GAMIFICATION SYSTEM — REPLIT PLANNING PROMPT

PCBancard Field Sales Intelligence Suite (brochuretracker.com)

⚠️ CRITICAL RULES — READ BEFORE DOING ANYTHING

Rule 1: DO NOT BREAK EXISTING FUNCTIONALITY

This application is a production system with active users. Before modifying ANY existing file, component, route, database table, or API endpoint:

- Read the existing code thoroughly first
- Understand what it does and what depends on it
- Make ADDITIVE changes (new files, new tables, new columns) rather than destructive changes
- If you must modify an existing file, make surgical edits — do not rewrite entire files
- Test that existing features still work after your changes

Rule 2: ASK QUESTIONS WHEN UNCERTAIN

If you encounter any of these situations, **STOP and ask me before proceeding**:

- You're unsure how an existing feature stores its data
- You need to modify a shared component and aren't sure what depends on it
- The existing code uses a pattern you don't understand
- You need to choose between multiple implementation approaches that have different tradeoffs
- You're about to delete, rename, or restructure existing code
- You can't find where something is implemented

- A database migration might affect existing data

Rule 3: SCOPE — ONLY TOUCH THESE FEATURES

This project is ONLY about adding gamification (progress tracking, XP, badges, certificates) to the features listed below. Do NOT modify, refactor, or “improve” anything outside this scope:

IN SCOPE (the training/coaching features shown in the screenshots):

- Sales Coach hub
- Daily Edge
- Teach Me the Presentation (8 modules, ~25 lessons)
- Interactive AI Training (Live Roleplay Simulator, Objection Gauntlet, Scenario Trainer, Delivery Analyzer)
- EquipIQ (AI equipment advisor, equipment knowledge quizzes)
- 2026 Sales Process (Phases: Prospecting, Discovery, Proposal & Close, Onboarding + tabs: Practice, Objections, Industries, Contacts)
- AI-Powered Prospecting / Sales Spark
- Agent Profile page (add progress dashboard here)
- Admin/Team Management (add agent progress views here)

OUT OF SCOPE — DO NOT TOUCH:

- Brochure tracking / QR scanning
- Merchant CRM / Merchant Profiles
- Statement Analyzer
- Proposal Generator
- Inventory Control
- Referral Management
- Deal Pipeline
- Authentication / Login system (don’t change how auth works)
- E-signature system
- Any feature not listed in the IN SCOPE section above

PROJECT OVERVIEW

What We're Building

A "Progress + Points + Badges + Certificates" gamification layer that sits on top of the existing training features. The goal is to make agents WANT to interact with training daily, track their competency, and give managers visibility into agent development.

What Already Exists (Technical Context)

Stack: React frontend, Node.js/Express backend, PostgreSQL database, hosted on Replit

Existing Database Tables (DO NOT MODIFY these — only ADD to them or create new tables):

- `users` — agent accounts (id, email, name, role, created_at)
- `user_permissions` — role-based access control with columns: user_id, organization_id, role (admin/manager/agent), agent_stage (trainee/active/senior), feature_overrides (JSONB), set_by, notes
- `chat_sessions` — roleplay sessions (id, user_id, scenario_type, started_at, ended_at, score, feedback JSONB)
- `chat_messages` — individual messages within roleplay sessions
- `knowledge_content` — knowledge base for AI coaching

Existing Permission System:

- Roles: admin, manager, agent
- Agent stages: trainee, active, senior
- Feature overrides stored as JSONB
- Permission middleware already protects routes
- Admin can toggle feature access per user

Existing Admin Views:

- Team Management panel exists
- User panel exists where admin can view individual agents
- Permission toggle section exists per agent

Things That DON'T Exist Yet (you need to build):

- Progress tracking for lesson/module completion (Teach Me the Presentation shows a "Completed" state in the UI, but you need to investigate how/where this is stored — it may be client-side only)
- Quiz score persistence for EquipIQ
- Roleplay scoring persistence (the `chat_sessions.score` and `chat_sessions.feedback` columns exist but may not be populated)
- Any XP/points system
- Any badge/achievement system
- Any certificate generation
- Any agent-facing progress dashboard
- Any admin view of agent training progress

IMPORTANT: Before building, examine the existing codebase to determine:

1. How does "Teach Me the Presentation" currently track lesson completion? (Is there a DB table? LocalStorage? React state?)
2. How does the EquipIQ quiz currently work? Where do quiz results go?
3. What data gets written to `chat_sessions` after a roleplay? Is `score` populated?
4. What's currently on the Profile page?
5. What existing components/patterns does the app use for cards, modals, progress bars, etc.?
6. What PDF library (if any) is already installed? Check `package.json` for `jspdf` , `pdfkit` , `@react-pdf/renderer` , `puppeteer` , etc.

Use whatever you find. Match existing patterns. Don't introduce new UI libraries unless nothing exists.

THE 5-LEVEL RECOGNITION LADDER

Each level represents 20% of total possible progress. Agents advance by earning XP AND meeting minimum Skill Score thresholds.

Level 1: Field Scout (0–20%)

- **Icon idea:** Compass
- **Color:** Slate/gray
- **Tagline:** "You've entered the arena."
- **What it means:** Agent has completed onboarding basics and first training reps
- **Unlocks:** Access to all training modules, Daily Edge, Sales Spark

Level 2: Pipeline Builder (20–40%)

- **Icon idea:** Stacked blocks / pipeline
- **Color:** Blue
- **Tagline:** "Building daily motion."
- **What it means:** Agent is prospecting regularly and doing structured practice
- **Unlocks:** Advanced roleplay personas, streak bonus multipliers

Level 3: Objection Breaker (40–60%)

- **Icon idea:** Shield with crack / shield-break
- **Color:** Orange/amber
- **Tagline:** "Calm under pressure."
- **What it means:** Agent can handle objections and maintain conversation control
- **Unlocks:** Scenario Trainer advanced mode, Delivery Analyzer access

Level 4: Closer (60–80%)

- **Icon idea:** Handshake / checkmark
- **Color:** Green
- **Tagline:** "Landing commitments."
- **What it means:** Agent can run a complete presentation flow and close
- **Unlocks:** Priority support queue, advanced proposal tools (if available)

Level 5: Residual Architect (80–100%)

- **Icon idea:** Crown / blueprint

- **Color:** Purple/gold
- **Tagline:** “Building long-term portfolio value.”
- **What it means:** Agent can repeatably produce outcomes and mentor others
- **Unlocks:** Full platform access, mentor badge, certificate of mastery

---

SCORING MODEL — TWO LAYERS

Layer 1: XP (Points) — “Work Done”

XP is an additive ledger that only goes up. It measures effort and engagement. Every meaningful action earns XP. Every XP award is an immutable ledger entry (prevents disputes, enables future monetization).

XP Awards by Feature:

Teach Me the Presentation:

- Lesson completed: 25 XP
- Lesson completed with quick-check score ≥ 80%: +10 XP bonus
- Lesson completed with quick-check score ≥ 90%: +25 XP bonus (instead of the +10)
- Module completed (all lessons in one of the 8 modules): +150 XP bonus
- All 8 modules completed: +500 XP milestone bonus

Interactive AI Training:

- Live Roleplay session completed: 60 XP base
    - Performance bonus: +0 to +40 XP based on AI rubric score (if roleplay scoring exists)
    - Anti-gaming: Session must have ≥ 6 conversation turns OR ≥ 3 minutes duration to count
- Objection Gauntlet — per objection completed: 15 XP
    - All 12 objections cleared in one run: +50 XP bonus
- Scenario Trainer — per scenario completed: 40 XP
    - “Best outcome” path chosen: +20 XP bonus

- Delivery Analyzer — full run-through completed: 80 XP
  - All required presentation stages detected: +20 XP bonus

EquipIQ:

- Equipment knowledge quiz completed: 50 XP base
  - Score bonus: (score_percentage ÷ 2) XP — example: 92% score = +46 XP
- Equipment advisor chat session (meaningful interaction): 10 XP

2026 Sales Process:

- Sales process phase reviewed/verified (Prospecting, Discovery, Proposal & Close, Onboarding): 30 XP each
- All 4 phases completed: +100 XP bonus
- Practice session completed (Interactive Sales Practice within 2026 Sales Process): 40 XP

Daily Edge:

- Daily Edge completed: 15 XP per day
- Streak bonuses (consecutive days):
  - 3-day streak: +25 XP
  - 7-day streak: +100 XP
  - 14-day streak: +250 XP
  - 30-day streak: +500 XP
- Streak breaks: Streak counter resets to 0. Previous streak bonus XP is kept (already in ledger).

Sales Spark:

- Sales Spark prompt submitted: 10 XP (max 3 per day)

Anti-Gaming Caps:

- Maximum 400 XP per day from Teach Me the Presentation lessons/modules (prevents grinding through all lessons meaninglessly in one sitting — but still allows a productive binge day)
- Roleplay XP only counts if session meets minimum turn/time thresholds (stated above)

- Same quiz cannot award XP more than once per 24 hours (retakes for practice are fine, but XP only on first daily attempt)
- Sales Spark capped at 3 XP-earning submissions per day

Layer 2: Skill Score (0–100) — "How Good They Are"

Skill Score is a weighted proficiency metric that managers can trust. Unlike XP (which only goes up), Skill Score reflects CURRENT competency and can go down if an agent stops practicing or scores poorly on recent attempts.

Weighted Components:

Component	Weight	What It Measures	Source
Presentation Mastery	30%	Teach Me the Presentation completion % + quiz scores	Lesson completions + quick-check scores
Objection Handling	25%	Gauntlet performance + Scenario Trainer outcomes	Gauntlet scores + scenario best-outcome rates
Roleplay Performance	25%	AI roleplay rubric scores (rolling average of last 10 sessions)	chat_sessions.score
Equipment Knowledge	10%	EquipIQ quiz scores (best recent score per vendor)	Quiz results
Process Knowledge	10%	2026 Sales Process completion + practice scores	Phase completions + practice session scores

Calculation Notes:

- Use rolling averages where applicable (last 10 roleplays, last 5 quizzes per category) so the score reflects recent performance, not all-time history
- If an agent has no data for a component, that component contributes 0 to their score (not excluded from the weighted average — this incentivizes covering all areas)
- Recalculate on every relevant event (not on a schedule)
- Store the computed score and its component breakdown so dashboards can show drill-down

Badge Progression Thresholds:

Level	Name	XP Minimum	Skill Score Minimum	Completion %

1	Field Scout	0	0	0–20%
2	Pipeline Builder	500	20	20–40%
3	Objection Breaker	1500	40	40–60%
4	Closer	3000	55	60–80%
5	Residual Architect	5000	75	80–100%

"Completion %" = percentage of all trackable training items completed at least once.

An agent must meet BOTH the XP minimum AND the Skill Score minimum to earn the badge. If their Skill Score drops below the threshold (because recent roleplay scores declined, for example), they keep the badge but get a visual indicator that their score needs attention.

EVENT TAXONOMY — EVERY TRACKABLE ACTION

Each event below should be logged as an immutable record. The system needs to capture these events from existing features. **IMPORTANT:** You will need to examine the existing code for each feature to determine WHERE to hook into and emit these events. If a feature doesn't currently have a clear completion signal, ASK ME how it should work rather than guessing.

Event: `lesson_completed`

- **Source:** Teach Me the Presentation
- **Fields:** user_id, lesson_id, module_id, quick_check_score (nullable), completed_at
- **Validation:** Lesson content must have been rendered/opened for at minimum 30 seconds (prevents instant-clicking through)
- **UI Feedback:** Green checkmark on lesson card, brief "🎉 +25 XP" toast notification, progress bar update on module card

Event: `module_completed`

- **Source:** Teach Me the Presentation
- **Trigger:** Automatically when all lessons in a module show `lesson_completed`
- **Fields:** user_id, module_id, completed_at

- **UI Feedback:** Module card shows completed state, "+150 XP Module Bonus!" celebration toast, check if this triggers a badge level-up

Event: `quiz_completed`

- **Source:** EquipIQ Equipment Knowledge Quiz
- **Fields:** user_id, quiz_id, vendor_focus, difficulty, score_percent, answers_json, completed_at
- **Validation:** Quiz must have been started (answers cannot be submitted without questions being displayed)
- **UI Feedback:** Score display with "+50 XP + [bonus] XP" toast, grade indicator (A/B/C/D/F based on percentage)

