**SONIC: Auditory Database**

**A PROJECT REPORT**

***Submitted by***

JHANTU SAMUI(MAKAUT, West Bengal)

SUCHANA SINGH(MCKVIE, LILUAH)

RONIT SINGH(MCKVIE, LILUAH)

Internship at CDAC, Kolkata

***Under the guidance of***

Mr Madhab Pal

**Centre for Development of Advanced Computing(C-DAC)**

Plot - E, GP Block, Sector V, Bidhannagar, Kolkata-700091

YEAR: 2023-2024

**Acknowledgment**

I would like to express my sincere gratitude to all those who have contributed to the successful completion of this Node.js application project using the Sequelize ORM. Their support and collaboration have been invaluable in achieving our goals.

First and foremost, I extend my thanks to Mr. Madhab Pal, our esteemed guide, for his unwavering support, invaluable guidance, and encouragement throughout the project. His expertise and mentorship played a crucial role in shaping our understanding of audio processing and database management.

A special note of appreciation goes to our fellow teammates for their collaboration, dedication, and collective effort in bringing this project to fruition. Their commitment and teamwork significantly contributed to the success of this endeavor.

I would also like to express gratitude to the entire CDAC Kolkata team for providing us with resources, facilities, and a conducive learning environment during our internship. Their support played a crucial role in our professional development.

Lastly, I extend my thanks to all the individuals who supported and encouraged us throughout this journey. Their unwavering support was a constant source of motivation.

This project would not have been possible without the collective efforts and support of each individual mentioned above. Their contributions have been instrumental in the successful completion of the Node.js application project, ensuring its adherence to best practices in terms of data integrity, security, and modularization..

**TABLE OF CONTENT**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Name of content** | **Page No.** |
| 1. | Introduction | 4-5 |
| 2. | Objectives | 6-7 |
| 3. | Proposed solution | 8-11 |
| 4. | Methodology and implementation details | 12-14 |
| 5. | Figure | 14-20 |
| 6. | Conclusion | 21-22 |
| 7. | Future Scope | 22-24 |

**INTRODUCTION**

In the evolving landscape of audio processing and data management, the need for a centralized platform to organize, store, and access audio data has become increasingly evident. The SONIC project, an acronym for "System for Organized and Networked Integrated Audio Content," emerges as a sophisticated solution aimed at addressing this requirement. This introduction sets the stage for understanding the key objectives, significance, and structure of the SONIC project.This project involves the development of a Node.js application using the Sequelize ORM for managing various types of users and administrators in a relational database. The application adheres to best practices in terms of data integrity, security, and modularization. The key components of the project include database models, middleware functions, routes, and an Express.js server.

**1. Background:**

With the proliferation of audio data across various domains such as research, education, and entertainment, there arises a demand for a unified system capable of efficiently managing diverse audio files and associated information. The SONIC project is conceived as a comprehensive response to this demand.

**2. Project Overview:**

SONIC is a web-based platform developed using cutting-edge technologies, including React for the frontend, Node.js and Express for the backend, and MySQL for database management. It acts as a centralized hub where users can contribute audio files, transcriptions, and related details, as well as retrieve and interact with the stored audio content.

**3. Objectives:**

The primary objectives of the SONIC project encompass establishing a centralized audio database, enabling user contributions, facilitating robust audio search capabilities, supporting audio visualization, and implementing various playback controls. Additionally, the project aims to ensure data security, create an administrative panel with diverse roles, and generate insightful reports.

**4. Scope of the Project:**

SONIC targets two primary user groups: contributors and seekers. Contributors have the option to upload individual audio files with details or submit a zip file containing multiple audio files, transcriptions, and a CSV file for bulk contributions. Seekers can utilize search functionalities, visualize audio waveforms, and access playback controls for downloaded audio files.

**5. Significance:**

The significance of the SONIC project lies in its potential to revolutionize how audio data is managed and accessed. By providing a user-friendly interface for both contributors and seekers, SONIC contributes to the advancement of audio processing technologies, research methodologies, and educational practices.

**6. Structure of the Report:**

The project report is structured to provide a comprehensive overview of the SONIC project, including its objectives, proposed solution, implementation details and results, conclusions, and future scope. Each section is carefully crafted to offer insights into the project's development, functionality, and potential impact.

As we delve deeper into the subsequent sections, the report will unfold the intricate details of SONIC's architecture, features, and outcomes, shedding light on its role in shaping the future of audio data management.

