

SignBridge Project Rollback Document



Project Title:

SignBridge – Making Gestures Accessible





Objective:

To build a cutting-edge Chrome browser extension that translates sign language into text and optionally voice in real-time during video conferencing. Designed for use with platforms like Google Meet, Zoom Web, and Microsoft Teams Web, SignBridge enhances accessibility by enabling seamless communication between sign language users and others.



Core Features:

-  **Live Gesture Capture:** Access user's webcam to capture real-time hand gestures.
 - **Gesture Recognition:** Use **MediaPipe** or **TensorFlow.js** to identify sign language gestures.
 - **Real-Time Translation Overlay:** Display interpreted text over video call screens.
 - **Text-to-Speech (TTS):** Optionally convert translated text to speech using native browser APIs.
 - **Cross-Platform Support:** Integrates with Google Meet, Zoom Web, Microsoft Teams Web.
 -  **One-Click Installation:** Lightweight Chrome extension with intuitive UI.
-



Project Architecture

```
chrome-extension/  
├─ manifest.json      → Extension config and permissions  
├─ content.js         → Core logic: webcam stream, gesture recognition,  
overlay              overlay  
├─ popup.html/.js     → Extension UI: start/stop buttons, model toggle  
├─ model/             → ML model files (MediaPipe/TF.js)  
├─ tts.js             → Handles browser-based TTS  
├─ overlay.js         → On-screen overlay rendering logic  
└─ styles.css         → UI and overlay styling
```



Timeline: 13 July – 17 August

Team & Roles:

Arjit – Core Developer / Systems Engineer

- Setting up Chrome Extension environment.
- Managing webcam streams and overlay logic.
- Building and integrating Text-to-Speech using browser APIs.
- Modularizing code (content.js into reusable modules).
- Ensuring cross-platform compatibility (Chrome, Edge).
- Packaging and preparing final submission.

Apoorv – Machine Learning Engineer

- Researching and integrating **MediaPipe Hands** or **TensorFlow.js** models.
- Building and refining gesture recognition pipeline.
- Ensuring model robustness across different lighting and hand conditions.
- Optimizing performance for real-time predictions.
- Handling model edge cases and false positives.

Ishita – UI/UX Designer & Brand Strategist

- Designing and developing `popup.html` UI with responsive styling.
- Creating engaging icons, controls, and theme styling.
- Crafting the feedback interface and landing page design.
- Assisting in final presentation and pitch deck designs.

Sree – QA, Documentation, and Design Assistant

- Structuring and maintaining `manifest.json` with proper permissions.
- Styling overlays, popup interfaces, and mobile responsiveness.
- Performing extensive testing on various platforms.
- Writing technical documentation (README, walkthrough, permissions).
- Contributing to presentation polish and feature testing logs.



17 Day-wise Plan with Role-specific Tasks

July 13 – Planning & Kickoff

- All: Set goals, clarify roles, share references, and finalize architecture.

July 14 – Project Setup

- **Arjit:** Initialize folder structure, build webcam prototype in `content.js`
- **Sree:** Setup `manifest.json`, ensure correct permissions and folder logic

- **Apoorv:** Test MediaPipe Hands functionality
- **Ishita:** Sketch `popup.html` wireframe and layout plan

July 15 – Gesture Recognition Core

- **Apoorv:** Integrate MediaPipe with live webcam in `content.js`
- **Arjit:** Sync MediaPipe outputs with UI overlay placeholders

July 16 – Overlay + Text-to-Speech

- **Arjit:** Create `tts.js`, integrate TTS into recognition loop
- **Ishita:** Design on-screen text overlay components
- **Sree:** Add styling for overlays and placement positions

July 17 – Modular Refactor

- **Arjit:** Break `content.js` into independent modules (TTS, overlay, recognition)
- **Apoorv:** Abstract gesture detection logic into its own module for reusability

July 18 – UI Polish

- **Ishita:** Build final `popup.html`, add controls for starting/stopping gesture tracking
- **Sree:** Enhance UI/UX with transitions, styling fixes

July 19 – Extension Packaging

- **Sree:** Finalize `manifest.json`, test extension packaging
- **Arjit:** Add `background.js` (if needed), resolve load-time bugs

July 20 – Bug Fixing Sprint

- **All:** Test on Google Meet, Zoom Web, Teams Web
- **Arjit:** Fix bugs with overlays and permission issues
- **Apoorv:** Address edge case inaccuracies in gesture detection
- **Sree:** Log recurring bugs and prepare test cases

July 21 – Cross-Browser Testing

- **Arjit:** Chrome, Edge testing + fix device-specific issues
- **Ishita:** Brave browser interface checks
- **Sree:** Compatibility report documentation

July 22 – Feature Freeze

- **All:** Stop core feature changes. Lock scope. Polish only.

July 23-25 – Landing Page Development

- **Arjit:** Build responsive static site with extension redirect
- **Ishita:** Design aesthetics (fonts, icons, flow)

- **Sree:** Ensure mobile responsiveness, form styling

July 26-30 – Documentation & Pitch Prep

- **Sree:** Write README, technical walkthrough
- **Ishita:** Build pitch deck, visual assets
- **Arjit:** Record final demo video for GitHub and store

Aug 1-5 – User Testing Phase

- All: Share build for feedback
- **Arjit:** Tune TTS parameters, responsiveness
- **Apoorv:** Retrain or fine-tune model
- **Sree:** Gather UI feedback, report adjustments

Aug 6-10 – Submission Finalization

- **Arjit:** Final packaging for Web Store
- **Sree:** Double-check permission rules, documentation

Aug 11-16 – Final Polish

- All: Polish UI, transitions, bugs
- **Arjit:** Backup demo in case of failure

Aug 17 – 🏆 Submission Day

📖 Summary Table

Name	Primary Tasks	Load
Arjit	Core logic, webcam, TTS, modular refactor, packaging, testing	🏆 High
Apoorv	ML model integration, accuracy optimization, modular abstraction	🏆 High
Ishita	Popup UI/UX, overlay elements, pitch design, branding	🏈 Medium
Sree	Styling, manifest setup, testing, documentation, presentation support	🎨 Light

🧱 Bonus Features (Stretch Goals)

- 🎤 Transcription log of all signs
 - 🗣️ Multi-language support for TTS
 - 🗣️ Whisper Mode (low-volume speech or text-only mode)
 - 📷 Analytics for gesture recognition and frequency
-

Final Deliverables

- Fully functional Chrome Extension
- Public Landing Page with Web Store link
- Organized GitHub Repository with demo
- README, walkthrough, and pitch deck
- Polished demo video

Let's Build SignBridge Together! 