# **Product Requirements Document (PRD)**

## **1. Overview**

**Project:** CENA Self‑Tape Club for Actors  
 **Vision:** Provide actors with a structured monthly self‑tape practice cycle—administrators configure prompts, actors submit videos, tutors review and give feedback, actors view results and book sessions—all within a cohesive web app.

## **2. Objectives & Success Metrics**

* **Objective 1:** Enable admins to configure each month’s theme easily.  
   **Metric:** Admin can publish a new theme in ≤5 minutes; adoption by 100% of admin users.
* **Objective 2:** Streamline actor submission and review flow.  
   **Metric:** ≥80% of actors submit by deadline; average time from submission to feedback ≤48 hours.
* **Objective 3:** Facilitate tutor feedback and booking.  
   **Metric:** Tutors return feedback for assigned actors; ≥20 private sessions booked via WhatsApp link.
* **Objective 4:** Deliver a frictionless auth and payment flow.  
   **Metric:** <5% drop-off at sign-up; support three paid plans with conversion ≥3%.

## **3. Stakeholders & Personas**

| **Role** | **Persona** | **Needs & Goals** |
| --- | --- | --- |
| **Admin** | Theme Curator | Easily configure theme, upload assets, monitor system status. |
| **Actor** | Self‑Tape Participant | View monthly theme, download monologues, submit self‑tapes, review feedback. |
| **Tutor** | Acting Coach | Access assigned actors, view submissions, record/upload feedback, manage schedule. |
| **End‑User** | Prospective Subscriber | Discover value on landing page, sign up, choose plan. |

## **4. Scope & Phases**

### **Phase 0: Admin Theme Configuration**

* UI: /admin/theme form (title, description, header image, video URL, PDF uploads).
* API: GET/POST /api/admin/theme (in-memory MVP, later persistent).
* Data: Temporary storage; later integrate DB or object storage.

### **Phase 1: Actor Submission**

* UI: /theme page for signed‑in actors: display theme, show monologue links, submission form (paste YouTube links).
* API: POST /api/theme/submissions; GET /api/theme/submissions?actor=<id>.

### **Phase 2: Tutor Review**

* UI: /dashboard/[tutorId]: list assigned actors with completed submissions.
* UI: /dashboard/[tutorId]/actor/[actorId]: view actor’s submissions, record feedback (video link + transcription), send.
* API: POST /api/feedback; GET /api/feedback?actor=<id>.

### **Phase 3: Actor Feedback & Booking**

* UI: /feedback or enhanced /profile: show tutor info, embedded feedback video, transcription, WhatsApp booking button.
* Integration: WhatsApp link with wa.me/<number>?text=<predefined>.

## **5. Functional Requirements**

### **Authentication & Authorization**

* Clerk integration with email/password + Google SSO.
* Roles: Admin, Tutor, Actor; route guards via Next.js middleware.
* /sign-in, /sign-up, /account, /profile pages provided by Clerk.

### **Payments & Plans (MVP)**

* Three plans (Basic, Pro, Enterprise) displayed in PricingGrid.
* Integrate Stripe Checkout with Brazilian payment methods (PIX).
* Post‑payment webhook to Clerk metadata to grant resource access.

### **Content Management (Admin)**

* Upload image/PDF/file storage (initially in-memory, later S3 or equivalent).
* Embed external YouTube video via URL input.

### **Submission & Feedback Flow**

* Form for actors to submit up to 5 video links.
* Tutor review interface with input fields for feedback link + transcript.
* Notification: email or in-app notifier when feedback posted.

## **6. Non‑Functional Requirements**

* **Performance:** Pages render under 1s; media via CDN.
* **Scalability:** Decouple storage; use serverless functions; plan for growth.
* **Security:** HTTPS everywhere; Clerk‑managed sessions; validate file uploads.
* **Accessibility:** WCAG AA compliance for forms and video embeds.

## **7. Technical Architecture**

* **Frontend:** Next.js App Router, Shadcn UI (Tailwind CSS), Clerk auth.
* **Backend/API:** Next.js API routes (serverless), in-memory MVP → migrate to database (Postgres via Prisma).
* **Storage (MVP):** In-memory → object storage for images/PDFs.
* **Notifications:** Email via SendGrid; explore WebPush for in-app.

## **8. User Flows**

1. **Admin publishes theme** → hits /api/admin/theme → data stored.
2. **Actor visits /theme** → fetch theme via GET /api/admin/theme → views assets → submits link.
3. **Tutor logs in** → visits /dashboard/<id> → views actors with submissions → posts feedback → triggers notification.
4. **Actor checks /feedback** → sees tutor video + transcription + booking link.

## **9. Roadmap & Milestones**

| **Milestone** | **Target Date** | **Deliverable** |
| --- | --- | --- |
| Complete Phase 0 MVP | +1 week | Admin theme configuration form & API |
| Complete Phase 1 MVP | +2 weeks | Actor theme page & submission API |
| Complete Phase 2 MVP | +3 weeks | Tutor dashboard & feedback API |
| Complete Phase 3 MVP | +4 weeks | Actor feedback page & booking link |
| Payment integration | +5 weeks | Stripe checkout & plan gating |

## **10. Next Steps**

1. **Validate Phase 0** with admin users (done).
2. **Prototype Phase 1** actor theme page.
3. **Iterate on feedback** and adjust UI.
4. **Build Phase 2** tutor interface using existing sidebar layout.

*End of PRD*