7. **Database:**

MySQL serves as the relational database management system for SONIC. It stores audio files, transcriptions, user details, and other relevant information. Sequelize ORM is utilized for database interactions, ensuring data integrity and security. The database schema is designed to accommodate diverse data types and relationships, providing a structured foundation for efficient data management.

8**. Frontend:**

The frontend is developed using ReactJS, a powerful JavaScript library for building user interfaces. It provides an intuitive and responsive user experience, allowing contributors and seekers to interact seamlessly with the platform. The frontend incorporates role-based access control, ensuring that users only have access to functionalities relevant to their roles.

**OBJECTIVES**

* **Establish a Centralized Audio Database:**
* Create a platform to centralize and organize audio data, allowing users to contribute and access audio files, transcriptions, and related details.
* **Enable User Contributions:**
* Provide a user-friendly interface for contributors to upload single WAV files along with transcription files and relevant details through a form.
* Allow contributors to upload a zip file containing multiple WAV files, their corresponding transcription files, and a CSV file with detailed information.
* **Facilitate Audio Search:**
* Implement a robust search functionality for users to find relevant audio files based on criteria such as gender, age group, or other relevant parameters.
* **Support Audio Visualization:**
* Integrate a feature to visualize audio waveforms, enabling users to have a graphical representation of the audio data.
* **Implement Audio Player Functionality:**
* Provide users with playback controls, including play, pause, restart, skip forward, and skip backward options, enhancing the user experience.
* **Enable Audio Download:**
* Allow users to download audio files directly from the platform based on their search criteria.
* **Offer Zooming Functionality:**
* Implement zoom in and zoom out features for the audio waveform, allowing users to focus on specific sections of the audio data.
* **Ensure Data Security and Management:**
* Implement robust security measures to safeguard user-contributed data.
* Manage and organize data efficiently, ensuring accurate storage of audio files, transcriptions, and associated details.
* **Create an Admin Panel:**
* Develop an admin panel with different roles (data preparation admin, data preparation user, super admin, report admins, report users, testing admin, testing user, training admin, training user) each having specific responsibilities and access levels.
* **Generate Reports and Analytics:**
* Implement reporting functionalities for admins to generate insights into data usage, contributor activity, and overall system performance.
* **Ensure Scalability and Future Expansion:**
* Design the system with scalability in mind, accommodating potential future growth in terms of data volume and user interactions.
* **Provide Documentation and Training:**
* Create comprehensive documentation for users and administrators to ensure ease of use and system maintenance.
* Conduct training sessions for administrators to effectively manage the system.

These objectives cover a range of functionalities, from user contributions to advanced features and administrative capabilities, aligning with the goals of building a comprehensive audio database system.

**Proposed Solution**

**Problem Statement Recap**

The SONIC project addresses the need for a comprehensive audio database system, catering to both contributors and users. Contributors seek a seamless platform to submit audio files with associated transcriptions, while users aim to efficiently search, visualize, and download relevant audio data based on specific criteria. the admin panel implementation for the SONIC project involved addressing challenges related to RBAC, user interface design, security, backend integration, scalability, testing, and documentation. Successfully overcoming these challenges was crucial to delivering a functional, secure, and user-friendly admin panel that meets the diverse needs of the SONIC project stakeholders.

**Objective Alignment**

The proposed solution aligns with the following key objectives:

* **Efficient Data Contribution:**
* Enable contributors to submit individual audio files or batch uploads with corresponding transcriptions and metadata.
* **User-Friendly Interface:**
* Develop an intuitive user interface for contributors to easily navigate and input data, enhancing the overall user experience.
* **Robust Search and Retrieval:**
* Implement a robust search functionality for users to find audio files based on criteria such as gender, age group, and other relevant attributes.
* **Audio Visualization and Playback:**
* Provide users with the capability to visualize audio waveforms and utilize playback functionalities for a richer user experience.
* **Multi-functional Tools:**
* Integrate tools for zooming, playing, pausing, and restarting audio, allowing users to interact with the audio data dynamically.
* **Role-Based Access Control (RBAC):**
* Defining and implementing different roles (data preparation admin, data preparation user, super admin, report admins, report users, testing admin, testing user, training admin, training user) with specific permissions can be complex.

**Technical Overview**

The SONIC project leverages a full-stack solution, utilizing React for the frontend, Node.js and Express for the backend, and MySQL for database management. The system incorporates a scalable architecture to accommodate a growing database of audio files.

**System Architecture**

The system is divided into distinct layers:

* **Frontend:** React components for an interactive and responsive user interface.
* **Backend:** Node.js and Express to handle data processing, storage, and retrieval.
* **Database:** MySQL to store audio file metadata, transcriptions, and user information.

