

Frontend Documentation

Authentication

Front-end (Client) establishes a secure HTTPS connection with back-end (Server) to create a token-based session. Client uses public key of server's key pair to generate a key.

Client sends raw user credentials, server hashes and encrypts before storing credentials. Server uses the same algorithms to decipher.

Library Used

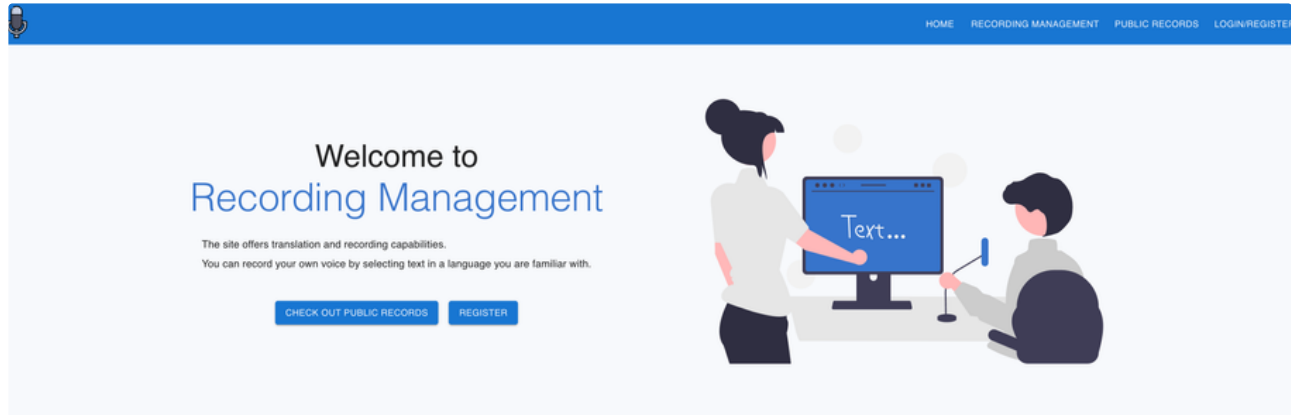
- React: the framework
- React MUI: the component library
- Axios: library to make ajax requests
- react-router: used for client side routing
- react-media-recorder: the functionality to record audio
- zustand: to share state between components without prop drilling
- TypeScript: provide type safety for JavaScript
- Vite: modern packaging tool for frontend
- ESLint: code linter
- Prettier: code formatter

Folder Structure

```
1 Web/                                # frontend project is in the web folder
2   └─ dist                           # the packaged files
3   └─ public                         # the public static files
4   └─ src/                           # the source code
5       └─ assets/                    # the asset files
6           └─ react.svg
7       └─ components/                # the React components
8           └─ Header.tsx
9           └─ PublicRecordsList.tsx
10          └─ Recorder.tsx
11          └─ TaskList.tsx
12       └─ views/                     # the views for the application
13           └─ HomeView.tsx
14           └─ LoginView.tsx
15           └─ ProfileView.tsx
16           └─ PublicRecordsView.tsx
17           └─ RecordingView.tsx
18           └─ RegisterView.tsx
19           └─ RootView.tsx
20       └─ api.ts
21       └─ App.tsx
22       └─ entity.ts
23       └─ env.d.ts
24       └─ main.tsx
25       └─ store.ts
26       └─ T&D.ts
27       └─ utils.tsx
28       └─ vite-env.d.ts
```

Frontend Pages

1. Home: welcome texts



Human Audio Recording Tool

Designed to revolutionize the way you record audio. It allows users to easily create new audio files or update existing ones, providing a seamless experience for all your recording needs. Whether you're recording a project that doesn't yet have an audio file or re-recording an existing file, our tool makes the process simple and efficient.



