PHONIX:

AUTOMATING SPEECH SOUND DISORDER RELATED THERAPIES AND DATA TRACKING FOR SPEECH-LANGUAGE PATHOLOGISTS

SRI BHARGAVA BHAMIDI

bhamidi@buffalo.edu

in collaboration with

AMIR NASSERELDINE QINGXIAO ZHENG JIAJIE LI

ALISON HENDRICKS ABBIE OLSZEWSKI

May 2026

A supervised research project report

submitted to the Faculty of the Graduate School

The University at Buffalo, State University of New York

Advisor: DR. JINJUN XIONG

Department of Computer Science and Engineering

Acknowledgments

I am deeply grateful to my advisor, Dr. Jinjun Xiong, for his guidance, support, and mentorship throughout this research at the University at Buffalo. His direction shaped both the technical scope and the clinical aspirations of the Phonix platform, and his confidence in the work made it possible to carry the project from prototype to a deployed clinical system.

Abstract

Speech-Language Pathologists (SLPs) treating children with speech sound disorders spend substantial time outside of therapy itself: searching for word lists that simultaneously satisfy a target phoneme, a word position, an imageability constraint, an age-of-acquisition ceiling, and a frequency band; preparing distinct materials for each evidence-based intervention approach a clinician deploys; and manually tracking per-trial accuracy across sessions. This report presents **Phonix**, a deployed web platform that automates intervention-material generation and in-session data tracking for five evidence-based phonological and motor-speech therapies: Minimal Pair Therapy, Contextual Utilization, Multiple Oppositions, the Cycles Approach, and Rapid Syllable Transition Treatment.

The system is backed by a 289,000-word English lexicon annotated per phoneme with articulatory features (place, manner, and voicing for consonants; height, backness, and rounding for vowels), syllabification, frequency, age of acquisition, three independent imageability scores, concreteness, Wiktionary-derived semantics, and WordNet hypernymy. Each therapy module implements a distinct algorithmic backbone: a nine-factor shaping phoneme scorer, an IPA-first pseudo-word generator with controllable stress patterns, a ten-stage MongoDB rime-aggregation pipeline, a bombardment-and-probe pipeline for the Cycles Approach, and a filtered minimal-pair lookup. An embedded Trial Collector captures every clinical observation an SLP makes during a session, including the SODA error taxonomy and cue level, and feeds a reporting engine composed of seven analyzers that produce per-phoneme, per-position, and goal-tracking summaries with explicit confidence labels.

The frontend is a Next.js single-page application; the backend is a FastAPI service with a repository-pattern MongoDB layer and a self-hosted Supabase PostgreSQL database protected by row-level security. The main therapy platform is deployed in production at `phonix-therapy.xlabub.com`, with a companion public Tools site for discovery utilities at `phonix.xlabub.com`.

Contents

Chapter 1

Introduction

1.1 The Clinical Infrastructure Gap

Speech-Language Pathologists (SLPs) carry caseloads in the dozens to hundreds and serve children with a wide range of speech sound disorders. They are deeply trained in phonological systems, articulatory mechanics, and developmental trajectories. They know how to choose the right intervention approach, whether that is Minimal Pair contrastive therapy, the Cycles Approach, Contextual Utilization, Multiple Oppositions, or Rapid Syllable Transition Treatment, and how to adjust it in response to a child during a session. What slows them down is not clinical judgment; it is the work of *assembling materials*.

Delivering an evidence-based phonological intervention requires word lists that satisfy several constraints at once: a target phoneme, a position in the word (initial, medial, or final), a level of imageability so the child can picture the referent, an age of acquisition appropriate for the child, and a frequency band that supports recognition. Most SLPs build these lists from textbooks, photocopied flashcards, dated online word-finders, or memory. Each new child and each new session restarts the work. Preparation effort is largely non-reusable, scales linearly with caseload, and represents a structural gap in *clinical infrastructure*, not in clinical knowledge. Figure 1.1 contrasts the manual workflow with what Phonix offers in its place.

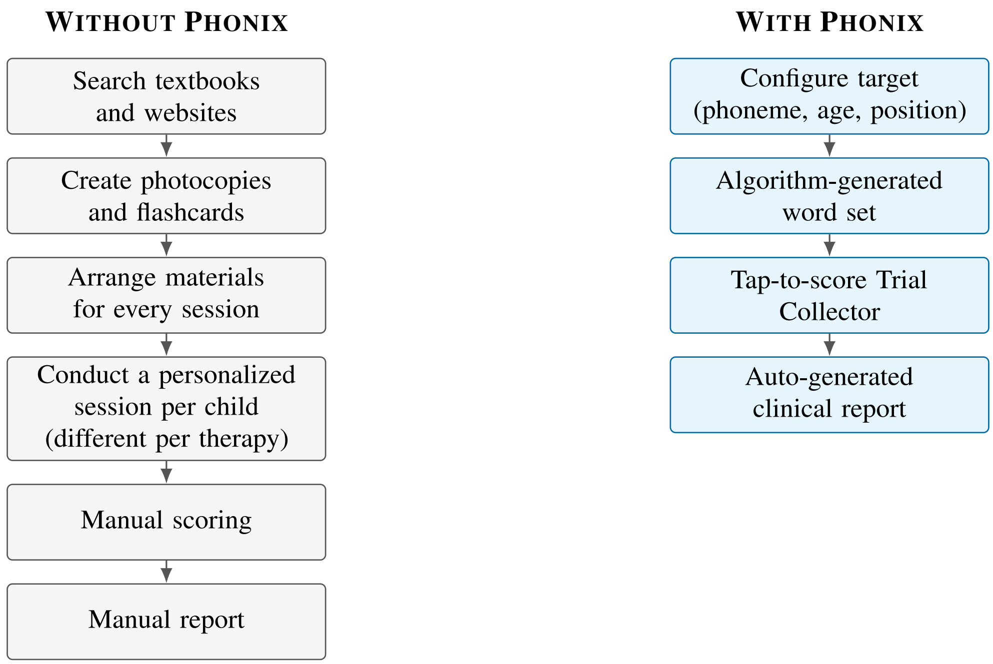

Figure 1.1. The clinical infrastructure gap. The manual workflow on the left scales linearly with caseload because every step restarts for every child and every therapy; Phonix collapses the same flow into four reusable, queryable steps.

1.2 Contributions of This Work

This report presents Phonix, a web-based clinical platform that addresses that infrastructure gap. The main therapy platform is deployed in production at `phonix-therapy.xlabub.com`, accompanied by a public Tools site for discovery utilities at `phonix.xlabub.com`. The contributions of this work are:

1. A 289,000-word English lexicon annotated per-phoneme with articulatory features, syllabification, frequency, age of acquisition, three independent imageability scores [1, 4], concreteness, Wiktionary-derived semantics [14], and WordNet hypernymy [6], combined into a single MongoDB representation that supports clinically-relevant queries in real time.
2. Five algorithmically-distinct therapy modules implementing Minimal Pair Therapy [12], Contextual Utilization [3], Multiple Oppositions [13], the Cycles Approach [8], and Rapid Syllable Transition Treatment [5].

3. An embedded Trial Collector with a three-layer data model that cleanly separates clinician input, automatically-derived therapy context, and session metadata.
4. A reporting engine composed of seven analyzers that use a custom linear regression implementation and threshold-based pattern detection to produce per-phoneme, per-position, per-session, and goal-tracking summaries from raw trial data, with explicit confidence labels.
5. A deployed multi-tenant system serving SLPs with Supabase-backed authentication [10], row-level-security-isolated client data, and an administrative console for API-key management and analytics.

Chapter 2

System Architecture

2.1 High-Level Topology

Phonix is a three-tier web system. The client is a Next.js [11] single-page application built with React, Tailwind, and a shadcn/Radix UI primitives layer. The API server is a FastAPI [9] service. Three storage backends sit behind the API: MongoDB [7] for the phonolinguistic corpus, a self-hosted Supabase instance [10] (PostgreSQL plus Supabase Auth) for client and session data, and MinIO for AI-generated word illustrations. A legacy Solr instance is retained for narrow autocomplete fallbacks. The system is deployed under PM2 process management on a single host. Figure 2.1 shows the topology.

