Yestoryd rAI System - Final Design Document

Intelligent Chat for Child Reading Development

Version: 1.0
Date: December 21, 2025
Authors: Amit Kumar Rai & Claude
Status: Ready for Implementation

Table of Contents

---

1. Mission & Philosophy

1.1 Core Principle

> **"Every response should contribute to a child's reading development."**

The rAI system exists to serve one primary purpose: **improving children's reading abilities**. All features, queries, and responses must ultimately support this mission.

1.2 Value Proposition by Role

Role	rAI Value
Parents	Understand your child's learning journey. Participate actively in their development. Know what's working and what needs attention.
Coaches	Deliver better sessions through preparation. Track what teaching methods work for each child. Get AI-powered lesson planning assistance.
Admin	Spot platform-wide learning patterns. Identify at-risk children early. Make data-driven decisions that impact Yestoryd's educational mission.

1.3 What rAI is NOT

- ❌ Not a financial reporting tool
- ❌ Not a replacement for dashboards
- ❌ Not a general-purpose chatbot
- ❌ Not a support ticket system

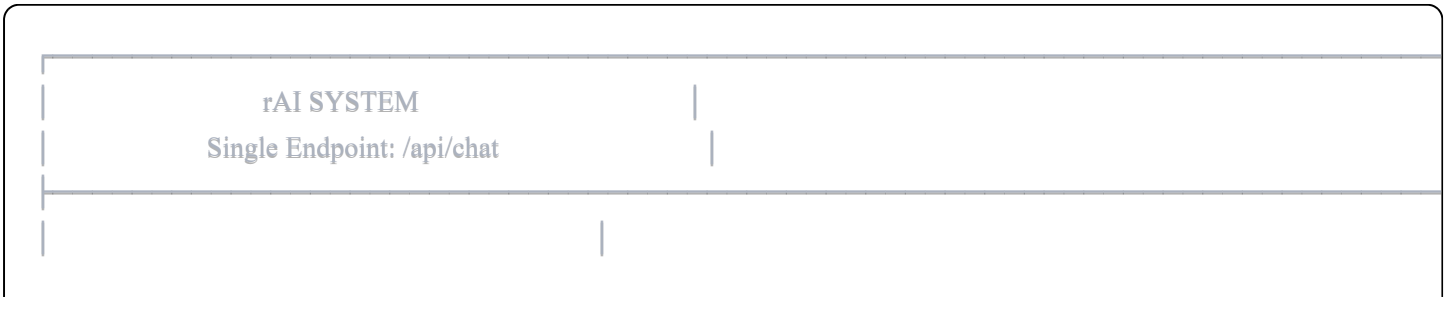
1.4 Design Principles

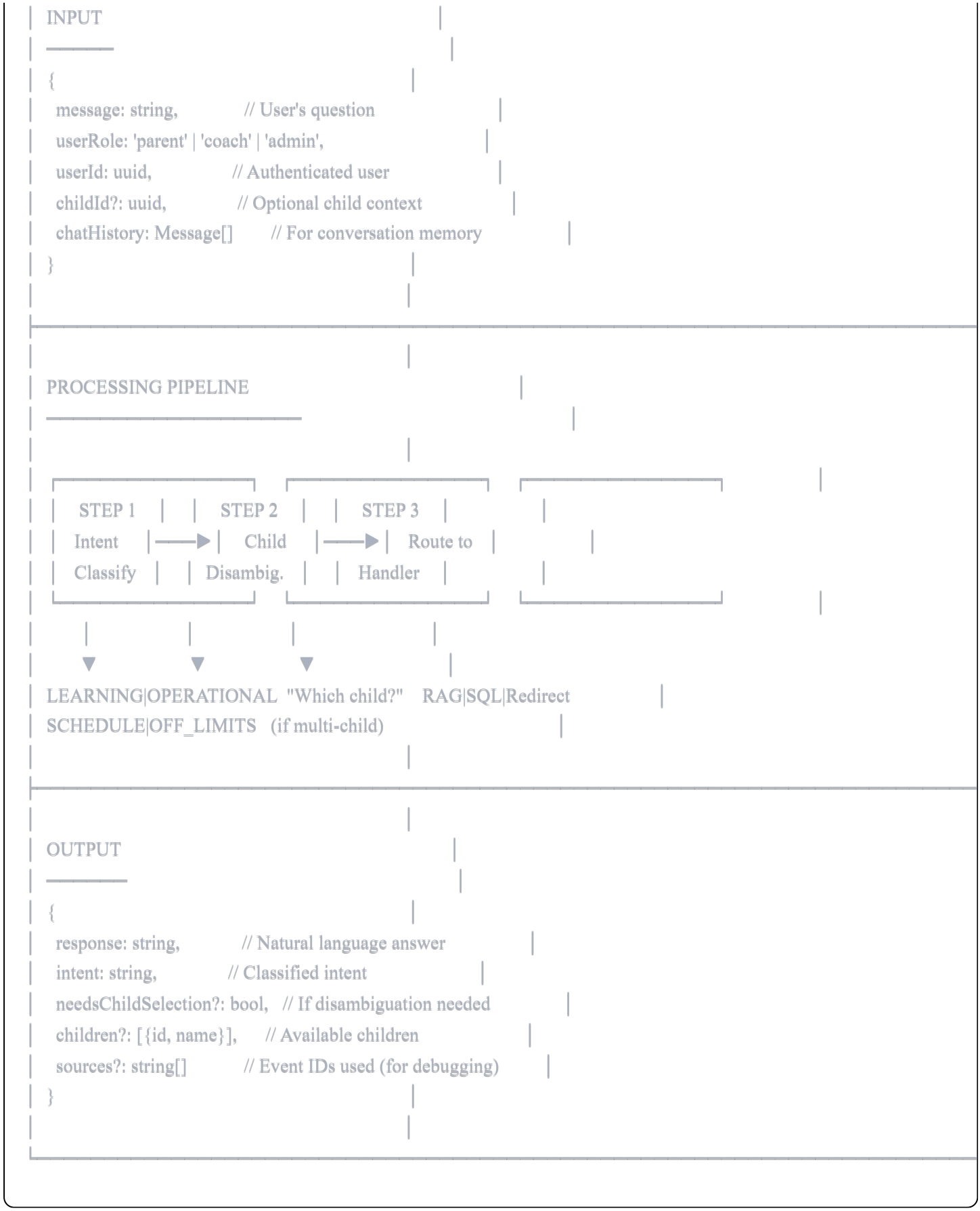
1. **Learning-Focused**: Every feature must serve child development
2. **Fast Operational Access**: Simple questions get instant SQL answers
3. **Role-Appropriate**: Users only see data relevant to their role
4. **Anti-Hallucination**: Never invent data - only use verified information
5. **Conversation-Aware**: Maintain context within a chat session
6. **Cost-Conscious**: Optimize for scale (10,000 children target)