**Features and Functionalities**

* **Data Contribution:**
* Contributors can submit single audio files with transcriptions or batch uploads containing multiple files and a CSV file for streamlined data entry.
* **User Authentication:**
* Implement user authentication to distinguish contributors and users, ensuring secure access to relevant functionalities.
* **Search and Retrieval:**
* Develop a powerful search algorithm to retrieve audio files based on specified criteria, providing users with tailored results.
* **Audio Visualization:**
* Utilize the WaveSurfer.js library for dynamic audio waveform visualization, enhancing the user's ability to interact with the audio data.
* **Playback and Tools:**
* Enable users to play, pause, restart, and apply various tools for audio manipulation, enhancing the overall user experience.
* **User Management:**
* The user management system provides comprehensive functionality for efficient administration. Admins, including a super admin, can seamlessly add users, specifying details such as name, email, password, and relevant information. The system supports role-based user addition, enabling assignment to specific admins for data preparation, training, testing, or report administration. User information can be updated by admins, including modifications to attributes based on user ID, encompassing details like name, email, password, and phone number. Admins also have the capability to delete users from the system, with deletion based on user ID and accompanied by appropriate error handling. The search functionality empowers admins to perform effective operations, allowing for the precise location of users through filtering based on various criteria, ensuring a robust and flexible user management experience.

**Innovation and Uniqueness**

* **Dynamic Audio Visualization:**
* The integration of WaveSurfer.js allows for real-time audio waveform visualization, offering users a unique and interactive experience.
* **Scalability:**
* The system is designed with scalability in mind, accommodating a growing database of audio files and users.
* **Role-Based Access and Dashboard Customization:**

The admin panel features a role-based access system, assigning distinct roles (e.g., data preparation, training, testing, report, super admin) with tailored dashboards for each. This ensures a streamlined user experience by providing relevant information and tools based on the admin's responsibilities upon login..

* **Secure Middleware Authentication:**

The use of middleware technology for login adds an extra layer of security. The system not only verifies email and password but also checks for specific roles and subroles before granting access. This ensures that only authorized personnel can access sensitive areas of the application.

* **Dynamic CRUD Operations with Role Restriction:**

The admin panel allows for dynamic CRUD operations, enabling administrators to add, update, and delete users within their respective domains (data preparation, training, testing, report).

**Challenges and Mitigations**

* **Large File Handling:**

Address potential challenges related to large file uploads through asynchronous processing and optimized storage strategies.

* **Search Algorithm Optimization:**

Implement efficient algorithms for fast and accurate search results, minimizing response times even with a vast audio database.

* **Security Vulnerabilities:**

One of the major challenges in an admin panel is the potential for security vulnerabilities. Storing and managing sensitive information such as user credentials and access control details can make the system susceptible to attacks like SQL injection, cross-site scripting (XSS), and unauthorized access. The use of plain text passwords and lack of proper encryption or hashing mechanisms can expose sensitive data, posing a significant security risk.

**Methodology and implementation details**

**System Architecture**

The SONIC project adopts a robust and scalable architecture to fulfill its objectives. The system is divided into three main layers: the frontend, backend, and database.

**Frontend**

The frontend is developed using React, a popular JavaScript library for building user interfaces. React's component-based architecture provides modularity and flexibility. The user interface is designed to be intuitive, ensuring a seamless experience for both contributors and users.

**Backend**

Node.js and Express.js are employed to build the backend infrastructure. Node.js offers a non-blocking, event-driven architecture, making it well-suited for handling concurrent operations. Express.js, a web application framework for Node.js, streamlines the development of robust APIs and ensures efficient routing.

**APIs and Endpoints**

The backend exposes APIs and endpoints to facilitate various functionalities, including data contribution, user authentication, and audio file retrieval. The APIs are RESTful, adhering to best practices for communication between the frontend and backend.

**Database**

MySQL serves as the relational database management system, storing metadata related to audio files, transcriptions, and user information. The database schema is designed to efficiently organize and retrieve data. Proper indexing and normalization practices are implemented to ensure optimal performance.

**Data Contribution Workflow**

**Single File Upload**

* **Form Submission:**
* Contributors access a user-friendly form to submit details such as gender, age group, and the audio file along with its transcription in a single WAV and TXT pair.
* **Backend Processing:**
* Upon submission, the backend processes the form data, validates the inputs, and stores the metadata and files in the database and server, respectively.

