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
03-31-2025	Parisha Nizam	Updated Section 1-3, and Section 5
03-31-2025	Promish Kandel	Updated Section 4
03-31-2025	Parisha Nizam and Promish Kandel	Reviewed and Edited Document
03-31-2025	Parisha Nizam	Added Section 6; Future Goals

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/src/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/src/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

TeleHealth Insights backend is built on a microservice architecture allowing the backend to be extendend by adding new services. This sections discusses each of the current services and their various API endpoints.
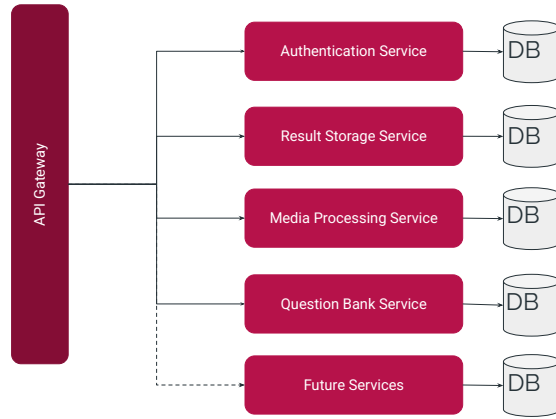
50

Figure 1: Microservice architecture of our backend

4.1 Authentication Service

This service is responsible for handling user authentication and account management for different user roles. It is accessible at:
https://telehealth-insights.onrender.com/auth.
Note that the API endpoint will change if you decide to use another deployment platform than then one stated in the tech stack.

4.1.1 Client

Handles creating an initial client account for parents to signup on.

POST /auth/client/add-client

- **Endpoint:** https://telehealth-insights.onrender.com/auth/client/add-client

- **Description:** Creates a new client account and attach client account to clinician

- **Body Parameters (JSON):**

    - `clinicianUsername`: The username of the associated clinician.
    - `firstName`: The client's first name.
    - `lastName`: The client's last name.

4.1.2 Clinician

Handles routes related to clinician registration and login.

8

POST /auth/clinician/signup

- **Endpoint:** https://telehealth-insights.onrender.com/auth/clinician/signup

- **Description:** Registers a new clinician account.

- **Body Parameters (JSON):**

  - `firstname`: The clinician's first name.
  - `lastname`: The clinician's last name.
  - `username`: The desired username (unique identifier).
  - `email`: Email for the clinician account.
  - `password`: Password for the clinician account.

POST /auth/clinician/login

- **Endpoint:** https://telehealth-insights.onrender.com/auth/clinician/login

- **Description:** Logs in an existing clinician.

- **Body Parameters (JSON):**

  - `username`: Clinician's username.
  - `password`: Clinician's password.

POST /auth/clinician/logout

- **Endpoint:** https://telehealth-insights.onrender.com/auth/clinician/logout

- **Description:** Logs out the current clinician session.

- **Body Parameters (JSON):** None

4.1.3 Parent

Handles routes related to parent registration, login, logout, and account details.

POST /auth/parent/signup

- **Endpoint:** https://telehealth-insights.onrender.com/auth/parent/signup

- **Description:** Registers a new parent account.

- **Body Parameters (JSON):**

  - `email`: Email address of the parent.
  - `username`: Unique username for the parent.
  - `password`: Password for the parent.
  - `confirmPassword`: Confirmation of the password.
  - `clinicianUsername`: The associated clinician's username.
  - `securityCode`: A code used to validate or secure the signup.

POST /auth/parent/login

- **Endpoint:** https://telehealth-insights.onrender.com/auth/parent/login

- **Description:** Logs in an existing parent account.

- **Body Parameters (JSON):**

    – `username`: Parent's username.

    – `password`: Parent's password.

POST /auth/parent/logout

- **Endpoint:** https://telehealth-insights.onrender.com/auth/parent/logout

- **Description:** Logs out the current parent session.

- **Body Parameters (JSON):** None

GET /auth/parent/account-details/`:username`

- **Endpoint:** https://telehealth-insights.onrender.com/auth/parent/account-details/:username

- **Description:** Retrieves account details for the specified parent username.

- **URL Parameters:**

    – `:username`: The username of the parent.

4.2   Media Processing Service

Handles uploading and processing of media files, as well as retrieving processed media. It is accessible at: https://telehealth-insights.onrender.com/media.

POST /media/

- **Endpoint:** https://telehealth-insights.onrender.com/media/

- **Description:** Uploads and processes a video file and associated audio files.

- **Form Data Parameters (multipart/form-data):**

    – `videoFile`: The main video file to be uploaded.

    – `audioFiles`: Up to 100 audio files.

GET /media/:parentUsername/:folderName/:assessmentId

- **Endpoint:** https://telehealth-insights.onrender.com/media/:parentUsername/:folderName/:assessmentId

- **Description:** Fetches a processed video by the specified filename convention.