---

2. System Architecture

2.1 High-Level Architecture





2.2 Component Overview

Component	Technology	Purpose
Chat Endpoint	Next.js API Route	Single entry point for all chat
Intent Classifier	Gemini Flash Lite	Lightweight query classification

Component	Technology	Purpose
RAG Handler	pgvector + Gemini	Learning-focused semantic search
Operational Handler	Direct SQL	Fast structured data queries
Schedule Handler	Direct SQL	Calendar and session queries
Response Generator	Gemini Flash Lite	Natural language formatting

2.3 Data Flow

---

3. Intent Classification

3.1 Intent Categories

Intent	Description	Handler
LEARNING	Questions about child progress, development, teaching strategies	RAG (Vector Search + AI)
OPERATIONAL	Counts, coach info, program details, payment status	Direct SQL Query
SCHEDULE	Session times, calendar, upcoming/past sessions	Direct SQL Query
OFF_LIMITS	Earnings, other users' data, sensitive information	Polite Redirect

3.2 Classification Prompt

```
You are an intent classifier for a children's reading education platform.

Classify the following query into exactly ONE category:
- LEARNING: Child progress, recommendations, session summaries, teaching tips
- OPERATIONAL: Counts, coach info, program info, payment status
- SCHEDULE: Session times, calendar, when is next session
- OFF_LIMITS: Earnings, other users' data, platform metrics

User role: {role}
Query: {message}

Respond with JSON: {"intent": "LEARNING|OPERATIONAL|SCHEDULE|OFF_LIMITS", "entities": ["child names or topics mentioned"]}
```

3.3 Classification Examples

Query	Role	Intent	Entities
"How is Sima doing?"	Parent	LEARNING	["Sima"]