Event: `roleplay_completed`

- **Source:** Interactive AI Training → Live Roleplay Simulator
- **Fields:** user_id, session_id (FK to chat_sessions), persona_id, difficulty, turn_count, duration_seconds, ai_score (nullable), completed_at
- **Validation:** turn_count ≥ 6 OR duration_seconds ≥ 180
- **UI Feedback:** Session summary card with score breakdown, "+60 XP [+ bonus]" toast

Event: `gauntlet_completed`

- **Source:** Interactive AI Training → Objection Gauntlet
- **Fields:** user_id, objections_attempted, objections_passed, perfect_run (boolean), scores_per_objection (JSON), completed_at
- **Validation:** At least 1 objection must have been attempted with a response submitted
- **UI Feedback:** Gauntlet results card, per-objection pass/fail, "+[XP] earned" toast, perfect run celebration if all 12 cleared

Event: `scenario_completed`

- **Source:** Interactive AI Training → Scenario Trainer
- **Fields:** user_id, scenario_id, choices_made (JSON), best_outcome_achieved (boolean), completed_at
- **Validation:** All decision points in the scenario must have received a selection

- **UI Feedback:** Outcome reveal with explanation, "+40 XP [+ bonus]" toast

Event: `delivery_analysis_completed`

- **Source:** Interactive AI Training → Delivery Analyzer
- **Fields:** user_id, stages_detected (JSON array), stages_total, coverage_percent, ai_feedback (text), completed_at
- **Validation:** User must have submitted content for analysis (not empty)
- **UI Feedback:** Stage coverage visualization (which stages were hit/missed), "+80 XP [+ bonus]" toast

Event: `sales_process_phase_completed`

- **Source:** 2026 Sales Process → Phases tab
- **Fields:** user_id, phase_name (prospecting/discovery/proposal_close/onboarding), completed_at
- **Validation:** Examine existing code to determine what constitutes "completing" a phase view. If there's no current mechanism, ASK ME.
- **UI Feedback:** Phase icon turns green/checked in the progress stepper, "+30 XP" toast

Event: `sales_process_practice_completed`

- **Source:** 2026 Sales Process → Practice tab → Interactive Sales Practice
- **Fields:** user_id, stage_practiced, completed_at
- **Validation:** Practice session must have had meaningful interaction
- **UI Feedback:** "+40 XP" toast

Event: `daily_edge_completed`

- **Source:** Daily Edge
- **Fields:** user_id, focus_topic, completed_at
- **Validation:** Examine existing code to determine what constitutes "completing" a Daily Edge. The user likely needs to have opened/read/listened to it. If unclear, ASK ME.
- **UI Feedback:** "+15 XP" toast, streak counter update ("🔥 3-day streak!")

Event: `streak_awarded`

- **Source:** System (triggered by daily_edge_completed when streak threshold is hit)
- **Fields:** user_id, streak_length, bonus_xp, awarded_at
- **UI Feedback:** Celebration modal/toast ("🔥 7-Day Streak! +100 XP Bonus!")

Event: `badge_earned`

- **Source:** System (triggered when XP + Skill Score + Completion % all meet threshold)
- **Fields:** user_id, badge_level (1-5), badge_name, earned_at
- **UI Feedback:** Full-screen or modal celebration ("🎉 You've earned Pipeline Builder!"), badge icon displayed prominently

Event: `sales_spark_used`

- **Source:** Sales Spark / AI-Powered Prospecting
- **Fields:** user_id, prompt_text (optional, for analytics), completed_at
- **Validation:** User submitted a prompt and received a response
- **UI Feedback:** "+10 XP" subtle toast (max 3 per day)

DATABASE SCHEMA — NEW TABLES

IMPORTANT: These are NEW tables. Do NOT modify existing tables. Create these alongside what exists. Use the existing `users.id` as the foreign key reference.

