✈ Salesforce Airline Management System AMS (Developer Edition – Free Org)

Phase 1: Problem Understanding & Industry Analysis

- Scope small: focus on Ticket Booking, Flight Scheduling, Passenger Management, Refund Requests.
- Stakeholders in Dev Org →
- Admin (you)
- Booking Agent (internal user)
- Passenger (Customer Community not available → use Contact records).
- Keep use cases simple: single airline, limited routes, 1–2 demo flights.

___

Phase 2: Org Setup & Configuration

- Edition: Developer Edition (free).
- Company profile → Skyline Airlines.
- Users → Admin + 1 Booking Agent (max allowed).
- Roles & Profiles →
- Admin
- Airline Staff (Agent)
- Passenger (use Contact records only, no login).
- OWD → Bookings private, Flights public.
- No sandbox → use Change Sets in same org or unmanaged package for backup/demo.

___

Phase 3: Data Modeling & Relationships

Use Custom Objects (all free in Dev Org):

- Flight ✈ (Custom)
- Passenger (Contact with extra fields)
- Booking (Junction → Flight ↔ Passenger)
- Ticket (Custom, lookup Booking)
- Payment (Custom, lookup Booking)

Relationships:

- Flight → Booking (Master-Detail).
- Passenger → Booking (Lookup).
- Ticket → Booking (Lookup).

Keep it lean → Don't model Crew, Airport, Loyalty unless needed (to save object/field limits).

Phase 4: Process Automation

- Validation Rule: Prevent booking if flight capacity full.
- Flow Builder (instead of Workflow, since Workflow is legacy):
- Auto-create Ticket after Booking confirmed.
- Send Email Alert (Booking Confirmation).
- Auto-cancel Booking if not paid in 24 hrs (Scheduled Flow).

___

Phase 5: Apex Programming

In Dev Edition, Apex is free:

- Trigger: Update available seats after booking.
- Trigger: Prevent duplicate booking.
- Batch Apex: Archive old flights (demo with a few test records).
- Future Method: Send async notifications (just log in debug).
- Test Classes: Write basic unit tests to ensure 75%+ coverage.

___

Phase 6: User Interface

- Lightning App Builder → Airline Agent Console (1 App).
- Record Page → Passenger 360 view (Bookings, Tickets, Payments related lists).
- LWC Components (basic):
- Flight Search
- Seat Selection (demo only, no full seat map).
- Flight Status Board (show random status).

___

Phase 7: Integration

Developer Edition has API access, but limited daily calls:

- Mock external integration (instead of real payment gateway).
- Create a REST Apex Class for “Book Flight” API (for demo).
- Use Platform Events → Flight delay alerts.

___

Phase 8: Data Management

- Import sample data with Data Import Wizard (since storage is limited).
- 10 demo Flights, 20 demo Passengers.
- Use Duplicate Rules for Passengers.
- No sandbox → Use Unmanaged Package for backup.

___

Phase 9: Reporting & Security

- Reports:
- Bookings by Flight
- Revenue by Flight
- Passenger Trends
- Dashboard:
- Flight Occupancy (use chart)
- Monthly Revenue
- Security:
- Role Hierarchy (Admin > Agent).

- Login IP not needed (1 org user).
- Audit Trail → enabled.

___

Phase 10: Demo & Presentation

- End-to-End Demo (inside free org):
- Passenger record → Book flight → Confirm → Ticket auto-created → Payment recorded.
- Show Flow + Trigger execution (capacity check, seat update).
- Dashboards → Flight Occupancy & Revenue.
- Deliverables → ERD, Flow screenshots, Apex test coverage, dashboard screenshots.