Query	Role	Intent	Entities
"How many children do I have?"	Coach	OPERATIONAL	[]
"When is my next session?"	Parent	SCHEDULE	[]
"What are my earnings?"	Coach	OFF_LIMITS	[]
"Prepare me for tomorrow's session with Aarav"	Coach	LEARNING	["Aarav"]
"Who is my coach?"	Parent	OPERATIONAL	[]
"What's my schedule today?"	Coach	SCHEDULE	[]
"How much does the program cost?"	Parent	OPERATIONAL	[]

4. Role-Based Access & Boundaries

4.1 Parent Access

Allowed Queries

Category	Query Type	Data Source	Example
LEARNING	Child progress	RAG	"How is my child doing?"
LEARNING	Recommendations	RAG	"What books should my child read?"
LEARNING	Session summaries	RAG	"What happened in the last session?"
LEARNING	Home practice tips	RAG + General AI	"How can I help at home?"
OPERATIONAL	Coach information	SQL	"Who is my coach?"
OPERATIONAL	Program details	SQL (site_settings)	"How many sessions are included?"
OPERATIONAL	Payment status	SQL	"Is my enrollment active?"
SCHEDULE	Next session	SQL	"When is the next session?"
SCHEDULE	Session history	SQL	"How many sessions completed?"

Off-Limits

Topic	Response
Other children's data	"I can only share information about your enrolled children."
Coach earnings	"I don't have access to that information."
Platform statistics	"I focus on your child's learning journey."

Multi-Child Handling

If parent has multiple children enrolled and asks about "my child":

> "I see you have Sima and Aarav enrolled. Which child are you asking about?"

4.2 Coach Access

Allowed Queries

Category	Query Type	Data Source	Example
LEARNING	Student progress	RAG	"How is Aarav doing?"
LEARNING	Session preparation	RAG	"Prepare me for tomorrow's session with Sima"
LEARNING	Teaching strategies	RAG + General AI	"What phonics games work for struggling readers?"
LEARNING	What works analysis	RAG	"What teaching methods work best with Sima?"
OPERATIONAL	Student count	SQL	"How many children do I have?"
OPERATIONAL	Session count	SQL	"How many sessions this month?"
OPERATIONAL	Program structure	SQL	"How many sessions per enrollment?"
SCHEDULE	Today's schedule	SQL	"What's my schedule today?"
SCHEDULE	Week's schedule	SQL	"How many sessions this week?"

Off-Limits

Topic	Response
Earnings/payouts	"For earnings information, please check your Earnings dashboard."
Other coaches' data	"I can only share information about your assigned students."
Unassigned children	"I don't have access to that student's information."

4.3 Admin Access

Allowed Queries

Category	Query Type	Data Source	Example
LEARNING INSIGHTS	At-risk children	Cached Insights	"Which children are at risk?"
LEARNING INSIGHTS	Common struggles	Cached Insights	"What are children struggling with?"
LEARNING INSIGHTS	Coach effectiveness	Cached Insights	"Which coaches have best outcomes?"
LEARNING INSIGHTS	Learning trends	Cached Insights	"How are scores trending?"
OPERATIONAL	Enrollment metrics	SQL	"How many enrollments this month?"
OPERATIONAL	Session metrics	SQL	"What's the session completion rate?"
OPERATIONAL	Coach workload	SQL	"How many sessions per coach?"
OPERATIONAL	Assessment metrics	SQL	"How many assessments this week?"
SCHEDULE	Platform schedule	SQL	"How many sessions today?"

Off-Limits

Topic	Response
Financial/Revenue data	"For revenue and payout information, please check the Revenue dashboard."
Personal contact details	Only when needed for follow-up on learning issues

5. Data Sources & Handlers

5.1 LEARNING Handler (RAG)

Purpose: Answer questions about child development, progress, and learning

Data Source: `learning_events` table with vector embeddings

Flow:

```
1. Generate embedding for user query (text-embedding-004)
2. Vector similarity search in learning_events
   - Filter by role (parent: own child, coach: assigned children)
   - Match threshold: 0.5
   - Return top 10-15 results
3. Build context from retrieved events
4. Include chat history for conversation memory
5. Generate response with Gemini (learning-focused prompt)
```

Key Tables:

- `learning_events` (primary - with embeddings)
- `children` (child profiles)
- `scheduled_sessions` (session context)

Anti-Hallucination Prompt:

```
You are rAI, a reading development assistant for Yestoryd.

CRITICAL RULES:
1. ONLY use data from the context provided below
2. NEVER invent scores, dates, or session details
3. If data is not available, say "I don't have that information yet"
4. Focus on learning progress and development
5. Provide actionable recommendations when relevant

Context:
{retrieved_events}

Conversation History:
{chat_history}

User Query: {message}
```

5.2 OPERATIONAL Handler (Direct SQL)

Purpose: Fast answers to structured data questions

Data Source: Predefined SQL templates

Flow:

1. Match query to SQL template using keyword detection
2. Execute query with role-based filtering
3. Format results in natural language

SQL Templates:

```sql
```

```
-- PARENT QUERIES --

parent_coach_info:
  SELECT c.name, c.email
  FROM coaches c
  JOIN children ch ON ch.assigned_coach_id = c.id
  WHERE ch.parent_id = $1

parent_payment_status:
  SELECT amount, status, created_at
  FROM payments
  WHERE parent_id = $1
  ORDER BY created_at DESC LIMIT 1

parent_program_info:
  SELECT setting_value
  FROM site_settings
  WHERE setting_key IN ('program_price', 'program_duration', 'session_count')

parent_session_count:
  SELECT
    COUNT(*) FILTER (WHERE status = 'completed') as completed,
    COUNT(*) as total
  FROM scheduled_sessions ss
  JOIN children c ON ss.child_id = c.id
  WHERE c.parent_id = $1


-- COACH QUERIES --

coach_child_count:
  SELECT COUNT(*) as count
  FROM children
  WHERE assigned_coach_id = $1 AND status = 'enrolled'

coach_session_count_month:
  SELECT COUNT(*) as count
  FROM scheduled_sessions
  WHERE coach_id = $1
    AND status = 'completed'
    AND scheduled_date >= date_trunc('month', NOW())

coach_session_count_total:
  SELECT
    COUNT(*) FILTER (WHERE status = 'completed') as completed,
    COUNT(*) FILTER (WHERE status = 'scheduled') as upcoming
```

```
  FROM scheduled_sessions
  WHERE coach_id = $1


-- ADMIN QUERIES --

admin_enrollment_count:
  SELECT
    COUNT(*) FILTER (WHERE created_at >= date_trunc('month', NOW())) as this_month,
    COUNT(*) FILTER (WHERE created_at >= date_trunc('week', NOW())) as this_week,
    COUNT(*) as total
  FROM enrollments WHERE status = 'active'

admin_session_metrics:
  SELECT
    COUNT(*) as total_sessions,
    COUNT(*) FILTER (WHERE status = 'completed') as completed,
    ROUND(100.0 * COUNT(*) FILTER (WHERE status = 'completed') / COUNT(*), 1) as completion_rate
  FROM scheduled_sessions
  WHERE scheduled_date >= date_trunc('month', NOW())

admin_coach_workload:
  SELECT
    c.name as coach_name,
    COUNT(ss.id) as session_count
  FROM coaches c
  LEFT JOIN scheduled_sessions ss ON c.id = ss.coach_id
    AND ss.scheduled_date >= date_trunc('month', NOW())
  WHERE c.status = 'active'
  GROUP BY c.id, c.name
  ORDER BY session_count DESC
```

5.3 SCHEDULE Handler (Direct SQL)

Purpose: Session timing and calendar queries

Data Source: `scheduled_sessions` table

SQL Templates:

```
sql
```

```
-- PARENT SCHEDULE --

parent_next_session:
  SELECT
    ss.scheduled_time,
    ss.session_type,
    ss.google_meet_link,
    c.child_name
  FROM scheduled_sessions ss
  JOIN children c ON ss.child_id = c.id
  WHERE c.parent_id = $1
    AND ss.scheduled_time > NOW()
    AND ss.status = 'scheduled'
  ORDER BY ss.scheduled_time
  LIMIT 1

parent_upcoming_sessions:
  SELECT
    ss.scheduled_time,
    ss.session_type,
    ss.google_meet_link,
    c.child_name
  FROM scheduled_sessions ss
  JOIN children c ON ss.child_id = c.id
  WHERE c.parent_id = $1
    AND ss.scheduled_time > NOW()
    AND ss.status = 'scheduled'
  ORDER BY ss.scheduled_time
  LIMIT 5


-- COACH SCHEDULE --

coach_today_schedule:
  SELECT
    c.child_name,
    ss.scheduled_time,
    ss.session_type,
    ss.google_meet_link
  FROM scheduled_sessions ss
  JOIN children c ON ss.child_id = c.id
  WHERE ss.coach_id = $1
    AND DATE(ss.scheduled_time) = CURRENT_DATE
    AND ss.status = 'scheduled'
  ORDER BY ss.scheduled_time
```

```
coach_week_schedule:
  SELECT
    DATE(ss.scheduled_time) as session_date,
    COUNT(*) as session_count
  FROM scheduled_sessions ss
  WHERE ss.coach_id = $1
    AND ss.scheduled_time >= date_trunc('week', NOW())
    AND ss.scheduled_time < date_trunc('week', NOW()) + INTERVAL '7 days'
    AND ss.status = 'scheduled'
  GROUP BY session_date
  ORDER BY session_date


-- ADMIN SCHEDULE --

admin_today_sessions:
  SELECT
    COUNT(*) as total_sessions,
    COUNT(DISTINCT coach_id) as active_coaches
  FROM scheduled_sessions
  WHERE DATE(scheduled_time) = CURRENT_DATE
    AND status = 'scheduled'
```