Before creating migrations, check if there's an existing migration system (files named like `001_*.sql`, `002_*.sql` in a migrations folder, or a migration runner). If so, follow that pattern. If not, create a single SQL file and ASK ME how I want to run it.

```
-- ============================================================
-- GAMIFICATION SCHEMA — ADDITIVE ONLY
-- Does not modify existing tables
-- ============================================================

-- 1. TRAINING PROGRESS — tracks completion of individual training items
CREATE TABLE IF NOT EXISTS training_progress (
  id SERIAL PRIMARY KEY,
  user_id INTEGER NOT NULL REFERENCES users(id) ON DELETE CASCADE,

  -- What was completed
  item_type VARCHAR(50) NOT NULL, -- 'lesson', 'module', 'quiz', 'roleplay', 'gau
  item_id VARCHAR(100) NOT NULL,  -- identifier for the specific item (lesson slu

  -- Context
  parent_id VARCHAR(100),         -- e.g., module_id for a lesson, vendor for a q
  difficulty VARCHAR(20),         -- if applicable

  -- Scores
  score_percent DECIMAL(5,2),     -- 0.00 to 100.00
  score_raw JSONB,                -- raw scoring data (per-objection scores, stag

  -- Session linkage
  session_id INTEGER REFERENCES chat_sessions(id), -- for roleplays, link to exis

  -- Timing
  duration_seconds INTEGER,
  turn_count INTEGER,             -- for roleplays

  -- Metadata
  completed_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

  -- Indexes will go below
  CONSTRAINT valid_item_type CHECK (item_type IN (
    'lesson', 'module', 'quiz', 'roleplay', 'gauntlet', 'scenario',
    'delivery_analysis', 'sales_process_phase', 'sales_process_practice',
    'daily_edge', 'sales_spark'
  ))
);

CREATE INDEX idx_training_progress_user ON training_progress(user_id);
CREATE INDEX idx_training_progress_type ON training_progress(user_id, item_type);
CREATE INDEX idx_training_progress_item ON training_progress(user_id, item_type,
CREATE INDEX idx_training_progress_date ON training_progress(user_id, completed_a


-- 2. XP LEDGER — immutable log of every point earned
CREATE TABLE IF NOT EXISTS xp_ledger (
  id SERIAL PRIMARY KEY,
  user_id INTEGER NOT NULL REFERENCES users(id) ON DELETE CASCADE,

  -- What earned the XP
  event_type VARCHAR(50) NOT NULL, -- matches event names from taxonomy above
  event_id INTEGER REFERENCES training_progress(id), -- link to the training_prog

  -- Points
  xp_base INTEGER NOT NULL DEFAULT 0,      -- base XP for the action
  xp_bonus INTEGER NOT NULL DEFAULT 0,     -- bonus XP (score bonus, streak bonus,
  xp_total INTEGER NOT NULL DEFAULT 0,    -- xp_base + xp_bonus (denormalized for

  -- Why the bonus
  bonus_reason VARCHAR(200),              -- human-readable: "Score ≥ 90%", "7-da

  -- Anti-gaming
  daily_cap_category VARCHAR(50),         -- 'teach_me', 'roleplay', 'sales_spark

  -- Metadata
  created_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()
);

CREATE INDEX idx_xp_ledger_user ON xp_ledger(user_id);
CREATE INDEX idx_xp_ledger_user_date ON xp_ledger(user_id, created_at DESC);
CREATE INDEX idx_xp_ledger_daily_cap ON xp_ledger(user_id, daily_cap_category, cr


-- 3. SKILL SCORES — computed proficiency snapshots
CREATE TABLE IF NOT EXISTS skill_scores (
  id SERIAL PRIMARY KEY,
  user_id INTEGER NOT NULL REFERENCES users(id) ON DELETE CASCADE,

  -- Overall score
  total_score DECIMAL(5,2) NOT NULL DEFAULT 0, -- 0.00 to 100.00

  -- Component breakdown
  presentation_mastery DECIMAL(5,2) DEFAULT 0,  -- 30% weight
  objection_handling DECIMAL(5,2) DEFAULT 0,    -- 25% weight
  roleplay_performance DECIMAL(5,2) DEFAULT 0,  -- 25% weight
  equipment_knowledge DECIMAL(5,2) DEFAULT 0,   -- 10% weight
  process_knowledge DECIMAL(5,2) DEFAULT 0,     -- 10% weight

  -- Completion tracking
  total_items_available INTEGER DEFAULT 0,
  total_items_completed INTEGER DEFAULT 0,
  completion_percent DECIMAL(5,2) DEFAULT 0,

  -- XP summary
  total_xp INTEGER DEFAULT 0,

  -- Current badge level
  badge_level INTEGER DEFAULT 0, -- 0=none, 1=Scout, 2=Builder, 3=Breaker, 4=Clos
  badge_name VARCHAR(50),

  -- Streak
  current_streak_days INTEGER DEFAULT 0,
  longest_streak_days INTEGER DEFAULT 0,
  last_daily_edge_date DATE,

  -- Timestamps
  calculated_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

  UNIQUE(user_id)
);

CREATE INDEX idx_skill_scores_user ON skill_scores(user_id);
CREATE INDEX idx_skill_scores_badge ON skill_scores(badge_level);
CREATE INDEX idx_skill_scores_xp ON skill_scores(total_xp DESC);


-- 4. BADGES EARNED — historical record of badge achievements
CREATE TABLE IF NOT EXISTS badges_earned (
  id SERIAL PRIMARY KEY,
  user_id INTEGER NOT NULL REFERENCES users(id) ON DELETE CASCADE,

  badge_level INTEGER NOT NULL,   -- 1-5
  badge_name VARCHAR(50) NOT NULL,

  -- Snapshot at time of earning
  xp_at_award INTEGER,
  skill_score_at_award DECIMAL(5,2),
  completion_at_award DECIMAL(5,2),

  earned_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

  UNIQUE(user_id, badge_level)
);


-- 5. CERTIFICATES — generated certificate records
CREATE TABLE IF NOT EXISTS certificates (
  id SERIAL PRIMARY KEY,
  user_id INTEGER NOT NULL REFERENCES users(id) ON DELETE CASCADE,

  badge_level INTEGER NOT NULL,
  badge_name VARCHAR(50) NOT NULL,
  agent_name VARCHAR(255) NOT NULL,

  -- Verification
  verification_code VARCHAR(50) UNIQUE NOT NULL, -- random unique code for QR/ver

  -- PDF storage
  pdf_url TEXT,                    -- URL/path to generated PDF
  generated_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()
);

CREATE INDEX idx_certificates_user ON certificates(user_id);
CREATE INDEX idx_certificates_verify ON certificates(verification_code);


-- 6. STREAKS — daily engagement tracking
CREATE TABLE IF NOT EXISTS streak_log (
  id SERIAL PRIMARY KEY,
  user_id INTEGER NOT NULL REFERENCES users(id) ON DELETE CASCADE,
  activity_date DATE NOT NULL,

  -- What counted for this day
  daily_edge_completed BOOLEAN DEFAULT FALSE,
  any_training_completed BOOLEAN DEFAULT FALSE,

  created_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

  UNIQUE(user_id, activity_date)
);

CREATE INDEX idx_streak_log_user_date ON streak_log(user_id, activity_date DESC);
```