**Batch Upload with CSV**

* **Zip File Submission:**
* Contributors may choose to upload a zip file containing multiple WAV files, corresponding TXT files, and a CSV file with detailed metadata.
* **CSV Parsing:**
* The backend parses the CSV file to extract information such as file names, file paths, gender, age group, and transcription file names.
* **Database Storage:**
* The extracted data is stored in the database, and the audio files are saved on the server.

**User Authentication**

User authentication is implemented to differentiate contributors from users. Contributors have access to data contribution functionalities, while users can search, visualize, and download audio files. And for admin , by providing authenticate details admin can log in and perform task.

**Search and Retrieval**

The search functionality enables users to find audio files based on specified criteria. The backend processes search queries, retrieves relevant data from the database, and presents results to the user. Every admin search under their allocated users with each of single their details (like name, email, phone no).

**Audio Visualization**

WaveSurfer.js, a JavaScript library for audio visualization, is integrated into the frontend to dynamically display audio waveforms. Users can visualize, play, pause, and manipulate audio files directly within the interface.

**Playback and Tools**

The frontend incorporates controls for audio playback, including play, pause, restart, and additional tools for zooming in and out on the audio waveform. These features enhance the user's ability to interact with and analyze audio data.

**Technologies Used**

* **Frontend:** React
* **Backend:** Node.js, Express.js
* **Database:** MySQL
* **Audio Visualization:** WaveSurfer.js

**Figures**

**Admin Login : -** In the admin panel, the process of Admin login involves verifying the provided email and password, checking the role and subrole, and granting access to the respective dashboard based on the admin's role, such as data preparation, training, testing, report, or super admin .

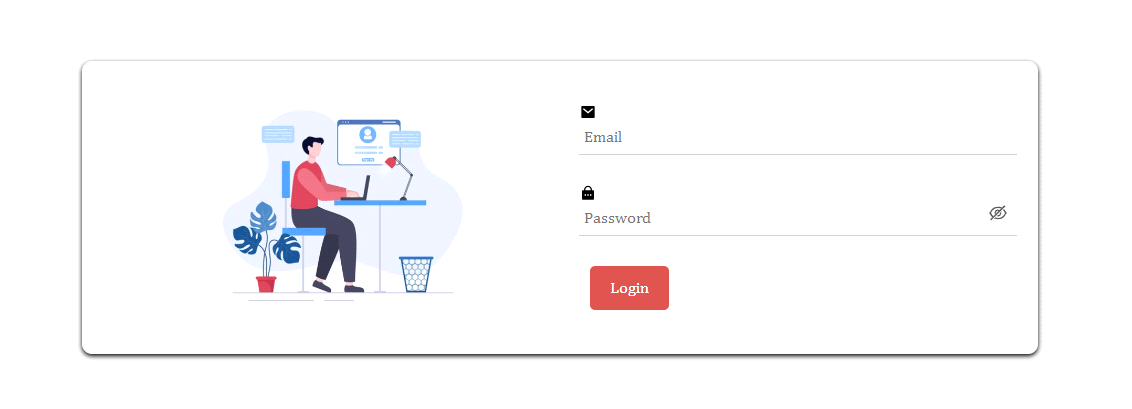


Figure-1: Admin Login

**Data Preparation Admin Dashboard : -**

In the Data Preparation Admin panel, administrators can perform CRUD operations on users. They can add users with role-based details, view, update, and delete user information, and utilize effective search functionalities.

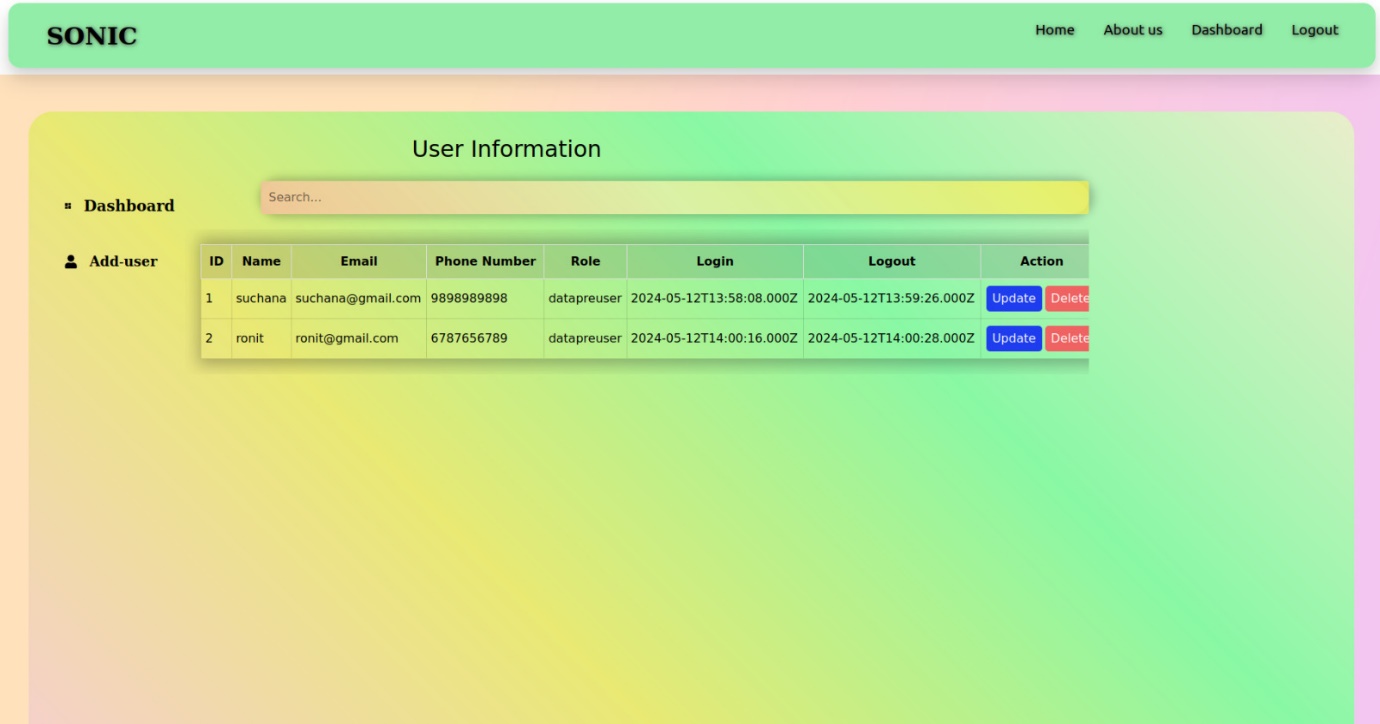
****

Figure-2: Data preparation Admin Dashboard

**Super Admin : -**

In the admin panel, the Super admin process involves transferring super admin rights by updating the active status and email of the existing super admin to deactivate and adding a new super admin with updated credentials, enabling seamless system control and security management. And also add the user and can see how meny users are allocated under which admin.

