**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.

**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) | * *Requirement:* The system shall allow users to log in as either a student or a transcriber. * *Success Measure:* Users log in and see the correct dashboard for their role. |
| 2. Students can join lectures via a join\_code | * *Requirement:* Students shall be able to enter a code to join a lecture session. * *Success Measure:* Student enters code → system confirms → student joins session. |
| 3. Real-time transcription streaming | * *Requirement:* The system shall display transcriber input in real-time to all connected students. * *Success Measure:* Text typed by the transcriber appears instantly for the student. |
| 4. Final transcription storage | * *Requirement*: System shall save a transcript once a session ends. * *Success* *Measure*: Transcript available on student dashboard post-session |
| 5. Student Note-Taking\* | * *Requirement: Students shall be able to add personal notes alongside the live transcription.* * *Success Measure: Notes are typed ad saved locally or to the student’s account; can be reviewed after the session.* |
| 6. Accessibility features | * Requirement: All major features shall be usable with keyboard only, and transcripts shall allow text resizing and sufficient contrast. * Success Measure: Students can navigate all screens without a mouse and adjust text size/contrast to read content comfortably. |

**Section 2b:**

Stretch Goals

| **Streach**  **Goals** | **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 Real-Time Notification or Reminders | 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:

| **Requirement** | **Description** | **Acceptance Criteria** | **Hours Expected** | **Due Date** |
| --- | --- | --- | --- | --- |
| 1. User Authentication & Role-Based Dashboard | Allow users to sign up and log in as Student, Transcriber, or Admin. Display correct dashboard based on role. | Users log in and are directed to the appropriate dashboard without errors. | 16 | Oct 18 |
| 2. Student Join Lecture via Join Code | Students enter a code to join a live session. | Student enters code → system confirms → student joins session successfully. | 12 | Oct 25 |
| 3. Real-Time Transcription Streaming | Transcriber input appears live to all connected students. | Text typed by transcriber appears immediately for all students in session. | 20 | Nov 1 |
| 4. Final Transcript Storage | Merge all real-time chunks into a final transcript after session ends. | Transcript stored in database and accessible in student dashboard. | 16 | Nov 15 |
| 5. Student Note-Taking | Students can add personal notes alongside live transcription. | Notes are saved and retrievable after the session. | 16 | Nov 29 |
| 6. Accessibility Features | Keyboard navigation, text resizing, color contrast adjustments. | Students can navigate without mouse, resize text, and apply high-contrast theme. | 12 | Dec 6 |
| 7. Admin Session Management (Stretch Goal) | Admins can create lectures, assign transcribers, monitor logs. | Admin dashboard shows correct session setup, assigned transcribers, and log history. | 10 |  |
| 8. Multi-Transcriber Support (Stretch Goal) | More than one transcriber can join same session. | Students see combined live input from multiple transcribers. |  |  |

I kept stretch goals blank

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

*Workflow:*

● **Scheduling (Admin):** The admin sets up lectures. They choose a transcriber for each lecture and make a join code. The transcriber sees the schedule on their dashboard. The student gets the join code to join the lecture.

● **Starting the Session (Transcriber):** The transcriber clicks on their scheduled lecture to start it. This opens the session and a secure channel for live transcription.

● **Joining the Session (Student):** The student enters the join code or clicks “Join Lecture.” If they are new, the system signs them up automatically. They are then connected to the live session.

● **Live Transcription:** The transcriber types what is said in the lecture. Students see it in real time. More than one transcriber can work together if needed.

● **Taking Notes (Student):** Students can write their own notes while reading the live transcript.

● **Privacy:** Only students see the transcriber’s words. Transcriber audio and video are private.

● **Ending the Session:** When the lecture is finished, all typed text is combined into a final transcript. Students can read it right away with their personal notes.

● **Accessibility:** Students can make text bigger, change colors, or switch between light and dark mode. They can also use only the keyboard to move around the site.

● **Admin Control:** The admin manages schedules, making it easy for transcribers and safe for students. This keeps everything organized and private.

*Resources*:

| **Component** | **Tool / Technology** | **What It Does (Simple)** |
| --- | --- | --- |
| **User Interface** | SvelteKit, HTML, CSS, JavaScript | Loads pages fast and shows updates in real time |
|  | Tailwind CSS | Keeps design consistent and nice on all devices |
| **Server & Logic** | SvelteKit + Node.js | Connects frontend to database and handles live sessions |
|  | Vite | Helps build and update website faster |
| **Development & Testing** | Git / GitHub | Tracks changes and helps teamwork |
|  | Postman / HTTPie | Tests system to make sure it works |
| **Live Session** | WebRTC | Sends lecture audio and video to students quickly |
|  | Session Tokens | Only authorized users can join |
|  | Privacy | Transcriber audio/video is private |
| **Database & Storage** | Supabase / PostgreSQL | Stores users, lectures, enrollments, live & final transcripts, notes |
|  | UUIDs | Creates unique, hard-to-guess IDs |
|  | Encryption | Keeps passwords and sensitive info safe |
| **Accessibility** | Frontend Features | Change text size, color, contrast; keyboard navigation; screen reader support |

**Data on the Wire**

During live sessions, WebRTC streams one-way audio and video from the lecture to students, providing low-latency, real-time communication. Transcription chunks from transcribers are sent to the database as they are typed, ensuring students see live updates instantly. All data transmitted over the network is secured: TLS/HTTPS protects API calls, and SRTP (Secure Real-time Transport Protocol) encrypts media streams. Session tokensenforce access control, so only authorized students can join the session and access both the live stream and transcription data. This setup ensures real-time performance, privacy, and secure data delivery.

