Design Document: Mute Streamer Overload

Version: 0.1

Date: May 27, 2025

Author: Natus Anima

---

1. Introduction

Mute Streamer Overload is an application designed to empower mute streamers to communicate effectively with their audience while live-streaming video games. It provides a dynamic text display overlay, integrates with Twitch chat, and offers AI-powered voice narration, enabling a rich and interactive streaming experience for both the streamer and their viewers.

2. Goals

- To facilitate seamless communication for mute streamers.
- To enhance viewer engagement through accessible communication methods.
- To provide customizable and flexible communication tools.
- To integrate with existing streaming platforms (specifically Twitch).
- To create a user-friendly and intuitive interface.

3. Target Audience

- Mute streamers who wish to communicate with their audience via text and voice during live gameplay.
- Viewers of mute streamers who desire clearer and more immediate communication.

4. Features

4.1. Core Communication

- **Text Input:** A primary input box for the streamer to type messages.
- **Dynamic Text Display Overlay:**
  - A secondary, transparent window that overlays the game.
  - Displays the typed message a few words at a time, gradually revealing the full message.
  - Customizable text speed/reveal rate.
- **Twitch Chat Integration:** Duplicates the displayed message directly into the streamer's Twitch chat using their username.

- **AI Voice Narration:** Converts the typed text into speech using an AI voice, played through the streamer's audio output (which is then picked up by the stream).

4.2. Customization & Controls

- **Main Control Window:** A dedicated window for all application settings and controls.
- **Text Display Customization:**
    - **Font Selection:** Choose from a variety of installed fonts.
    - **Font Size:** Adjustable font size for readability.
    - **Font Color:** Customizable text color.
    - **Background Transparency/Color (Overlay):** Option to adjust the transparency of the overlay window, or set a solid background color.
- **Text Speed Control:** Slider or input field to control the speed at which words appear on the overlay.
- **AI Voice Selection (Stretch Goal/Future Iteration):** If multiple AI voices are available, allow selection.
- **Volume Control (AI Voice):** Adjustable volume for the AI narration.
- **Toggle Features:** Buttons to easily enable/disable:
    - Dynamic Text Overlay
    - Twitch Chat Integration
    - AI Voice Narration

4.3. User Experience

- **Intuitive Interface:** Clear labeling and logical grouping of controls.
- **Hotkey Support:** Customizable hotkeys for common actions (e.g., send message, clear text box, toggle overlay).
- **Preview Functionality:** A way to preview how the text will appear on the overlay without sending it live.

5. Technical Architecture (High-Level)

- **Application Framework:** (e.g., Electron for cross-platform desktop app, C# .NET for Windows, Python with a GUI library like PyQt/Tkinter) – *Recommendation based on ease of development and features: Electron or C# .NET for robust UI and system integration.*
- **Overlay Technology:**

    - Windows: Transparent window with WS_EX_LAYERED and SetLayeredWindowAttributes or a game overlay API (if applicable).
    - Cross-platform (Electron): Chromium's transparent option for windows.
- **Twitch Integration:** Twitch API (OAuth for authentication, Chat IRC for sending messages).
- **Text-to-Speech (TTS):**
    - **Local TTS:** Utilizing OS-level TTS engines (e.g., SAPI for Windows, AVFoundation for macOS). This is a good starting point for simplicity.
    - **Cloud-based TTS API:** (e.g., Google Cloud Text-to-Speech, Amazon Polly, Microsoft Azure Cognitive Services Speech) for higher quality and more diverse voices. This would require an API key and internet access. *Initial recommendation: Start with local TTS for offline functionality, and then explore cloud APIs as an enhancement.*
- **Configuration Management:** Saving and loading user preferences (font, size, speed, API keys, etc.).

6. User Interface (UI) Wireframes (Conceptual)

6.1. Main Control Window

- **Layout:** Two main sections: "Text Input & Send" and "Settings & Controls."
- **Text Input Area:**
    - Large multiline text box.
    - "Send Message" button.
    - "Clear" button.
- **Display Settings (Group):**
    - Font dropdown.
    - Font size slider/input.
    - Font color picker.
    - Text speed slider.
    - Overlay transparency slider.
- **Output Settings (Group):**
    - "Enable Twitch Chat" checkbox.
    - "Enable AI Voice" checkbox.
    - AI Voice volume slider.

    - (Optional: AI Voice selection dropdown).

- **Hotkey Settings (Group):**
    - List of customizable actions and their assigned hotkeys.
- **Preview Area:** Small section showing a sample of how the text will look.

6.2. Transparent Overlay Window

- **Appearance:** Minimalist, transparent background.
- **Content:** Only displays the dynamic text.
- **Position:** User should be able to drag and drop this window to position it anywhere on their screen.

7. Workflow

1. **Streamer opens Mute Streamer Overload.**
2. **Configures settings:** Adjusts font, size, speed, enables/disables Twitch chat/AI voice as desired.
3. **Starts their game and streaming software.**
4. **Positions the transparent overlay window** on their screen where it's visible to viewers but doesn't obstruct gameplay.
5. **Types a message** into the main control window's text input box.
6. **Clicks "Send Message" (or uses hotkey).**
7. **Application Logic:**
    - Splits the message into words/phrases.
    - Displays words sequentially on the transparent overlay, adhering to the set speed.
    - Sends the full message to Twitch chat via the streamer's account.
    - Sends the full message to the AI TTS engine, playing the audio through the selected output.
8. **Overlay completes message display.**
9. **Streamer continues to type and send new messages.**

8. Future Enhancements (Roadmap)

- **Emote Support:** Displaying Twitch emotes on the overlay.
- **Canned Responses/Quick Phrases:** Pre-configured common phrases for quick selection.

- **Customizable AI Voice Profiles:** Allow users to import or train their own AI voice (highly complex).
- **Multiple Overlay Profiles:** Save different overlay configurations for different games/stream types.
- **Discord Integration:** Send messages to a Discord channel.
- **OBS/Streamlabs OBS Integration:** Direct integration for source control (more robust than just an overlay window).
- **Theming/Skins:** Customizable look for the main control window.
- **Error Handling and User Feedback:** Clear messages for API connection issues, invalid inputs, etc.

9. Technical Considerations & Challenges

- **Overlay Stability:** Ensuring the transparent window remains on top of games, especially full-screen games, without causing performance issues.
- **Twitch API Rate Limits:** Managing API calls to avoid hitting rate limits.
- **TTS Quality:** Ensuring the AI voice is clear, natural-sounding, and understandable.
- **Offline Functionality:** If using local TTS, ensuring the app works without an internet connection. If using cloud TTS, handling offline scenarios gracefully.
- **Accessibility:** Ensuring the app is usable by streamers with various needs (e.g., keyboard-only navigation).
- **Cross-Platform Compatibility:** If targeting multiple operating systems, managing platform-specific differences.

10. Development Milestones

- **Phase 1: Core Functionality (MVP)**
  - Basic text input and display on a transparent window.
  - Manual text speed control.
  - Basic Twitch chat integration.
  - Local OS-based TTS integration.
- **Phase 2: Customization & Enhancements**
  - Font, size, color customization.
  - Overlay transparency.
  - UI for settings.
  - Hotkey support.

- **Phase 3: Robustness & Polish**
    - Error handling.
    - User profile saving.
    - Performance optimizations.
    - User testing and bug fixing.
- **Phase 4: Future Enhancements (Iterative)**
    - Cloud TTS integration.
    - Canned responses.
    - More advanced overlay features.

Priority	**Category**	**Feature/Task**	**Description**	**Notes**
1 (Critical)	Core Communication	Text Input Box	A user interface element where the streamer types their message.	Essential for any communication.
1 (Critical)	Core Communication	Transparent Overlay Window (Basic)	A non-interactive, always-on-top window that can display text.	The fundamental output mechanism for viewers.
1 (Critical)	Core Communication	Basic Text Display on Overlay	Display the full text from the input box onto the transparent window.	Initial step, without dynamic reveal yet.
1 (Critical)	Core Communication	"Send Message" Button/Action	A button or hotkey to trigger sending the typed message.	Triggers all output actions.
1 (Critical)	Output - Voice	Basic AI Voice Narration (Local TTS)	Convert typed text to speech using an OS-	Easier to implement than cloud TTS.

Priority	Category	Feature/Task	Description	Notes
			level text-to-speech engine.	
1 (Critical)	Output - Chat	Basic Twitch Chat Integration	Send the typed message to the streamer's Twitch chat using their username.	Requires Twitch API authentication.
2 (High)	Core Communication	Dynamic Text Reveal on Overlay	Display text a few words at a time, gradually revealing the message.	Improves readability and engagement.
2 (High)	Customization	Text Speed Control (Overlay)	Slider/input to adjust rate of text appearance.	Key for dynamic display customization.
2 (High)	User Interface	Main Control Window Layout	Design and implement the primary control panel for streamers.	Central hub for interaction.
2 (High)	Customization	Font Selection & Size (Overlay)	Choose font family and size for text.	Improves aesthetics and readability.
2 (High)	Customization	Font Color (Overlay)	Choose color for overlay text.	Enhances visibility.
2 (High)	Usability	Basic Hotkey Support	Hotkey for "Send Message" and possibly "Clear Input."	Improves workflow.
3 (Medium)	Customization	Overlay Transparency/Color Control	Adjust overlay transparency or	Better game/stream

Priority	Category	Feature/Task	Description	Notes
			background color.	layout integration.
3 (Medium)	Output - Voice	AI Voice Volume Control	Adjust narration volume.	Important for audio balance.
3 (Medium)	Usability	"Clear Input Box" Button/Hotkey	Quick way to clear the input.	Convenience feature.
3 (Medium)	System	Save/Load User Settings	Persist user preferences across sessions.	Crucial for usability.
4 (Low)	Output - Chat	Twitch Authentication Persistence	Securely remember login to avoid repeated authentication.	Improves setup time.
4 (Low)	User Interface	Preview Functionality (Overlay)	Small display area to preview text display.	Lets streamer tweak without going live.
4 (Low)	Output - Voice	AI Voice Selection (Cloud TTS)	Choose different voices/providers if using cloud TTS.	Requires API key management.
5 (Enhancement)	Core Communication	Emote Support (Overlay & Chat)	Display Twitch emotes properly.	Adds personality.
5 (Enhancement)	Usability	Canned Responses/Quick Phrases	Pre-defined common messages.	Speeds up repetitive communication.
5 (Enhancement)	Usability	Drag-to-Position Overlay Window	Allow users to reposition the overlay via drag.	Flexible for layout.

Priority	Category	Feature/Task	Description	Notes
5 (Enhancement)	Advanced Output	OBS/Streamlabs Source Integration	Provide overlay as a browser source or plugin.	Robust, integrated solution.
5 (Enhancement)	Advanced Output	Discord Integration	Send messages to a Discord channel.	Extends communication.
6 (Future/Complex)	Advanced Customization	Multiple Overlay Profiles	Save/load different overlay settings.	Useful for multi-game streamers.
6 (Future/Complex)	Advanced Output	Customizable AI Voice Profiles	Train/import unique voices.	Highly complex (ML involved).
6 (Future/Complex)	User Interface	Theming/Skins for Main Window	Custom visual themes for the control panel.	Aesthetic only, low priority.