Family Leads Schema Design Documentation

Project: CareLinkAI - Family ↔ Marketplace Lead/Inquiry Flow
Phase: Phase 1 - Database Schema & Models
Date: December 7, 2025
Branch: `feature/family-leads-mvp`
Migration: `20251207154010_add_family_and_lead_models`

Executive Summary

This document details the database schema changes for implementing a Family-to-Marketplace inquiry system that enables families to submit leads for both Aides (Caregivers) and Providers. The implementation uses a polymorphic pattern to support both target types within a single `Lead` model.

Schema Changes Overview

1. Enhanced Family Model

The existing `Family` model has been extended with additional fields to capture care context and primary contact information at the time of lead creation.

New Fields Added

Field Name	Type	Nullable	Description
`primaryContactName`	String	Yes	Name of the primary family contact
`phone`	String	Yes	Contact phone number
`relationshipToRecipient`	String	Yes	Relationship to care recipient (e.g., “Spouse”, “Child”, “Sibling”, “Friend”)
`recipientAge`	Int	Yes	Age of the care recipient
`primaryDiagnosis`	Text	Yes	Primary medical diagnosis or care need
`mobilityLevel`	String	Yes	Mobility status: “Independent”, “Needs Assistance”, “Wheelchair”
`careNotes`	Text	Yes	Additional care context and notes

Rationale

- **Snapshot Data**: These fields provide a quick reference snapshot of the family’s care situation without requiring joins to other tables
- **Lead Context**: When a family creates a lead, we can pre-populate inquiry forms with this context
- **Operator Efficiency**: Operators can quickly assess family needs when reviewing leads
- **Progressive Enhancement**: Fields are optional to accommodate families at different stages of the care journey

New Relationship

- `leads` → `Lead[]` : One-to-many relationship with Lead model

---

2. New Lead Model

A new `Lead` model has been created to track Family inquiries directed at Aides or Providers in the marketplace.

Model Structure

```
model Lead {
  id         String         @id @default(cuid())
  familyId   String
  targetType LeadTargetType // AIDE or PROVIDER

  // Polymorphic target references
  aideId     String?
  providerId String?

  status     LeadStatus     @default(NEW)
  message    String?        @db.Text

  // Care details snapshot
  preferredStartDate    DateTime?
  expectedHoursPerWeek  Int?
  location              String?

  // Operator management
  operatorNotes         String?  @db.Text
  assignedOperatorId    String?

  // Soft delete support
  deletedAt             DateTime?

  // Timestamps
  createdAt DateTime @default(now())
  updatedAt DateTime @updatedAt
}
```

Field Descriptions

Core Fields:
- `id` : Primary key (CUID)
- `familyId` : Foreign key to Family who created the lead
- `targetType` : Enum distinguishing between AIDE and PROVIDER leads
- `status` : Current lead status (see LeadStatus enum)

Polymorphic Pattern:
- `aideId` : Set when targetType=AIDE, references Caregiver.id
- `providerId` : Set when targetType=PROVIDER, references Provider.id
- **Constraint**: Only one of aideId or providerId should be populated based on targetType

Family Input Fields:
- `message` : Initial inquiry message from the family
- `preferredStartDate` : When the family needs care to begin
- `expectedHoursPerWeek` : Estimated care hours needed
- `location` : Care location (city, state, or address)

Operator Management:
- `operatorNotes` : Internal notes for operator use only (not visible to families)
- `assignedOperatorId` : Optional assignment to specific operator for follow-up

Data Management:
- `deletedAt` : Soft delete timestamp (allows lead archiving without data loss)

- `createdAt`: Lead creation timestamp
- `updatedAt`: Last modification timestamp

---

3. New Enums

LeadTargetType

```
enum LeadTargetType {
  AIDE
  PROVIDER
}
```

Purpose: Distinguishes the type of marketplace entity the lead targets.

Values:
- `AIDE`: Lead is for a Caregiver/Aide in the marketplace
- `PROVIDER`: Lead is for a Provider (agency, service organization)

LeadStatus

```
enum LeadStatus {
  NEW
  IN_REVIEW
  CONTACTED
  CLOSED
  CANCELLED
}
```

Purpose: Tracks the lifecycle of a lead through the operator workflow.