5.4 OFF_LIMITS Handler

Purpose: Polite redirects for out-of-scope queries

Response Templates:

```
typescript
```

```
const OFF_LIMITS_RESPONSES = {
  earnings: {
    coach: "For earnings and payout information, please check your Earnings dashboard. Is there anything else about your stud
    admin: "For revenue and financial data, please check the Revenue dashboard. Would you like to know about learning metri
  },

  other_users: {
    parent: "I can only share information about your enrolled children. Is there something specific about {childName} you'd lik
    coach: "I can only provide information about students assigned to you. Would you like to know about one of your students?
  },

  support_needed: "Would you like me to connect you with support? You can reach us on WhatsApp: 918976287997",

  unknown: "I'm not sure I can help with that. I'm best at answering questions about children's reading development. Is there s
};
```

---

6. Conversation Memory

6.1 Purpose

Enable natural multi-turn conversations where users can refer to previous messages:

> User: "How is Sima doing?"
> rAI: "Sima has improved her fluency to 58 WPM..."
> User: "What about her reading speed?"
> rAI: (Knows "her" = Sima) "Sima's reading speed..."

6.2 Implementation

typescript

```typescript
interface ChatMessage {
  role: 'user' | 'assistant';
  content: string;
  timestamp: Date;
  childId?: string;      // Track which child was discussed
  intent?: string;       // Track query type
}

interface ChatRequest {
  message: string;
  userRole: 'parent' | 'coach' | 'admin';
  userId: string;
  childId?: string;
  chatHistory: ChatMessage[];  // Last 10 messages
}
```

6.3 Context Extraction

typescript

```typescript
function buildConversationContext(history: ChatMessage[]): string {
  // Take last 5 exchanges (10 messages)
  const recent = history.slice(-10);

  return recent.map(msg =>
    `${msg.role === 'user' ? 'User' : 'rAI'}: ${msg.content}`
  ).join('\n');
}

function extractChildFromHistory(history: ChatMessage[]): string | null {
  // Find most recently mentioned child
  for (let i = history.length - 1; i >= 0; i--) {
    if (history[i].childId) {
      return history[i].childId;
    }
  }
  return null;
}
```

6.4 Memory Limits

Limit	Value	Rationale
History depth	10 messages	Balance context vs. token cost

Limit	Value	Rationale
Session duration	30 minutes	After inactivity, start fresh
Child context	Persists in session	Don't ask "which child" repeatedly

7. Admin Weekly Insights

7.1 Overview

Pre-computed insights reduce cost and improve response time for complex Admin queries.

Schedule: Every Sunday at 2:00 AM IST (via QStash)

Storage: `admin_insights` table

7.2 Insight Types

Insight	Computation	Use Case
`at_risk_children`	Children without session in 14+ days	Early intervention
`common_struggles`	Aggregate focus areas from sessions	Curriculum planning
`coach_effectiveness`	Progress ratings by coach	Quality monitoring
`learning_trends`	Week-over-week score changes	Platform health

7.3 Database Schema

```sql
```

```
CREATE TABLE admin_insights (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  insight_type TEXT NOT NULL UNIQUE,
  insight_data JSONB NOT NULL,
  computed_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
  valid_until TIMESTAMP WITH TIME ZONE,

  CONSTRAINT valid_insight_type CHECK (
    insight_type IN ('at_risk_children', 'common_struggles',
                     'coach_effectiveness', 'learning_trends')
  )
);

-- Index for fast lookup
CREATE INDEX idx_admin_insights_type ON admin_insights(insight_type);
```