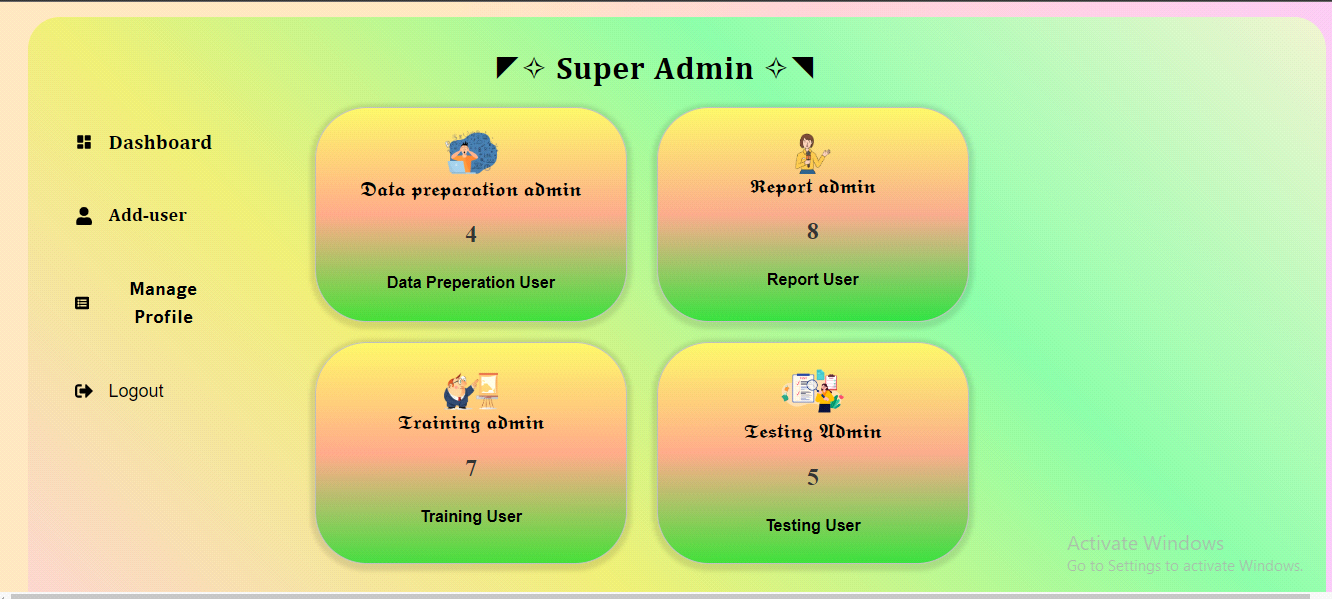


Figure-3 : Super Admin Dashboard

**Update Super Admin : -**

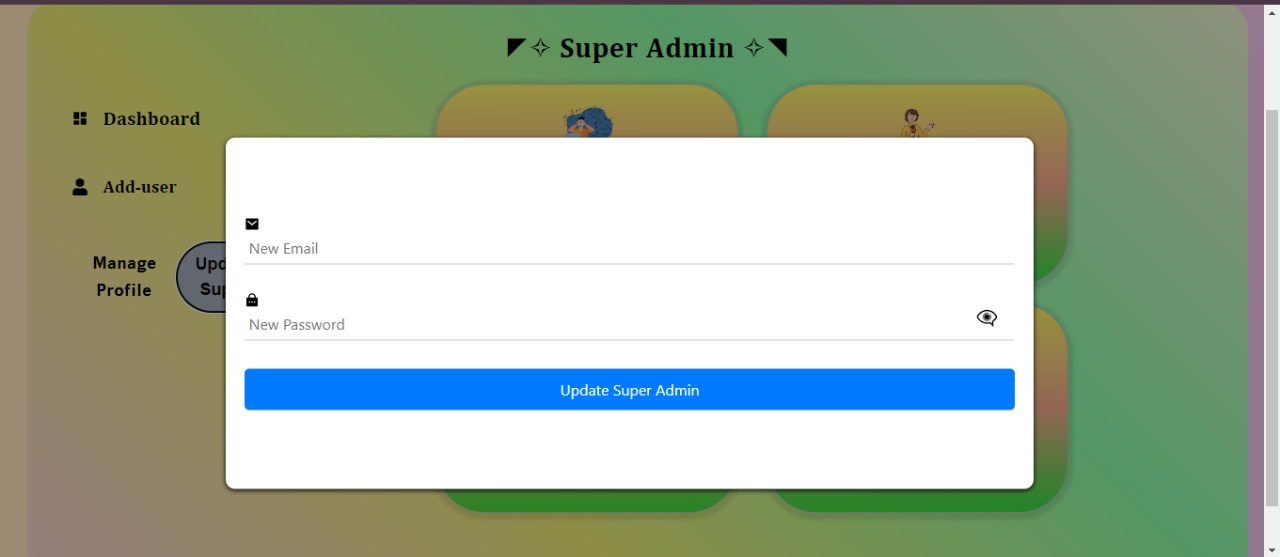
****

Figure-4 : Update Super Admin

**Database : -**

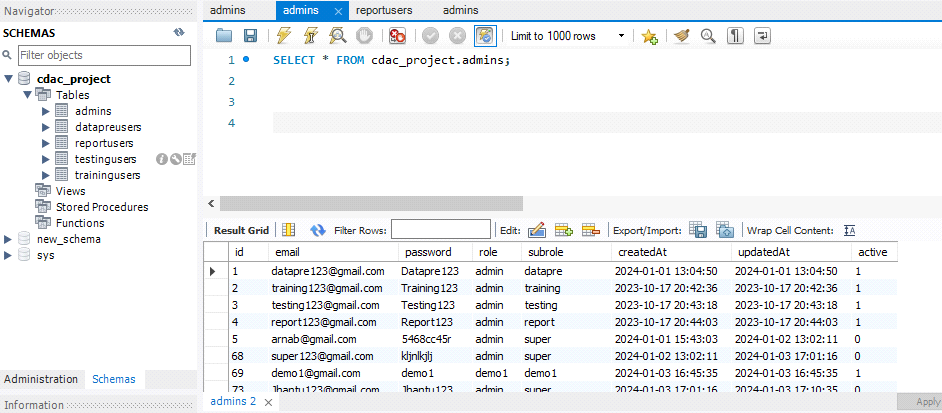


