AbleAssist

By:
Sohan Pranay
Aditi Bera

Purpose;

The purpose of this project is to develop a web-based assistant that enables disabled and normal users to use technology with the help of voice, hand signs or gestures and text. accessible to everyone. It focuses on breaking the communication gap faced by the disabled users to interact easily with digital devices.

Languages used:

The use of coding languages helped us create a very interactive and useful website for all kinds of users. We used frontend + backend languages for the same. They are:

Frontend:

1. HTML (HyperText Markup Language)
2. CSS (Cascading Style Sheets)
3. JS (JavaScript)
4. API's/Libraries (MediaPipe, Web search API, Browser API's)
5. ML (Machine Learning)

Backend:

1. JS (JavaScript) i.e. Node.js and Express.js
2. JSON (Data storage)

We have used the frontend and backend languages to make this website in the coding platform **Visual Studio Code(VSCode).**

Now, the question is: **Why did we use these languages?**
The website creation works on specific frontend and backend tools, each tool has its own purpose in fulfilling a set of user required tasks, The frontend mainly focuses on building the UI/UX (User Interface/User Experience), while the backend works as the

Brain and body of the website. Now Let us focus on where we have used the set of languages mentioned above in making our project,

1. **HTML**

   Purpose:

   Html is used to create the structure of our website(Home page and Multi-Mode page).

   Creates pages like:

   - index.html (home page)
   - sign.html (sign language page)
   - multimode.html (voice + gesture page)

   Defines:

   - Camera section
   - Voice assistant button
   - Search bars
   - Text boxes where detected letters/voice appear
     Buttons like Start, Train, Search, Gesture Book

2. **CSS (Frontend – Design and Accessibility)**

   Purpose:

   CSS controls how the website looks and feels.

   What it does in our project?

   Layout:

   - Two-column design (camera on left, output on right)

   Accessibility:

   - Large buttons
   - High-contrast colors

   Animations:

   - Collapsing camera feed
   - Opening text box automatically

   Responsive design:

   - Works on laptop, tablet, phone

3. **JavaScript (Frontend – Logic & Interaction)**

   Purpose:

   JavaScript is the brain of the front end.

What it does in our project?

- Starts camera and microphone
- Handles button clicks

Converts:

- Voice ↠ text
- Gesture ↠ letter
- Sends data to the backend, receives responses and updates the UI.

4. **MediaPipe (Frontend ML Library)**

Purpose:

MediaPipe provides real-time hand and face tracking.

What it does in our project?

Detects:

- Hand landmarks (21 points)
- Finger positions
- Gives raw data needed for gesture recognition
- Works directly in the browser

Why is MediaPipe frontend ML?

- Runs in JavaScript
- Uses camera input
- No server required for detection

5. **Machine Learning Logic (Frontend + Backend)**

Where is ML used?

Both frontend and backend

Frontend ML:

Uses MediaPipe to:

- Capture hand landmark coordinates
- Normalize data
- Sends data to backend for training / prediction

Backend ML:

- Stores training samples

Associates:

- Landmark data ↠ label (A, B, C)
- Compares live gesture with saved data
- Returns predicted letter

Frontend captures ↠ Backend learns ↠ Backend predicts

6. **Node. js (Backend – Runtime)**

Purpose:

Node.js lets you run JavaScript on the server.

What it does in our project?

Runs backend files:

- server.js
- routes/

Handles API requests:

- Save training data
- Predict gesture
- Process search queries

7. **Express. js (Backend – Framework)**

Purpose:

Express makes backend development simple and structured.

What it does in our project?

Creates APIs like:

- /train
- /predict
- /search

Handles requests from frontend

Sends JSON responses

8. **JSON (Data Exchange Format)**

Purpose:

JSON is used to send data between frontend and backend.

What it does in our project?

Sends:

- Hand landmark arrays
- Labels (A, B, Space)

- Search queries

Receives:

- Predicted letters
- URLs
- Status messages

9. **Browser APIs (Built-in Web Technologies)**

Purpose:

Browser APIs allow access to hardware.

Used APIs in your project

- getUserMedia() ↠ camera and microphone
- SpeechRecognition ↠ voice to text
- Window.open() ↠ open websites
- LocalStorage ↠ temporary saving

Steps on how we train our model:

Our model has a gesture book in the Multimode page, where we have placed an easily understandable sign image for each alphabet. Each alphabet has its corresponding sign gesture which makes it easy for disabled users who have no idea about the sign language gestures.

Firstly, we turn the training mode ON which makes it save all the gestures we show on the camera. We trained the video using live hand gestures. Data collected from the camera is converted into landmark points by showing the hand signs and simultaneously selecting the alphabet corresponding to what the hand sign represents to store its data.

It stores this data in the backend of our website where it is readily accessible by the user whenever needed and the user does not have the need to train the model again for the same.

CONCLUSION:

Finally, we used two applications in making this project into a live working website **Netlify** + **Render**. In Render, we push the GitHub repository of the

project and navigate the backend to it. The generated backend link is then edited in our main “index.html” webpage file which connects the front end and backend, then we push the same GitHub repository into the Netlify to make a complete working website link.
So, our website is basically made to easily access digital technology using voice, handsigns, text etc for the disabled or normal users. We have tried not to differentiate between the kinds of users and created an attractive and useful website for all.

“Accessibility is not a feature, it is a right.”