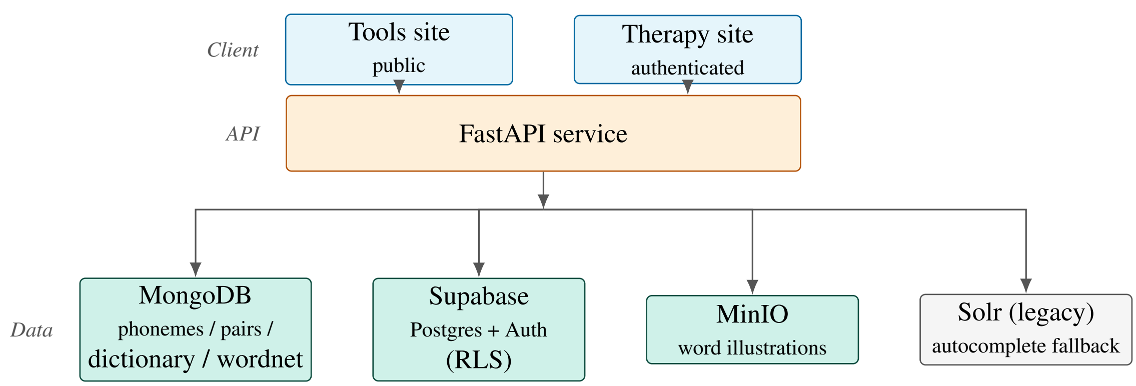

Figure 2.1. Phonix system architecture. Two frontend deployments share a codebase but are configured at build time as a public Tools site or an authenticated Therapy site. All traffic terminates at the FastAPI service; the frontend never speaks to Supabase directly.

2.2 Frontend: Two Halves of One Codebase

The frontend is organized into two logical halves. The first is a set of public discovery utilities for word search, articulation search, autocomplete, minimal and maximal pair lookup, and consonantcluster exploration; these are the tools an SLP reaches for during clinical preparation. The second is the authenticated clinical workflow itself, containing the five therapy modules and the reporting UI. The two halves live in the same codebase but ship as two separate production deployments, chosen at build time, so the same components, design tokens, and shared utilities back both sites without duplication. Figure 2.2 shows the Therapy site after login; Figure 2.3 shows a representative Tools site result.

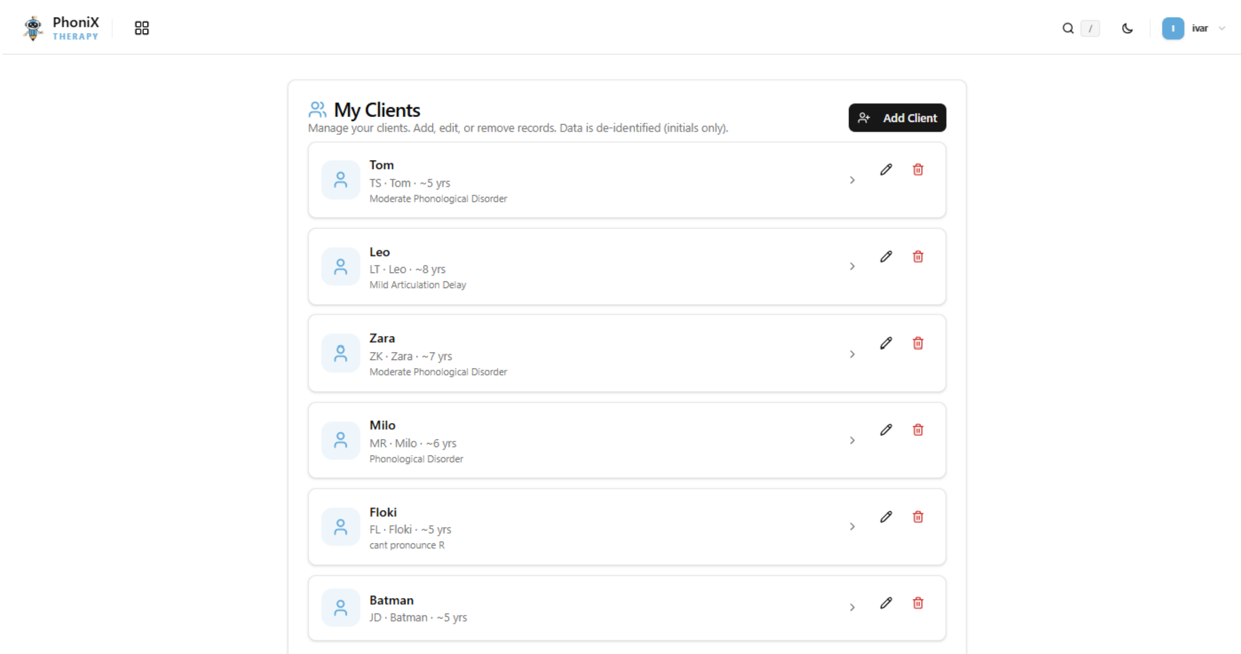

Figure 2.2. Therapy site dashboard with the client list.

2.3 Backend: Layered Services and Repositories

The backend is organized in three layers. Routes parse and validate every incoming request, hand it to a service that contains the clinical or business logic, and the service in turn delegates database access to a repository. The repository is the only place that knows the shape of a MongoDB query; if a query has to change because an index moves or a field is renamed, only the repository changes, and every caller continues to see the same interface. A single `MongoService` facade preserves a unified entry point for legacy code that depends on it, while each repository can be unit-tested and

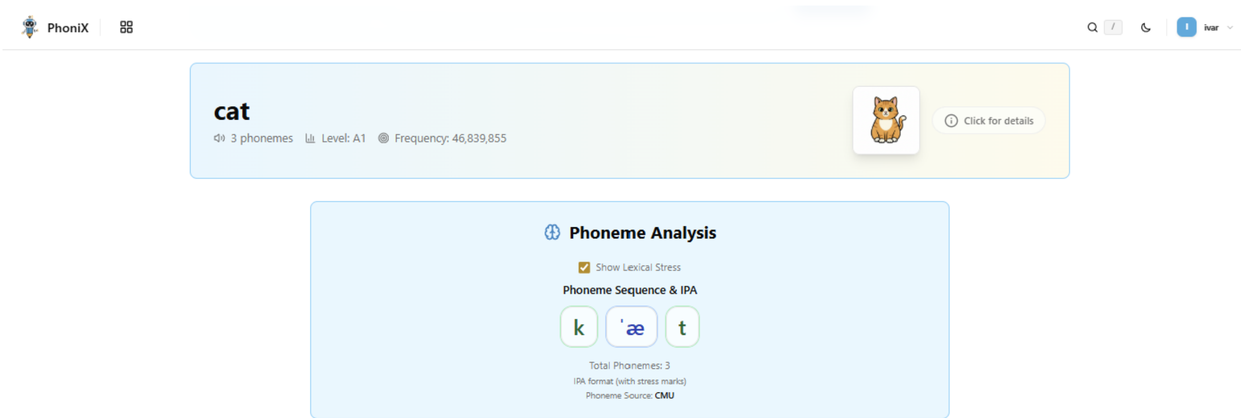

Figure 2.3. A representative result from the Tools site word search, showing the phoneme detail modal for the word "cat".

swapped independently. The therapy modules, the reporting analyzers, the search endpoints, and the administrative console all sit at the service layer, and all of them reach the data layer through this same pattern. Table 2.1 lists the principal libraries that the system rests on.

Table 2.1. Principal technology stack.

Tier	Component	Library	Version
Frontend	Framework	Next.js	15
	UI library	React	19
	Styling	Tailwind CSS	4
Backend	Framework	FastAPI	0.104
	Validation	Pydantic	2.5
	MongoDB driver	PyMongo	4.6
	Supabase client	supabase-py	2.3
Data	Document store	MongoDB	6
	Relational	PostgreSQL (Supabase)	15
	Object store	MinIO	–

2.4 The Middleware Chain

Four middlewares wrap every request, in a documented order (Figure 2.4). CORS handling is first. Then an analytics middleware fires-and-forgets a row to a MongoDB `api_usage_logs` collection; the log write is wrapped in a permissive `try/except` so a logging failure never blocks a request. Next, a user-session middleware resolves an HttpOnly cookie named `phx_session` against a server-side session collection; the routes that require authentication are listed in a `PROTECTED_ROUTE_PREFIXES` constant. Finally an API-key middleware enforces rate limits, per-endpoint permissions, and lifetime quotas for external integrators; it contains a deliberate *Origin/Referer bypass* so that frontend traffic originating from an approved domain is allowed without a key, while direct API consumers must supply one.

