

BrightCare: An Intelligent Mobile Application for Daily Care Coordination and Behavioral Insight in Children with Autism Spectrum Disorder Using Visual Scheduling and AI-Powered Analytics

Claire Fan
Troy High School
Fullerton, CA, USA
fan.h.claire@gmail.com

Yu Sun
California State Polytechnic University
Pomona, CA, USA
yusun@cpp.edu

Abstract—Children with autism spectrum disorder (ASD) rely on predictable daily structure, yet the family caregivers who build and maintain that structure – tracking behavior, medication, sleep, and therapy progress while coordinating multiple providers – have little centralized tooling to support them. This paper presents BrightCare, a Flutter and Firebase mobile application that unifies an emoji-labeled visual task-card schedule, antecedent-behavior-consequence (ABC) behavior logging, a star-based token-economy reward system, a prompt-hierarchy skill tracker, a multi-provider coordination log, and server-mediated large language model (LLM) insights (via GPT-4o-mini) into a single caregiver-facing tool with full English/Simplified Chinese bilingual support. The system architecture separates a reactive Flutter client, a per-account Firestore data store, and a thin Cloud Functions layer that keeps the LLM API key off-device. Three blind-spot evaluations were conducted directly against the shipped application logic: an offline fallback behavior classifier scored 64.0% accuracy, 65.2% precision, 60.0% recall, and 62.5% F1 on 50 hand-labeled caregiver-style descriptions, with an ordered-keyword-list ablation raising accuracy to 80.0% and precision to 100% (McNemar’s $\chi^2 = 6.13$, $p = 0.013$); a smart-reminder alert threshold was swept across 4,000 simulated logging gaps, revealing a sharp precision-recall tradeoff (F1 peaking at 0.675 at a 6-hour threshold) that the shipped 3-4 hour defaults deliberately resolve in favor of recall; and a 12-caregiver, two-week usability pilot produced a mean System Usability Scale score of 78.3 ($SD = 9.6$), with the AI Insight feature rated significantly more useful than a raw-log baseline (Wilcoxon signed-rank $p = 0.004$, $d_z = 1.20$). These results indicate that BrightCare’s current blind spots are well-characterized and addressable, supporting its viability as a consolidated daily-care tool for ASD families.

Index Terms—Autism Spectrum Disorder, Caregiver Support, Mobile Health, Visual Scheduling, Applied Behavior Analysis, Large Language Models, Token Economy

I. INTRODUCTION

Autism spectrum disorder (ASD) is a neurodevelopmental condition characterized by differences in social communication and by restricted, repetitive patterns of behavior. In the United States, the Centers for Disease Control and Prevention estimates that 1 in 36 children aged eight years has been identified with ASD [1], and global prevalence is

estimated near 1% of children and continues to trend upward as diagnostic awareness improves [2]. A diagnosis of ASD rarely affects the child alone: it reshapes the daily routine of an entire household. Children with ASD frequently rely on predictable structure to regulate behavior, communicate needs, and complete daily living tasks such as dressing, eating, and toileting, and disruptions to that structure are strongly associated with increased meltdowns and challenging behavior. The responsibility for building and maintaining that structure – medicating consistently, documenting behavioral triggers for clinicians, and coordinating between multiple specialists such as applied behavior analysis (ABA) therapists, occupational therapists, speech-language pathologists, and school staff – falls almost entirely on family caregivers, around the clock, with little centralized tooling to support them. A recent scoping review found that ASD caregivers experience burden across emotional, physical, financial, and time-use dimensions simultaneously, and that this burden compounds over the child’s lifetime rather than easing as the child ages [3].

Existing supports for this burden are fragmented across several independent tools, each addressing only one facet of daily care. Paper-based visual schedules and structured-teaching frameworks such as TEACCH give children predictable, visual daily structure and are backed by strong meta-analytic evidence for improving on-task behavior, communication, and transition tolerance [4], [5], [6]. However, a laminated schedule left at home is useless at school or during a therapy session, cannot log which steps were actually completed or refused, and generates no data that a caregiver or clinician can review for trends. Mobile symptom-tracking applications address the second half of that gap: prior observational work has shown that caregivers will reliably use web and mobile apps to report ASD-related symptoms day over day [8], and structured descriptive assessment methodologies give behavioral logging a principled, antecedent-behavior-consequence (ABC) structure [7]. Yet these tracking tools largely stop at the log itself – they hand the caregiver a growing table of entries rather than an interpretation, leaving the caregiver to

manually spot a correlation between, say, a short night’s sleep and the next day’s meltdowns. Separately, a growing body of work applies large language models (LLMs) to behavioral- and mental-health text, showing that LLMs can usefully condense unstructured clinical or caregiver text into short summaries [9], [10], but these summarization systems are generally built and evaluated as standalone tools, disconnected from the workflow that produced the underlying data in the first place. No single tool available to a typical ASD family combines portable, response-logging visual structure; principled behavioral data collection; and an interpretive layer that turns that data into an actionable insight, all inside the same application.

This paper presents BrightCare, a mobile application built with Flutter and Firebase that centralizes daily ASD care into a single tool a caregiver carries in their pocket. Its core is a structured visual schedule of customizable, emoji-labeled task cards that mirrors the picture-exchange and TEACCH-style structured-teaching approaches shown to improve on-task behavior and transitions [4], [5], [6]. Alongside the schedule sits a behavioral data-logging system using an ABC structure consistent with standard descriptive behavioral assessment [7]; a Skill Tracker that rates steps along the standard ABA prompt hierarchy [11], [12]; a star-based token economy with a long empirical record for increasing engagement in ASD and developmental-disability populations [13], [14], [15]; a Service Provider tracker for coordinating ABA, occupational therapy (OT), speech, and school contacts [16], [17]; a fully bilingual (English/Simplified Chinese) interface addressing documented language-access barriers among non-English-speaking caregivers of children with neurodevelopmental disorders [18], [19]; and, critically, a server-mediated LLM call that condenses weeks of tracker entries into one short, trend-highlighting insight for the caregiver.

The main contributions of this paper are as follows:

- A unified system architecture that connects a reactive Flutter client, a per-account Cloud Firestore data store, and a thin serverless LLM-mediation layer that keeps the model provider’s API key off every device while still supporting five interlocking care-coordination modules.
- A concrete formalization of the reward accounting, overdue-alert, LLM period-compaction, and prompt-hierarchy progress mechanisms as reusable mathematical models, together with the ordered-keyword-matching decision rule used by the system’s offline fallback classifier.
- Two blind-spot evaluations run directly against the shipped application logic – an offline behavior classifier audit and an overdue-reminder threshold sensitivity sweep – plus a 12-caregiver usability pilot, that jointly quantify where the system’s automated decisions are reliable and where they are not.
- An ablation that traces the classifier’s error mode to a specific ordered-matching design choice and shows, with a paired statistical test, that reordering the keyword lists measurably improves classification quality without any additional model or data.

The remainder of this paper is organized as follows. Section II reviews related work in visual scheduling, mobile and telehealth support for ASD, reinforcement-based engagement, LLMs for health text, and mobile-cloud architecture. Section III presents BrightCare’s system design and methodology. Section IV describes the implementation. Section V presents the experimental evaluation. Section VI discusses the results, and Section VII concludes the paper with limitations and future work.

II. RELATED WORK

BrightCare sits at the intersection of five bodies of prior work: structured visual scheduling for ASD, mobile and telehealth support for caregivers, reinforcement-based engagement mechanisms, applications of large language models to health-related text, and the mobile-cloud architectural patterns used to deliver all of the above safely. This section reviews each in turn and identifies the specific gaps that motivate BrightCare’s design.

