IRCTC IRM Super App — Feature Overview

IRCTC IRM Super App — Feature Overview

A single Streamlit app that unifies forecasting, fare optimization, seat/coach operations, compliance constraints, revenue simulation, and audit-ready persistence for IRCTC trains.

Key capabilities (by tab)

Dashboard
Booking pace curve (0–30 days to departure).
Capacity snapshot per class (total, effective after overbooking + no-shows).

Forecast & EMSR
Baseline demand by OD with season/competition controls.
Optional ML forecast from uploaded CSV (XGBoost/GBR).
EMSR-b protection levels & booking limits per fare bucket; colorful OD heatmap.

Multi-leg Optimizer
OD allocations that respect segment capacity.
Priority rules: Longest first, Shortest first, Proportional to .
Shows allocated seats, spill, and remaining segment capacity.

Coach Revenue (MC/PRS)
Monte Carlo revenue per coach using PRS fare rules: distance slabs + per-km base, slab multipliers.
Superfast surcharge, reservation fee, GST.
Dynamic surge by occupancy bands.
Tatkal / Premium Tatkal windows and uplifts.
Outputs mean/P10/P90 revenue and mean load; bar viz per coach.

Seatmaps (True & Tools)
True berth patterns: SL/3A (LB/MB/UB + SL/SU), 2A (LB/UB + SL/SU), 1A cabins, CC 3–2.
Women-only bays (ratio or PRS config).
Ops tools: Swap seats and RAC→CONF squeeze, plus import/export.

Family Seating & Conflicts
Synthetic family allocator with child/elderly lower-berth protection and a women-only booking switch (all-female enforcement).
Conflict analytics: severity-weighted heatmap (child_upper=2.0, elder_upper=1.5, women_only=3.0, mixed_gender=4.0).
Seat-level drill-down for violations.

Rules (YAML)
View/edit defaults (pricing multipliers, Tatkal policy, overbooking).
Download/upload rules for ops governance.

DB Sync (PostgreSQL)
Version rules and version seatmaps; save family allocations linked to seatmap_version_id for audit trails.
SQLAlchemy models: rules_versions, seatmap_versions, seat_rows, family_allocations.

REST Hooks
Calls FastAPI optimizer endpoints (/emsr, /multileg) with live payloads.
Reads PRS endpoints when available (/fare_rules, /inventory).

Exports
One-click ZIP of CSV outputs (forecast, EMSR, multileg, revenue).
Optional PDF summary (ReportLab).

How the core logic works (quick)
Demand: baseline function with seasonality + competition modifiers; optional ML model when CSV is provided.
EMSR-b: computes protection levels and booking limits per fare bucket using cumulative demand mean/variance and a critical fractile step.
Multi-leg: greedily assigns OD seats using selected priority rule under per-segment capacity.
Revenue (per coach): Monte Carlo sim of extra bookings & no-shows; each showing passenger draws an OD distance → fare via PRS tariff (distance slab, surcharges, Tatkal/PT window, surge, GST). Aggregates to P10/mean/P90.

Integrations & configuration
PRS/UTS:
GET /fare_rules (optional) → overrides default tariff.
GET /inventory?train=&date;= → women-only bays & coach config.
Set PRS API base URL and token in the sidebar.
Optimizer API: set base URL (defaults to http://localhost:8010).
Database: set DATABASE_URL to enable versioning/persistence.

Audit & governance
Seatmaps saved as immutable versions; family allocations carry a foreign key to seatmap_version_id.
Rules can be versioned and re-loaded.
Conflict heatmaps + seat-level tables support compliance audits (women-only, child/elderly protection).

Demo tips
Start with Dashboard, tweak seasonality/competition to see pace and capacity impact.
In Forecast & EMSR, upload a small CSV to show ML lift; then show EMSR booking limits shifting.
Use Multi-leg to illustrate segment feasibility and spill.
Jump to Coach Revenue and increase hours-to-departure / occupancy to show PRS dynamic surge effects.
In Seatmaps, toggle women-only bays and demo Swap / RAC→CONF.
In Family Seating, enable women-only booking and show how conflicts/heatmap respond; then Persist in DB Sync and note the linked version ID.

Runbook (quick)
pip install streamlit pandas numpy altair pyyaml requests
pip install sqlalchemy psycopg2-binary reportlab scipy scikit-learn xgboost
streamlit run irctc_rirm_super_app.py