Status Meanings:

Status	Description	Typical Actions
`NEW`	Just created, not yet reviewed	Operator needs to triage
`IN_REVIEW`	Operator is evaluating the lead	Research, internal discussion
`CONTACTED`	Operator has reached out to family/provider	Awaiting response, scheduling
`CLOSED`	Lead successfully converted or resolved	Archive, report metrics
`CANCELLED`	Family cancelled or lead no longer valid	Document reason, archive

Status Transitions:
- NEW → IN_REVIEW → CONTACTED → CLOSED
- NEW → CANCELLED (family cancels early)

- IN_REVIEW → CANCELLED (lead deemed invalid)
- CONTACTED → CANCELLED (family withdraws)

Design Decisions & Rationale

1. Why Polymorphic Pattern?

Decision: Use a single `Lead` model with polymorphic foreign keys rather than separate `AideLead` and `ProviderLead` models.

Rationale:
- **Unified Workflow**: Operators manage all leads in one interface
- **Consistent API**: Single set of endpoints for CRUD operations
- **Reduced Duplication**: Shared fields (status, message, operatorNotes) aren't duplicated
- **Simplified Queries**: Get all leads for a family with one query
- **Future Extensibility**: Easy to add new target types (e.g., FACILITY, EQUIPMENT_RENTAL)

Trade-offs:
- Requires application-level validation to ensure targetType matches populated foreign key
- Slightly more complex queries when filtering by aide/provider
- **Mitigation**: Use database indexes and application-layer checks

2. Why Soft Delete?

Decision: Include `deletedAt` field for soft deletion rather than hard deletes.

Rationale:
- **Audit Trail**: Maintain history of all lead activity for compliance
- **Analytics**: Include deleted leads in historical reporting
- **Recovery**: Operators can "undelete" if cancelled by mistake
- **Relationships**: Preserve referential integrity (foreign keys remain valid)

Implementation:
- Application queries should filter `WHERE deletedAt IS NULL` by default
- Add index on `deletedAt` for performance
- Admin tools can show deleted leads for audit purposes

3. Why Snapshot Care Context?

Decision: Store care details (preferredStartDate, expectedHoursPerWeek, location) directly on Lead rather than always querying Family model.

Rationale:
- **Point-in-Time Accuracy**: Captures family's needs at time of inquiry (may change later)
- **Performance**: Operators can list/filter leads without joining to Family and Resident tables
- **Flexibility**: Family can create multiple leads with different parameters
- **Historical Record**: If family's situation changes, old leads still reflect original context

4. Optional Operator Assignment

Decision: Make `assignedOperatorId` optional rather than required.

Rationale:
- **Flexible Workflow**: Supports both assigned and pool-based lead distribution

- **Initial Triage**: New leads can exist unassigned until reviewed
- **Team Capacity**: Allows reassignment if operator unavailable
- **Round-Robin**: Enables automated assignment logic in future

5. Status Enum Design

Decision: Keep status enum focused on operator workflow rather than business outcomes.

Rationale:
- **Operator-Centric**: Reflects internal process stages
- **Clear Semantics**: Each status has unambiguous meaning
- **Linear Progression**: Status transitions follow expected workflow
- **Not Over-Engineered**: Avoids excessive substates (e.g., "CONTACTED_AWAITING_CALLBACK")

Future Consideration: May add `outcomeType` field (HIRED, NOT_INTERESTED, NO_RESPONSE) for closed leads to improve analytics.

Database Relationships

Relationship Diagram

```
Family (1) ──────< (many) Lead
                          │
                          ├──> (0..1) Caregiver (when targetType=AIDE)
                          ├──> (0..1) Provider (when targetType=PROVIDER)
                          └──> (0..1) User [Operator] (assignedOperatorId)
```

Foreign Key Constraints

From Model	To Model	FK Column	Constraint	On Delete
Lead	Family	`familyId`	Required	CASCADE
Lead	Caregiver	`aideId`	Optional	CASCADE
Lead	Provider	`providerId`	Optional	CASCADE
Lead	User	`assignedOperat-orId`	Optional	SET NULL

On Delete Behaviors:
- **CASCADE for Family**: If family deleted, all leads deleted (GDPR compliance)
- **CASCADE for Aide/Provider**: If aide/provider removed, leads also removed
- **SET NULL for Operator**: If operator user deleted, lead remains but assignment cleared

---

Indexes

The following indexes have been added to optimize query performance:

Index Name	Columns	Purpose
`Lead_familyId_idx`	`familyId`	Get all leads for a family
`Lead_targetType_idx`	`targetType`	Filter by AIDE vs PROVIDER
`Lead_aideId_idx`	`aideId`	Get all leads for a specific aide
`Lead_providerId_idx`	`providerId`	Get all leads for a specific provider
`Lead_status_idx`	`status`	Filter leads by status (operator dashboard)
`Lead_assignedOperatorId_idx`	`assignedOperatorId`	Get leads assigned to operator
`Lead_createdAt_idx`	`createdAt`	Sort leads by creation date
`Lead_deletedAt_idx`	`deletedAt`	Efficiently filter out deleted leads

Query Optimization Examples

```sql
-- Operator dashboard: Get NEW leads
SELECT * FROM "Lead" WHERE status = 'NEW' AND "deletedAt" IS NULL
ORDER BY "createdAt" DESC;

-- Family profile: Get all leads for family
SELECT * FROM "Lead" WHERE "familyId" = ? AND "deletedAt" IS NULL;

-- Aide profile: Get all inquiries for aide
SELECT * FROM "Lead"
WHERE "targetType" = 'AIDE' AND "aideId" = ? AND "deletedAt" IS NULL;
```

Migration Details

Migration File

Path: `prisma/migrations/20251207154010_add_family_and_lead_models/migration.sql`

Migration Steps

1. **Create Enums**
   - `LeadTargetType` (AIDE, PROVIDER)
   - `LeadStatus` (NEW, IN_REVIEW, CONTACTED, CLOSED, CANCELLED)

2. **Alter Family Table**
   - Add 7 new optional columns for care context

3. **Create Lead Table**
   - All columns as specified in model
   - Primary key on `id`

4. **Create Indexes**
   - 8 indexes for query optimization

5. **Add Foreign Keys**
   - 4 foreign key constraints with appropriate ON DELETE behaviors

Running the Migration

On Production (Render):

```
# Via Render Dashboard Shell:
npx prisma migrate deploy

# Or via local connection:
DATABASE_URL="<render-prod-url>" npx prisma migrate deploy
```

On Development:

```
npx prisma migrate dev
```

Rollback Plan

If migration needs to be rolled back:

```
-- Drop Lead table and enums
DROP TABLE "Lead";
DROP TYPE "LeadStatus";
DROP TYPE "LeadTargetType";

-- Remove Family columns
ALTER TABLE "Family"
  DROP COLUMN "primaryContactName",
  DROP COLUMN "phone",
  DROP COLUMN "relationshipToRecipient",
  DROP COLUMN "recipientAge",
  DROP COLUMN "primaryDiagnosis",
  DROP COLUMN "mobilityLevel",
  DROP COLUMN "careNotes";
```

Validation Rules (Application Layer)

The following validation rules should be enforced in the application:

Lead Creation

1. **Polymorphic Integrity**
   - If `targetType = AIDE`, then `aideId` must be set and `providerId` must be null
   - If `targetType = PROVIDER`, then `providerId` must be set and `aideId` must be null

2. **Foreign Key Validity**
   - `aideId` must reference an existing, active Caregiver
   - `providerId` must reference an existing, active Provider
   - `familyId` must reference an existing Family
   - `assignedOperatorId`, if set, must reference a User with role=OPERATOR

3. **Field Validation**
   - `message`: Max 5000 characters
   - `location`: Max 255 characters
   - `expectedHoursPerWeek`: If set, must be between 1 and 168 (hours in a week)
   - `preferredStartDate`: If set, must be today or future date

4. **Status Transitions**
   - NEW → IN_REVIEW: Allowed
   - IN_REVIEW → CONTACTED: Allowed
   - CONTACTED → CLOSED: Allowed
   - * → CANCELLED: Allowed from any status
   - CLOSED → : Not allowed (terminal state)
   - CANCELLED → : Not allowed (terminal state)

Soft Delete

- Only leads with status = CLOSED or CANCELLED can be soft deleted
- `deletedAt` timestamp must be set to current time
- Deleting a lead does not cascade to related entities

API Implications

Expected Endpoints (Phase 2)

Based on this schema, Phase 2 will implement:

Family-facing:
- `POST /api/leads` - Create new lead
- `GET /api/leads` - Get family's leads
- `GET /api/leads/[id]` - Get lead details
- `PATCH /api/leads/[id]` - Update lead (limited fields)
- `DELETE /api/leads/[id]` - Cancel lead (soft delete)

Operator-facing:
- `GET /api/operator/leads` - List all leads (with filters)
- `GET /api/operator/leads/[id]` - Get lead details
- `PATCH /api/operator/leads/[id]` - Update status, assign, add notes
- `PATCH /api/operator/leads/[id]/assign` - Assign to operator

Admin-facing:
- `GET /api/admin/leads/deleted` - View soft-deleted leads
- `POST /api/admin/leads/[id]/restore` - Restore deleted lead

Sample JSON Schema

Lead Creation Request:

```
{
  "targetType": "AIDE",
  "aideId": "clx123abc",
  "message": "Looking for caregiver with dementia experience",
  "preferredStartDate": "2025-01-15T00:00:00Z",
  "expectedHoursPerWeek": 20,
  "location": "Seattle, WA"
}
```

Lead Response:

```
{
  "id": "lead_xyz789",
  "familyId": "fam_abc123",
  "targetType": "AIDE",
  "aideId": "clx123abc",
  "status": "NEW",
  "message": "Looking for caregiver with dementia experience",
  "preferredStartDate": "2025-01-15T00:00:00.000Z",
  "expectedHoursPerWeek": 20,
  "location": "Seattle, WA",
  "operatorNotes": null,
  "assignedOperatorId": null,
  "createdAt": "2025-12-07T15:40:00.000Z",
  "updatedAt": "2025-12-07T15:40:00.000Z"
}
```

Testing Considerations

Unit Tests

1. **Model Validation**
   - Test polymorphic integrity (aideId/providerId exclusivity)
   - Test field length constraints
   - Test date validations

2. **Status Transitions**
   - Test valid transitions (NEW → IN_REVIEW → CONTACTED → CLOSED)
   - Test invalid transitions (CLOSED → CONTACTED)
   - Test cancellation from any status

3. **Soft Delete**
   - Test deletedAt filtering
   - Test restoration logic

Integration Tests

1. **Lead Creation**
   - Family creates aide lead
   - Family creates provider lead
   - Invalid targetType scenarios

2. **Lead Management**
   - Operator lists leads (with pagination)

- Operator updates lead status
- Operator assigns lead
- Operator adds internal notes

3. **Query Performance**
   - Test index usage with EXPLAIN ANALYZE
   - Test pagination on large datasets
   - Test filtering combinations

Security Considerations

Access Control

- **Families**: Can only view/edit their own leads
- **Operators**: Can view/edit all leads, update status
- **Admins**: Full access including soft-deleted leads
- **Aides/Providers**: Cannot view leads directly (privacy)

Data Privacy

- `operatorNotes` must never be exposed to families
- Family's message should be visible to assigned operator only
- Soft-deleted leads require admin access

HIPAA Compliance

- `primaryDiagnosis`, `careNotes` contain PHI (Protected Health Information)
- Must be encrypted at rest
- Audit logs should track all lead access
- Retention policy: Keep deleted leads for 7 years per HIPAA requirements

---

Future Enhancements

Phase 2 Candidates

1. **Lead Source Tracking**: Add `source` field (WEB, MOBILE, REFERRAL, ADMIN)
2. **Communication History**: Add `LeadMessage` model for 2-way messaging
3. **Outcome Tracking**: Add `outcomeType` and `outcomeNotes` for closed leads
4. **SLA Tracking**: Add `firstResponseAt`, `resolvedAt` for operator performance metrics
5. **AI Matching**: Add `matchScore` field for AI-suggested aide/provider matches

Data Analytics

Future reporting queries:
- Lead conversion rate by targetType
- Average time to contact
- Operator assignment balance
- Lead source effectiveness
- Geographic distribution of leads

Appendix: Related Models

Existing Models Referenced

- **Family**: Enhanced with care context fields
- **Caregiver**: References via `aideId` when targetType=AIDE
- **Provider**: References via `providerId` when targetType=PROVIDER
- **User**: References via `assignedOperatorId` for operator assignment

Similar Models (for context)

- **Inquiry**: Existing model for Family → Home inquiries
- Uses `homeId` foreign key
- Similar status workflow
- Has `internalNotes` and `tourDate` fields
- Could be harmonized with Lead model in future refactor

Changelog

Date	Change	Author
2025-12-07	Initial schema design and migration	System

Approval & Sign-off

- [x] Schema reviewed for data integrity
- [x] Indexes designed for query performance
- [x] Foreign key constraints validated
- [x] Soft delete pattern implemented
- [x] Migration SQL generated
- [x] Documentation completed

Next Phase: Phase 2 - Backend APIs (Lead CRUD endpoints)