**  
  
Data State**

**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):**

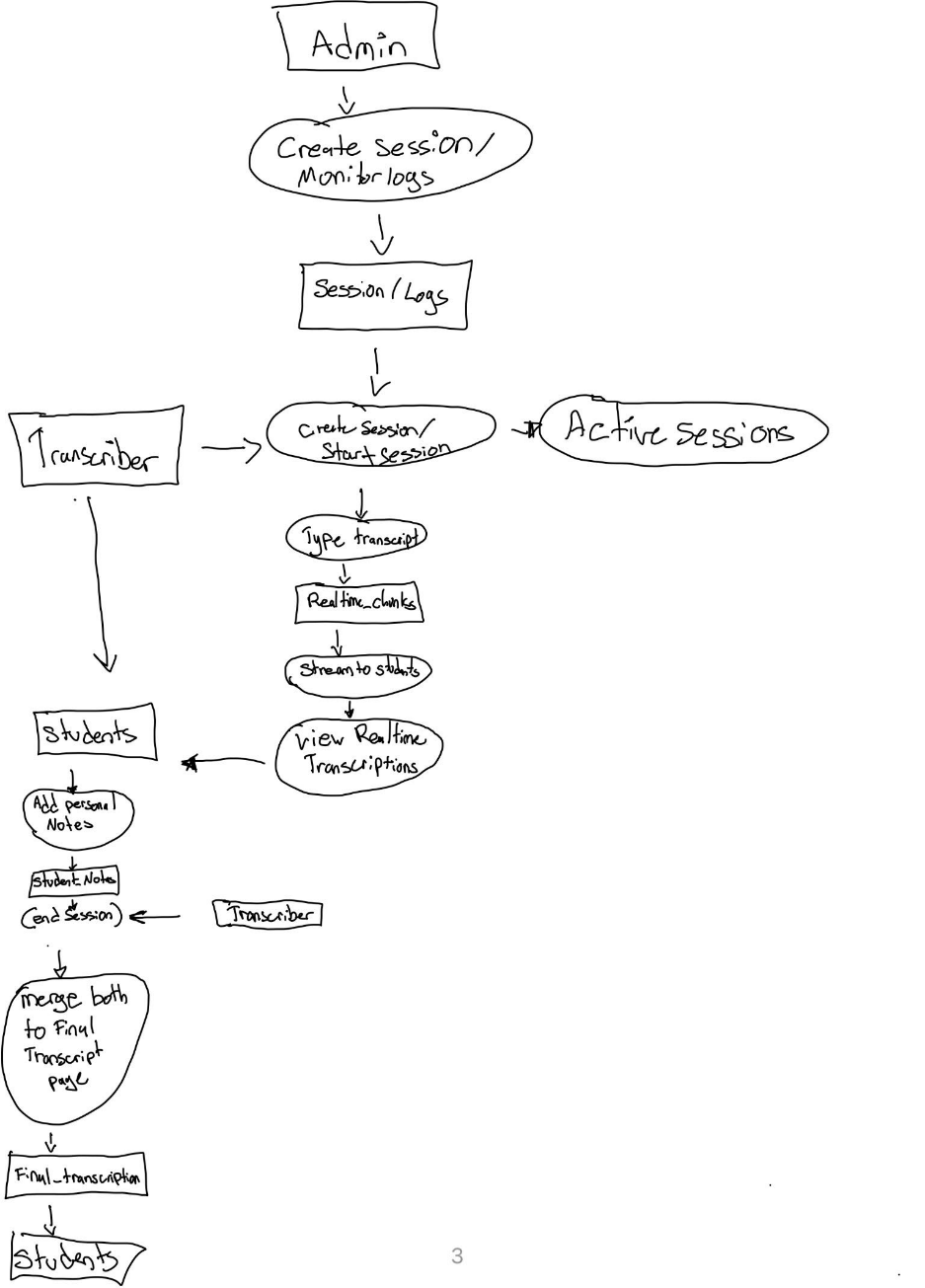
* Assign transcribers to lectures.
* Monitor active sessions and student attendance.
* Access logs of completed sessions and transcripts.

**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:**

****

:

A screenshot of a whiteboard

AI-generated content may be incorrect.

A whiteboard with text and words

AI-generated content may be incorrect.Entity Relationship Diagram:  
A diagram of a website

AI-generated content may be incorrect.

A diagram of a flowchart

AI-generated content may be incorrect.

Header and Footer:

A whiteboard with writing on it

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a whiteboard

AI-generated content may be incorrect.Transcript View (Rough idea of the layout-created using Figma Make):  
Clients View Wireframe:

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer screen

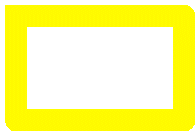
AI-generated content may be incorrect.

A screenshot of a computer screen

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.



**Section 4:** Verification:

**Demo:**

**Testing:**

**Sources/Citation/Resources** Links: