# Developer Guide
# Software Engineering

Team #22, TeleHealth Insights
Mitchell Weingust
Parisha Nizam
Promish Kandel
Jasmine Sun-Hu

Table 1: Revision History

| **Date** | **Developer(s)** | **Change** |
|---|---|---|
| Date1 | Name(s) | Description of changes |
| Date2 | Name(s) | Description of changes |
| ... | ... | ... |

# Contents



# 1 Introduction

Welcome to the developer guide for **TeleHealth Insights**, a web-based platform designed to facilitate at-home bilingual language assessments for children with speech difficulties. This project was developed to bridge the gap between traditional in-clinic assessments and accessible, remote care.

The platform empowers parents to administer structured language assessments from home while ensuring that Speech Language Pathologists (SLPs) have secure and organized access to comprehensive assessment data. The system is built with three key user roles in mind: *Parents*, *Children*, and *Clinicians*.

TeleHealth Insights consists of two Dashboards: **Parent Dashboard** and **Clinician Dashboard** with a tailored interface focused on usability, engagement, and efficiency.

## 1.1 Project Overview

**TeleHealth Insights** enables parents to conduct speech assessments with their children in the comfort of their own homes, without requiring real-time clinician

supervision.

The assessments, based on the MERLS framework, are available in both **English** and **Mandarin**, and consist of three test types:

- **Matching Tests:** Select the picture that best corresponds to the audio prompt.
- **Repetition Tests:** Repeat the sentence played in the audio as accurately as possible.
- **Quantifier Tests:** Choose the image that best represents the quantity or description mentioned in the audio.

**Parent Interface Features:**

- **Guided Tutorials:** Step-by-step instructions to help parents guide their children through the assessment process.
- **Assessment Execution:** Parents can select a test type, configure video and audio recording hardware, and conduct assessments independently.
- **Results Dashboard:** A centralized view of current and historical test results for ongoing progress tracking.

**Clinician Interface Features:**

- **Client Monitoring:** View individual client results by test type or overall performance.
- **Media Review:** Watch video recordings of test sessions and evaluate corresponding answers.
- **Manual Grading:** Review and grade repetition tests where automated scoring may not apply.
- **Bias Detection:** Utilize integrated tools to detect anomalies, inconsistencies, or potential cheating.
- **Clinical Notes:** Add notes and observations for each session directly within the platform.
- **Add Clients:** Add New clients to the clinician's list within the clinic.

This developer guide provides a detailed breakdown of system architecture, components, APIs, and user flow to help you effectively build, maintain, or extend the platform.

# 2 Technology Stack

## 2.1 Frontend

The frontend of the application is built using modern web technologies to ensure a responsive and accessible user experience across platforms. The technologies include:

- **JavaScript:** Core language for client-side logic.
- **React:** A component-based library for building dynamic and interactive UIs.
- **Tailwind CSS:** Utility-first CSS framework for rapid UI development and consistent styling.
- **Vite:** A fast build tool and development server that improves performance and module handling.
- **Bootstrap:** A UI toolkit used selectively for layout and responsiveness in specific components.

## 2.2 Backend

The backend handles API routing, media processing, authentication, and external integrations. It includes the following technologies:

- **Node.js:** A JavaScript runtime for building scalable server-side applications.
- **Express.js:** A minimalist web framework used to create RESTful API endpoints.
- **MediaPipe:** A framework used to process video input and detect facial landmarks.
- **DeepGram:** A speech-to-text API integrated for transcription and voice analysis.

## 2.3 Storage and Cloud Services

- **AWS S3:** Used for secure storage of user-submitted media files (e.g., video and audio recordings), assessment results, and assessment content such as images and audio prompts. All data is stored with fine-grained access control to ensure security and privacy. It serves as the central hub for both data submission and retrieval across the platform, supporting features such as result dashboards and clinician grading.

The system uses the following dedicated S3 buckets for service-specific storage:

- **telehealth-clinicians:** Stores clinician-related data and configurations.
- **telehealth-media-processing:** Handles video/audio files submitted during assessments for backend processing.
- **telehealth-parents:** Stores parent-side submission data and metadata.
- **telehealth-question-storage:** Contains the visual and audio question content used in assessments.
- **telehealth-result-storage:** Stores all processed assessment results and analytics for rendering dashboards and clinician review.

### 2.4 Deployment

- **Netlify:** Used to deploy and host the frontend application with continuous integration and automatic build on Git updates.
- **Render:** Used to deploy the backend server and manage API availability.

## 3 Set-Up Information

### 3.1 Repository

The source code for this project is publicly available on GitHub:

- **Repository URL:** https://github.com/parishanizam/TeleHealth

### 3.2 Cloning the Repository

To get started, clone the repository to your local machine using the following command:

```
git clone https://github.com/parishanizam/TeleHealth.git
```

### 3.3 Running the Application Locally

#### 3.3.1 Backend

Navigate to the backend directory and install dependencies:

```
cd TeleHealth/backend
npm install
npm start
```

#### 3.3.2 Frontend

In a separate terminal, navigate to the frontend directory and start the development server:

```
cd TeleHealth/frontend
npm install
npm run dev
```

The frontend will typically be served at http://localhost:5173, and the backend will run on http://localhost:3000.

### 3.4 Deployment Notes

Once deployed, ensure all instances of local URLs (e.g., `http://localhost:3000`) are updated to their corresponding production endpoints:

- **Backend Production URL:** https://telehealth-insights.onrender.com/
- **Frontend Production URL:** https://telethealthinsights.netlify.app/

Make sure API calls from the frontend are pointing to the Render backend URL in production.

## 4 Backend Architecture

### 4.1 Authentication Service

### 4.2 Media Processing Service

### 4.3 Question Bank Service

### 4.4 Result Storage Service

## 5 Bias Detection Feature

To ensure assessment integrity in at-home settings, the system includes a built-in **Bias Detection Module** that leverages both audio and video analysis. The goal is to identify possible unintentional guidance or intervention by parents during a child's test session.

### How It Works

- **Facial Detection:** Using **MediaPipe**, the system analyzes video frames to detect multiple faces. If more than one face is consistently present during a test, the system flags it for clinician review.

  The facial detection logic can be customized in the `facedetection.py` and

`videoProcessing.js` files located in the `media-processing-service`. These files handle MediaPipe integration, face counting logic, and video frame analysis.

- **Speech Monitoring:** Audio from assessments is transcribed in real-time using **Deepgram**'s Speech-to-Text API. The system scans the transcript for a list of predefined "cheating phrases" (e.g., *"choose this," "correct answer," "that one"*), which may indicate external prompting.

  The list of flagged "cheating phrases" can be customized in the `audioProcessing.js` file located in the `media-processing-service/helpers` directory.

- **Flagging System:** When suspicious behavior is detected—such as background speech patterns, coaching commands, or visible parental presence—a bias alert is generated. This is shown on the clinician dashboard, along with access to the original media and transcripts of these flags for manual validation.

- **Clinician Review:** Clinicians are prompted to review flagged submissions. They can override or confirm the bias detection results and leave notes for context.