- **URL Parameters:**

  - :parentUsername: The username of the parent.
  - :folderName: The folder where the media is stored.
  - :assessmentId: The ID of the assessment.

GET /media/history/:parentUsername

- **Endpoint:** https://telehealth-insights.onrender.com/media/history/:parentUsername

- **Description:** Retrieves the media processing history for the given parent username.

- **URL Parameters:**

  - :parentUsername: The username of the parent.

4.3   Question Bank Service

Provides access to a set of questions, including retrieval of all questions for a language/test type or a specific question by ID. It is accessible at: https://telehealth-insights.onrender.com/questions.

GET /questions/:language/:testType

- **Endpoint:** https://telehealth-insights.onrender.com/questions/:language/:testType

- **Description:** Retrieves all questions for a specific language and test type.

- **URL Parameters:**

  - :language: The language of the question. Curently we have English or Mandarin.
  - :testType: The type of test i.e Matching, Repetition.

GET /questions/:language/:testType/:id

- **Endpoint:** https://telehealth-insights.onrender.com/questions/:language/:testType/:id

- **Description:** Retrieves a specific question by ID for the given language and test type.

- **URL Parameters:**

  - :language: The language of the question. Curently we have English or Mandarin.
  - :testType: The type of test i.e Matching, Repetition.
  - :id: The unique ID of the question.

4.4   Result Storage Service

Manages the storage and retrieval of assessment results, including history and specific question results. It is accessible at: https://telehealth-insights.onrender.com/resultstorage.

POST /resultstorage/submit-assessment

- **Endpoint:** https://telehealth-insights.onrender.com/resultstorage/submit-assessment

- **Description:** Submits an assessment result for storage.

- **Body Parameters (JSON):**

  - parentUsername: The parent's username.
  - name: Childs name.
  - assessment_id: Id of assessment to be stored under. This is just an integer
  - questionBankId: Id of the question bank used.
  - results: This is a list of information, such as timestamps collected by the system.

GET /resultstorage/assessment-history/:username

- **Endpoint:** https://telehealth-insights.onrender.com/resultstorage/assessment-history/:username

- **Description:** Retrieves the assessment history for a particular user.

- **URL Parameters:**

  - :username: The username of the account whose assessment history is requested.

GET /resultstorage/results/:parentUsername/:assessmentId

- **Endpoint:** https://telehealth-insights.onrender.com/resultstorage/results/:parentUsername/:assessmentId

- **Description:** Retrieves results for a specific assessment.

- **URL Parameters:**

  - :parentUsername: The parent's username.
  - :assessmentId: The ID of the assessment.

GET /resultstorage/results/:parentUsername/:assessmentId/:question_id

- **Endpoint:** https://telehealth-insights.onrender.com/resultstorage/results/:parentUsername/:assessmentId/:question_id

- **Description:** Retrieves result details for a specific question within an assessment.

- **URL Parameters:**

  - :parentUsername: The parent's username.
  - :assessmentId: The ID of the assessment.
  - :question_id: The unique ID of the question.

POST /resultstorage/results/:parentUsername/:assessmentId/:question_id/bias

- **Endpoint:** https://telehealth-insights.onrender.com/resultstorage/results/:parentUsername/:assessmentId/:question_id/bias

- **Description:** Updates or saves bias modifications for a specific question result.

- **URL Parameters:**

  - :parentUsername: The parent's username.
  - :assessmentId: The assessment ID.
  - :question_id: The specific question ID.

POST /resultstorage/results/:parentUsername/:assessmentId/:question_id/mark

- **Endpoint:** https://telehealth-insights.onrender.com/resultstorage/results/:parentUsername/:assessmentId/:question_id/mark

- **Description:** Updates or saves mark modifications for a specific question result.

- **URL Parameters:**

  - :parentUsername: The parent's username.
  - :assessmentId: The assessment ID.
  - :question_id: The specific question ID.

POST /resultstorage/results/:parentUsername/:assessmentId/:question_id/notes

- **Endpoint:** https://telehealth-insights.onrender.com/resultstorage/results/:parentUsername/:assessmentId/:question_id/notes

- **Description:** Updates or saves notes for a specific question result.

- **URL Parameters:**

    - :parentUsername: The parent's username.
    - :assessmentId: The assessment ID.
    - :question_id: The specific question ID.

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.

6 Future Goals

As the platform evolves, we aim to enhance both the engagement and functionality of the system with the following planned features:

- **Motivational Progress Checkpoints:** Introduce gamified elements and milestone feedback to encourage children to complete assessments and stay motivated throughout their therapy journey. Add Motivational audio messages as checkpoints.

- **Improved Test Flow with Queued Tests:** Enable a smoother, uninterrupted assessment experience by allowing multiple tests to be queued and completed in sequence without requiring the parent to manually restart or reconfigure settings between tests.

- **Storytelling Test:** Add a new type of language assessment where children narrate a story based on visual prompts or sequences. This will help clinicians evaluate expressive language skills, creativity, and vocabulary usage in a more open-ended context.