Figure-5 : Database

**Middleware Function:-**

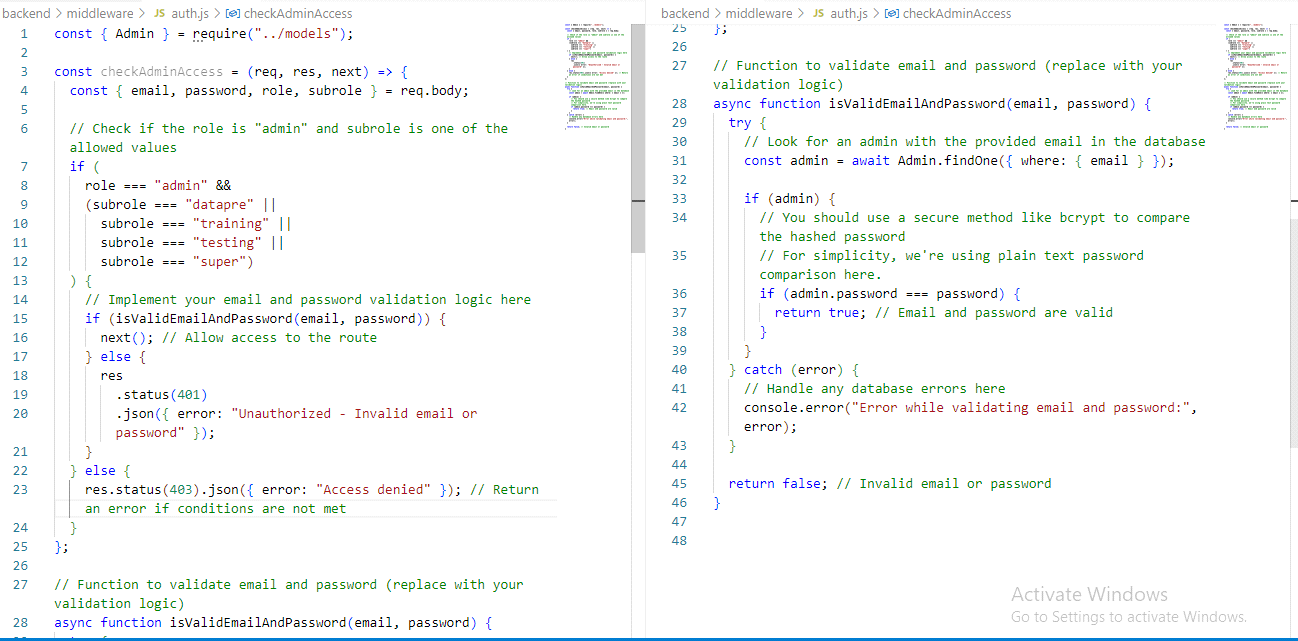


Figure-6: Middleware for Login

**Search button for users:-**

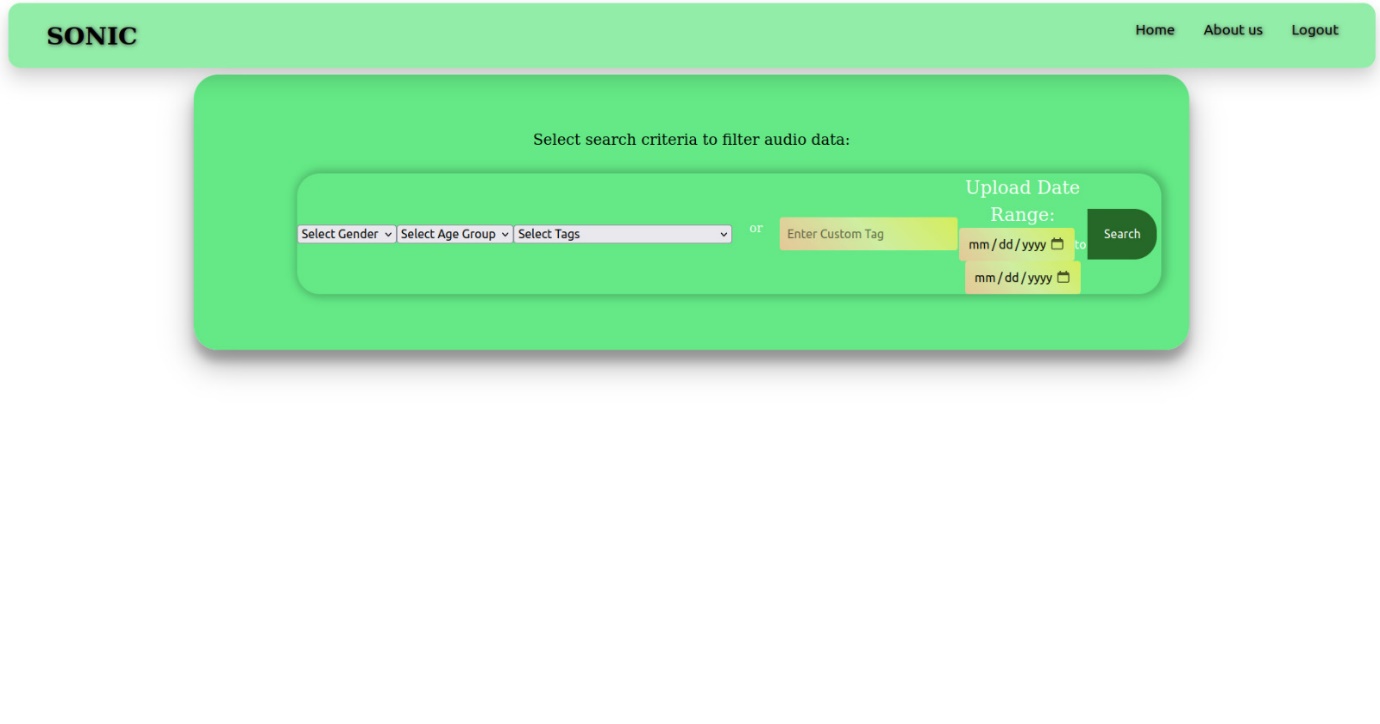