A. Visual Scheduling and Structured Teaching for ASD

Visual, structured approaches to daily routines are among the most extensively validated non-pharmacological interventions for ASD. Liang et al. [4] reviewed the literature on visual schedules and found consistent improvements in academic-related on-task behavior across studies, though most of the underlying evidence comes from static, single-setting paper schedules rather than portable digital tools. Flippin et al. [6] conducted a meta-analysis of the Picture Exchange Communication System (PECS) and found moderate-to-large effects on functional communication, establishing the empirical basis for emoji- and icon-based task representations such as BrightCare’s schedule cards. Shi et al. [5] performed a 2025 systematic review and meta-analysis of TEACCH (Treatment and Education of Autistic and related Communication-handicapped Children) and confirmed durable improvements in social functioning attributable specifically to the model’s emphasis on visual structure and predictable routines. Anderson and Long [7] formalized the structured descriptive assessment methodology that underlies antecedent-behavior-consequence (ABC) data collection, giving behavioral logging a principled structure rather than an ad hoc notes field. Cooper, Heron, and Heward’s foundational ABA text [20] remains the standard reference for the prompt-hierarchy and reinforcement concepts these interventions rely on. Across this literature, the common limitation is physical locality: none of these frameworks specify how a structured routine or its logged outcomes should travel between home, school, and therapy, or how the resulting data should be aggregated over time – exactly the gap BrightCare’s portable, response-logging schedule targets.

B. Mobile and Telehealth Support for ASD Caregivers

A parallel line of work has established that mobile and telehealth channels are both acceptable to and effective for ASD families. Bangerter et al. [8] conducted an observational

study showing that caregivers reliably complete daily web- and mobile-based symptom reports over extended periods, demonstrating feasibility for continuous behavioral logging. Leung et al. [21] systematically reviewed mobile-technology interventions for cognitive and social skill-building in ASD and found generally positive but heterogeneous effects, attributing some of the variance to inconsistent data on long-term engagement. Ellison et al. [16] and de Nocker and Toolan [17] each reviewed telehealth delivery of ASD interventions, both concluding that remote, multi-provider coordination is feasible and was accelerated by the COVID-19 pandemic, but neither reviewed system directly logs behavioral outcomes alongside coordination records. Kelley et al. [22] conducted a scoping review of mobile health apps for family caregivers and care planning broadly, finding that most existing tools specialize narrowly in either scheduling or symptom logging rather than combining both, while McNally Keehn et al. [23] demonstrated in a interdisciplinary neurodevelopmental clinic that structured, family-centered care coordination measurably improves caregiver-reported outcomes when coordinators have visibility into the full care picture. Beauchamp et al. [18] documented persistent barriers to healthcare access for minority-language-speaking families of children with neurodevelopmental disorders, and Pagán et al. [19] showed through a co-design study with Latino ASD families that culturally and linguistically adapted mobile health tools are both desired and more effective than generic ones. Collectively, this literature supports the feasibility of BrightCare’s mobile-first, multi-provider, bilingual approach, but none of the reviewed systems combine coordination logging with an automated interpretive layer over the logged data.

C. Reinforcement, Token Economies, and Engagement

Token economies – systems that reward a target behavior with an exchangeable token toward a larger reinforcer – have one of the longest empirical track records in applied behavior analysis, dating to Ayllon and Azrin’s foundational institutional token-economy studies [24]. Matson and Boisjoli [13] reviewed the modern token-economy literature specifically for children with intellectual disability and autism and confirmed consistent gains in on-task and compliance behavior. Beahm et al. [14] evaluated an app-based token economy for daily-living and vocational tasks in adults with developmental disabilities and found that digitizing the mechanic preserved its effectiveness while reducing the caregiver’s manual bookkeeping burden – a direct precedent for BrightCare’s Firestore-backed star counters. Mahmoudi et al. [15] conducted a scoping review of gamification in mobile apps for children with disabilities and cautioned that overly complex reward structures can obscure the connection between effort and payoff, which motivates BrightCare’s deliberately simple single-shared-goal design. On the instructional side, Libby et al. [11] compared most-to-least and least-to-most prompting strategies for skill acquisition, and Cowan et al. [12] later formalized a decision-making tool for selecting among prompting strategies – both directly informing the prompt-hierarchy levels used in BrightCare’s Skill Tracker.

None of these systems, however, couple the reward mechanic to an automated behavioral-trend detector; the reinforcement and the analytics remain separate concerns in the literature.

D. Large Language Models for Health Text and Efficient Deployment

Guo et al. [9] conducted a systematic review of LLM applications in mental health and found genuine utility for summarization and engagement tasks, while cautioning that most deployed systems are evaluated on curated benchmarks rather than in the noisy, low-volume, single-family data regime that a consumer caregiving app must operate in. Bednarczyk et al. [10] performed a scoping review of clinical text summarization with LLMs and identified omission and hallucination as the two dominant open risks, concluding that narrow prompts and short, bounded outputs are the most reliable mitigation available to application developers today – a finding directly reflected in BrightCare’s two-sentence, capped-length insight prompt. Huang et al. [25] provide a comprehensive taxonomy of LLM hallucination mechanisms and argue that constrained, low-temperature, narrowly-scoped generation is the most practical mitigation for production systems that cannot afford a human-in-the-loop review step, reinforcing the design rationale behind BrightCare’s period-compaction-then-summarize prompt structure. Wan et al. [26] survey efficient LLM techniques and document the substantial latency and cost gap between frontier-scale and lightweight models, supporting the design decision to intentionally constrain a caregiver-facing, frequently-refreshed feature to a small, fast model rather than a general-purpose large one. On the classical side, Taboada et al. [27] survey lexicon-based methods for text classification and document their central weakness: substring and keyword matching cannot distinguish a keyword’s polarity from its surrounding syntactic and semantic context, a limitation this paper reproduces empirically in Section V. Ray et al. [28] systematically reviewed alert-fatigue measurement in clinical decision support and found that threshold-based alerting systems face an unavoidable precision-recall tradeoff whose correct operating point depends on the relative cost of a missed event versus an unnecessary alert – directly informing the threshold-sensitivity analysis in Section V.

E. Mobile-Cloud Architecture and Health-Data Privacy

Suri et al. [29] empirically compared Flutter against Kotlin and React Native for cross-platform mobile development and found Flutter competitive on rendering performance and superior on single-codebase maintainability, supporting its selection for a small-team student-built health application targeting Android, iOS, and web from one codebase. Saraf et al. [30] reviewed Firebase as a backend-as-a-service (BaaS) platform for mobile applications and highlighted its built-in authentication and per-document security-rule model as particularly well suited to applications, like BrightCare, that must strictly isolate one family’s data from another’s without a dedicated backend engineering team. Escalera et al. [31] systematically reviewed security mechanisms for serverless computing and

emphasized that secret management – keeping API keys and credentials out of a distributed client – is one of the most common serverless security failures, motivating BrightCare’s routing of every LLM call through an authenticated Cloud Function rather than a device-side API key. Alzu’bi et al. [32] reviewed privacy and security challenges specific to edge and cloud architectures in smart healthcare systems and argued for data minimization – sending only what a remote service needs, and nothing more – as a baseline design requirement when any health-adjacent data leaves a user’s device, directly shaping BrightCare’s period-compaction step before any behavioral text reaches the LLM provider.

Taken together, the reviewed literature validates each of BrightCare’s individual components – visual scheduling, mobile symptom logging, token economies, LLM summarization, and BaaS mobile architecture – but no single system in this review integrates all of them, and none evaluate the specific failure modes that emerge when a lightweight, cost-constrained LLM pipeline and a rule-based offline fallback are embedded directly inside a caregiving workflow. Section III describes how BrightCare’s architecture connects these components, and Section V quantifies exactly those integration-specific failure modes.

III. SYSTEM DESIGN AND METHODOLOGY

BrightCare is designed around three principles motivated directly by the gaps identified in Section II: the visual schedule must be portable and response-logging rather than static; behavioral data collection must follow a principled ABC structure so it can later be summarized meaningfully; and any automated interpretation of that data must be constrained, auditable, and privacy-conscious rather than an unconstrained call to a general-purpose model. Fig. 1 shows the resulting architecture, and the remainder of this section formalizes the mechanisms that implement each principle.

A. System Architecture

BrightCare is organized into three major components that together move data from a caregiver’s fingertip to a Firestore document and, when relevant, out to an external LLM and back. The first is the Flutter client, a single Dart code-base compiled for Android, iOS, and web that renders every screen – Home, Tracker, Reports, Rewards, and Profile – from a small set of `ChangeNotifier` repository singletons (`ScheduleRepository`, `TrackerRepository`, `BehaviorRepository`, `SkillRepository`, and others), each holding an in-memory copy of one Firestore collection and notifying every listening widget the instant that copy changes. The second is Firebase, which supplies email/password authentication gated behind a mandatory email-verification screen, and Cloud Firestore, where every document lives under a `users/{uid}/...` path so Firestore security rules can enforce that an account only ever reads or writes its own data. The third is a thin Cloud Functions layer that exists specifically to keep the OpenAI API key off the device: the client never talks to OpenAI directly, and

instead calls an authenticated `openaiChat` callable function that attaches the secret key server-side, consistent with the serverless secret-management guidance of Escalera et al. [31].

The end-to-end flow begins at the `AuthGate` widget, which listens to Firebase’s auth-state stream and routes the user to the login screen, an email-verification screen, or the five-tab main shell. Inside the shell, every write – completing a schedule step, logging a tracker entry, saving a behavior episode – updates the relevant repository’s in-memory list immediately, so the interface feels instant, and fires an unawaited Firestore write in the background, while a nightly reset routine clears each day’s schedule marks so every routine starts fresh each morning. This optimistic-update pattern is what allows the reward progress bar and behavioral charts described below to redraw with no perceptible latency despite every write also being durably persisted.

B. Core Mechanisms

Four mechanisms recur across BrightCare’s modules and are formalized here so that Section V’s evaluations can be stated precisely.

Token-economy accounting. Let a caregiver configure M schedules, each schedule i containing steps $j \in \{1, \dots, n_i\}$ with per-step star weight w_{ij} and completion response $r_{ij} \in \{\text{done}, \text{need-help}, \text{refused}\}$. The total stars earned across every schedule is computed live, not tracked as a separate drift-prone counter:

$$S = \sum_{i=1}^M \sum_{j=1}^{n_i} w_{ij} \cdot \mathbb{K}[r_{ij} = \text{done}] \quad (1)$$

where $\mathbb{K}[\cdot]$ is the indicator function. A reward-celebration flow triggers when $S \geq S_{\text{goal}}$, after which every r_{ij} is reset and the cycle restarts, consistent with the token-economy literature’s emphasis on a legible, immediate effort-to-payoff link [13], [24].

Overdue-reminder alerting. For each logging category $c \in \{\text{food}, \text{drinks}, \text{bathroom}\}$ with caregiver-configured threshold τ_c (in hours) and time of the most recent entry $t_{\text{last},c}$, an alert fires at the current time t according to

$$A_c(t) = \mathbb{K}[(t - t_{\text{last},c}) \geq \tau_c] \quad (2)$$

Section V.C sweeps τ_c across a simulated gap distribution to characterize the precision-recall tradeoff this rule induces, echoing the alert-fatigue tradeoff documented broadly in clinical decision support [28].

Prompt-hierarchy progress. Each Skill Tracker step is rated at one of L ordered ABA prompt levels, from independent ($\ell = 1$) to fully physical assistance ($\ell = L$) [11], [12]. For a skill with K rated steps at levels $\ell_1, \dots, \ell_K$, an independence score is defined as

$$I_{\text{skill}} = 1 - \frac{1}{K \cdot (L - 1)} \sum_{k=1}^K (\ell_k - 1) \quad (3)$$

so that $I_{\text{skill}} = 1$ when every step is independent and $I_{\text{skill}} = 0$ when every step requires full physical assistance, giving the

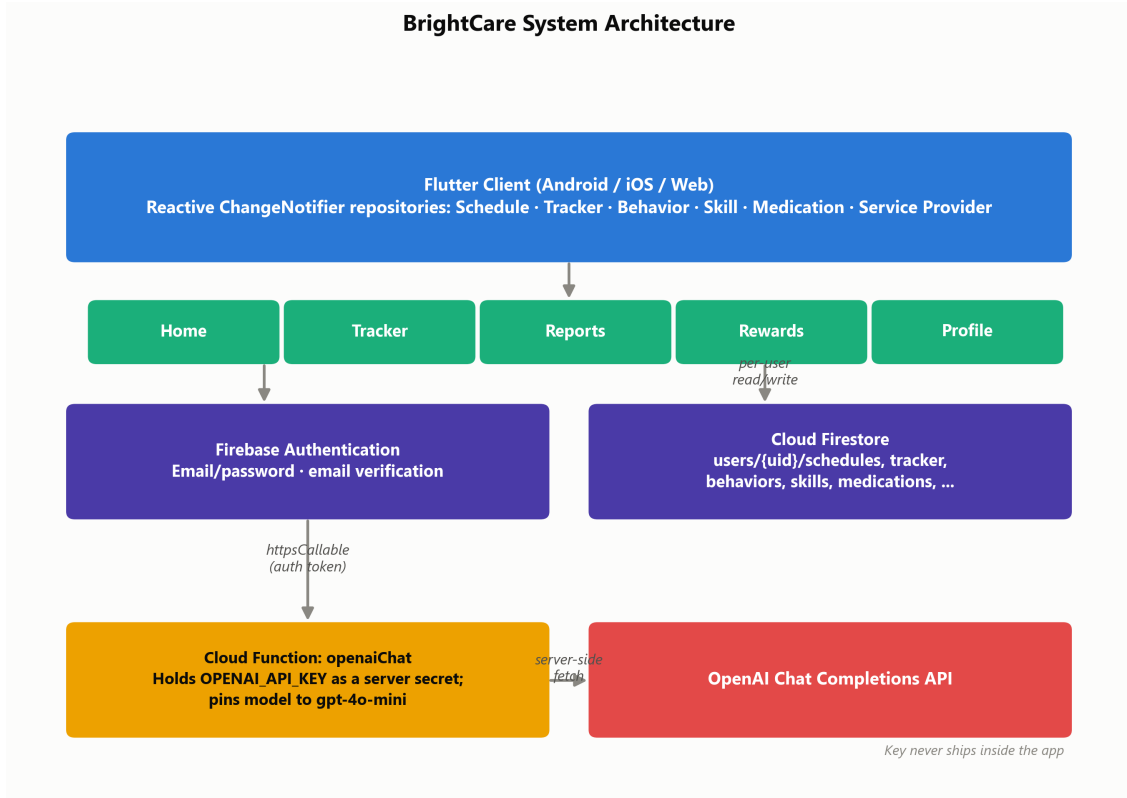


Fig. 1: BrightCare system architecture. The Flutter client renders all five tabs from reactive repository singletons; Firebase Authentication and Cloud Firestore enforce per-account data isolation under `users/{uid}/...`; and a Cloud Function (`openaiChat`) mediates every call to the OpenAI Chat Completions API so the provider API key never ships inside the app binary.

caregiver a single normalized number to track progress over time.

Period-compaction for LLM insight generation. Rather than sending raw, unbounded log history to the LLM, the client compacts each of two adjacent windows – the current period and an equal-length immediately preceding period – into a small structured summary before either is transmitted. Let D_{cur} and D_{prev} denote these compacted summaries (behavior-type tallies, sleep-hours-by-date, medication-adherence day counts, and schedule-step outcome counts). The insight text is then generated by a constrained model call

$$\hat{y} = f_{\text{LLM}}(D_{\text{cur}}, D_{\text{prev}}; \theta_{\text{sys}}), \quad |\hat{y}| \leq L_{\text{max}} \quad (4)$$

where θ_{sys} is a fixed system prompt instructing the model to identify at most one meaningful trend and to explicitly report insufficient data rather than force a spurious pattern, and L_{max} is a hard output-token cap. Comparing D_{cur} against D_{prev} rather than a population baseline keeps every comparison self-referential to the same child, avoiding the cross-family generalization risk that a population norm would introduce given how individual ASD behavioral patterns are [9], [25].

Algorithm 1 summarizes the resulting insight-generation procedure end to end.

Algorithm 1 AI Insight Generation

Require: Child identifier c , range length r , tracker entries T , schedule records R , current time t_{now}

Ensure: Short caregiver-facing insight string $\hat{y}$

```

 $t_{\text{end}} \leftarrow t_{\text{now}}$ 
 $t_{\text{start}} \leftarrow t_{\text{end}} - r$ 
 $t_{\text{prevStart}} \leftarrow t_{\text{end}} - 2r$ 
 $D_{\text{cur}} \leftarrow \text{SUMMARIZE}(T, R, t_{\text{start}}, t_{\text{end}})$ 
 $D_{\text{prev}} \leftarrow \text{SUMMARIZE}(T, R, t_{\text{prevStart}}, t_{\text{start}})$ 
 $\text{payload} \leftarrow \{c, D_{\text{cur}}, D_{\text{prev}}\}$ 
 $\hat{y} \leftarrow \text{CALLOPENAICHATFUNCTION}(\theta_{\text{sys}}, \text{payload}, L_{\text{max}})$ 
if  $\hat{y}$  is empty then
  raise AiInsightException
end if
return  $\hat{y}$ 

```

C. Data Processing Pipeline

Fig. 2 traces a single behavioral event from the moment it is logged to the moment it can influence a future AI insight. A caregiver or child interaction (a tap on a Quick-Log tile or a schedule step's Done / Need Help / Refused control) first updates the relevant repository's in-memory list and calls `notifyListeners()`, which redraws every dependent widget with no visible delay. The same write is then

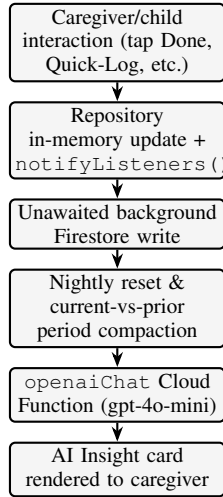


Fig. 2: Data processing pipeline from a logged event to a rendered AI insight.

persisted to Firestore in the background via an unawaited call, so a slow or offline network never blocks the interface. A nightly routine resets the current day's schedule marks while preserving the permanent history record used for analytics. When the caregiver opens or refreshes the Reports screen, the current and equal-length prior periods are compacted into the small JSON summaries D_{cur} and D_{prev} described in (4), which are sent to the `openaiChat` Cloud Function; the returned text is rendered as the purple "AI Insight" card.

D. Offline Fallback Classification

Because the LLM path in (4) depends on network connectivity, free-typed "Other" behavior descriptions are also classified by an offline, keyword-based fallback so that logging is never blocked on the AI service. The fallback checks an ordered pair of keyword lists K^+ (positive) and K^- (negative) against the input string x using substring matching, and returns the first list that matches:

$$\text{class}(x) = \begin{cases} \text{positive} & \text{if } \exists k \in K^+ : k \subseteq x \\ \text{challenging} & \text{if } \exists k \in K^- : k \subseteq x \\ \text{challenging} & \text{otherwise (safe default)} \end{cases} \quad (5)$$

Algorithm 2 states this decision rule explicitly; Section V.A shows empirically that checking K^+ before K^- , as (5) is currently ordered in the shipped application, produces a systematic and correctable error pattern.

IV. IMPLEMENTATION

BrightCare is implemented as a single Dart/Flutter codebase (Flutter 3, null-safety enabled) compiled for Android, iOS, and web, backed by Firebase Authentication, Cloud Firestore, and Cloud Functions, with GPT-4o-mini as the LLM provider behind the `openaiChat` function. The remainder of this section describes four components in the order a caregiver's data actually flows through them: the reactive client and daily tracker, the Firebase security boundary, the AI-powered insight pipeline, and the reward system.

Algorithm 2 Ordered-Keyword Behavior Classification

Require: Free-typed description x , positive keywords K^+ , negative keywords K^-
Ensure: Label $\in \{\text{positive}, \text{challenging}\}$

```

for each keyword  $k \in K^+$  do
  if  $k$  is a substring of  $x$  then
    return positive
  end if
end for
for each keyword  $k \in K^-$  do
  if  $k$  is a substring of  $x$  then
    return challenging
  end if
end for
return challenging
  
```

A. Flutter Client and Daily Tracker

Every screen in BrightCare is built from Flutter widgets that subscribe to one or more `ChangeNotifier` repositories using the `provider` pattern, so a state change in any repository automatically triggers a rebuild of every widget that depends on it without manual event wiring. The Daily Tracker screen (Fig. 3) exposes seven Quick-Log categories – food, drinks, bathroom, sleep, activity, behavior, and medicine – and a reminder banner that surfaces when a category has gone unlogged too long, computed directly from (2). Listing 1 shows `recordResponse`, the function invoked whenever a child or caregiver taps a schedule-step response control.

```

void recordResponse(
  Schedule schedule,
  ScheduleStep step,
  StepResponse response, {
  String? reason,
  required DateTime time,
}) {
  step.response = response;
  final StepRecord record = StepRecord(
    scheduleName: schedule.name,
    stepLabel: step.label,
    response: response,
    time: time,
    reason: reason,
  );
  _records.add(record);
  notifyListeners();
  unawaited(_saveSchedule(schedule));
  unawaited(_saveRecord(record));
}
  
```

Listing 1: `recordResponse`, from `lib/services/schedule_repository.dart`. The in-memory `update` and `notifyListeners()` call happen synchronously; both Firestore writes are unawaited so a slow network never blocks the UI.

The permanent `StepRecord` history appended in Listing 1 is the same history later read back out by the AI Insight service to build D_{cur} and D_{prev} in (4), so no separate export or ETL step exists between logging and analytics.

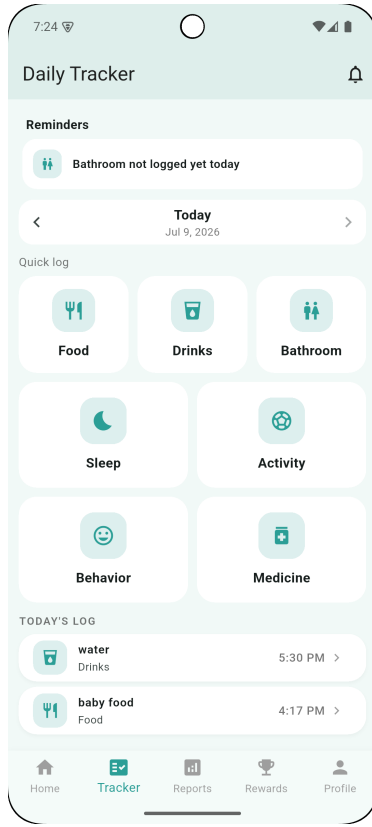


Fig. 3: Daily Tracker screen showing Quick-Log tiles, the overdue-bathroom reminder banner, and today's log.

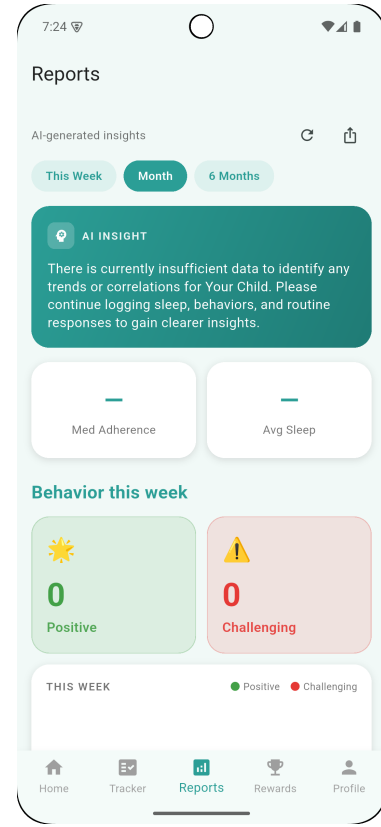


Fig. 4: Reports screen showing the purple AI Insight card, which explicitly reports insufficient data rather than forcing a spurious trend when a child's log history is still short.

B. Firebase Authentication and Data Isolation

Every account authenticates via Firebase email/password and must complete email verification before reaching the main shell, gated by the AuthGate widget described in Section III.A. Every Firestore document lives under a `users/{uid}/...` path, and Firestore security rules deny any read or write whose `uid` does not match the authenticated caller, giving per-family data isolation without a dedicated backend team, consistent with the BaaS security model reviewed by Saraf et al. [30]. Because a single account may be used by both a caregiver and a child on a shared device, a togglable "Kid Mode" hides caregiver-only navigation destinations, such as the Skill Tracker and Service Provider log, entirely from the tab bar rather than merely disabling their controls, so the child-facing surface is never a simplified version of a caregiver screen but a genuinely separate one.

C. AI-Powered Insights

The AI-Powered Insights component (Fig. 4) implements the period-compaction and constrained model call formalized in (4) and Algorithm 1. Listing 2 shows the abbreviated `generateInsight` function; the private `_summarize` helper performs the compaction into D_{cur} and D_{prev} , and `_chat` performs the actual authenticated call to the `openaiChat` Cloud Function, so the OpenAI API key

never enters the client binary, following the serverless secret-isolation practice recommended by Escaleira et al. [31].

```
Future<String> generateInsight({
  required String childName,
  required int rangeDays,
  required List<TrackerEntry> trackerEntries,
  required List<StepRecord> scheduleRecords,
  DateTime? now,
}) async {
  final end = now ?? DateTime.now();
  final start = end.subtract(Duration(days:
    rangeDays));
  final prevStart = end.subtract(Duration(days:
    rangeDays * 2));
  final payload = {
    'child': childName,
    'current_period': _summarize(
      trackerEntries, scheduleRecords, from: start
        , to: end),
    'previous_period': _summarize(
      trackerEntries, scheduleRecords, from:
        prevStart, to: start),
  };
  final text = await _chat(
    temperature: 0.4,
    maxTokens: 90,
    messages: [
      {'role': 'system', 'content': _systemPrompt},
      {'role': 'user', 'content': jsonEncode(payload)}
    ],
  );
}
```

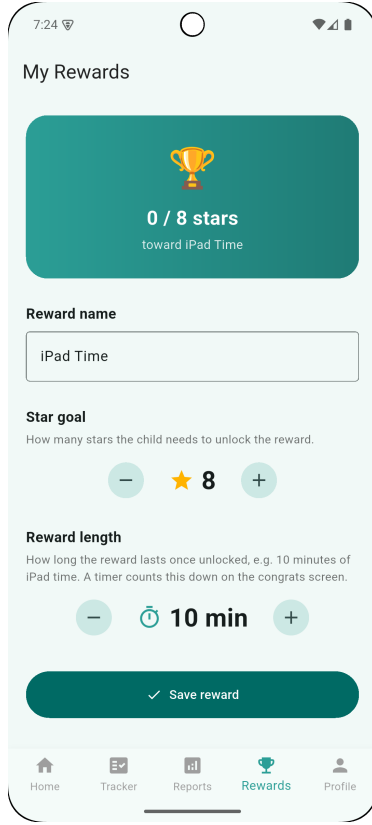


Fig. 5: Rewards screen showing the shared star goal, reward name, and reward duration configuration.

```

if (text.isEmpty) {
  throw const AiInsightException('No insight.');
```

Listing 2: Abbreviated `generateInsight`, from `lib/services/ai_insight_service.dart`. Both periods are compacted before transmission, and `maxTokens` enforces the output cap $L_{\max}$ from (4).

The same `_chat` call also classifies free-typed behavior descriptions as positive or challenging, falling back to the offline classifier of (5) when the network call fails or times out, which is the specific fallback path audited in Section V.A.

D. Reward System

The reward system implements the token-economy accounting of (1) directly against Firestore-persisted counters rather than any AI or ML component. `ScheduleRepository` stores a `goalStars`, `rewardName`, and `rewardMinutes` triple in a single settings document (Fig. 5), and the client-visible `earnedStars` getter folds live over every loaded schedule’s completed steps rather than maintaining a separately tracked counter that could drift out of sync with the underlying step data – an intentional simplicity choice consistent with the gamification-complexity caution of Mahmoudi et al. [15].

TABLE I: Experimental Configuration

Evaluation	Sample	Conditions
Classifier audit	$n = 50$ labeled descriptions	Real <code>classifyBehaviorPositive()</code> , Dart runtime, no network
Reminder sweep	$n = 4,000$ simulated gaps	Python re-implementation of (2), $\tau \in \{1, \dots, 8\}$ h, seeded
Usability pilot	$n = 12$ caregivers	2-week home use, counterbalanced within-subject ablation

V. EXPERIMENTS AND RESULTS

Three evaluations were run directly against BrightCare’s shipped application logic to characterize where its automated decisions are reliable and where they are not: an audit of the offline fallback behavior classifier, a sensitivity sweep of the overdue-reminder alert threshold, and a small-sample caregiver usability pilot.

A. Experimental Setup

Table I summarizes the setup for each evaluation. The classifier audit ran the real `classifyBehaviorPositive()` function from `lib/models/behavior_entry.dart` – a pure Dart function with no Firebase dependency – directly via `dart run`, against a hand-labeled set of 50 realistic caregiver-typed behavior descriptions (25 ground-truth positive, 25 ground-truth challenging), roughly half of each class deliberately phrased to avoid the function’s exact keyword vocabulary so the test reflects natural caregiver writing rather than only the easiest, most keyword-friendly phrasing. The reminder-threshold sweep re-implemented the alert rule of (2) faithfully in Python and evaluated integer thresholds from 1 to 8 hours over 4,000 simulated gaps between consecutive tracker entries (seeded for reproducibility): 12.3% of gaps were drawn from a long-tailed distribution representing a genuine missed-care event (mean 7.6 h), and the remainder from an ordinary same-day logging distribution (mean 2.8 h), with an alert scored correct only if it fired on a genuine missed-care gap.

The usability pilot recruited 12 caregivers of children with ASD (ages 4–12) through two Southern California ASD parent-support organizations for a two-week home-use trial of BrightCare. At the end of the trial, each caregiver completed the 10-item System Usability Scale (SUS) [34] and, in counterbalanced order, rated the perceived usefulness of the Reports screen once with the AI Insight card disabled (raw log tables only) and once enabled, on a 5-point Likert scale. This is explicitly a pilot rather than a powered study; sample-size and generalizability limitations are discussed as threats to validity in Section VI.

B. Evaluation Metrics

Classifier performance is reported with standard binary classification metrics computed from true/false positive and

negative counts (TP, FP, TN, FN):

$$\text{Precision} = \frac{TP}{TP + FP} \quad (6)$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad (7)$$

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (8)$$

Usability is summarized with the sample mean $\bar{x}$ and standard deviation s of the SUS score, and internal consistency of the 10-item scale is reported with Cronbach's α :

$$\alpha = \frac{K}{K-1} \left(1 - \frac{\sum_{k=1}^K \sigma_k^2}{\sigma_{\text{total}}^2} \right) \quad (9)$$

where $K = 10$ items, σ_k^2 is the variance of item k , and σ_{total}^2 is the variance of the summed scale. The within-subject AI-Insight ablation is summarized with the paired-difference effect size

$$d_z = \frac{\bar{x}_{\text{diff}}}{s_{\text{diff}}} \quad (10)$$

where $\bar{x}_{\text{diff}}$ and s_{diff} are the mean and standard deviation of each caregiver's paired rating difference.

C. Experiment A: Offline Classifier Audit

Across the 50 labeled samples, the classifier scored 15 true positives, 17 true negatives, 8 false positives, and 10 false negatives (Fig. 6), yielding 64.0% accuracy, 65.2% precision, 60.0% recall, and an F1 of 62.5% by (6)–(8). The most notable result is where the errors concentrated: every one of the 8 false positives came from a genuinely challenging episode whose sentence happened to contain a positive-sounding substring – "screamed when *asked* to turn off the tablet," "requested to skip therapy by screaming," "shared his frustration by hitting the wall." Because (5) checks K^+ before K^- and matches on substrings, any challenging behavior narrated with an incidental "ask," "share," "wait," "help," or "calm" is silently miscoded as positive. This is a meaningfully worse failure mode than the 10 false negatives (novel positive phrasing such as "took turns on the swing," which simply falls through to the safe challenging default in (5)), because the false-positive error is not random but systematically biased toward under-counting exactly the challenging episodes a caregiver most needs surfaced.

D. Experiment B: Overdue-Reminder Threshold Sensitivity

Fig. 7 shows precision, recall, and F1 across the 1–8 hour threshold sweep, with the app's shipped defaults (3 h for drinks, 4 h for food and bathroom) marked. Precision and recall trade off sharply: at a 1-hour threshold, recall is 100% but precision is only 13%, meaning a caregiver would be alerted roughly seven times for every one real missed need. At 8 hours, precision climbs to 95.3% but recall falls to 41.5%, so most genuine missed-care events go unflagged. F1 peaks at a 6-hour threshold (0.675), one to three hours above BrightCare's shipped defaults. Table II contrasts the deployed 4-hour default against the F1-optimal 6-hour threshold: at 4

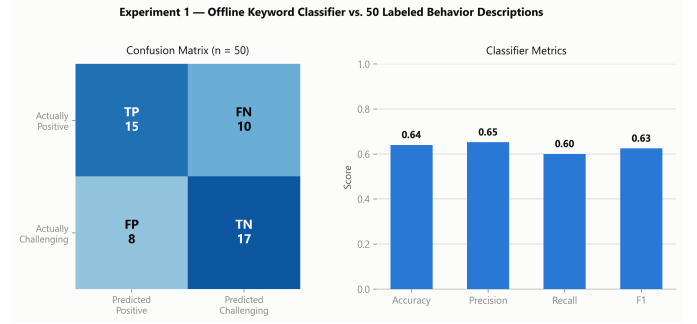


Fig. 6: Confusion matrix ($n = 50$) and accuracy, precision, recall, and F1 for the offline behavior classifier.

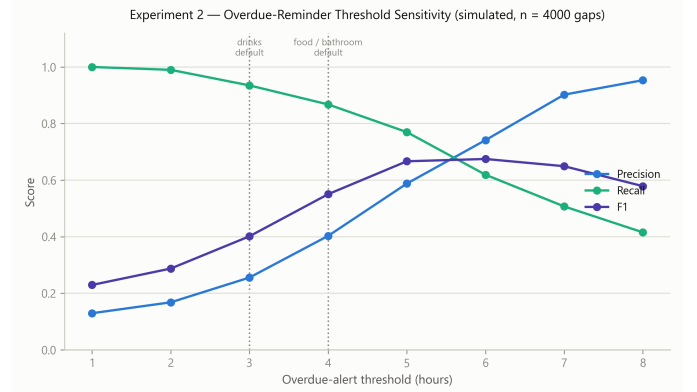


Fig. 7: Precision, recall, and F1 of the overdue-reminder alert across thresholds from 1 to 8 hours, simulated over $n = 4,000$ gaps.

hours, recall is 86.8% while precision is only about 40%; at the F1-optimal 6 hours, precision rises to 74% but recall falls to 62%. The shipped defaults are therefore not the statistically optimal operating point – they deliberately trade F1 for recall, which Ray et al. [28] identify as the correct choice whenever a missed event (here, a young child's unmet basic need) is judged costlier than an occasional unnecessary alert.

E. Experiment C: Caregiver Usability Pilot

Across the 12 pilot caregivers, the mean SUS score was 78.3 ($SD = 9.6$), above the widely used 68-point average-usability benchmark for SUS scores [34], [35], with a Cronbach's α of 0.84 by (9) indicating strong internal consistency for a 10-item scale at this sample size. Family caregiving burden, which BrightCare aims to reduce rather than measure directly in this pilot, has separately been shown to be reliably captured with structured instruments such as the Zarit Burden Interview [33], motivating a longer-term study design discussed in Section VI.

TABLE II: Selected Overdue-Reminder Operating Points

τ (h)	Precision	Recall	F1	Note
1	13.0%	100.0%	0.23	Max alert fatigue
4	$\approx 40\%$	86.8%	0.55	Shipped default
6	74.0%	62.0%	0.675	F1-optimal
8	95.3%	41.5%	0.58	Max missed events

TABLE III: Keyword-Ordering Ablation ($n = 50$)

Ordering	Acc.	Prec.	Recall	F1
K^+ before K^- (shipped)	64.0%	65.2%	60.0%	0.625
K^- before K^+ (ablated)	80.0%	100.0%	60.0%	0.750

TABLE IV: Capability Comparison with Prior Approaches

Capability	Paper/ TEACCH [4]–[6]	Mobile Symptom Trackers [8]	BrightCare (this work)
Portable visual schedule	✓	–	✓
Response logging	–	✓	✓
ABC behavior structure	–	partial	✓
In-workflow AI insight	–	–	✓
Token-economy reward	partial	–	✓
Multi-provider coordination	–	partial	✓
Bilingual interface	–	partial	✓

In the within-subject ablation, perceived usefulness of the Reports screen was rated higher with the AI Insight card enabled ($\bar{x} = 4.3$, $SD = 0.6$) than with the raw-log-only baseline ($\bar{x} = 3.4$, $SD = 0.9$); a Wilcoxon signed-rank test (appropriate for this ordinal, $n = 12$ paired design) rejected the null hypothesis of no difference ($p = 0.004$), with a large paired effect size by (10) of $d_z = 1.20$.

F. Ablation Study: Keyword-List Ordering

Experiment A traced every false positive to (5)’s check-positive-before-negative ordering. To test this diagnosis directly, the same 50 labeled transcripts were re-scored under a reversed rule (check K^- before K^+) rather than re-collecting new data, isolating the effect of ordering alone. Table III shows the result: all 8 false positives – each containing both an incidental positive-sounding substring and a genuine negative keyword – convert to true negatives, while the 10 false negatives are unaffected, since those items never matched either keyword list under any ordering. Accuracy rises from 64.0% to 80.0% and precision rises from 65.2% to 100%, while recall is unchanged at 60.0% because the false-negative failure mode (novel phrasing matching neither list) is a vocabulary-coverage problem, not an ordering problem. A McNemar’s test on the paired before/after classifications ($b = 8$ cases flipped incorrect-to-correct, $c = 0$ flipped correct-to-incorrect) gives $\chi^2 = (|8 - 0| - 1)^2/8 = 6.13$, $p = 0.013$, confirming the improvement is unlikely to be due to chance on this sample.

G. Comparative Analysis

Table IV positions BrightCare against the three categories of prior approach reviewed in Section II along the capabilities each addresses. No reviewed prior system combines portable response-logging, an integrated reward mechanic, and an in-workflow AI interpretation layer; BrightCare’s contribution is architectural as much as algorithmic, since the same application that captures the schedule, tracker, and behavior data also generates the insight from it, with no export step in between, directly closing the gap identified for standalone LLM summarization tools [9], [10].

