**Title Page – SRS Project Name**

Team Members: Cayleigh Leishman

Description of project:

Clario is a web-based platform designed to provide live transcription of lectures for students, ensuring accessibility for deaf or hard-of-hearing students. Transcribers provide live text transcription, students can view it in real time, and admins manage users, lectures, and live sessions. The platform emphasizes real-time performance, multi-admin support, and historical access to finalized transcripts.

**Section 1:**

Technologies Used:  
**Frontend:**

* **SvelteKit:** Provides a reactive, component-based framework for building a fast, interactive web application.
* **HTML5, CSS3, JavaScript/TypeScript:** Standard web technologies for layout, styling, and client-side functionality.
* **Tailwind CSS / Optional UI Libraries:** For rapid styling and consistent UI components.

**Backend / Server:**

* **Supabase:** Handles authentication, database management (PostgreSQL), and real-time data updates.
* **Node.js (via SvelteKit endpoints):** Serves API routes, handles server-side logic, and connects frontend with the database.

**Database & Data Management:**

* **PostgreSQL (via Supabase):** Stores user profiles, lectures, enrollment, transcriptions, session data, and notes.
* **Realtime Database / Supabase Realtime:** Supports live updates for transcription chunks in WebRTC sessions.

**Real-Time Communication:**

* **WebRTC:** Enables secure, low-latency streaming of audio and video from lecture sessions to students without sharing transcriber audio/video.
* **Session Tokens & Authentication:** Ensures secure, role-based access to live sessions for students, transcribers, and admins.

**Additional Tools & Libraries:**

* **UUID:** Generates unique IDs for database records and session tokens.
* **Vite:** Optimizes the frontend development workflow.
* **Git / GitHub:** Version control for collaborative development and deployment tracking.
* **Postman / HTTPie:** Testing APIs and database endpoints during development.

**Accessibility & UX Tools:**

* Color contrast and theme selection for visually impaired users
* Keyboard navigation and screen reader support planning

**Problem Statement:**  
While attending BYU-I, I experienced challenges with existing transcription services. Sometimes, transcribers had their audio on without realizing it, which compromised their privacy, and as a deaf student, I couldn’t detect this. Additionally, transcribers had to remember to manually send transcripts to students after class, creating extra workload and potential delays. Students, meanwhile, struggled with disorganized or incomplete notes, making it difficult to follow lectures effectively. Existing solutions did not provide a unified platform that addressed these challenges for both students and transcribers.

**Solution:**

Clario solves these problems by providing an integrated platform that combines **real-time transcription, audio/video streaming, and note-taking** in a single web application. For students, it delivers accurate, structured transcriptions and allows hands-on note-taking without the need for additional software. For transcribers, Clario ensures privacy by streaming audio one-way only to students, eliminating the risk of accidentally broadcasting their own audio or video. Additionally, all transcriptions are automatically saved and linked to students’ accounts, removing the need for transcribers to manually distribute notes after class. This unified platform streamlines lecture participation, protects transcriber privacy, and ensures students receive clear, organized transcripts.

**Section 2a:**

| **Requirement** | **Success Measure / Demonstration** |
| --- | --- |
| 1. Authenticate users with roles (client, transcriber, admin) | Users can log in and access functionality specific to their role (see their dashboard, edit transcription or read only, etc)  **Test:** Attempt student, transcriber, and admin logins; verify access. |
| 2. Students can join lectures via a join\_code | Students input code → automatically enrolled if not already → join WebRTC session.  **Test:** Student joins session and sees live transcription. |
| 3. Real-time transcription streaming | Live transcription appears in sequence as transcriber types.  **Test:** Compare transcriber input with student view in real time. |
| 4. Student enrollment tracking | Ensure students are enrolled in lectures via student\_enrollments.  **Test:** Enrollment record exists after join; duplicates prevented. |
| 5. Final transcription storage | After session ends, chunks merge into final\_transcriptions.  **Test:** Verify merged transcript matches all real-time chunks. |
| 6. Accessibility features | Students can adjust background and text colors for readability.  **Test:** User changes colors → interface updates correctly and is readable. |

**Section 2b:**

Stretch Goals

| **Requirement** | **Success Measure / Demonstration** |
| --- | --- |
| 1. Multi-admin support | Admins can simultaneously manage lectures, users, and sessions.  **Test:** Two or more admins perform management tasks without conflicts. |
| 2. Admin session management | Admins can assign transcribers, start/end sessions, and monitor activity.  **Test:** Admin dashboard shows accurate session status. |
| 3. Multi-transcriber support | More than one transcriber can join the same session.  **Test:** Student see combined live input from multiple transcribers. |
| 4. Accessibility features | Platform meets basic accessibility standards (e.g., screen reader support, high contrast, keyboard navigation).  **Test:** Test with screen readers, keyboard-only navigation, and color contrast tools |

**Section 2c:**   
Weekly schedule:

**Section 3:** Design Overview of the Product.

Workflow:   
The Clario workflow is designed to support both students and transcribers in a seamless, integrated environment. A typical lecture session begins when a transcriber starts a session by selecting a lecture from the platform. The system creates an active session record and generates a unique session token, which allows students to securely join the live lecture. Students enter the lecture’s join code, which automatically enrolls them if they are not already registered, and connects them to the WebRTC session.

During the session, the transcriber provides live transcription, which is sent in real time to all students, while students can simultaneously take their own notes within the platform. The transcription is streamed one-way, ensuring the transcriber’s audio and video remain private. Multiple transcribers can join the same session if needed, contributing collaboratively to the transcription. Once the session ends, the system automatically merges all transcription chunks into a final transcript stored in the database, making it immediately accessible to students. This workflow eliminates the need for transcribers to manually distribute notes, reduces setup complexity, and provides a structured, accessible learning experience for students.

