DESIGN DOCUMENT

FOR

Insight Diagnostic & Vision Therapy Platform

Version *0.2.3*

Prepared by

Student Name(s): Anki Youngblood

Instructor: **Prof. Thoms**

Course: **COMP 491 Capstone Preparation**

Date: *May 4, 2025*

System Background

Problem / Need Description

Binocular Vision Dysfunction is an umbrella term for conditions causing misalignment of one or both eyes which results in decreased or absent depth perception. To perceive and judge depth properly our eyes use some very handy properties of physics. Using the images from your left and right eyes your brain performs a very important task called fusion. Fusion is the act of combining the two images from slightly different angles to create the illusion of depth. What you are really seeing is two separate images layered on top of each other allowing you to “triangulate” where objects are in 3D space compared to your eyes.

Currently people who lack stereoscopic vision/depth perception have several options of treatment, some very much better than others. The most invasive option is strabismus surgery which uses adjustable stitches on cut ocular muscles to reposition the eye, the muscle is cut under general anesthesia, although the adjustable stitches are moved while the patient is awake. The noninvasive methods include base prism lenses, convex prism lenses, or vision therapy. Currently a singular company called NeuroLens, with the product NeuroLenses, has the patent for convex prism lenses and uses a proprietary measuring algorithm to give you a NeuroLens number. This number can be used by your neuro-ophthalmologist to order the NeuroLenses according to your needs.

The initial appointment to see a neuro-ophthalmologist requires a deposit because of the effort and time the comprehensive exam requires. The deposit can be either the entire cost of the appointment or a percentage up front. The price upfront ranges from $100 – full appointment cost, with the full appointment cost being about $580 on average for neurolens partners (yes they partner with ophthalmologists). Patients will usually require <u>at least</u> two pairs of neurolenses throughout their treatment, which cost $700 dollars a pair. That means within the standard treatment span of 6-8 weeks you will have spent $2000, with a guarantee of only 50% symptom improvement for one of the symptoms, possibly still needing to purchase new lenses in the future.

System overview

NeuroLens also has a patent on the device they call N3, this device is a simple Business to Business (B2B) Virtual Reality (VR) headset with eye tracking built in. The headset is made by a company called Pico and sold on their website under the name Pico Neo 3 Pro Eye. NeuroLens purchases these devices, then modifies the headsets operating system to launch their measurement software on startup. The device performs several stages of testing in order to get a control of your eyes base alignment, then using that control to measure if misalignment is present. The stages of measurement are calibration, base tropia measurement right, fine tune right, then again for the left side.

This application is an open source phoria and tropia measurement program I have fittingly internally code-named “NeuroLens Killer.” The Insight Diagnostic Platform and NeuroLens’ N3 device program are both very simple. The VR headset acts as a vehicle for graphics to be displayed separately for each eye. The ability to show the eyes non-fusible images (left and right see something different) allows the program to easily manipulate the eyes in a way that can expose deficiencies. Dissociated phoria, for example, can be measured by showing the left and right eye non-fusible images to find one of the eyes resting position. This resting position can then be compared to the movement of each eye establishing an amount of offset (measured in diopters or arcseconds) that correlates to the misalignment when the eyes are at rest. The nature of the VR headset also allows simulated misalignment or correction as if the user was wearing a pair of prescription glasses. This allows the program to both diagnose more accurately and correct vision using games that require the user to force their eyes to participate in order to see a fully fused image.

Insight will utilize these tools to measure the misalignment in a user and tailor the vision therapy experience to their specific misalignment or insufficiency. Other devices can also easily be supported due to the use of OpenXR, which is a standard library that works across most VR Headsets. Providing relatively low cost alternatives and a therapy based approach allows users to choose the market Neurolens has built around their corrective lenses, or an open platform that requires more effort at almost no cost. The user could simply install the program on their eye tracking capable VR headset and perform a diagnostic test and therapy that would otherwise cost 1000s of dollars.

User characteristics

Insight is aimed to provide an alternative to expensive in-office diagnostics and corrective lenses. The laymen user has the most to gain from the device, this inherently hinders the amount of benefit a professional user may get because of the diversion of laymen users to a self run test. Laymen users will be able to use an off the shelf VR device and free application to diagnose and measure any Binocular Vision Dysfunction. Professional users will be able to use the same off the shelf devices to improve their diagnostic testing in the office, but it does not drastically change their access to affordable diagnostic options.

Laymen Users

Jamie is a college student who recently began noticing recurring headaches and eye strain while studying. She doesn't have access to a specialist but owns a Meta Quest Pro that she sometimes uses for games. After reading about BVD online, she downloads the Insight Diagnostic Platform in hopes of better understanding her symptoms. Jamie values privacy and ease of use, and she has no clinical background. She is able to follow the program's on-screen guidance to calibrate the headset and complete the diagnostic tests. The session gives her a report indicating an EXOphoria, helping her feel validated and informed when she later visits an optometrist. For Jamie, the app's success depends on clear visuals, guided steps, low jargon, and no required internet access.

Dana is a 35-year-old single parent of two. Her 10-year-old daughter, Maya, often struggles to read and frequently complains of blurry vision and headaches - but routine eye exams haven't revealed any issues. After researching online and stumbling across a post about Binocular Vision Dysfunction, Dana learns about the Insight Diagnostic Platform. While she can't afford a $1,000 VR headset with built-in eye tracking, she finds that her local public library recently received a Vive Pro Eye through a community tech grant. The library's assistive technology room allows patrons to reserve the headset for 30-minute private sessions. Dana signs up for a session, helps Maya through the fitting and calibration process, and walks her through the on-screen instructions. The program produces results suggesting an ESOphoria, giving Dana something concrete to share with a local optometrist for follow-up care. For Dana, success means access: having a free, guided, and dignified way to understand her daughter's struggles.

Professional User

Dr. Lee runs a clinic with limited access to proprietary diagnostic tools due to cost and the requirement to join a partner program. She purchases several consumer VR headsets for her office and uses Insight to quickly screen patients for binocular misalignment before referring them for more intensive follow-up. She appreciates being able to store anonymized diagnostic results and use the calibration and gaze data to cross-validate her clinical impressions. Her needs are accuracy, reliable data output, and the ability to export results to her clinic's Electronic Medical Record (EMR) system. For Dr. Lee, success is a reliable, fast, and low-maintenance diagnostic tool that can be integrated into her workflow.

Mr. Alvarez works at a public middle school and runs intervention assessments for students struggling academically. He's not an optometrist, but he’s trained to use digital tools for early screening. After getting approval from administration, he uses Insight with a donated VR headset to screen students who report headaches, trouble copying from the board, or skipping lines while reading. The app provides instant results and allows him to print a simple PDF to send home to parents. For Mr. Alvarez, success means short, reliable tests that are easy for students to understand, require no internet, and provide actionable recommendations. This allows the school to flag potential BVD cases and refer families for care without needing an in-house specialist.

Product functions

Function 1: Session Setup & Optional User Identification

Description: Handles starting the app, optionally collecting user email for identification, managing temporary data, and includes the end-of-session flow for demographics and consent-based persistent saving.

App Requirements (Device: Meta Quest Pro): Renders UI for splash screen, optional email input, demographic questions, and consent prompt. Executes logic for temporary data storage during the session. Executes conditional logic at session end to either save data persistently or discard it based on user consent. If saving, handles data and file writing with unique filenames.

Network Requirements: The user must have an internet connection and access to the Meta Horizon store in order to download and run the application. The application can also be sideloaded by advanced users, but this is not recommended.

Data Requirements (Local Storage on Device): Uses temporary storage (device memory or temporary files) for all session data (measurements, email if entered, demographics) during the active session. Uses persistent storage (`Application.persistentDataPath`) only if the user consents at the beginning of the session. If consent is given, stores complete session data (measurements, demographics, email if provided, calibration info) in a non-human-readable binary format.

Function 2: Headset Fit, IPD, & Eye Tracking Calibration

Description: Guides the user through proper headset fit and manual IPD adjustment, performs the temporary eye tracking calibration required for measurements, and provides success/failure feedback. Successful calibration is mandatory to proceed.

App Requirements (Device: Meta Quest Pro): Renders UI for fitting guidance (reactive graphic-based), IPD adjustment assistance (reactive graphic-based, text, clarity test target), and calibration stimulus (random dot). Executes logic for the calibration sequence. Processes feedback from the eye tracking SDK regarding calibration success/failure. Enforces calibration redo loop upon failure. Utilizes the Quest Pro eye tracking SDK/API.

Data Requirements (Local Storage on Device): Generates a temporary eye tracking calibration model (personalized mapping). This model is stored only in active memory for the duration of the current session. No persistent storage of calibration data occurs.