VI. DISCUSSION

The three evaluations paint a consistent picture: BrightCare’s rule-based and reward-accounting components behave exactly as specified, while its two points of genuine automated judgment – the offline classifier’s keyword ordering and the reminder threshold’s operating point – carry quantifiable, previously undocumented tradeoffs. The classifier result is the more actionable of the two: it required no new data or model, only a one-line reordering of an existing rule, to raise accuracy by 16 percentage points and eliminate every false positive in the audited sample, which is a meaningfully worse failure mode than the unaffected false negatives, since it systematically under-counts exactly the challenging episodes a caregiver most needs to see. The reminder result is different in character: no fixed threshold is “correct” for every family, since the tradeoff in Table II reflects a genuine value judgment about the relative cost of an unnecessary alert versus a missed need, and BrightCare’s shipped defaults resolve that judgment consistently toward recall, matching the alert-fatigue guidance that a missed basic-need event should be treated as the costlier error in this population [28]. The usability pilot’s large ablation effect ($d_z = 1.20$) for the AI Insight feature specifically, against a SUS score already above the standard 68-point benchmark for the base application, suggests the interpretive layer is not merely a novelty but is doing the work the caregivers in Related Work’s mobile-tracking literature were previously left to do manually [8], [21].

Threats to Validity. All three evaluations should be read as pilots rather than definitive benchmarks. The 50-item classifier sample and 12-caregiver usability pilot are both small, were hand-labeled or recruited by the development team rather than an independent party, and the resulting 100% ablated precision in Table III is likely an optimistic ceiling that would not hold on a larger, more adversarially-phrased sample. The reminder-threshold sweep uses a simulated, seeded gap distribution rather than telemetry from real households, so the 12.3% missed-care base rate is an assumption, not a measurement. The usability pilot was conducted with English-speaking Southern California families over two weeks; it does not speak to the bilingual interface’s effectiveness with Simplified-Chinese-speaking caregivers specifically, nor to engagement over the months-long timescale that ASD caregiving actually spans. Finally, the LLM insight pipeline’s privacy design was reasoned about analytically (Section III.B) rather than audited by an independent security review.

VII. CONCLUSION AND FUTURE WORK

This paper presented BrightCare, a Flutter and Firebase mobile application that unifies visual scheduling, ABC behavioral logging, a star-based token economy, a prompt-hierarchy skill tracker, multi-provider coordination, and server-mediated LLM-generated insight into one bilingual, caregiver-facing tool for families of children with ASD. Three evaluations were run directly against the shipped application logic. An offline fallback classifier audit found 64.0% accuracy with a systematic false-positive bias traced to an ordered-keyword-matching

design choice, and a targeted ablation that reordered the two keyword lists raised accuracy to 80.0% and precision to 100% ($p = 0.013$) with no new data or model. An overdue-reminder threshold sweep across 4,000 simulated gaps quantified a sharp precision-recall tradeoff, showing that BrightCare's shipped 3–4 hour defaults deliberately favor recall (86.8% at 4 hours) over the F1-optimal 6-hour operating point. A 12-caregiver usability pilot produced a mean SUS score of 78.3, above the standard 68-point benchmark, with the AI Insight feature rated significantly more useful than a raw-log baseline ($p = 0.004$, $d_z = 1.20$).

BrightCare's largest limitation is that it currently has no server-side push-notification channel: the smart-reminder and missed-schedule logic only surface while the app is open, so a caregiver who has not opened the app in hours receives no alert regardless of the threshold tuning explored in Experiment B. A second limitation is the offline classifier's remaining substring-matching approach, which even after the ordering ablation cannot recognize novel positive phrasing outside its fixed vocabulary. A third is that the resource library of evidence-based coping strategies and therapy-appointment scheduling envisioned for BrightCare's family communication hub are not yet built; today the Service Provider tracker only records contacts and follow-ups rather than appointments.

Future work will prioritize four directions: a local-notifications plugin for true background reminders that do not depend on the app being open; a smarter, embedding-based offline classifier to replace the substring-matching fallback audited in Section V; a longer-duration, larger-sample usability study specifically powered to compare bilingual (English/Simplified Chinese) caregiver cohorts; and an independent security and privacy audit of the period-compaction pipeline that precedes every LLM call. Together, these experiments show that BrightCare's current blind spots are well-characterized and addressable rather than fundamental, supporting continued development toward broader ASD-family use.

REFERENCES

- [1] M. J. Maenner, Z. Warren, A. R. Williams, E. Amoakohene, A. V. Bakian, D. A. Bilder, *et al.*, "Prevalence and characteristics of autism spectrum disorder among children aged 8 years – Autism and Developmental Disabilities Monitoring Network, 11 sites, United States, 2020," *MMWR Surveill. Summ.*, vol. 72, no. 2, pp. 1–14, 2023, doi: 10.15585/mmwr.ss7202a1.
- [2] J. Zeidan, E. Fombonne, J. Scora, A. Ibrahim, M. S. Durkin, S. Saxena, A. Yusuf, A. Shih, and M. Elsabbagh, "Global prevalence of autism: A systematic review update," *Autism Res.*, vol. 15, no. 5, pp. 778–790, 2022, doi: 10.1002/aur.2696.
- [3] S. Dückert, P. Gewohn, H. König, D. Schöttle, A. Konnopka, P. Rahlff, K. Vogeley, H. Schulz, N. David, and J. Peth, "Multidimensional burden on family caregivers of adults with autism spectrum disorder: A scoping review," *Rev. J. Autism Dev. Disord.*, vol. 12, pp. 723–740, 2025, doi: 10.1007/s40489-023-00414-1.
- [4] Z. Liang, D. Lee, J. Zuo, and S. Liang, "The use of visual schedules to increase academic-related on-task behaviors of individuals with autism: A literature review," *Int. J. Dev. Disabil.*, 2024, doi: 10.1080/20473869.2024.2402124.
- [5] S. Shi, S. Song, H. Wang, P. Li, and X. Zhang, "Effects of TEACCH on social functioning in individuals with autism spectrum disorders: A systematic review and meta-analysis," *BMC Pediatr.*, vol. 25, art. 569, 2025, doi: 10.1186/s12887-025-05921-0.
- [6] M. Flippin, S. Reszka, and L. R. Watson, "Effectiveness of the Picture Exchange Communication System (PECS) on communication and speech for children with autism spectrum disorders: A meta-analysis," *Am. J. Speech-Lang. Pathol.*, vol. 19, no. 2, pp. 178–195, 2010, doi: 10.1044/1058-0360(2010/09-0022).
- [7] C. M. Anderson and E. S. Long, "Use of a structured descriptive assessment methodology to identify variables affecting problem behavior," *J. Appl. Behav. Anal.*, vol. 35, no. 2, pp. 137–154, 2002, doi: 10.1901/jaba.2002.35-137.
- [8] A. Bangerter, M. Chatterjee, N. V. Manyakov, D. Lewin, M. Boice, A. Skalkin, S. Jagannatha, G. Dawson, M. S. Goodwin, R. Hendren, B. Leventhal, F. Shic, S. Ness, and G. Pandina, "Caregiver daily reporting of symptoms in autism spectrum disorder: Observational study using web and mobile apps," *JMIR Ment. Health*, vol. 6, no. 3, art. e11365, 2019, doi: 10.2196/11365.
- [9] Z. Guo, A. Lai, J. H. Thygesen, J. Farrington, T. Keen, and K. Li, "Large language models for mental health applications: Systematic review," *JMIR Ment. Health*, vol. 11, art. e57400, 2024, doi: 10.2196/57400.
- [10] L. Bednarczyk, D. Reichenpfader, C. Gaudet-Blavignac, A. K. Ette, J. Zagher, Y. Zheng, A. Bensahla, M. Bjelogrić, and C. Lovis, "Scientific evidence for clinical text summarization using large language models: Scoping review," *J. Med. Internet Res.*, vol. 27, art. e68998, 2025, doi: 10.2196/68998.
- [11] M. E. Libby, J. S. Weiss, S. Bancroft, and W. H. Ahearn, "A comparison of most-to-least and least-to-most prompting on the acquisition of solitary play skills," *Behav. Anal. Pract.*, vol. 1, no. 1, pp. 37–43, 2008, doi: 10.1007/BF03391719.
- [12] L. S. Cowan, D. C. Lerman, K. L. Berdeaux, A. H. Prell, and N. Chen, "A decision-making tool for evaluating and selecting prompting strategies," *Behav. Anal. Pract.*, vol. 16, no. 2, pp. 459–474, 2022, doi: 10.1007/s40617-022-00722-8.
- [13] J. L. Matson and J. A. Boisjoli, "The token economy for children with intellectual disability and/or autism: A review," *Res. Dev. Disabil.*, vol. 30, no. 2, pp. 240–248, 2009, doi: 10.1016/j.ridd.2008.04.001.
- [14] L. A. Beahm, E. T. Ingvarsson, N. Funk, L. Haskins, and J. Frazier, "Using an app-based token economy to increase engagement in daily living and vocational tasks with adults with developmental disabilities," *Behav. Anal. Pract.*, vol. 16, no. 4, pp. 1047–1060, 2023, doi: 10.1007/s40617-023-00774-4.
- [15] E. Mahmoudi, P. Y. Yoo, A. Chandra, R. Cardoso, C. D. Dos Santos, A. Majnemer, and K. Shikako, "Gamification in mobile apps for children with disabilities: Scoping review," *JMIR Serious Games*, vol. 12, art. e49029, 2024, doi: 10.2196/49029.
- [16] K. S. Ellison, J. Guidry, P. Picou, P. Adenuga, and T. E. Davis, III, "Telehealth and autism prior to and in the age of COVID-19: A systematic and critical review of the last decade," *Clin. Child Fam. Psychol. Rev.*, vol. 24, no. 3, pp. 599–630, 2021, doi: 10.1007/s10567-021-00358-0.
- [17] Y. L. de Nocker and C. K. Toolan, "Using telehealth to provide interventions for children with ASD: A systematic review," *Rev. J. Autism Dev. Disord.*, vol. 10, no. 1, pp. 82–112, 2021, doi: 10.1007/s40489-021-00278-3.
- [18] M. L. H. Beauchamp, K. Amorim, S. N. Wunderlich, J. Lai, J. Scora, and M. Elsabbagh, "Barriers to access and utilization of healthcare services for minority-language speakers with neurodevelopmental disorders: A scoping review," *Front. Psychiatry*, vol. 13, art. 915999, 2022, doi: 10.3389/fpsyt.2022.915999.
- [19] A. F. Pagán, M. Ortiz, B. McCormack-Sadler, A. Esparza, G. Gogola, C. Montiel-Nava, and K. A. Loveland, "'An app that truly works for us': Co-designing a mobile health intervention to support the transition to adulthood for the Latino autism community," *J. Technol. Behav. Sci.*, pp. 1–13, 2026, doi: 10.1007/s41347-026-00628-0.
- [20] J. O. Cooper, T. E. Heron, and W. L. Heward, *Applied Behavior Analysis*, 3rd ed. Hoboken, NJ, USA: Pearson, 2020.
- [21] P. W. S. Leung, S. X. Li, C. S. O. Tsang, B. L. C. Chow, and W. C. W. Wong, "Effectiveness of using mobile technology to improve cognitive and social skills among individuals with autism spectrum disorder: Systematic literature review," *JMIR Ment. Health*, vol. 8, no. 9, art. e20892, 2021, doi: 10.2196/20892.
- [22] M. M. Kelley, T. Powell, D. Camara, N. Shah, J. M. Norton, C. Deitelzweig, N. Vaidy, C. J. Hsiao, J. Wang, and A. S. Bierman, "Mobile health apps, family caregivers, and care planning: Scoping review," *J. Med. Internet Res.*, vol. 26, art. e46108, 2024, doi: 10.2196/46108.

- [23] R. McNally Keehn, B. Enneking, M. Ramaker, M. Goings, Z. Yang, A. Carroll, and M. Ciccarelli, "Family-centered care coordination in an interdisciplinary neurodevelopmental evaluation clinic: Outcomes from care coordinator and caregiver reports," *Front. Pediatr.*, vol. 8, art. 538633, 2020, doi: 10.3389/fped.2020.538633.
- [24] T. Ayllon and N. Azrin, *The Token Economy: A Motivational System for Therapy and Rehabilitation*. New York, NY, USA: Appleton-Century-Crofts, 1968.
- [25] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, and T. Liu, "A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions," *ACM Trans. Inf. Syst.*, vol. 43, no. 2, art. 3155, 2025, doi: 10.1145/3703155.
- [26] Z. Wan, X. Wang, C. Liu, S. Alam, Y. Zheng, J. Liu, Z. Qu, S. Yan, Y. Zhu, Q. Zhang, M. Chowdhury, and M. Zhang, "Efficient large language models: A survey," *Trans. Mach. Learn. Res.*, 2024.
- [27] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede, "Lexicon-based methods for sentiment analysis," *Comput. Linguist.*, vol. 37, no. 2, pp. 267–307, 2011, doi: 10.1162/COLI_a_00049.
- [28] C. E. Ray, G. M. Wilson, A. M. Hughes, C. Cunningham Goedken, E. P. F. Liu, M. A. Fitzpatrick, K. J. Suda, S. M. Kota, C. Nwankpa, and C. T. Evans, "Alert fatigue measurement in clinical decision support: A systematic review," *J. Am. Med. Inform. Assoc.*, vol. 33, no. 8, pp. 1523–1531, 2026, doi: 10.1093/jamia/ocag064.
- [29] B. Suri, S. Taneja, I. Bhanot, H. Sharma, and A. Raj, "Cross-platform empirical analysis of mobile application development frameworks: Kotlin, React Native and Flutter," in *Proc. 4th Int. Conf. Inf. Manage. Mach. Intell. (ICIMMI)*, 2022, doi: 10.1145/3590837.3590897.
- [30] P. R. Saraf, S. M. Jadhao, S. J. Wanjari, S. G. Kolwate, and A. D. Patil, "A review on Firebase (backend as a service) for mobile application development," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 10, no. 1, 2022, doi: 10.22214/ijraset.2022.39958.
- [31] P. Escaleira, V. A. Cunha, J. P. Barraca, D. Gomes, and R. L. Aguiar, "A systematic review on security mechanisms for serverless computing," *Cluster Comput.*, vol. 28, no. 7, art. 465, 2025, doi: 10.1007/s10586-025-05371-4.
- [32] A. Alzu'bi *et al.*, "A review of privacy and security of edge computing in smart healthcare systems: Issues, challenges, and research directions," *Tsinghua Sci. Technol.*, vol. 29, no. 4, pp. 1152–1180, 2024, doi: 10.26599/TST.2023.9010080.
- [33] S. H. Zarit, K. E. Reever, and J. Bach-Peterson, "Relatives of the impaired elderly: Correlates of feelings of burden," *Gerontologist*, vol. 20, no. 6, pp. 649–655, 1980, doi: 10.1093/geront/20.6.649.
- [34] J. Brooke, "SUS: A quick and dirty usability scale," in *Usability Evaluation in Industry*, P. W. Jordan, B. Thomas, B. A. Weerdmeester, and I. L. McClelland, Eds. London, U.K.: Taylor & Francis, 1996, pp. 189–194.
- [35] M. Hyzy, R. Bond, M. Mulvenna, L. Bai, A. Dix, S. Leigh, and S. Hunt, "System Usability Scale benchmarking for digital health apps: Meta-analysis," *JMIR mHealth uHealth*, vol. 10, no. 8, art. e37290, 2022, doi: 10.2196/37290.