Product Requirements Document (PRD)

Project: EduFocus Adaptive

1. Project Overview

Project EduFocus Adaptive is a next-generation educational application designed to deliver a hyper-personalized, accessible learning environment for children with cognitive and physical disabilities, particularly ADHD and severe motor impairments.

The platform's vision is to transcend the "one-size-fits-all" model of digital education. Its core is a dual-engine system:

1. **An AI Adaptive Lesson Engine** that generates and dynamically adjusts interactive learning content in real-time, adapting to the user's focus, performance, and learning style.
2. **A Computer Vision-Based Control System** that enables full application navigation and interaction *entirely* through eye-tracking, eliminating the need for any traditional input devices.

By creating a self-contained, autonomous learning loop, EduFocus Adaptive aims to foster scholastic independence, build confidence, and provide an engaging, frustration-free educational experience.

2. Problem Statement & Motivation

The Problem: Digital learning tools remain profoundly inaccessible for a significant cohort of students.

1. **The Cognitive Barrier (ADHD):** Standard educational software, with its static content and rigid progression, fails to retain the focus of neurodivergent students. This lack of dynamic stimulation leads to disengagement, frustration, and a perception of failure, widening the learning gap (Kofler, et al., 2013).
2. **The Physical Barrier (Physical Disabilities):** The vast majority of applications are built around mouse, keyboard, or touchscreen interactions. For children with conditions like Cerebral Palsy, Muscular Dystrophy, or spinal cord injuries, these interfaces represent an insurmountable barrier, effectively excluding them from modern digital learning.

The Motivation: A critical gap exists at the intersection of these two challenges. While some tools address ADHD and others address physical accessibility, few (if any) provide a holistic solution for a child who faces *both*. EduFocus Adaptive is motivated by the principle of "radical inclusion"—that technology must adapt to the unique needs of every user, not the other way around. By pairing adaptive AI with gaze-based interaction, we can create a system where the child's focus and cognition are the *only* prerequisites for learning.

3. Target Users

The system is designed for a single user archetype, creating a fully self-contained and personalized environment.

Primary Persona: The Student

- **Name:** Alex
- **Age:** 10
- **Conditions:** ADHD (Inattentive Type), Cerebral Palsy (affecting fine motor control of all four limbs).
- **Abilities:** Bright and inquisitive. Good head and eye control. Can maintain focus on topics of interest but struggles with "dry" or repetitive material.
- **Needs:**
    - A "zero-touch" interface that allows for full interaction without hand movement.
    - Learning content that is "novel," "responsive," and "challenging-but-not-impossible."
    - Immediate, positive feedback, but a low-penalty environment for "mistakes" or "distraction."
    - Full autonomy within the learning environment.
- **Frustrations:**
    - Being "locked out" of games or apps due to physical input requirements.
    - Timed tests that create anxiety.
    - Content that is either too easy (boring) or too hard (frustrating) for long stretches.
    - Small click targets and complex menu navigation.

4. Objectives & Key Features

Objectives

1. **Personalization (AI):** To provide a 1:1, AI-driven learning path that continuously adapts content difficulty and style to each user.
2. **Accessibility (CV):** To enable 100% of the application's functionality via a software-only, webcam-based eye-tracking system.
3. **Engagement (UX):** To maximize on-task focus and lesson completion rates through gamified mechanics and an ADHD-friendly interface.
4. **Autonomy (System):** To create a "closed-loop" system where the student can independently manage their learning—from calibration to lesson completion—without external help.

Key Features

- **F1: AI Adaptive Lesson Engine:** The core pedagogical feature. The AI generates, presents, and modifies lessons on the fly based on user inputs.
- **F2: Eye-Tracking Control System:** The sole interaction modality. This includes gaze-based navigation, selection ("dwell-to-click"), and scrolling.
- **F3: Simple Eye-Tracking Calibration Wizard:** A one-time, visually guided setup process that tunes the CV model to the user's specific eye movements and environment.
- **F4: Gamified Micro-Lesson Structure:** Educational content is broken into short, 3-5 minute "missions" or "challenges" to maintain focus and provide rapid feedback.
- **F5: Real-time Progress & Focus Tracking:** A system that passively monitors performance (speed, accuracy) and focus (gaze-on-task vs. gaze-wander) to feed adaptation data back to the AI engine.

5. Functional Requirements

User & Calibration

- **FR-5.1:** The system shall support a single "Student" user profile, stored locally on the device.
- **FR-5.2:** On first launch, the system *must* initiate the Eye-Tracking Calibration Wizard.
- **FR-5.3:** The Calibration Wizard shall consist of the user "looking at" a series of 5-9 on-screen targets.
- **FR-5.4:** The user must be able to re-initiate calibration at any time from the main menu.

Eye-Tracking & Interaction

- **FR-5.5:** The system shall render a "gaze cursor" on-screen that mirrors the user's eye movement.
- **FR-5.6:** Selection shall be performed by "Dwell-to-Click": holding the gaze cursor on an interactive element for a configurable duration (default: 1.5 seconds).
- **FR-5.7:** A visual indicator (e.g., a closing ring) must show the dwell-to-click progress.
- **FR-5.8:** The system shall support gaze-based scrolling (e.g., looking at the top/bottom 10% of the screen).

AI & Learning

- **FR-5.9:** The system shall present a simple "Subject" selection menu (e.g., Math, Reading, Science).
- **FR-5.10:** Upon subject selection, the AI Engine shall generate a single, interactive micro-lesson (e.g., a multiple-choice question, a "match the pair" task).
- **FR-5.11:** All lesson content (text, instructions) must be automatically converted to speech (Text-to-Speech).
- **FR-5.12:** The AI must adapt the *next* lesson based on the user's interaction with the current one (see Section 9: AI Design System).
- **FR-5.13:** The system must provide immediate, positive, and non-distracting audio-visual feedback for correct answers.

- **FR-5.14:** For incorrect answers, the system shall provide a gentle, non-punitive "nudge" (e.g., "Good try! Let's look at that again...") and a simplified hint.

6. Non-Functional Requirements

- **NFR-6.1: Performance**
    - **Eye-Tracking:** Gaze-to-cursor latency must be < 150ms to feel responsive.
    - **AI Generation:** A new micro-lesson (question + options) must be generated and displayed in < 4 seconds.
    - **System:** The application must run efficiently (<= 30% CPU load) on mid-range hardware with a standard 720p/1080p webcam.
- **NFR-6.2: Privacy & Security**
    - **CRITICAL:** All camera video processing and eye-tracking analysis *must* be performed 100% on-device. No video frames or biometric data (e.g., face mesh) shall be sent to any server.
    - Only anonymized, non-biometric data (e.g., lesson prompts, performance statistics) may be sent to the backend for AI generation and storage.
- **NFR-6.3: Accessibility (UI/UX)**
    - The app must adhere to a "low-clutter," high-contrast design.
    - All interactive elements ("gaze targets") must have a minimum size of 100x100dp.
    - All text must be rendered in a clear, sans-serif, or dyslexic-friendly font.
    - No core functionality shall be "hidden" in complex menus.
- **NFR-6.4: Robustness**
    - The eye-tracking system must be robust to variations in lighting (e.g., daytime indoor) and user accessories (e.g., glasses).

7. User Experience (UX) & Interface Design

- **Core Principle:** "Focus by Design"
- **Visual Design:**
    - **Palette:** Use calm, cool color palettes (soft blues, greens, lavenders) to reduce cognitive load. Avoid harsh, alerting colors (e.g., bright reds, yellows) even for errors.
    - **Layout:** "One thing at a time." Each screen presents a single question, a single choice, or a single instruction.
- **Interaction Design:**
    - **Flow:** The user is guided from one task to the next in a linear, predictable flow.
    - **Feedback:** Feedback is positive and "intrinsic" (e.g., a puzzle piece fitting into place) rather than "extrinsic" (e.g., loud sounds, flashing banners) which can be distracting.
- **Focus Aids:**
    - The system shall detect gaze-wander (e.g., gaze off-task for > 10 seconds).

  - In response to gaze-wander, the system will *not* punish, but rather "nudge" (e.g., a subtle "breathing" highlight on the main task, a quiet audio prompt: "Still working on this one?").

8. Technology Stack & System Architecture

- **Frontend/Application (Client): Flutter**.
  - *Reasoning:* Excellent for cross-platform (desktop, tablet) UI. Provides direct, low-level access to the device camera feed and good performance for real-time rendering.
- **Computer Vision (CV) Engine (Client): TensorFlow Lite / OpenCV**.
  - *Reasoning:* A custom or pre-trained eye-tracking model (e.g., from MediaPipe's Face Mesh) can be run on-device via TFLite. OpenCV can be used for initial feed processing (e.g., grayscale, region-of-interest extraction).
- **AI Lesson Generation (Backend): FastAPI (Python) + Gemini API**.
  - *Reasoning:* A lightweight FastAPI backend serves as a secure broker. It receives requests from the client (e.g., "get level 3 math question"), crafts a detailed prompt, and queries a powerful generative model (like Gemini) to get a JSON-formatted lesson.
- **Database (Backend): Firebase Firestore**.
  - *Reasoning:* Stores user profile data (e.g., current level, subject preferences) and performance metrics. Real-time and simple to integrate.

System Architecture Flow:

1. **(Client):** Flutter app captures camera feed.
2. **(Client):** On-device TensorFlow model processes the feed, determines gaze coordinates, and moves the UI cursor.
3. **(Client):** User's gaze "dwells" on the "Math" button. A "click" event is triggered.
4. **(Client -> Backend):** Client sends an API request: `POST /getLesson {"userId": "alex123", "subject": "math", "metrics": {"last_score": 0.8, "last_speed": 4.5}}`
5. **(Backend):** FastAPI server validates the request and crafts a prompt for the AI: "Generate a 4th-grade math question, word problem style, medium difficulty... Format as JSON."
6. **(Backend -> LLM API):** The backend calls the Gemini API.
7. **(LLM API -> Backend):** The LLM returns a JSON object: `{"question": "...", "options": [...], "correct": 2}`
8. **(Backend -> Client):** The backend passes the JSON to the client.
9. **(Client):** The Flutter app parses the JSON and renders the new lesson.

9. AI Design System

This section defines the logic for the "AI Adaptive Lesson Engine" (F1).

Input Trigger (User Behavior)	AI Analysis (Inference)	AI Action (Adaptation)
Performance: User answers 3+ consecutive questions correctly.	User is "mastering" this level.	**Increase Difficulty:** Advance to the next "level" (e.g., 2-digit to 3-digit addition).
Performance: User answers 2+ consecutive questions incorrectly.	User is "struggling."	**Decrease Difficulty:** Break the problem down. Offer a simpler, foundational question.
Speed: User answers correctly and very quickly (< 3 seconds).	User is "bored" or "finds this too easy."	**Increase Novelty:** Change the question *style* (e.G., from multiple-choice to a "fill-in-the-blank").
Speed: User is "stuck" on a question (e.g., > 30 seconds).	User is "confused" or "losing focus."	**Offer Hint:** Proactively display a simple, textual, or visual hint.
Focus: Gaze tracking detects "gaze-wander" (off-task > 10 seconds).	User is "distracted."	**Trigger Focus-Aid:** Initiate a gentle visual/audio "nudge" to re-engage the user *before* they fail.
Preference: User consistently answers "visual" problems faster than "text" problems.	User has a "visual learning preference."	**Personalize Style:** Prioritize generating more visual-based problems in future lessons and store this preference.

10. Evaluation Metrics & Success Criteria

- **Accessibility:**
  - **Metric:** Task Completion Rate (TCR) using 100% eye-tracking.
  - **Success:** 95% of users can complete the entire loop (calibrate -> select subject -> finish lesson) without assistance.
- **Engagement:**
  - **Metric:** "Focus Time Ratio" (Time gaze-on-task) vs. "Wander Time."
  - **Success:** A session-over-session *increase* in the Focus Time Ratio for a given subject.
- **Learning:**
  - **Metric:** Rate of Mastery (e.g., number of lessons/time taken to progress from "Level 1" to "Level 2").
  - **Success:** Measurable progression in at least one subject over a 1-week period.
- **AI Performance:**
  - **Metric:** Frequency of "frustration" states (e.g., multiple incorrect answers in a row).
  - **Success:** The system's adaptations (Section 9) lead to a *reduction* in frustration states over time.