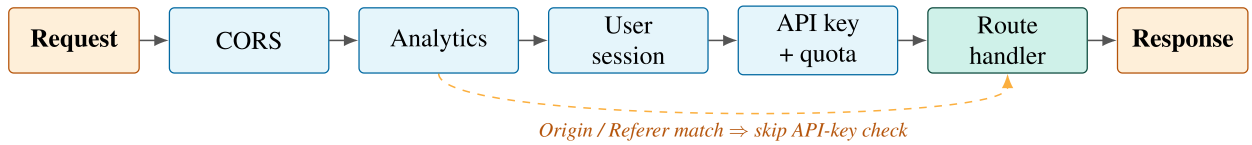

Figure 2.4. The middleware chain. Frontend traffic from an approved Origin or Referer bypasses the API-key gate (orange dashed); direct API consumers traverse the full chain.

2.5 Data and Authentication Boundaries

A strict invariant is enforced in the deployed system: the frontend never calls Supabase directly. Authentication is performed against the FastAPI service, which mints an opaque session cookie; the cookie is the only credential carried over the network from the client. Figure 2.5 shows the login flow.

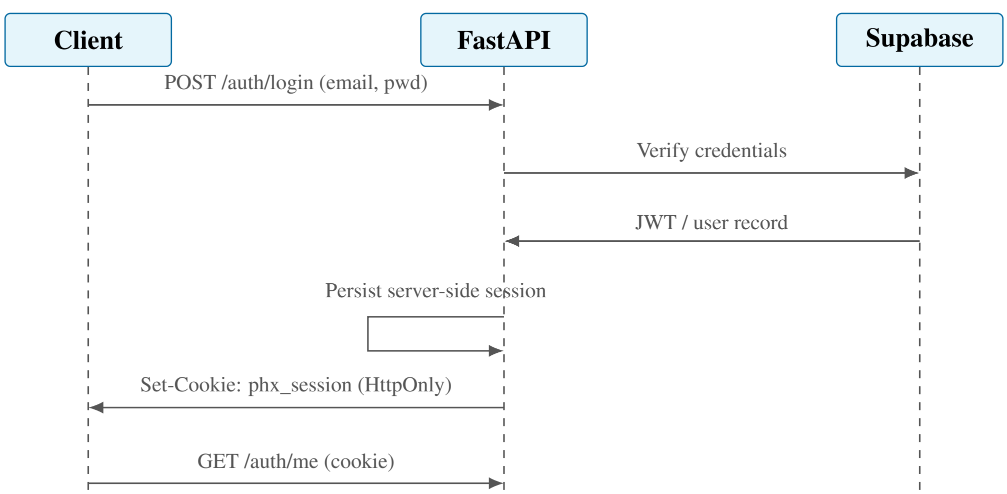

Figure 2.5. Login sequence. All Supabase calls are server-side; the client only ever sees an opaque `phx_session` cookie.

Chapter 3

Data Foundation

3.1 The Annotated Lexicon

The Phonix lexicon contains approximately 289,000 English word forms, each annotated as a single clinical object. Every word carries a primary ARPABET phoneme transcription [2] with stress digits preserved, aligned arrays of articulatory features at every phoneme position, syllabification produced by the Pyphen library, word-level frequency drawn from a large web corpus, part-of-speech tags, and syllable counts. Where available, each word additionally carries an age-of-acquisition rating [4], three independent imageability scores (Brysbaert, MRC, and CPB), a concreteness rating [1], and a Fry readability rank.

A single word object therefore exposes eight annotation surfaces. The word *cat*, for example, is stored as a document carrying phonemes /k æ t/; per-position articulation (Velar / Stop / Voiceless at the onset, Open / Front / Unrounded at the vowel, Alveolar / Stop / Voiceless at the coda); developmental ratings (age of acquisition 2.8 years, frequency 18,432); imageability from three sources; the Wiktionary definition "a small domesticated carnivorous mammal"; part of speech (Noun); syllable structure (one syllable); and the WordNet semantic ancestry `animal` → `domestic animal` → `cat`. Every word in the lexicon receives this same shape.

The annotated lexicon is stored in MongoDB rather than a relational database because the per-word arrays (six articulatory features at up to a dozen phoneme positions) and the per-word nested objects (definitions, examples, synonyms, antonyms) are awkward to normalize into rows and would multiply the join cost of every therapy query.

3.2 The MongoDB Collections

Four core collections back the system; Figure 3.1 sketches the join points to the Supabase tables, and Table 3.1 summarizes the collection sizes and roles.

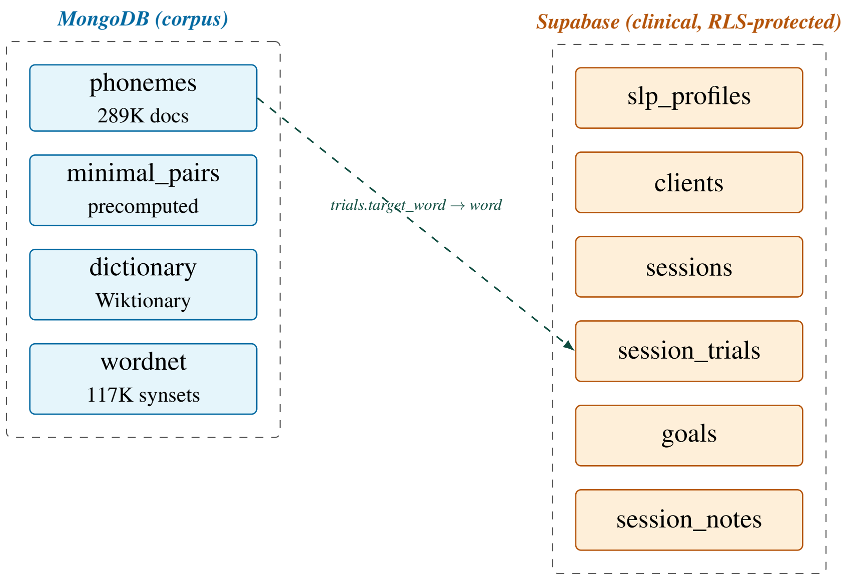

Figure 3.1. Data model overview. Phonolinguistic data lives in MongoDB; client and session data live in Supabase Postgres with Row-Level Security. Joins between the two are performed in application code at report time, not at the database layer.

Table 3.1. The four core MongoDB collections.

Collection	Approx. size	Primary use
`phonemes`	289,072 docs	Word inventory; per-phoneme articulation; clinical filters
`minimal_pairs`	precomputed	Fast contrastive-pair lookup for Minimal Pair Therapy
`dictionary`	Wiktionary	Definitions, examples, etymologies (kaikki.org)
`wordnet`	117,659 docs	Synset graph for semantic navigation and category trees

`phonemes`. The primary word inventory. Indexes are deliberately denormalized for query patterns: a text index on `word`, a multi-key index on the `phoneme` array (the most-queried field), a unique index on the legacy Solr-origin `id`, and compound indexes on `(phoneme, freq)` and `(POA, MOA, freq)` for the articulation search.

`minimal_pairs`. A precomputed pair index. Each document represents one ordered pair of words differing in exactly one phoneme position, with the differing position, the type of difference (manner, place, voicing, or multi-feature), and a `minimal_type` label of strict, not-strict, or maximal. Precomputing this collection trades storage for lookup speed; at therapy time a contrastive query becomes a single indexed read.

`dictionary`. A cleaned dump of the kaikki.org machine-readable Wiktionary export [14]. Used to enrich the `phonemes` collection with definitions, examples, sound recordings, etymologies, and cross-references.

`wordnet`. A compact synset graph derived from WordNet 3.0 [6], with each synset augmented by a precomputed `all_parents` list and a `descendant_count`. This shape supports a fast Lowest-Common-Ancestor query for the hypernym-path visualization used in the search tools.

3.3 The Supabase Tables

Client and session data are kept separate from the phonolinguistic corpus, in a self-hosted Supabase PostgreSQL instance. Six tables are in use: `slp_profiles` stores SLP accounts; `clients` stores only initials, a nickname, a birth month and year (not exact date of birth), a diagnosis, and an optional avatar identifier, minimizing identifiable information by design; `sessions` stores one row per therapy session with the session configuration and a JSON blob of session data; `session_trials` stores one row per individual correct/incorrect tap with the SODA error classification, the cue level, and the self-correction flag; `goals` stores IEP-style targets with baseline, target, and current accuracy plus a status enum; and `session_notes` stores free-text observations categorized as observation, behavior, recommendation, or parent feedback. All tables enable Row-Level Security; the active policy filters every read and write by the authenticated SLP's UUID.

Chapter 4

Therapy Modules

4.1 A Unified Therapy Workflow

All five therapy modules in Phonix follow the same four-step workflow shown in Figure 4.1: a Configuration step in which the SLP selects a target phoneme, the child's age, and therapy-specific parameters; a Results step that displays the algorithmically-generated word sets; a Practice step that drives in-session trial collection through illustrated word cards; and a Report step that summarizes session accuracy and routes the trial data into the reporting engine described in Chapter 5. Each module realizes this workflow in its own Next.js page module under `frontend/src/app/(therapy)/`, with module-specific configuration UI, results visualization, and report sections. Table 4.1 summarizes the five modules at a glance.

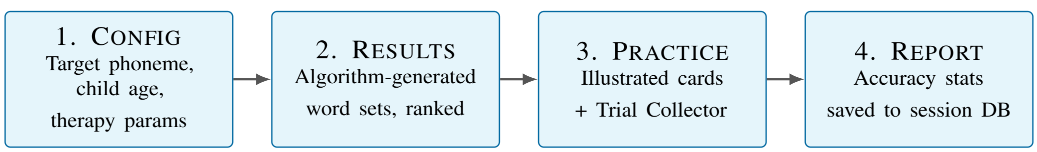

Figure 4.1. The unified four-step therapy workflow. All five Phonix modules realize this same pipeline; only the algorithm in steps 1–2 differs across modalities.

Table 4.1. Therapy modules at a glance.

Therapy	Indication	Algorithmic backbone
Minimal Pair	Phonological substitution	Precomputed pair index lookup
Contextual Utilization	Articulation, shaping	Nine-factor shaping phoneme scorer
Multiple Oppositions	Phoneme collapse	Ten-stage rime aggregation pipeline
Cycles Approach	Unintelligibility, multiple processes	Bombardment + multi-column probe
ReST	Childhood Apraxia of Speech	IPA-first pseudo-word generator

4.2 Minimal Pair Therapy

The minimal-pair module is the simplest in terms of data flow because it consumes the precomputed `minimal_pairs` collection directly. The SLP selects a target phoneme, one or more positions (initial, medial, final), and optionally a contrast phoneme, an age-of-acquisition ceiling, a pair-type filter (strict, not-strict, or maximal), and a difference-feature filter (manner, place, voicing, or multiple). A single indexed MongoDB query returns the candidate pairs, which are sorted by imageability descending and presented for selection (Figure 4.2). From the SLP's perspective the result is what would have taken twenty minutes of textbook searching, returned in well under a second.

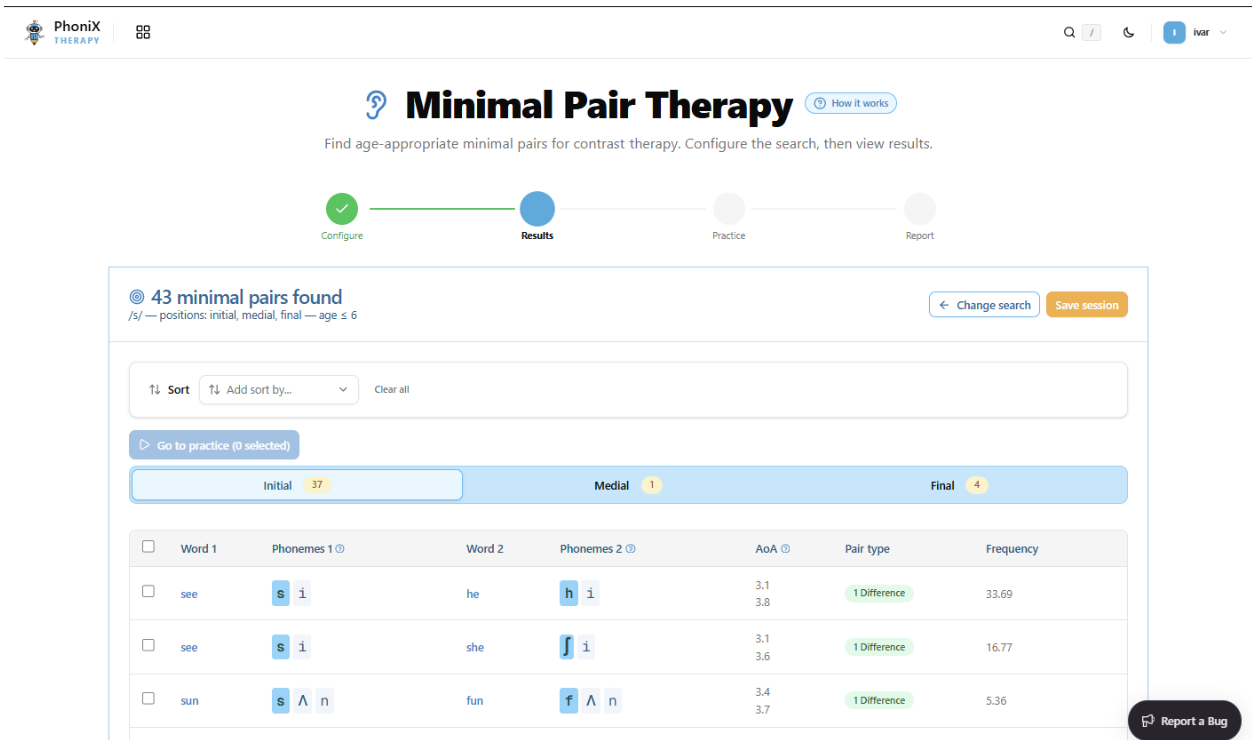

Figure 4.2. Minimal Pair Therapy results screen.

4.3 Contextual Utilization: A Nine-Factor Shaping Phoneme Scorer

Contextual Utilization required the most novel algorithm in the system. The clinical task is: given a target phoneme that the child cannot produce, suggest a *shaping phoneme* that the child probably *can* produce and that shares enough articulation with the target to bridge to it. The Phonix implementation scores every candidate on nine factors. Three address articulatory similarity to the target: a pattern-specificity score (a stop-to-fricative bridge, for instance, receives the maximum 15 points), an extra manner-pattern bonus of up to 20 points reserved for the most powerful clinical patterns (stop-to-fricative and stop-to-affricate), and a voicing-match score worth up to 10 points. Three address whether the child can plausibly produce the candidate: a clinical-difficulty score (up to 8 points for sounds rated easy), a 5-point bonus if the candidate is in a hardcoded "commonly mastered" set (T, D, P, B, M, N, K, HH), and an age-appropriateness score (up to 10 points when the child's age clears the candidate's mastery age by at least two years). The remaining three are minor terms: a motor-complexity score (up to 6), a phonological-process alignment bonus (up to 4) when the candidate participates in the same phonological process as the target, and a small visual-similarity score (up to 5) computed against a pre-tabulated similarity matrix. The total score for any candidate is the sum of these nine; ties are broken by ARPABET alphabetical order. After ranking, results are organized by the *following vowel height*, because high front vowels facilitate forward tongue placement for many target consonants. Figure 4.3 shows the result page.

4.4 Multiple Oppositions: Rime Aggregation on the Phonemes Collection

The Multiple Oppositions implementation queries the `phonemes` collection directly through a ten-stage MongoDB aggregation pipeline. The pipeline:

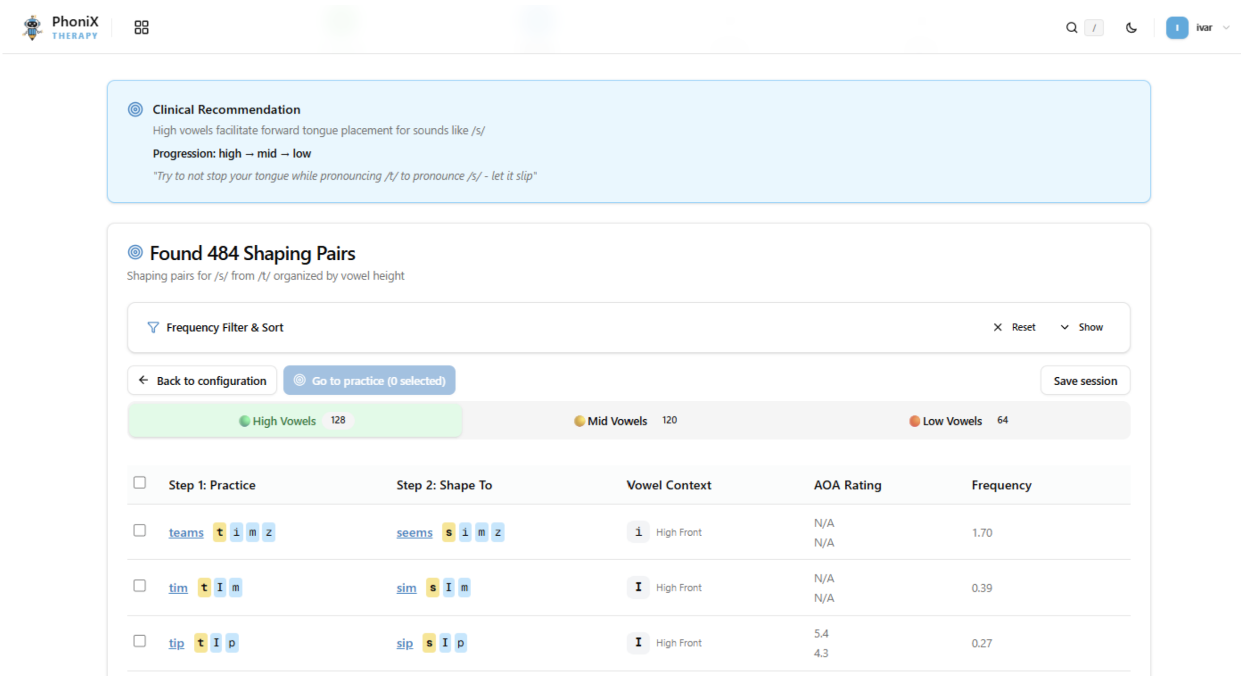

Figure 4.3. Contextual Utilization results page showing the ranked shaping phonemes and the recommended word pairs organized by following-vowel height.

1. filters to documents with two or more phonemes, frequency at least 10, word length at least 3, and age of acquisition no greater than the child's age plus three (or unknown);
2. projects an `onset` (the first phoneme) and a `rime` (the remaining phonemes);
3. restricts to documents whose onset is one of the required onsets (one error sound plus all selected target sounds);
4. builds a canonical rime key by joining the rime phonemes and substitutes a high sentinel for unknown age-of-acquisition values so they sort last;
5. sorts by age of acquisition ascending then frequency descending, and groups by (rime, onset), keeping the best word per pair;
6. groups again by rime alone, collecting all onsets present for that rime;
7. requires that the rime's onset set covers *all* required onsets;
8. computes a per-rime minimum frequency (a "weakest link" criterion);
9. sorts rimes by minimum frequency descending, then average age of acquisition ascending,

then total frequency descending;

10. projects the final response shape.

The pipeline returns rhyming word families such as "cold/sold/told/polled." The result for a session is shown in Figure 4.4. When no real word exists for the child's error sound under a given rime, a pseudo-word is generated by concatenating the error sound's ARPABET letter with the rime's grapheme rendering, for example *dap* from `D` plus `[AE1, P]`.

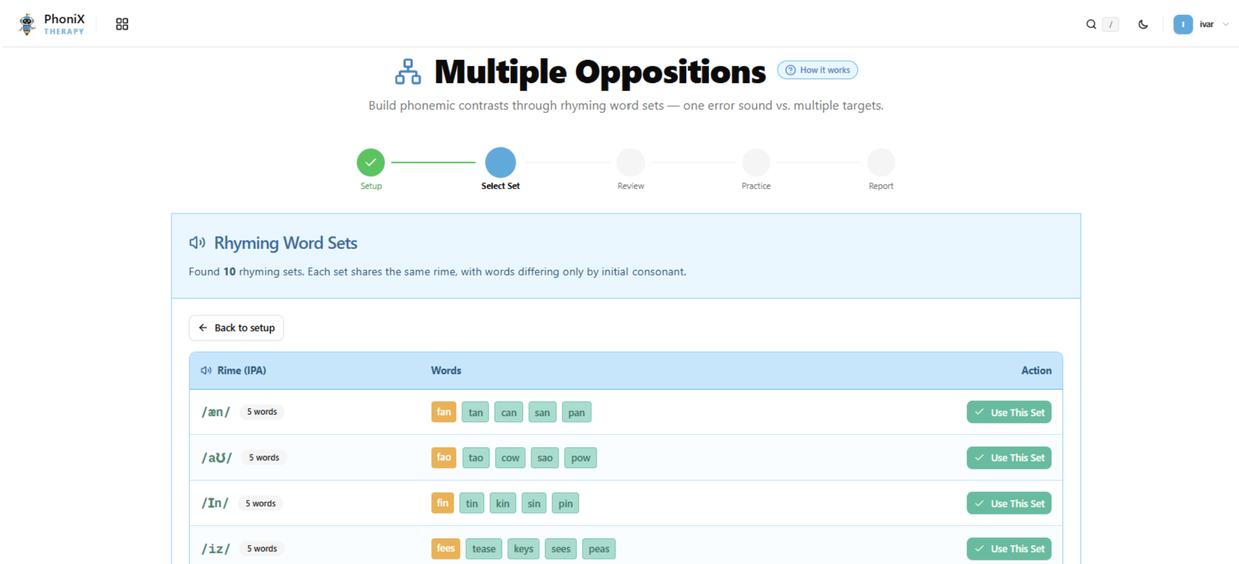

Figure 4.4. Multiple Oppositions rime sets view for an error sound collapsing several target sounds.

4.5 The Cycles Approach

Cycles requires four artifacts per session: a bombardment list of high-frequency words the child hears but does not have to produce, a smaller set of card candidates the child does practice, a multi-column stimulability probe that tests several phonemes within a pattern, and a parent home-program card. Phonix generates the bombardment list by sampling the top-200 imageability-then-frequency-ranked words for the target pattern, deduplicating, sorting by frequency descending, oversampling at three times the desired count, shuffling, and taking the requested number. The shuffle is deliberate: it ensures that successive re-fetches return different but equally clinically appropriate lists, supporting variety across sessions. Card candidates are selected from the top-50 words by position, deduplicated and sorted by imageability descending, capped at ten. The multi-column probe is driven by a hardcoded option set per phonological pattern: the Final-Consonant-Deletion pattern probes five common final consonants; the S-Cluster pattern probes seven clusters; Fronting Velars probes five velar candidates.

The frontend implements the Cycles protocol as the nine-step state machine shown in Figure 4.5. Figure 4.6 shows the practice step in the deployed UI.

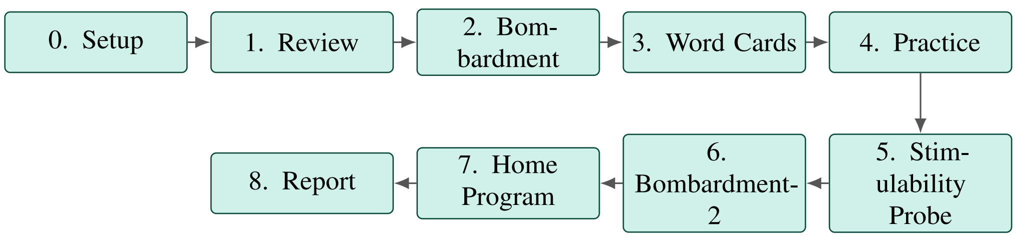

Figure 4.5. The Cycles Approach state machine as implemented in the frontend: nine steps from Setup through the parent Home Program card to the Report.

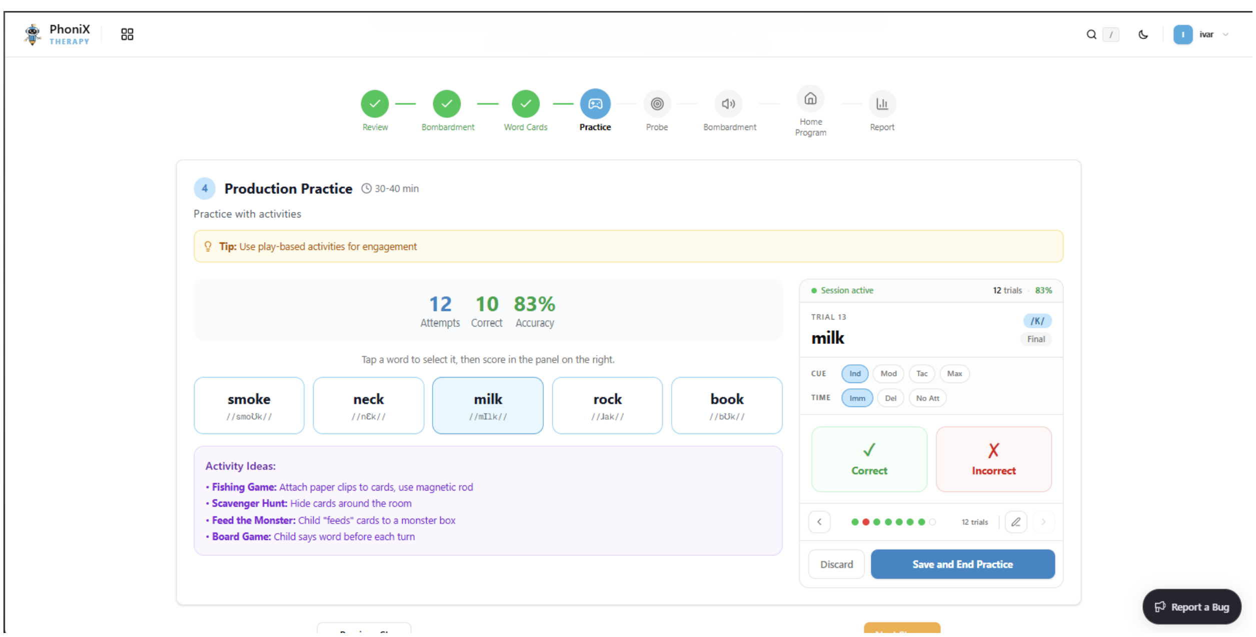

Figure 4.6. Cycles Approach Practice step with an illustrated word card and the trial scoring strip.

4.6 ReST: IPA-First Pseudo-Word Generation

ReST presented the toughest engineering challenge of the five modules. The system must generate phonotactically valid pseudo-words with controllable syllable count, controllable stress pattern (strong-weak, weak-strong, weak-weak, and others), and a configurable excluded-phoneme list (typically excluding R, L, and TH for younger children). The generator is two-stage, illustrated in Figure 4.7. The first stage operates in IPA: each syllable's shape is drawn from {CV, CVC, VC}; onset and coda phonemes are drawn from constrained pools (the onset pool excludes /ŋ/, the coda pool includes it); stress digits are appended to vowels (primary stress is '1', weak stress is '0' with vowels biased toward schwa). The second stage converts the IPA pseudo-word to a readable orthographic rendering through a hand-curated ARPABET-to-grapheme map, an explicit Sydney-style choice that prefers *fedar* to *fadra*, with similarity-based rejection of words too close to others already in the generated list.

The practice queue is then constructed by duplicating each pseudo-word eight times and Fisher-Yates-shuffling the result, so a five-word list produces forty productions in random order (Figure 4.8). Feedback is delivered to the child every ten productions, in keeping with the motor-learning theory underlying ReST: random practice schedules and intermittent feedback are known to produce better motor-learning outcomes than blocked practice and continuous feedback, despite feeling slower in the moment.

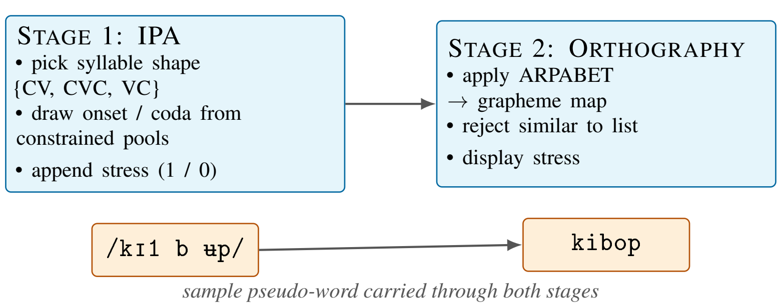

Figure 4.7. The ReST two-stage pseudo-word generator. IPA construction first (left), orthographic rendering second (right). The sample word *kibop* is shown traversing both stages.

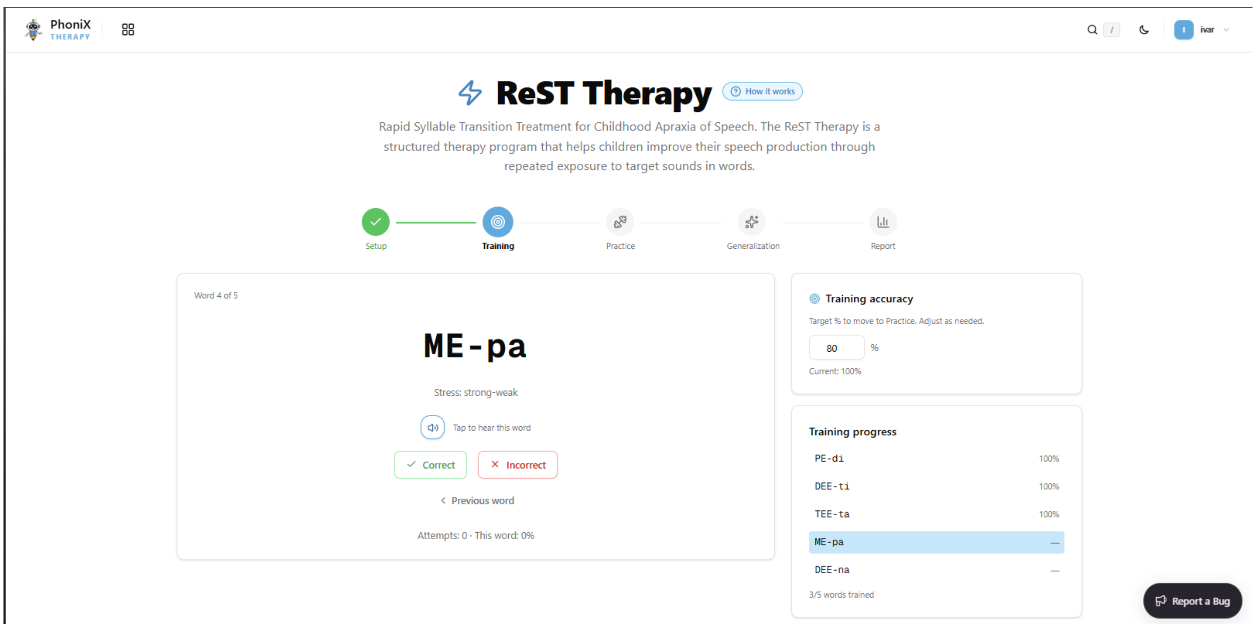

Figure 4.8. ReST Practice phase showing a pseudo-word card with IPA and stress display.

Chapter 5

The Clinical Reporting Engine

5.1 The Trial Collector: A Three-Layer Data Model

Every therapy session in Phonix feeds a Trial Collector, a frontend library implemented as a React hook that buffers per-tap trial data and submits it to the backend at session end. The collector cleanly separates three layers of information that a trial actually contains, illustrated in Figure 5.1.

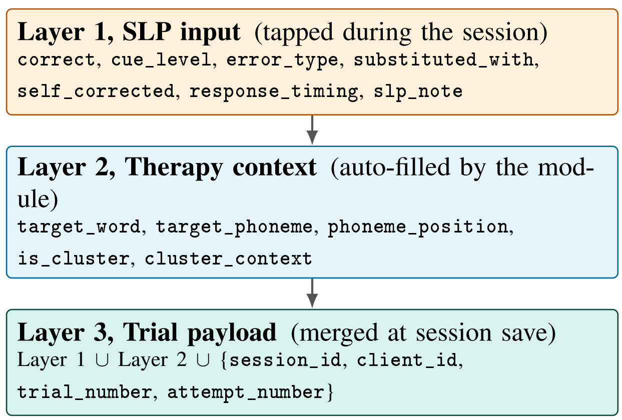

Figure 5.1. The three-layer trial data model. Layers 1 and 2 are populated independently during the session; Layer 3 is the merged payload persisted to Supabase `session_trials` when the SLP commits the session.

Layer 1: SLP input. The fields the clinician taps as the child speaks: a boolean correctness flag, a cue level (independent, model, tactile, max-cue), an error-type label drawn from the SODA taxonomy (Substitution, Omission, Distortion, Addition), the substituted phoneme when applicable, a self-correction flag, a response-timing label (immediate, delayed, no attempt), and an optional free-text note.

Layer 2: therapy context. Information that the therapy module already knows and supplies automatically: the target word, the target phoneme, the phoneme position, a cluster flag, and the cluster context where applicable. The SLP never enters these by hand.

Layer 3: trial payload. The merger of layers 1 and 2 with the session identifier, the client identifier, a sequential trial number within the session, and an attempt number that counts how many times the same word has been tested in the same session.

The collector retains its buffer locally until the SLP commits the session at the end. Two guarantees are enforced. If a trial is deleted mid-session, all subsequent trials are atomically renumbered to keep the trial-number sequence gapless. The hook installs a `beforeunload` warning so an SLP cannot accidentally close the tab and lose buffered trials. A dev-mode context validator catches malformed therapy context, such as bad target words or missing target phonemes, before any trial enters the buffer, so corrupted data cannot reach the reporting engine.

5.2 The Seven Analyzers

The reporting engine is composed of seven independent analyzers, each implemented as a pure function that consumes the trial dataset for one client. Figure 5.2 shows the fan-out.

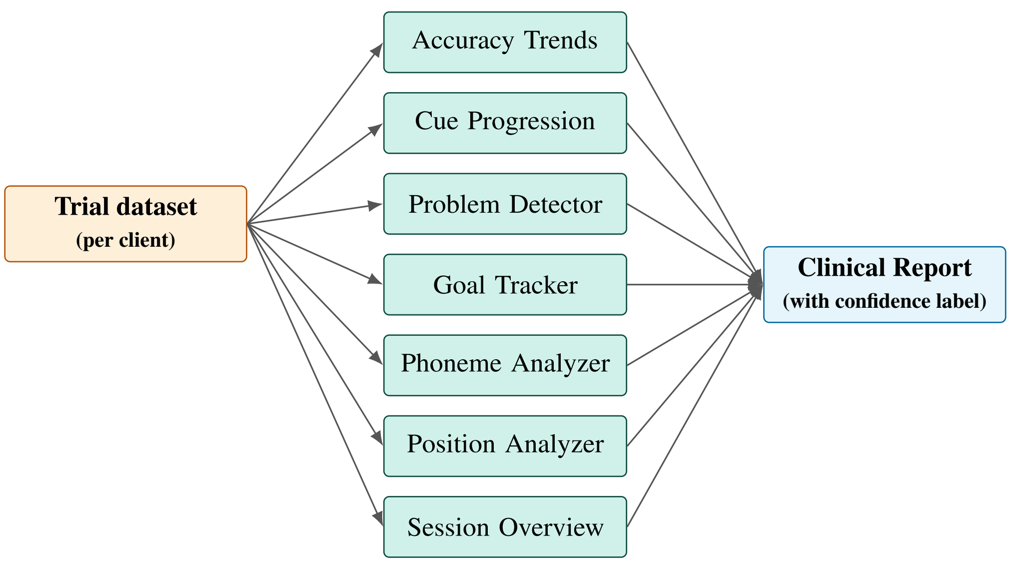

Figure 5.2. Fan-out of the reporting engine. The same per-client trial dataset feeds seven independent analyzers; each emits one section of the final clinical report. A confidence label propagates with the report based on dataset size.

Accuracy Trends. Computes per-session accuracy, a three-session rolling average, and a linear regression slope across the session series. A slope greater than $+2\%$ per session is classified as "improving"; less than -2% is "declining"; the middle band is "stable." A separate per-phoneme trends computation compares early-window accuracy to late-window accuracy with a change threshold of 10%, so a child can be improving overall while still showing stagnation on a single phoneme.

Cue Progression. Tracks the percentage of independent (no-cue) trials over time, the distribution of cue levels in each session, and the self-correction rate (errors the child caught and fixed themselves). A regression slope greater than $+3\%$ per session on independent-trial percentage is classified as "increasing independence."

Goal Tracker. For each declared IEP goal (a phoneme-position-therapy triple with a baseline accuracy and a target accuracy), compares current accuracy to baseline and target. A goal is "stalled" when current accuracy is no more than five percentage points above baseline, "near target" when the remaining gap is ten percentage points or less. Sessions-to-goal is estimated by dividing the remaining gap by the regression slope, capped at the configured maximum-sessions horizon.

Problem Detector. Flags stagnant phonemes (at least four sessions, slope below 2% per session, current accuracy below 70%), regressions (peak-minus-recent decline of 15% with current accuracy below 70%), persistent errors (at least three sessions with a phoneme present, persistence rate $\geq 60\%$), and dominant substitution patterns (at least five occurrences, one substitute accounting for at least 50% of errors; for example, 60% of /s/ errors going to /t/ is named as fronting).

Phoneme Analyzer, Position Analyzer, Session Overview. Break performance down by individual target phoneme, by initial/medial/final position, and by session, respectively. These rollups feed the Family/Parent report view planned as future work.

5.3 Sample Client Profiles for Reporting Coverage

To exercise every code path in the reporting engine, three synthetic client profiles are seeded into the database. The first profile represents a five-year-old with steady improvement over forty sessions and roughly 550 trials, exercising the improving-trend and goal-achievement paths. The second represents a six-year-old showing stagnation on /s/ in initial position and a regression on /k/, exercising the Problem Detector. The third represents a seven-year-old approaching graduation, with one goal already achieved and another near target, exercising the Goal Tracker's near-target and maintenance paths. The three profiles together form the demonstration dataset and serve as a regression test for the reporting engine across releases. Figure 5.3 shows a rendered sample clinical report.