Function 3: Distance Phoria Measurement (Dissociated & Associated)

Description: Objectively measures both dissociated (base alignment) and associated (fixation disparity using Nonius lines) phoria using eye tracking while the user views targets simulating a far distance (e.g. 6m). Fixation on the central target is monitored for validity during associated phoria measurement.

App Requirements (Device: Meta Quest Pro): Renders distinct stimuli for each eye for dissociated test (target vs. pattern). Renders binocular fusion target plus non-fusible target for associated test. Accesses calibrated eye tracking data (from Function 2) in real-time for both eyes. Executes logic to calculate gaze deviation relative to targets or fixation point. Executes logic for fixation monitoring (checking gaze within window around central target) during associated phoria test. Validates measurement data based on fixation status.

Data Requirements (Local Storage on Device): Outputs raw measurement results (e.g. angular deviation H/V) for both dissociated and associated tests at distance. This data is stored temporarily during the session, awaiting processing by Function 5 and final saving or discarding by Function 1.

Function 4: Near Phoria Measurement (Dissociated & Associated)

Description: Objectively measures both dissociated and associated phoria/fixation disparity using eye tracking while the user views targets simulating a near distance (e.g. 50cm). Fixation monitoring occurs during associated phoria measurement.

App Requirements (Device: Meta Quest Pro): Same capabilities as Function 3, but renders stimuli simulating a near viewing distance (adjusting vergence cues). Renders distinct stimuli for dissociated near test. Renders fusion target + non-fusible target for associated near test. Accesses calibrated gaze data, calculates deviations, monitors fixation, validates data.

Data Requirements (Local Storage on Device): Outputs raw measurement results (e.g. angular deviation H/V) for both dissociated and associated tests at near. This data is stored temporarily during the session.

Function 5: Deviation Calculation, PD Conversion, & Diagnosis Labeling

Description: Processes the raw angular deviation data from Functions 3 & 4. Converts these angles into Prism Diopters (PD) and determines the corresponding diagnostic directional label (e.g. EXO, ESO, HYPER, HYPO).

App Requirements (Device: Meta Quest Pro): Executes mathematical calculations: trigonometric functions (tangent) for PD conversion. Executes conditional logic to assign directional labels based on the sign/direction of the deviation values. Requires access to the temporary raw measurement data from Functions 3 & 4.

Data Requirements (Local Storage on Device): Outputs the final calculated results (e.g. "2.5 PD EXO", "0.5 PD R Hyper"). This processed data is stored temporarily during the session, ready to be saved persistently or discarded by Function 1 based on user consent.

Success Criteria

Success for the Insight Diagnostic Program is defined by its ability to produce accurate, consistent, and actionable diagnostic data regarding Binocular Vision Dysfunction. The system must reliably track and measure deviations using eye tracking across a wide range of users, ensuring both dissociated and associated measurements align with accepted clinical metrics. For lay users, success includes intuitive guidance through the testing process, with minimal jargon and no need for professional oversight. For professionals and researchers, success means having access to raw diagnostic data, accurate prism diopter conversions, and modular exportable results for integration with clinical records or research workflows. The system must allow for quick switching between user sessions without data overlap or privacy risk — especially critical in shared-device settings like libraries, clinics, or research labs. All data must be handled in accordance with HIPAA-level privacy standards, with user consent guiding persistent storage and personally identifying information treated with strict care. Finally, true success depends on trust and access: Insight must remain free, open-source, and affordable — designed to work on widely available consumer hardware while preserving clinical integrity. If the platform earns the trust of underserved users, clinicians, and researchers alike, and begins to democratize access to accurate BVD diagnostics, then it will have fulfilled its mission.

System Architecture

System Architecture

The Insight Diagnostic Program is designed as a modular, local-first system, designed to operate entirely on a standalone VR headset with built-in eye tracking. This structure was chosen to maximize user privacy, ensure accessibility without internet access, and support real-time eye tracking data without latency. The system is divided into five main subsystems:

1. **User Interaction & Session Manager** – Handles all user-facing elements including UI, onboarding, demographic questions, consent prompts, and calibration guidance. It also manages user sessions, handles temporary and persistent data storage, and enforces session flow control.

2. **Calibration & Device Layer** – Responsible for guiding the user through headset fit, IPD adjustment, and temporary eye tracking calibration. This layer interfaces directly with the VR platform's SDK (e.g., Meta Quest Pro Eye SDK or OpenXR eye tracking modules).

3. **Measurement Engine** – Core logic for rendering diagnostic stimuli to each eye and capturing real-time eye tracking data. This includes logic for dissociated and associated phoria tests at both distance and near viewing conditions. It handles data validation, fixation monitoring, and error detection.

4. **Data Processor & Diagnostic Logic** – Converts raw angular gaze deviations into clinically useful results, this means prism diopters and directional labels (e.g., EXO, ESO, HYPER). This logic is further segregated by discrete functions, allowing future diagnostic types or experimental measurements to be added by simply writing a new function for it.

5. **Data Persistence Layer** – Temporarily stores session data in memory and optionally serializes results to persistent storage (e.g., Binary files in `Application.persistentDataPath`) given prior user consent. It ensures user privacy by discarding unconsented sessions and isolating data per session into a non-human readable file format.

The system operates entirely offline and does not rely on cloud services. It was intentionally decomposed this way to isolate responsibilities, improve modularity, and support extension by researchers or clinicians. Future expansions such as vision therapy modules or cloud syncing for clinical deployments can be added on top of this base architecture without disrupting core diagnostics.

System Constraints

Hardware Requirements: The platform requires a VR headset with built-in eye tracking. At present, only a limited set of consumer-accessible devices meet this requirement, such as the Meta Quest Pro (which has been discontinued) or select Pico and Vive models. This inherently limits accessibility and user adoption, and is why the system needs to remain lightweight and compatible with OpenXR to support the widest range of eye-tracking capable headsets.

Offline Operation & Privacy: In order to meet HIPAA-level privacy standards and ensure accessibility in resource-limited environments (e.g., libraries, rural clinics), the diagnostic system and user interface must operate entirely offline. No cloud APIs, telemetry, or external analytics are present at this time. This constraint requires that all data be processed and stored locally on the headset, and that users are clearly prompted to opt-in before any personally identifiable information (PII) is saved.

SDK Interoperability: The program must support multiple eye tracking SDKs and conform to OpenXR standards to remain device-agnostic. Since manufacturers may implement eye tracking in different ways (e.g., Meta SDK vs. Tobii SDK vs. OpenXR SDK), Insight must include an abstraction layer to detect and standardize these different inputs, adding complexity to development and testing.

Performance & Real-time Requirements: Eye tracking data must be captured and interpreted in real time to ensure accurate gaze deviation measurement and proper calibration. Latency above acceptable thresholds (>30ms) could invalidate results, particularly in dynamic tests involving fixation monitoring. This requires careful performance optimization and efficient memory management within Unity or any other rendering engine used. The eye-tracking implementation must also operate above acceptable thresholds (>90hz).

User Interface Simplicity: The system must be usable by laypersons without technical or clinical training. UI/UX design is heavily constrained to use clear visual cues, voice or visual feedback, and zero-jargon explanations when giving a diagnostic result.

Limited Storage Resources: Because the system runs on headsets with limited internal storage, all session data must be kept as small as possible. Large datasets (e.g., full-resolution gaze heatmaps) must be optionally discarded or heavily compressed. Persistent session files must be small and stored in designated application-safe persistent file paths.

ASSUMPTIONS AND DEPENDENCIES

Supported Hardware Availability: The system assumes that the user has access to a compatible VR headset with built-in eye tracking. Currently, this includes devices like the Meta Quest Pro and Pico Neo 3 Pro Eye. Users without such devices cannot access the platform, and the diagnostic process will not run on devices lacking eye-tracking capabilities.

OpenXR & Eye Tracking SDK Support: The application depends on the availability and stability of vendor-specific eye tracking SDKs, such as Meta's XR Interaction SDK or Tobii's runtime (depending on the headset). The system also assumes these SDKs will remain compliant with OpenXR standards to allow future updates to continue to use proprietary SDKs. If SDKs are deprecated or fail to comply, the system's compatibility with multiple headsets could break.

Offline-Only Usage Environment: The system is designed to operate entirely offline for privacy reasons. It assumes that all required assets (binaries, calibration models, UI resources) are available on the device at runtime, and that no external internet connection or server access is required for function or updates.

End-User Technical Literacy is Low: It is assumed that most users, especially laypersons, will have little to no clinical or technical training. Therefore, the program's UX design is constrained to use simple instructions, clear feedback, and predictable session flows. Developers must avoid jargon, multi-step setup, or configuration screens that could confuse or deter users when creating extensions.

Session-Based Use Model: The application currently assumes that users will complete their diagnostics in a single session. While repeated use may be supported, data from each session is assumed to be ephemeral unless consent is granted for persistent storage.

Unity Runtime and VR Device OS Compatibility: The program is developed using the Unity engine and assumes continued support for monocular VR rendering within OpenXR.

Development Methods

The Insight Diagnostic Platform was developed using an iterative, prototype-driven approach. Given the experimental and technical nature of VR-based diagnostics, development prioritized early functional prototypes over documentation-heavy processes. Each core module — such as session flow, calibration, and phoria measurement — was first built as a standalone prototype and tested independently using simulated input data or real-time device output, before being integrated into a unified session framework.

The system was implemented in Unity using C#, chosen for its large VR development ecosystem (including tutorials), OpenXR compatibility, and rapid prototyping tools. Unity's support for cross-platform VR runtimes and eye-tracking SDKs allowed for flexible development targeting the Meta Quest Pro headset currently.

Version control and feature management were handled using Git and GitHub.

No formal framework such as Scrum or Extreme Programming was adopted in full, but elements of agile development — including short feedback loops, continuous refactoring, and incremental "feature delivery" — were used throughout. Given the project's dual nature as a capstone and open research tool, the methods used prioritized functionality, transparency, and future adaptability over strict adherence to a methodology.

Detailed System Design requirements

External interface requirements

Eye Tracking SDK Interface: The Insight Diagnostic Program interfaces directly with the eye tracking subsystem provided by the target VR headset. On the Meta Quest Pro, this is the Meta XR Eye Tracking SDK, which provides real-time gaze origin, direction, and fixation confidence values per eye (confidence values currently do not work). The app assumes access to calibrated eye tracking data post-session calibration function. This SDK is required to initiate and monitor calibration sequences, and continuously track gaze for all measurement functions.

OpenXR API Integration: To support cross-platform compatibility, Insight uses OpenXR for VR display and input management (custom gaze pointer is used). This includes stereoscopic rendering (per-eye render control), head pose tracking, and gaze input abstraction. OpenXR also facilitates fallback compatibility with non-Meta headsets that offer OpenXR-compliant eye tracking modules (e.g., Pico, Vive).

Local File System Access: The system writes diagnostic session data to the headset's internal storage using `Application.persistentDataPath` (Unity's standard local-safe directory). Session data — including calibration logs, gaze metrics, and result labels — is saved in binary files. If consent is not given, all session data is discarded from memory after the session. No cloud syncing or external APIs are used in the current version.

Headset Hardware Interface (Input/Output): The system receives user input via gaze-based UI selection (using eye tracking). It outputs visual information to the VR display using Unity's built-in pipeline (URP does not support per eye rendering control). Audio feedback (e.g., calibration pass/fail cues) may be played through the headset's built-in speakers.

Functional requirements

Session Setup & Optional User Identification

Description: Initializes the diagnostic session. Guides the user through optional identification (e.g., email input), asks demographic questions, and prompts for consent to save results.

Inputs:

- Gaze-based input
- User-provided email (optional)
- Consent selection (yes/no)
- Demographic form inputs (e.g., age, symptoms)

Outputs:

- Session metadata object (temp)
- Binary file (only if consent given)

System Components Used:

- UI renderer
- Input system
- Local file path

Headset Fit, IPD, & Eye Tracking Calibration

Description: Ensures proper headset placement and completes required eye-tracking calibration. Users cannot proceed until a valid calibration is completed.

Inputs:

- Headset position
- Eye gaze data
- Gaze selection to confirm fit

Outputs:

- Eye tracking calibration data (temp memory only)

System Components Used:

- Meta or OpenXR eye tracking SDK
- UI feedback (visual/audio)
- Calibration logic

Error Handling / Edge Cases:

- If calibration fails, the user is looped back with feedback to reattempt.
- If tracking is unstable (e.g., due to glasses), notify the user and offer retry or abort.

Distance Phoria Measurement (Dissociated & Associated)

Description: Displays targets at simulated far distance (~6m). Collects gaze deviation under dissociated (non-fusible) and associated (fusion + offset) conditions.

Inputs:

- Calibrated gaze data
- Headset orientation

Outputs:

- Raw angular deviation per eye (horizontal and vertical)

System Components Used:

- Stereo rendering system
- Gaze tracking system
- Target logic

Error Handling / Edge Cases:

- If user fails to fixate, system invalidates that trial and retries.

Near Phoria Measurement (Dissociated & Associated)

Description: Same as Function 3 but simulates near distance (e.g., 50cm) and adjusts vergence/focal cues accordingly.

Inputs / Outputs / Components:
(Same as above, with near-distance rendering)

Error Handling / Edge Cases:

- Same as Function 3, but allow slightly larger fixation window due to near viewing strain.

Deviation Calculation, PD Conversion, & Labeling

Description: Processes gaze deviation data into clinical labels (e.g., “2.5 PD EXO”). Converts angular deviation to prism diopters using trigonometric formulas.

Inputs:

- Angular deviation values from previous functions
- IPD (Interpupillary distance)
- Target coordinates

Outputs:

- Result string (e.g., "3.2 PD Left Hyper")
- Result object for export

System Components Used:

- Math logic engine
- Data processor
- UI result renderer

Error Handling / Edge Cases:

- If deviation is zero or below threshold, mark as "No measurable misalignment".
- Flag invalid inputs (e.g., missing calibration) before calculation.

Performance requirements

Calibration Latency: The system must complete eye tracking calibration within 30 seconds on average. If the calibration takes longer or fails repeatedly, the user should be guided with clear feedback and an option to retry or exit.

Real-Time Gaze Tracking: Gaze data must be processed with **<30ms latency** during all measurement phases. Tracking delay beyond this threshold may result in invalid or unusable measurement results, particularly in fixation-dependent tasks.

Fixation Monitoring Accuracy: The fixation validation algorithm must detect and flag deviations of >1.5 degrees from the center of the fusion target during associated phoria tests. Trials exceeding this should be excluded automatically.

Data Accuracy / Angular Resolution: The eye tracking system must resolve angular deviations with at least ±0.2° precision to support reliable prism diopter conversion. This aligns with clinical expectations for basic binocular deviation screening tools.

Memory Usage: Session memory footprint (including gaze buffers and temporary calibration data) must remain under 150MB RAM at runtime. All temporary data must be discarded after session end unless user consent is granted.

Session Duration: A full diagnostic session (including calibration, both near and distance tests, and results display) should complete in under 5 minutes for most users.

Storage Constraints: Persistent session data (if consented) must be stored in binary files under 1MB per session.

Design constraints

HIPAA-Aligned Privacy by Design: The system must follow privacy-first design principles aligned with HIPAA standards. No data is transmitted to cloud services, no identifying information is required, and all storage is consent-based and using non-human readable binary file types. Users must explicitly opt-in before any session data is saved to persistent storage.

Offline-First Architecture: All functionality, including calibration, measurement, data processing, and report generation, must function without internet connectivity. This ensures accessibility in clinical, school, or community environments with limited or no Wi-Fi.

Unity Game Engine (C#): The system is developed using the Unity engine and written in C#. Unity was selected for its robust support for OpenXR, VR development tools, cross-platform compatibility, and rapid prototyping capabilities. This constrains development to the capabilities and limitations of the Unity runtime.

OpenXR Compatibility Requirement: The application must support multiple headsets via the OpenXR API. Eye tracking integration is abstracted via conditional modules (e.g., Meta SDK, Tobii SDK) and must degrade gracefully if unsupported features are detected.

Consent-Gated Persistent Storage: The program must default to temporary session data only. Permanent storage is locked behind an explicit consent screen. If declined, all session data must be deleted with no manual action required by the user.

Linear User Flow with No Setup Menu: The diagnostic experience must follow a linear, non-changing path. No external settings menus, debug tools, or calibration parameters are exposed to end users. This enforces accessibility and simplicity for non-technical users, especially children or parents in library/public-use scenarios.

Minimalist, Text-Light UI: All UI screens must prioritize symbols, graphics, or spoken guidance over dense text. When text is needed, it must be written at about a 6th grade reading level or lower to support low-literacy and neurodivergent users.

Software System attributes

Security: Insight is designed to enforce strict data security by operating entirely offline. No cloud APIs, telemetry, or network dependencies are present, eliminating external attack surfaces. All diagnostic data is stored locally, and only when a user explicitly consents at the end of the session. Session files are stored with randomized filenames in protected directories and can not be accessed by users on the device. No personally identifying information is collected unless voluntarily entered by the user.

Reliability: The system is built to function consistently across multiple sessions and user types. All critical operations, such as eye tracking calibration and diagnostic measurement, include built-in validation and retry logic to ensure stable results. If gaze data becomes unreliable during testing, the system detects the issue and guides the user through recalibration or a clean restart without requiring developer intervention. Unreliable sessions will not provide a diagnostic result.

Maintainability: The system uses a modular software architecture with clean separation between user interface, calibration logic, measurement engines, and data processing. This design supports rapid debugging and low-friction extensibility when adding new diagnostic routines, supporting new hardware, or conducting research-based modifications.

Portability: The Insight application targets all VR headsets that implement OpenXR with eye tracking support. By abstracting hardware dependencies and adhering to cross-platform standards, the system can be ported between headsets with minimal architectural change. This allows the application to remain usable across future hardware iterations without reimplementation of core logic.

Availability: The application is locally installed and functions without internet access, ensuring 100% uptime as long as the device is operational. It can be deployed in libraries, schools, rural clinics, or homes without needing network connectivity. There is no login system, licensing requirement, or central server dependency, which supports fully independent usage.

Asset requirements

Visual Assets

- Stereoscopic stimuli used in diagnostic tests, including:
  - Dissociated phoria targets (per-eye unique images)
  - Associated phoria fusion targets (same as dissociated but not per-eye)
- Calibration dot animations and headset fitting guidance overlays
- UI elements for splash screen, consent prompts, session navigation, and result display
- High-contrast, icon-based controls designed for accessibility and gaze input

Audio Assets

- Feedback sounds for calibration success/failure
- Confirmation tones for test steps or results
- Optional voice prompts (planned) for accessibility or younger users

Rendering Architecture Insight intentionally does not use Unity's Universal Render Pipeline (URP), due to URP's limitation in handling distinct left/right eye rendering passes. Instead, the application uses Unity's built-in rendering pipeline, allowing full manual control over stereoscopic rendering behavior. This enables true per-eye stimulus control, which is essential for accurate phoria measurements.

Other requirements

Regulatory Awareness and Non-Medical Use Disclaimer: Insight is not currently certified as a medical device under FDA, CE, or other regulatory standards. As such, the software must include a disclaimer in its documentation and on-screen results indicating that it is for informational and research purposes only and is not intended to diagnose, treat, or cure any condition. This disclaimer should be presented prior to test execution and again in any result output.

Open Source Licensing Requirement: The software must be distributed under a GLP open-source license that allows modification, redistribution, and use for both research and non-commercial diagnostic purposes. Licensing terms must be visible in the code repository and within the application menu or about screen. Commercial use of the software and the underlying codebase are not permitted.

References

"Eye Tracking for Movement SDK for Unity." Meta Developers, https://developers.meta.com/horizon/documentation/unity/move-eye-tracking/. Accessed 22 Apr. 2025.

"OpenXR Specification." Khronos Registry, https://registry.khronos.org/OpenXR/specs/1.0/html/xrspec.html. Accessed 22 Apr. 2025.

Zimanyi, Gergely T. System for Measuring Binocular Alignment with Adjustable Displays and Eye Trackers. US Patent 20240407648, issued 22 Oct. 2024.

"VR Development in Unity." Unity Manual, https://docs.unity3d.com/2023.1/Documentation/Manual/VROverview.html. Accessed 22 Apr. 2025.

Scheiman, Mitchell, and Bruce Wick. Clinical Management of Binocular Vision: Heterophoric, Accommodative, and Eye Movement Disorders. 3rd ed., Lippincott Williams & Wilkins, 2008.

Rouse, Michael W., et al. "Reliability of Binocular Vision Measurements Used in the Classification of Convergence Insufficiency." Optometry and Vision Science, vol. 79, no. 4, 2002, pp. 254–264. https://doi.org/10.1097/00006324-200204000-00012.

"Validation of the Binocular Vision Dysfunction Questionnaire (BVDQ)." Vision Specialists of Michigan, https://www.vision-specialists.com/articles/validation-of-the-binocular-vision-dysfunction-questionnaire-bvdq/. Accessed 22 Apr. 2025.

"Binocular Vision Disorders." AccessScience, https://www.accessscience.com/content/article/a081450. Accessed 22 Apr. 2025.