C-LARA Platform

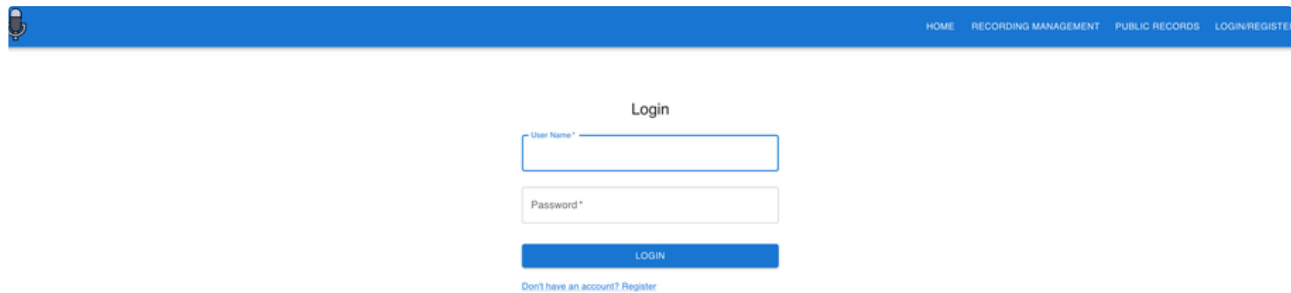
The C-LARA platform is a comprehensive solution designed to manage and facilitate the entire audio recording process. It allows users to send recording requests, manage tasks, download current content, and even delete recording tasks as needed. It is a one-stop platform for all your recording management needs.




Easy-to-use User Interface

Our tools and platforms are designed to be very user-friendly. The simple and clear interface design makes it easy for users to get started without having to spend a lot of time learning how to use it. Our goal is to allow users to focus more on their recording tasks rather than wasting time on complex operations.

2. Recording Management: the page where user can record an audio and upload to backend, and also manages his/her own audios.



3. Public Recordings: the page where the user can get access to the publicly available recordings.




HOME RECORDING MANAGEMENT PUBLIC RECORDS LOGIN/REGISTER

<input type="checkbox"/>	ID	Name	Audio	Text
<input type="checkbox"/>	1	Recording 1	<div><div>▶ 0:00 / 6:12</div><div></div><div>🔊 ⓘ</div></div>	They lived with their moth...
<input type="checkbox"/>	2	Recording 2	<div><div>▶ 0:00 / 7:05</div><div></div><div>🔊 ⓘ</div></div>	Once upon a time there w...

1-2 of 2 < >

4. Sign in/up: where the user can sign in/up to gain access to the system.



HOME RECORDING MANAGEMENT PUBLIC RECORDS LOGIN/REGISTER

Login

User Name *

Password *

LOGIN

Don't have an account? Register



Register

☐ Accept Terms & Conditions *

REGISTER

[Already have an account? Login](#)

5. Profile: allowed user change password or change user name and email, and logout account.




Good evening


[WANT TO CHANGE PASSWORD?](#) ▼

[EDIT PROFILE](#) ▼

LOGOUT

[HOME](#)[RECORDING MANAGEMENT](#)[PUBLIC RECORDS](#)[PROFILE](#)

Good evening


WANT TO CHANGE PASSWORD? 

Old Password*


New Password*

Confirm Password*


CHANGE PASSWORD


EDIT PROFILE 

LOGOUT

[HOME](#)[RECORDING MANAGEMENT](#)[PUBLIC RECORDS](#)[PROFILE](#)

Good evening

WANT TO CHANGE PASSWORD? 

EDIT PROFILE 

New Username*

New Email*

EDIT PROFILE

LOGOUT

Expected Schema:

- Post sign-in user

```
1 interface UserLogin {  
2   username: string;  
3   password: string;  
4 }
```

- Post sign-up user

```
1 interface UserSignUp {  
2   username: string;  
3   email: string;  
4   password: string;
```

```
5   isAdmin: boolean (default: False)
6 }
```

- Receive access token

```
1 interface UserLoginResponse {
2   isAdmin: boolean;
3   access: string;
4   refresh: string;
5 }
```

- Recording

```
1 interface Recording {
2   task_id: string;
3   block_id: string;
4   text: string;
5   file: string;
6 }
```

- Change Password

```
1 interface ChangePasswordDto {
2   current_password: string;
3   new_password: string;
4 }
```

- Edit Profile

```
1 interface EditProfileDto {
2   username: string;
3   email: string;
4 }
```

Expected Endpoints

- user sign up `POST /api/user/signup`
- user sign in `POST /api/user/login`
- get all recording task `GET /api/task`
- add recording task `POST /api/task/{task_id}/{block_id}`
- delete recording task with id `DELETE /api/recording/{task_id}/{block_id}`
- user change password `PUT /api/user/change-password`
- user edit profile `PUT /api/user/edit-profile`