Resources:  
Clario relies on a combination of software frameworks, libraries, cloud services, and developer tools to function efficiently. The **frontend** is built using **SvelteKit**, which provides a reactive framework for building fast and interactive web interfaces. Standard web technologies like **HTML5, CSS3, and JavaScript/TypeScript** handle layout, styling, and client-side logic, while **Tailwind CSS** ensures consistent and responsive UI design.

On the **backend**, Clario uses **Supabase**, which provides authentication, PostgreSQL database management, and real-time data streaming. Supabase Realtime allows live transcription chunks to be updated instantly for students. Server-side logic is handled via **SvelteKit endpoints** running on **Node.js**, connecting the frontend to the database and WebRTC sessions.

For **real-time communication**, Clario uses **WebRTC** to stream audio and video securely from the lecture to students, with transcriber audio remaining private. **Session tokens** and authentication protocols ensure secure, role-based access to all sessions.

Additional **tools and libraries** include **UUID** for unique record generation, **Vite** for frontend development optimization, and **Git/GitHub** for version control and collaboration. APIs and endpoints are tested using **Postman** or **HTTPie**. Accessibility considerations, including color contrast, theme selection, keyboard navigation, and screen reader support, are also incorporated to enhance usability.

These resources work together to create a **reliable, secure, and accessible platform** that meets the needs of both students and transcribers.

All user information and application data in Clario are stored securely in a **PostgreSQL database via Supabase**. This includes user profiles, roles, lecture details, student enrollments, real-time transcription chunks, final transcripts, and personal notes. Sensitive user data, such as authentication credentials, are managed by Supabase Auth, ensuring proper encryption and security standards. Data is structured to maintain relationships between lectures, students, and transcribers, enabling fast retrieval and accurate record-keeping. Historical data, such as final transcriptions and lecture records, are stored permanently for later access, while temporary data like active sessions and real-time chunks exist only for the duration of the session.

**Data on the Wire**

During live sessions, **WebRTC** streams one-way audio and video from the lecture to students and sends transcription chunks from transcribers to the database in real time. Data transmitted over the wire is secured using **TLS/HTTPS** for API calls and **SRTP (Secure Real-time Transport Protocol)** for WebRTC media streams. This ensures that sensitive session data, including live transcription text and session tokens, cannot be intercepted or tampered with. Only authorized students can access the live stream and transcription data based on their enrollment and session token authentication.

**Data State**

The data state in Clario follows a clear lifecycle:

1. **Pre-Session:** Lectures and student enrollments exist in the database; no active session is running.
2. **Session Start:** Transcriber creates an active session, generating a session token and setting is\_live = TRUE.
3. **Live Session:** Transcription chunks are streamed to the database in sequence, while students take notes locally. Real-time updates are pushed to all connected students via Supabase Realtime.
4. **Session End:** is\_live is set to FALSE in the active session record, and all transcription chunks are merged into a single entry in final\_transcriptions.
5. **Post-Session:** Students can access merged transcripts and personal notes, while historical session data remains stored for auditing or review.

This structured data flow ensures **data integrity, real-time synchronization, and secure storage** throughout the lifecycle of a lecture session.

**HMI/HCI/GUI**

Clario’s interface is designed to be **intuitive, accessible, and responsive**, ensuring that both students and transcribers can interact with the platform effectively.

**Student Interface:**

* **Lecture Dashboard:** Students see a list of upcoming lectures with join codes and status (live, upcoming, completed).
* **Live Lecture View:** A single-page layout where students can:
  + View **real-time transcription** in a scrolling pane.
  + Take **personal notes** in a side panel.
  + Watch **lecture video/audio** (one-way from the transcriber).
  + Adjust **accessibility settings** such as text size, color contrast, and theme (light/dark mode).
* **Final Transcripts:** After a session ends, students can access merged transcripts with search and download options.

**Transcriber Interface:**

* **Session Dashboard:** Shows lectures they are assigned to, along with active session status and student participation.
* **Live Transcription Pane:** Allows real-time transcription input without risk of broadcasting their personal audio or video.
* **Session Management Tools:** Start/end sessions, monitor student connections, and manage multiple transcribers in collaborative sessions.
* **Automated Transcript Handling:** All transcriptions are automatically stored in students’ accounts after the session.

**Admin Interface (Stretch Goal):**

* Assign transcribers to lectures.
* Monitor active sessions and student attendance.
* Access logs of completed sessions and transcripts.

**Accessibility Features:**

* Keyboard navigation for all major actions.
* Screen reader support for visually impaired users.
* Adjustable text size and color themes for readability.
* High contrast mode for low-vision users.

**Pictures / Diagrams**

**Mockup Ideas:**

1. **Student Live Lecture Screen:**
   * Left: Real-time transcription pane.
   * Right: Personal notes input panel.
   * Top: Video/audio player (one-way stream from transcriber).
   * Top-right corner: Accessibility options (text size, theme, contrast).
2. **Transcriber Live Session Screen:**
   * Central typing area for transcription input.
   * Panel showing connected students and session token info.
   * Buttons to start/end session, or add another transcriber.
3. **Data Flow Diagram:**
   * Shows flow from transcriber → realtime\_chunks table → students in live session → final\_transcriptions table.
4. **Workflow Diagram:**
   * Shows steps from lecture creation → enrollment → session start → real-time transcription → session end → transcript access.

Pictures/ Diagrams:

**Section 4:** Verification:

**Demo:**

**Testing:**

**Sources/Citation/Resources** Links: