Subprocessor	Function	Data shared / role
		Cardholder data (processed by Stripe); platform stores tokens/IDs only
Google Cloud / Play	In-app purchase verification (service account)	Subscription/purchase verification
Apple App Store	In-app purchase verification	Subscription/purchase verification
Netcore Smartech	Marketing / engagement & events	Contact + activity data
Zoom	Webinar hosting	Webinar registrant data
Firebase Cloud Messaging	Push notifications	Device push tokens
Credly	Digital badges / credentials	Recipient name + credential data
Certificate service	Certificate PDF generation	Recipient + course data
Miles LMS (CPA)	Course/alumni integration	Learner identifiers via signed webhooks

11. Security Posture Summary

The Miles Masterclass platform is built on a defense-in-depth architecture using managed, independently audited cloud providers. Security is enforced at every tier — edge, application, identity, and data — and the platform avoids holding the most sensitive data (card details) by delegating payments to Stripe.

Key controls in place include:

Domain	Controls in place
Transport & headers	TLS everywhere; HSTS (2-year, preload); enforced CSP and full security-header set
Identity & access	JWT with external SSO/OTP; database-enforced admin RBAC (RLS); MFA for administrative access; explicit token lifetimes with rotation and revocation
Application	Restrictive CORS allow-list; CSRF protections; API rate limiting; input validation and size limits; safe (JSON-only) task serialization
Data protection	Encryption in transit and at rest; application-layer field encryption for selected data; Stripe tokenization (no card data stored)
Secrets & pipeline	Runtime secrets from AWS Secrets Manager; least-privilege CI; automated secret, SCA, and SAST scanning; controlled rolling deployments
Monitoring & resilience	Centralized logging with alerting; sensitive-field redaction; uptime monitoring; autoscaling and fail-open caching

Overall posture

The architecture is well-structured, uses reputable managed providers with hardened configurations, and applies layered controls across identity, application, data, and infrastructure. Card data is never stored on-platform. The security program is governed by the companion Written Information Security Program and Incident Response Plan, and mapped to SOC 2, GDPR, and India's DPDP Act in the companion Compliance Mapping.

Appendix A — Technology Inventory

Layer	Technology
Web framework	Angular 21 (SSR via Express), Tailwind CSS 4
Web hosting	Vercel (edge network + serverless SSR)
API framework	Django 5.2, Django REST Framework 3.16, SimpleJWT 5.5
API runtime	Python 3.12, Gunicorn 23 (gthread)
Async	Celery 5.4 + Redis; django-celery-beat
Databases	PostgreSQL (primary), Supabase PostgreSQL (admin)
Storage / CDN	AWS S3 + CloudFront

Layer	Technology
Email	AWS SES
Identity	SimpleJWT, external Miles SSO (HS256), Supabase Auth (admin)
Payments	Stripe (+ Google Play / App Store IAP)
Container / CI-CD	Docker, AWS ECR/ECS, GitHub Actions, Vercel
Secrets	AWS Secrets Manager (runtime), git-ignored .env (local)

END OF DOCUMENT · Security Architecture Overview · Version 1.0 · 22 June 2026 · Confidential — Shareable under NDA. Maintained by the Information Security Office. Review at least annually or upon material architectural change.