7.4 Insight Computation

typescript

```
// Weekly cron job - runs Sunday 2 AM IST

const WEEKLY_INSIGHTS = [
  {
    type: 'at_risk_children',
    query: `
      SELECT
        c.id, c.child_name, c.parent_email,
        p.name as parent_name,
        MAX(ss.scheduled_date) as last_session,
        EXTRACT(DAY FROM NOW() - MAX(ss.scheduled_date)) as days_since
      FROM children c
      JOIN parents p ON c.parent_id = p.id
      LEFT JOIN scheduled_sessions ss
        ON c.id = ss.child_id AND ss.status = 'completed'
      WHERE c.status = 'enrolled'
      GROUP BY c.id, c.child_name, c.parent_email, p.name
      HAVING MAX(ss.scheduled_date) < NOW() - INTERVAL '14 days'
        OR MAX(ss.scheduled_date) IS NULL
      ORDER BY days_since DESC NULLS FIRST
    `,
    aiSummary: true  // Gemini generates summary
  },

  {
    type: 'common_struggles',
    query: `
      SELECT
        event_data->>'focus_area' as focus_area,
        COUNT(*) as session_count,
        AVG((event_data->>'engagement_level')::int) as avg_engagement
      FROM learning_events
      WHERE event_type = 'session'
        AND created_at > NOW() - INTERVAL '7 days'
        AND event_data->>'focus_area' IS NOT NULL
      GROUP BY event_data->>'focus_area'
      ORDER BY session_count DESC
      LIMIT 10
    `,
    aiSummary: true
  },

  {
    type: 'coach_effectiveness',
    query: `
      SELECT
```

```
      co.id as coach_id,
      co.name as coach_name,
      COUNT(DISTINCT le.child_id) as student_count,
      COUNT(le.id) as session_count,
      AVG(CASE
        WHEN le.event_data->>'progress_rating' = 'significant_improvement' THEN 4
        WHEN le.event_data->>'progress_rating' = 'improved' THEN 3
        WHEN le.event_data->>'progress_rating' = 'same' THEN 2
        WHEN le.event_data->>'progress_rating' = 'declined' THEN 1
        ELSE NULL
      END) as avg_progress_score
    FROM coaches co
    LEFT JOIN learning_events le
      ON co.id = le.coach_id AND le.event_type = 'session'
    WHERE co.status = 'active'
      AND (le.created_at > NOW() - INTERVAL '30 days' OR le.id IS NULL)
    GROUP BY co.id, co.name
    ORDER BY avg_progress_score DESC NULLS LAST
  `,
  aiSummary: true
 },

 {
  type: 'learning_trends',
  query: `
    SELECT
      date_trunc('week', event_date) as week,
      COUNT(*) as assessment_count,
      AVG((event_data->>'score')::numeric) as avg_score,
      AVG((event_data->>'wpm')::numeric) as avg_wpm
    FROM learning_events
    WHERE event_type = 'assessment'
      AND event_date > NOW() - INTERVAL '8 weeks'
    GROUP BY week
    ORDER BY week DESC
  `,
  aiSummary: true
 }
];
```

7.5 Example Insight Data

json

```
{
  "insight_type": "at_risk_children",
  "insight_data": {
    "children": [
      {
        "id": "uuid-1",
        "child_name": "Sima",
        "parent_name": "Priya",
        "parent_email": "priya@email.com",
        "days_since_session": 18,
        "risk_reason": "No session in 18 days"
      },
      {
        "id": "uuid-2",
        "child_name": "Aarav",
        "parent_name": "Rahul",
        "parent_email": "rahul@email.com",
        "days_since_session": 15,
        "risk_reason": "No session in 15 days"
      }
    ],
    "total_count": 5,
    "summary": "5 children showing disengagement. 2 haven't had sessions in 15+ days. Consider proactive parent outreach."
  },
  "computed_at": "2025-12-21T02:00:00Z",
  "valid_until": "2025-12-28T02:00:00Z"
}
```


---

8. Technical Specifications

8.1 AI Models

Purpose	Model	Tokens	Cost
Intent Classification	Gemini 2.5 Flash Lite	~120	₹0.001
Embedding Generation	text-embedding-004	~200	₹0.002
Response Generation	Gemini 2.5 Flash Lite	~650	₹0.01

8.2 Database Tables

Table	Role in rAI
learning_events	Primary RAG data source (with embeddings)
children	Child profiles, parent relationships
coaches	Coach profiles, assignments
parents	Parent accounts
scheduled_sessions	Session data for schedule queries
site_settings	Program info (price, duration)
admin_insights	Pre-computed weekly insights

8.3 Vector Search Configuration

```sql
```

```sql
-- Vector index (already created)
CREATE INDEX idx_learning_events_embedding
ON learning_events
USING ivfflat (embedding vector_cosine_ops)
WITH (lists = 100);

-- Search function
CREATE FUNCTION match_learning_events(
  query_embedding vector(768),
  match_threshold float DEFAULT 0.5,
  match_count int DEFAULT 10,
  filter_child_id uuid DEFAULT NULL
) RETURNS TABLE (
  id uuid,
  child_id uuid,
  event_type text,
  event_date timestamptz,
  event_data jsonb,
  ai_summary text,
  similarity float
) AS $$
  SELECT
    le.id,
    le.child_id,
    le.event_type,
    le.event_date,
    le.event_data,
    le.ai_summary,
    1 - (le.embedding <=> query_embedding) as similarity
  FROM learning_events le
  WHERE
    (filter_child_id IS NULL OR le.child_id = filter_child_id)
    AND 1 - (le.embedding <=> query_embedding) > match_threshold
  ORDER BY le.embedding <=> query_embedding
  LIMIT match_count;
$$ LANGUAGE SQL STABLE;
```

8.4 API Endpoint

typescript

```typescript
// POST /api/chat

interface ChatRequest {
  message: string;
  userRole: 'parent' | 'coach' | 'admin';
  userId: string;
  childId?: string;
  chatHistory?: ChatMessage[];
}

interface ChatResponse {
  response: string;
  intent: 'LEARNING' | 'OPERATIONAL' | 'SCHEDULE' | 'OFF_LIMITS';
  needsChildSelection?: boolean;
  children?: { id: string; name: string }[];
  sources?: string[];  // Event IDs for debugging
}
```

8.5 Security: Role-Based Filtering

Critical: All queries MUST filter by role at the SQL level.

```typescript
// Parent: Only own children
const parentFilter = `child_id IN (
  SELECT id FROM children WHERE parent_id = '${userId}'
)`;

// Coach: Only assigned children
const coachFilter = `child_id IN (
  SELECT id FROM children WHERE assigned_coach_id = '${userId}'
)`;

// Admin: No filter (sees all data)
const adminFilter = `TRUE`;
```

---

9. Cost Analysis

9.1 Per-Query Cost

Query Type	Average Cost
LEARNING (full RAG)	₹0.015
OPERATIONAL (SQL + format)	₹0.005
SCHEDULE (SQL + format)	₹0.005
OFF_LIMITS (template)	₹0.001

9.2 Query Distribution (Estimated)

Intent	Parent %	Coach %	Admin %
LEARNING	60%	50%	40%
OPERATIONAL	20%	25%	45%
SCHEDULE	15%	20%	10%
OFF_LIMITS	5%	5%	5%

9.3 Monthly Cost Projections

Scale	Parents	Coaches	Admin	Weekly Insights	Total
100 children	₹100	₹25	₹10	₹100	**₹235**
1,000 children	₹800	₹150	₹50	₹400	**₹1,400**
5,000 children	₹2,500	₹500	₹150	₹800	**₹3,950**
10,000 children	**₹3,000**	**₹750**	**₹250**	**₹1,200**	**₹5,200**

9.4 Cost Optimization Strategies

1. **Intent classification first** - Cheap filter before expensive RAG
2. **Predefined SQL templates** - No AI needed for operational queries
3. **Weekly insight caching** - Compute once, serve many times

4. **Token limits** - Cap response length at 150 tokens
5. **Conversation history limit** - Max 10 messages for context

---

10. Implementation Roadmap

Phase 1: Data Pipeline (Priority 1) - Day 1

Goal: Get learning data flowing into `learning_events`

Task	Status
Modify `/api/assessment/analyze` to save to learning_events	☐
Add embedding generation after assessment	☐
Test with real assessment	☐
Verify data appears in learning_events table	☐

Phase 2: Chat API Enhancement (Priority 2) - Day 1-2

Goal: Build the intent-based routing system

Task	Status
Add intent classifier function	☐
Build OPERATIONAL query handler with SQL templates	☐
Build SCHEDULE query handler	☐
Build OFF_LIMITS handler with redirects	☐
Add conversation memory support	☐
Add multi-child disambiguation	☐
Update LEARNING handler with new prompts	☐

Phase 3: UI Integration (Priority 3) - Day 2

Goal: Working chat on all dashboards

Task	Status
Fix ChatWidget on parent dashboard	☐
Verify coach AI assistant page	☐
Add chat to admin dashboard (if needed)	☐
Test conversation memory in UI	☐

Phase 4: Admin Insights (Priority 4) - Day 3

Goal: Weekly pre-computed insights for Admin

Task	Status
Create `admin_insights` table	☐
Build QStash cron job for weekly computation	☐
Build insight retrieval in chat handler	☐
Test insight queries	☐

Phase 5: Testing & Refinement - Day 3-4

Task	Status
End-to-end test: Parent journey	☐
End-to-end test: Coach journey	☐
End-to-end test: Admin journey	☐
Test multi-child scenarios	☐
Test conversation memory	☐
Verify anti-hallucination	☐

11. Appendix: Query Examples

11.1 Parent Query Examples

Query	Intent	Handler	Example Response
"How is Sima doing?"	LEARNING	RAG	"Sima has been making great progress! Her fluency improved from 45 to 58 WPM over the last 3 sessions. Coach Rucha noted she's doing particularly well with consonant blends. One area to focus on is 'th' sounds - you can practice words like 'the', 'this', 'that' at home."
"When is the next session?"	SCHEDULE	SQL	"Sima's next session is tomorrow (Monday) at 4:00 PM with Coach Rucha. Here's the Google Meet link: [link]"
"Who is my coach?"	OPERATIONAL	SQL	"Your coach is Rucha. You can connect with her during scheduled session times."
"How much did I pay?"	OPERATIONAL	SQL	"Your enrollment is active. You paid ₹5,999 on December 15, 2025 for the 3-month program."
"What are other children scoring?"	OFF_LIMITS	Redirect	"I can only share information about your enrolled children. Would you like to know more about Sima's progress?"

11.2 Coach Query Examples

Query	Intent	Handler	Example Response
"How many students do I have?"	OPERATIONAL	SQL	"You have 8 active students assigned to you."
"Prepare me for tomorrow's session with Aarav"	LEARNING	RAG	"For tomorrow's session with Aarav: Last session you worked on long vowels - he showed good progress but struggled with 'ea' words. His parent mentioned he practiced 4 days this week. Suggestion: Start with a quick 'ea' review, then introduce digraphs. The phonics memory game worked well with him last time."
"What's my schedule today?"	SCHEDULE	SQL	"Today you have 3 sessions: Sima at 10 AM, Aarav at 2 PM, Zara at 4 PM. [Meet links]"
"What are my earnings?"	OFF_LIMITS	Redirect	"For earnings and payout information, please check your Earnings dashboard. Is there anything about your students' learning I can help with?"

11.3 Admin Query Examples

Query	Intent	Handler	Example Response
"Which children are at risk?"	LEARNING	Cached Insights	"5 children haven't had sessions in 2+ weeks: Sima (18 days), Aarav (15 days), Zara (14 days). 2 parents raised concerns in the last week. Consider proactive outreach."
"How many enrollments this month?"	OPERATIONAL	SQL	"42 new enrollments this month. 156 total active children. Conversion rate from assessment: 8.2%."
"What are children struggling with most?"	LEARNING	Cached Insights	"This week, 40% of sessions focused on phonics remediation. Most common struggle areas: 'th' sounds (23%), long vowels (18%), digraphs (15%). Recommendation: Consider adding more phonics resources to the curriculum."
"What's the revenue this month?"	OFF_LIMITS	Redirect	"For revenue and financial data, please check the Revenue dashboard. Would you like to know about learning metrics instead?"

Document Approval

Role	Name	Date	Signature
Product Owner	Amit Kumar Rai		☐
Technical Lead	Claude	Dec 21, 2025	✅

Next Steps:

1. Review this document thoroughly
2. Confirm all boundaries and decisions
3. Proceed to implementation

Document Version: 1.0
Last Updated: December 21, 2025

Ready for Implementation: Pending Amit's Review