Hazard Analysis
Software Engineering

Team 13, Speech Buddies
Mazen Youssef
Rawan Mahdi
Luna Aljammal
Kelvin Yu

Table 1: Revision History

Date	**Developer(s)**	**Change**
October 7, 2025	Kelvin Yu	Added Sections 1-4
October 7, 2025	Mazen Youssef	Added Sections 5-7 (excl. fmea table)
October 8, 2025	Kelvin Yu	Added FMEA Table
...	...	...

Contents

1 Introduction

In the context of VoiceBridge, a hazard is any condition or event that could lead to harm, system failure, or loss of functionality during the collection, processing, and communication of speech data. The purpose of this analysis is to identify, evaluate, and propose mitigations for potential hazards before and during the system's lifecycle.

2 Scope and Purpose of Hazard Analysis

Our hazard analysis identifies potential risks that could compromise the VoiceBridge system throughout its development lifecycle and operational use.

Potential losses that could result from identified hazards include:

- Loss of user trust due to poor accuracy, delayed feedback, or confusing interactions.
- Loss of confidentiality if sensitive voice data, transcripts, or credentials are leaked.
- Loss of availability caused by hardware, software, or network failures that make the system unusable.
- Loss of integrity if documents or transcripts are modified, duplicated, or misinterpreted.

This scope ensures that both technical and human-interaction factors are considered when evaluating safety and usability risks.

3 System Boundaries and Components

VoiceBridge is a multi-component software system designed to convert and interpret speech reliably while protecting user data.

The main components analyzed for hazards are:

- Input Devices: Microphones and recording interfaces used to capture audio.
- ML Model Pipeline: The models responsible for phoneme recognition, intent detection, and command processing.
- Database/Storage: Secure repositories for user data, transcriptions, and system logs.
- Communication Layer: Real-time streaming, API connections, and message handling.

- User Interface: The front-end application where users interact and receive feedback.
- Third-Party Libraries and Services: External dependencies for authentication, hosting, and model frameworks.

Each of these components forms a boundary for potential hazards and has been evaluated for risks such as data loss, model inaccuracy, or accessibility failures.

4 Critical Assumptions

The following assumptions define the environment in which VoiceBridge operates and help focus this analysis on realistic and manageable risks:

- User devices and microphones are assumed to function normally, though temporary access loss or permission issues may occur and must be handled gracefully.
- Internet and cloud services are assumed to be available most of the time, but brief outages may happen. VoiceBridge must recover automatically.
- Users may provide imperfect audio (e.g., background noise or unclear pronunciation), and the system must tolerate and adapt to such inputs.
- Data storage systems and APIs are protected by authentication and encryption, reducing but not removing data-breach risks.
- Third-party and open-source libraries are assumed to be maintained and free from critical vulnerabilities, though version control and verification are required.

5 Failure Mode and Effect Analysis

5.1 Hazards Out of Scope

The following systems are important for VoiceBridge to work properly, but are not fully under our control. We can reduce their risks but not remove them entirely:

- **User devices and operating systems**: Microphones, drivers, or the computer's audio stack may fail or limit access. We mitigate via device detection and clear user prompts.
- **Cloud services and internet connection**: Model or network outages may affect availability. We use retry and queueing to reduce disruption.
- **Third-party services (e.g., login, hosting)**: External authentication or model-serving infrastructure may go down. The app should rely on cached state and switch to a safe, reduced-functionality mode.

5.2 Failure Modes and Effects Table

Component	What Can Go Wrong	What Happens	Why It Happens	How We Reduce the Risk	Ref.
Input Devices	Wrong mic chosen / permissions removed	No audio; session unusable	OS switched input; user denied access	Auto-detect device changes; show clear error messages	IR1, IMR1
Input Devices	Too much background noise	Poor accuracy / wrong commands	Loud space; gain too high	Noise suppression; confirm low-confidence results; keep use to medium–low background environments	IR2, IMR2
ML Model	Model doesn't adapt to accent or changes	Accuracy slowly drops; user loses trust	Low-quality datasets; poor fine-tuning	Provide re-training when accuracy drops below a threshold	IR3, ADR1
Communication	Weak or lost network	Delays; partial results	Poor Wi-Fi; server timeout	Show offline message	IR4, IMR3
Storage	Private data leaked	Privacy breach	Wrong permissions	Encrypt data; limit access	PRR1, ACR1
Auth / Access	Unauthorized entry	Data exposure; impersonation	Stolen credentials	MFA for admin; session limits	ACR2, ACR3, PRR2
Front-end Interface	Confirmation not clear or accessible	User acts by mistake	Poor contrast; missing cues	Visual + audio feedback; clear icons	IR5, IMR5, ACR1

Table 2: Hazard Identification and Risk Mitigation Table for VoiceBridge System

6 Safety and Security Requirements

We group VoiceBridge's requirements into five categories—**Access (ACR)**, **Integrity (IR)**, **Privacy (PRR)**, **Audit (ADR)**, and **Immunity (IMR)**—to match our hazard analysis. Each item connects back to one or more risks in Section 5.