Example JSON Objects

training_progress row (lesson completion):

```
{
  "id": 1,
  "user_id": 42,
  "item_type": "lesson",
  "item_id": "why-this-presentation-works",
  "parent_id": "module-1-psychology-foundation",
  "difficulty": null,
  "score_percent": 92.00,
  "score_raw": { "quick_check_answers": [true, true, false, true, true], "score":
  "session_id": null,
  "duration_seconds": 245,
  "turn_count": null,
  "completed_at": "2026-02-06T14:30:00Z"
}
```

xp_ledger row:

```
{
  "id": 1,
  "user_id": 42,
  "event_type": "lesson_completed",
  "event_id": 1,
  "xp_base": 25,
  "xp_bonus": 25,
  "xp_total": 50,
  "bonus_reason": "Quick-check score ≥ 90%",
  "daily_cap_category": "teach_me",
  "created_at": "2026-02-06T14:30:00Z"
}
```

skill_scores row:

```
{
  "id": 1,
  "user_id": 42,
  "total_score": 47.50,
  "presentation_mastery": 65.00,
  "objection_handling": 40.00,
  "roleplay_performance": 52.00,
  "equipment_knowledge": 30.00,
  "process_knowledge": 25.00,
  "total_items_available": 80,
  "total_items_completed": 34,
  "completion_percent": 42.50,
  "total_xp": 1850,
  "badge_level": 2,
  "badge_name": "Pipeline Builder",
  "current_streak_days": 5,
  "longest_streak_days": 12,
  "last_daily_edge_date": "2026-02-06",
  "calculated_at": "2026-02-06T14:30:00Z"
}
```