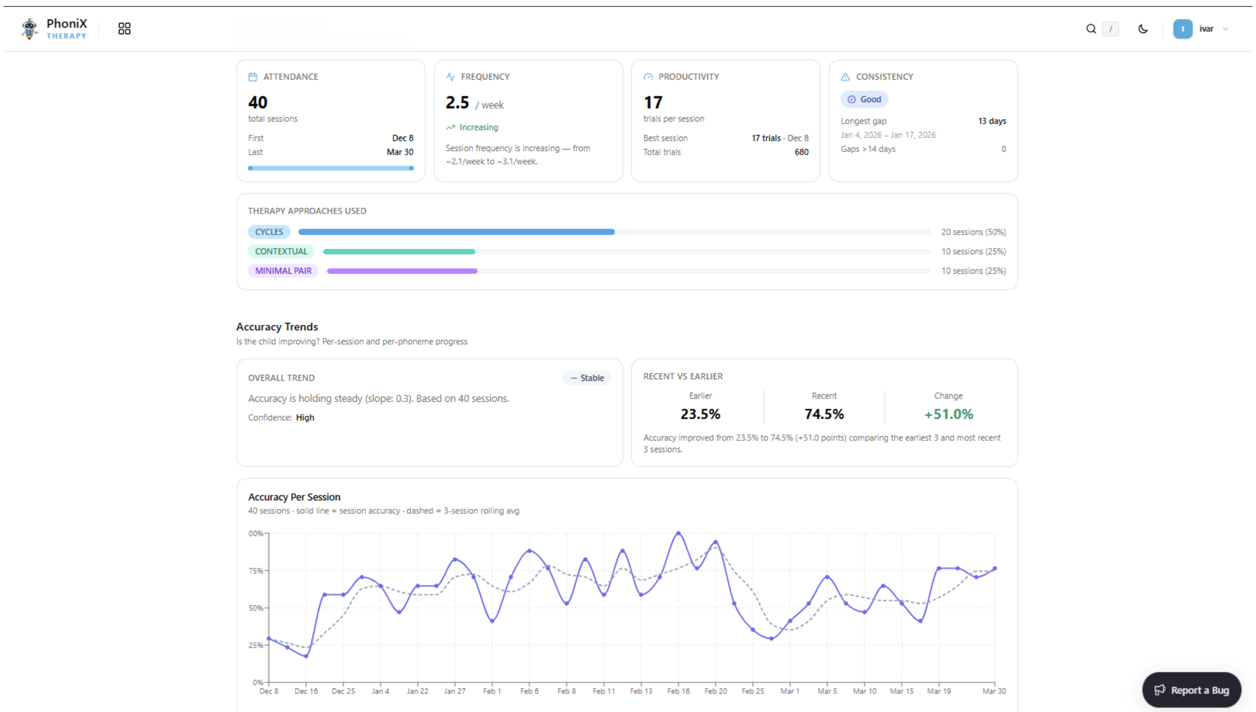

Figure 5.3. A rendered sample clinical report. Trend line, per-word accuracy bars, and cue distribution.

Chapter 6

Limitations and Future Work

This chapter sketches the directions in which Phonix can grow next. Each section names a concrete enhancement that builds on the current system rather than rebuilding it.

6.1 Parent-Facing Report View

The reporting engine already produces a structured payload from the seven analyzers. A natural extension is a parent-facing renderer that consumes the same payload and presents it in plain language, free of clinical jargon, with a focus on session-over-session progress and on what a parent can do at home. The infrastructure is in place: the analyzers are layered above a renderer, so the parent view is a presentation-layer addition rather than a data-layer change.

6.2 Improved Trial Data Collection Methodologies

The current Trial Collector captures every meaningful field a clinician taps during a session, but the tap surface itself can grow. Future work expands the collection methodology with one-handed scoring layouts for SLPs holding a card in the other hand, voice-driven trial entry so the clinician can score without breaking eye contact with the child, automatic timing capture from the moment a target is shown to the moment a response is logged, and structured fields for prosody, intelligibility, and effort that today live only in free-text notes. Richer trials feed richer reports without altering the analyzer interface.

6.3 Internationalization and Multilingual Coverage

Phonix currently serves English. The corpus, the articulation features, and the therapy algorithms have analogues in other languages. A natural next stage is multilingual coverage, beginning with Spanish given its prevalence in U.S. school-district caseloads, and extending the WordNet and Wiktionary integrations to use cross-lingual sources. Internationalizing the user interface itself is a straightforward translation pass once the underlying data layer supports a target language.

6.4 Accessibility for Broader Adoption

Full WCAG-compliant accessibility is the prerequisite for adoption beyond a pilot deployment. Future work extends the existing keyboard navigation and color contrast with ARIA landmarks across every therapy module, screen-reader-friendly trial scoring, and high-contrast and reduced-motion modes. This positions Phonix for use by SLPs with disabilities and within school districts that require accessibility certification.

6.5 An End-to-End Intelligent AI Pipeline

A longer-horizon direction is to thread intelligence through the entire clinical workflow rather than leaving each stage as an isolated tool. The pipeline begins at client check-in, where a reasoning model interprets intake notes, parent reports, and prior session summaries to propose a diagnosis and the most appropriate evidence-based therapy approaches for the child. Generative models then assemble personalized practice sessions tailored to the proposed diagnosis, the child's age, and stored progress, drawing on the same phonolinguistic corpus the current therapy modules use. After each session, generative models produce both the clinician report and the parent-facing summary directly from the trial data and from the structured notes the SLP leaves behind. The result is an assistive layer that lets the SLP supervise a coherent, evidence-grounded plan instead of stitching one together by hand for every child.

6.6 Continued SLP Exposure and Validation

The algorithmic backbones underlying the therapy modules, in particular the nine-factor shaping phoneme scorer, would benefit from broader SLP exposure and validation across multiple clinical contexts. A planned SLP consultation pack will collect structured expert feedback on the candidate rankings, the recommended word sets, and the report summaries, feeding the responses back into the threshold configuration and the scoring weights. Each round of exposure tightens the alignment between Phonix's recommendations and the judgment of practicing clinicians, and over time strengthens the empirical basis on which the system rests.

Chapter 7

Conclusion

This work presents Phonix, a phonolinguistic clinical platform that operationalizes a 289,000-word annotated lexicon into five evidence-based speech-language interventions and a reporting engine for in-session trial data. The main therapy platform serves Speech-Language Pathologists in production at `phonix-therapy.xlabub.com`, with a public Tools site at `phonix.xlabub.com` hosting the discovery utilities used during clinical preparation. The contributions are a queryable phonolinguistic corpus with per-phoneme articulatory annotation, three independent imageability sources, age-of-acquisition data, and WordNet semantic ancestry; five algorithmically-distinct therapy implementations with novel computational backbones; a clean three-layer trial collection model; and a deployed multi-tenant reporting engine, built end-to-end as a complete system.

The system is designed for incremental extension. The enhancements discussed in Chapter 6, namely a parent-facing report renderer, improved trial data collection methodologies, broader internationalization and accessibility, an end-to-end AI pipeline from client check-in through diagnosis and practice generation to reports, and continued SLP exposure and validation, can be added in stages without rebuilding the platform.

What Phonix demonstrates more broadly is that a curated phonolinguistic dataset combined with a thin clinical-logic layer is sufficient to recover hours of SLP preparation time per child per week. The SLP's role shifts from data assembler to clinical decision-maker, and the platform takes care of the rest.

Bibliography

[1] Marc Brysbaert, Amy Beth Warriner, and Victor Kuperman. Concreteness ratings for 40 thousand generally known English word lemmas. *Behavior Research Methods*, 46(3):904–911, 2014.

[2] Carnegie Mellon University. The CMU pronouncing dictionary. `http://www.speech.cs.cmu.edu/cgi-bin/cmudict`. Accessed 2026.

[3] Raymond D. Kent. Contextual facilitation of correct sound production. *Language, Speech, and Hearing Services in Schools*, 13(2):66–76, 1982.

[4] Victor Kuperman, Hans Stadthagen-Gonzalez, and Marc Brysbaert. Age-of-acquisition ratings for 30,000 English words. *Behavior Research Methods*, 44(4):978–990, 2012.

[5] Patricia McCabe, Donna Thomas, Elizabeth Murray, Laurelie Crocco, and Catherine Madill. Rapid syllable transition treatment – ReST. Technical report, The University of Sydney, 2017.

[6] George A. Miller. WordNet: A lexical database for English. *Communications of the ACM*, 38 (11):39–41, 1995.

[7] MongoDB, Inc. MongoDB: The developer data platform. `https://www.mongodb.com`. Accessed 2026.

[8] Raul F. Prezas and Barbara W. Hodson. The cycles phonological remediation approach. In A. L. Williams, S. McLeod, and R. J. McCauley, editors, *Interventions for speech sound disorders in children*, pages 137–158. Paul H. Brookes, 2010.

[9] Sebastián Ramírez. FastAPI: Modern, fast (high-performance) web framework for building APIs with Python. `https://fastapi.tiangolo.com`. Accessed 2026.

[10] Supabase, Inc. Supabase: The open source Firebase alternative. `https://supabase.com`. Accessed 2026.

[11] Vercel. Next.js: The React framework for the web. `https://nextjs.org`. Accessed 2026.

[12] Frederick F. Weiner. Treatment of phonological disability using the method of meaningful

minimal contrast: Two case studies. *Journal of Speech and Hearing Disorders*, 46(1):97–103, 1981.

[13] A. Lynn Williams. Multiple oppositions: Theoretical foundations for an alternative contrastive intervention approach. *American Journal of Speech-Language Pathology*, 9(4):282–288, 2000.

[14] Tatu Ylonen. Kaikki.org: Machine-readable Wiktionary data. `https://kaikki.org`. Accessed 2025–2026.