6.1 Access Requirements

ACR1: Important actions (like sending data or external messages) must use a clear, two-step confirmation to prevent mistakes. *(links to IR5/IMR7 — unclear confirmation)*

ACR2: Access to stored audio or transcripts is restricted by role-based permissions so only authorized users can view or edit sensitive data. *(links to Storage — Private data leaked)*

ACR3: Multi-factor authentication (MFA) is required for all administrator or privileged logins to prevent unauthorized entry. *(links to Auth/Access — Unauthorized entry)*

6.2 Integrity Requirements

IR1: VoiceBridge auto-detects microphone devices and alerts users if permissions are missing, offering a quick way to select a new device. *(links to Input Devices — Wrong mic chosen)*

IR2: Noise suppression must run before recognition; when confidence is low, users must confirm results. *(links to Input Devices — Too much background noise)*

IR3: The system tracks model accuracy and prompts retraining or calibration when accuracy drops below a set level. *(links to ML Model — Doesn't adapt to accent or changes)*

IR4: If the connection is lost, the app shows an offline message, pauses risky actions, and retries once the network returns. *(links to Communication — Weak or lost network)*

IR5: Confirmation screens and the overall user interface must be clear, accessible, and perceivable by text, audio, and assistive tools. *(links to Front-end Interface — Confirmation not clear)*

6.3 Privacy Requirements

PRR1: All stored data is encrypted with restricted access to protect privacy. *(links to Storage — Private data leaked)*

PRR2: Logs never store personal or raw speech data; only anonymized and redacted information is kept. *(supports IR2, PRR2)*

PRR3: Sessions expire automatically and are revoked if suspicious activity is detected. *(extends ACR3 — unauthorized access)*

PRR4: Users can request deletion of stored data; the system removes it after a defined time. *(future privacy maintenance)*

6.4 Audit Requirements

ADR1: The system records recognition accuracy over time and flags when results decline. *(links to ML Model — Accuracy drops)*

ADR2: Keep structured, redacted logs for reviews and debugging. *(links to Storage — Sensitive logging)*

ADR3: Create a Software Bill of Materials (SBOM) and block unsafe dependencies. *(links to library risk, supports IMR6)*

ADR4: Provide regular privacy reports showing data deletion and retention. *(future extension)*

6.5 Immunity Requirements

IMR1: The app recovers smoothly from device or permission changes without restarting. *(links to Input Devices — Wrong mic chosen)*

IMR2: Maintain good recognition accuracy even with background noise at normal levels. *(links to Input Devices — Too much noise)*

IMR3: The app retries failed network actions and cancels incomplete ones to handle weak connections gracefully. *(links to Communication — Weak or lost network)*

IMR4: Detect and drop repeated or expired requests to avoid replay attacks. *(future enhancement)*

IMR5: Ensure all confirmations can be seen or heard, including through screen readers. *(links to Front-end Interface — not perceivable)*

7 Roadmap

To show clear scope management, the lists below specify which requirements are included in the current VoiceBridge capstone build and which will be developed later.

7.1 Included in Current Capstone Delivery

Included: ACR1, IR1–IR2, IR4, PRR1–PRR3, ADR2, IMR1, IMR2–IMR3, IMR5

Why these:

- They target immediate, high-impact risks—audio input issues, background noise, privacy of stored data, and network reliability.
- They are practical within the capstone timeframe and provide clear, measurable improvements in usability, accuracy, and data protection.

7.2 Deferred to Future Work

Deferred: ACR2–ACR3 (role-based data permissions and multi-factor admin logins); IR3, IR5 (model accuracy tracking and full accessibility review); PRR4 (user-initiated data deletion and long-term retention policies); ADR1, ADR3–ADR4 (accuracy trend monitoring, dependency tracking, and privacy reports); IMR4 (replay-attack prevention and enhanced request validation)

Why deferred:

- These involve advanced monitoring, security infrastructure, and compliance features that require more time and integration than the current scope allows.
- They remain documented for follow-up implementation in future iterations or releases.

Appendix — Reflection

The purpose of reflection questions is to give you a chance to assess your own learning and that of your group as a whole, and to find ways to improve in the future. Reflection is an important part of the learning process. Reflection is also an essential component of a successful software development process.

Reflections are most interesting and useful when they're honest, even if the stories they tell are imperfect. You will be marked based on your depth of thought and analysis, and not based on the content of the reflections themselves. Thus, for full marks we encourage you to answer openly and honestly and to avoid simply writing "what you think the evaluator wants to hear."

Please answer the following questions. Some questions can be answered on the team level, but where appropriate, each team member should write their own response:

1. **What went well while writing this deliverable?**
   We split the work effectively across the team (writing, tables/figures, checks), agreed on clear owners for each section, and used short review cycles to keep everything consistent. This let us move quickly without stepping on each other's toes and made integration straightforward.

2. **What pain points did you experience, and how did you resolve them?**
   The hardest part was *identifying hazards*, brainstorming how they could affect our project, and then *translating those into concrete additional requirements*. We initially struggled to scope severity/likelihood consistently. We resolved this by aligning on a simple risk matrix with definitions, walking through realistic user stories, and iterating on a Failure Modes & Effects table. Each accepted hazard now maps to a mitigation and a verifiable requirement.

3. **Which risks were known before vs. discovered during this deliverable? How did the new ones come about?**
   Known before:

   - Cloud/network outages affecting availability.
   - User device/OS audio stack failures and limitations.

   Identified during the Hazard Analysis:

   - Data privacy in logs and error reports.
   - Latency spikes undermining "real-time" interaction.
   - Poor failure UX (unclear prompts/recovery paths) compounding minor faults.

   These emerged by systematically asking "what if?" at each step of the user flow, filling out the failure modes table, and tracing each failure to effects and necessary mitigations.

4. **Beyond physical harm, list at least two other risk types in software and why they matter.**

   - **Privacy & Security Risk:** Breaches or over-collection can expose sensitive data, create legal liability, and erode user trust.
   - **Reliability/Availability Risk:** Downtime or unstable behavior blocks users at critical moments; SLO breaches damage credibility.
   - **Compliance/Regulatory Risk:** Violations (e.g., data residency, consent) can lead to fines and forced rework.