UI/UX RECOMMENDATIONS — MOBILE-FIRST

Agent Dashboard (add to Profile page or create new Dashboard page)

Top Section — Status at a Glance:

- Circular progress ring showing completion % (large, centered)

- Current badge icon + name + level below the ring
- Next badge preview with "X% to go" indicator
- Current XP total displayed prominently
- Streak flame icon with day count

Middle Section — Today's Activity:

- "XP Today: [amount] / 400 cap" progress bar
- Next recommended action card ("Continue Module 3: Solution Positioning" or "Complete today's Daily Edge")
- Quick-launch buttons for the most impactful next step

Bottom Section — Badge Ladder:

- Horizontal or vertical stepper showing all 5 levels
- Filled/colored for earned badges, dimmed for unearned
- Tap on any badge to see requirements and progress toward it

Style Notes:

- Match the existing app's visual language (rounded cards, soft shadows, the blue/purple/orange/green accent colors visible in the screenshots)
- Don't make it look like a children's game — keep it professional but engaging
- Use the same card component pattern the app already uses
- Icons: use emoji or simple SVG icons consistent with what's already in the app (I can see the app uses emoji-style icons in many places)

Admin / Manager View (add to Team Management)

Per-Agent Summary Row (in the existing agent list):

- Add columns or badges showing: current badge level icon, Skill Score number, XP total, last active date, current streak
- Make each row clickable to drill into detail

Agent Detail View (new section within existing agent profile/management page):

- Skill Score breakdown chart (spider/radar chart or stacked horizontal bars showing all 5 components)

- XP history over time (simple line chart — last 30 days)
- Training completion checklist (which modules, quizzes, roleplays have been done)
- Recent activity feed (last 20 events from training_progress)
- Badge progression timeline
- Button to download/view certificates earned

Cohort View (optional but valuable):

- Leaderboard sorted by XP or Skill Score
- Filter by date range, badge level, agent stage
- Average Skill Score across all agents
- Completion rate by module (which modules are agents getting stuck on?)

CERTIFICATES

Certificate Content

Each certificate should contain:

- **Header:** PCBancard Field Sales Intelligence Suite
- **Title:** Certificate of Achievement (or level-specific: "Certificate of Mastery" for Level 5)
- **Badge:** Level name and icon (e.g., "Pipeline Builder" with stacked blocks icon)
- **Agent Name:** Full name from user profile
- **Date Earned:** Formatted date
- **Verification Code:** Unique alphanumeric code (e.g., "PCB-2026-A7X9K2")
- **QR Code** (optional): Encodes a URL like `brochuretracker.com/verify/[code]` that can confirm the certificate is authentic
- **Tagline:** Brief description of what this level represents

Certificate Generation

- Check `package.json` for existing PDF libraries first
- If none exist, suggest one of: `jspdf` (client-side), `pdfkit` (server-side), or `@react-`

`pdf/renderer` (React-based) — pick whichever fits the existing architecture best. ASK ME if you're unsure.

- Store generated PDF as a file or blob, link from `certificates` table
- Agent can download from their profile/dashboard
- Admin can view/download from the agent management page

When to Generate

- Automatically when a badge is earned
- Agent can also manually request re-generation from their profile
- Admin can trigger generation from the agent detail view

PERMISSIONS + VISIBILITY

How Gamification Interacts with Existing Permissions

- If a feature is toggled OFF for an agent (e.g., EquipIQ is disabled), that feature's training items should NOT count toward their completion percentage or Skill Score calculation. Adjust the "total items available" denominator accordingly.
- When a feature is toggled back ON, the agent doesn't lose any previously earned XP (it's immutable), but their Skill Score and completion % recalculate to include the newly available items.
- Daily Edge, Sales Spark, and basic training features should always be trackable regardless of permission level (since they're available to all stages).

Visibility Rules

- **Agents** see: their own dashboard, their own XP/badges/scores, their own certificates, a leaderboard (optional — if we decide to show it)
- **Managers** see: their own data + summary cards for all agents assigned to them, drill-down into any agent's training details
- **Admins** see: everything — all agents, all data, cohort analytics, full audit trail
- Respect the existing permission middleware. Add new route guards following the same pattern the app already uses.

BUILD ORDER — IMPLEMENTATION CHECKLIST

Do these in order. Each phase should be working before starting the next.

Phase 1: Foundation (Database + Scoring Engine)

- Examine existing codebase to understand current data storage patterns
- Create database migration with all new tables above
- Run migration safely (without affecting existing tables)
- Build server-side scoring engine:
    - `calculateSkillScore(userId)` function
    - `awardXP(userId, eventType, eventId, xpBase, xpBonus, bonusReason)` function
    - `checkBadgeProgression(userId)` function
    - `checkDailyCap(userId, category)` function
    - `updateStreak(userId)` function
- Build API routes for:
    - `GET /api/gamification/profile/:userId` — get agent's full gamification profile
    - `GET /api/gamification/leaderboard` — get ranked list (with permission checks)
    - `GET /api/gamification/admin/agent/:userId` — get detailed agent view (admin/manager only)
    - `POST /api/gamification/events` — internal endpoint to log events (called by other features, not directly by clients)
- ASK ME to verify the schema and API look correct before proceeding

Phase 2: Event Hooks (Connect Features to Scoring)

- **Teach Me the Presentation:** Find where lesson completion happens in the code. Add event emission for `lesson_completed` and `module_completed`. If completion isn't currently tracked, build it (likely needs a click/button that marks a lesson done + minimum time check).
- **EquipIQ Quiz:** Find where quiz results are calculated. Add event emission for `quiz_completed`. Ensure scores are persisted.

- **Interactive AI Training:** Find where roleplay sessions end, gauntlet results are shown, scenarios complete, and delivery analysis finishes. Add event emissions for each.
- **2026 Sales Process:** Find what interaction exists in the Phases and Practice tabs. Add appropriate event emissions. If "completing" a phase isn't currently a concept, ASK ME how it should work.
- **Daily Edge:** Find what "completing" a Daily Edge means in the current code. Add event emission for `daily_edge_completed`.
- **Sales Spark:** Find where a Sales Spark prompt gets submitted and a response is returned. Add event emission for `sales_spark_used`.
- For EACH feature above: if you can't find a clear completion point in the code, ASK ME rather than guessing.

Phase 3: Agent-Facing UI

- Build the agent progress dashboard component (progress ring, XP, streak, badge ladder)
- Add XP toast notifications (subtle, non-blocking — appears briefly when XP is earned)
- Add badge level-up celebration (modal or prominent toast when a new badge is earned)
- Integrate dashboard into the Profile page or create a new nav item
- Match existing app styling — use the same component patterns, colors, and spacing

Phase 4: Admin/Manager Views

- Add gamification summary data to existing agent list in Team Management
- Build agent training detail view (accessible from the existing agent management page)
- Add Skill Score breakdown visualization
- Add training completion checklist view
- Add recent activity feed
- Add cohort/leaderboard view (if time allows)

Phase 5: Certificates

- Determine PDF library to use (check existing dependencies first)
- Build certificate template with all required fields

- Build generation endpoint
- Add download button to agent profile
- Add admin view/download capability
- Generate verification codes

Phase 6: Testing + Polish

- Test all event hooks — does every feature correctly emit events?
- Test XP calculations — do all formulas produce correct results?
- Test daily caps — does the 400 XP/day cap on Teach Me work?
- Test streak logic — do consecutive days count correctly? Do gaps reset?
- Test badge progression — do badges award at correct thresholds?
- Test permission integration — do toggled-off features correctly exclude from scores?
- Test admin views — can managers see their agents? Can admins see everyone?
- Test on mobile — is the dashboard usable on a phone screen?
- Verify existing features still work exactly as before

FINAL REMINDERS

1. **ADDITIVE CHANGES ONLY.** New files, new tables, new components. Do not rewrite existing features.
2. **ASK before modifying shared code.** If you need to add an event hook inside an existing component, make the smallest possible change.
3. **Match existing patterns.** Look at how the app currently handles state, API calls, component styling, and database queries. Do the same thing.
4. **Mobile-first.** Every UI element must work well on a phone. This is a field sales app — agents use it on the go.
5. **Professional, not childish.** These are adult salespeople. The gamification should feel like an achievement system, not a kids' game. Think LinkedIn skill badges, not Candy Crush.
6. **When in doubt, ASK ME.** I would rather answer 20 questions than debug a broken

system.