****

Figure-7: User interface for searching

**Search results from database:-**

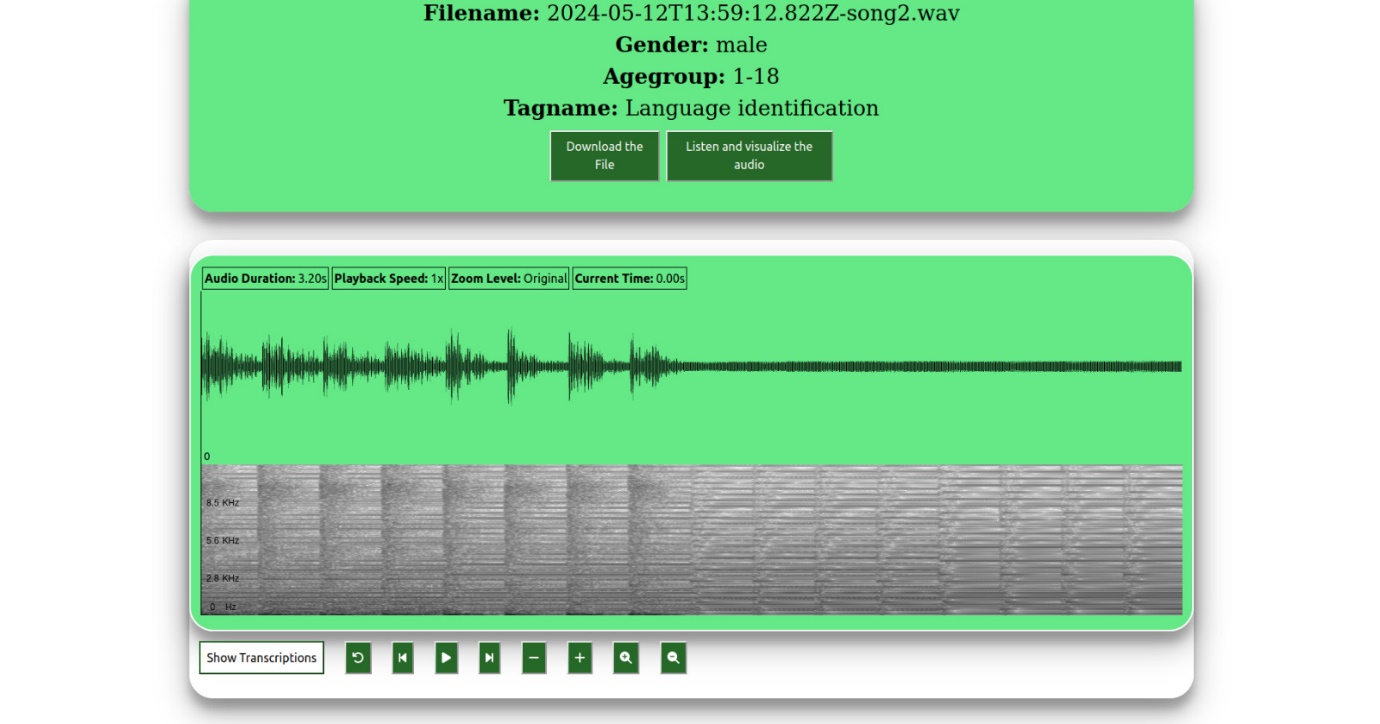


Figure-8: Available results

**Audio wave visualization and other functionalities:-**

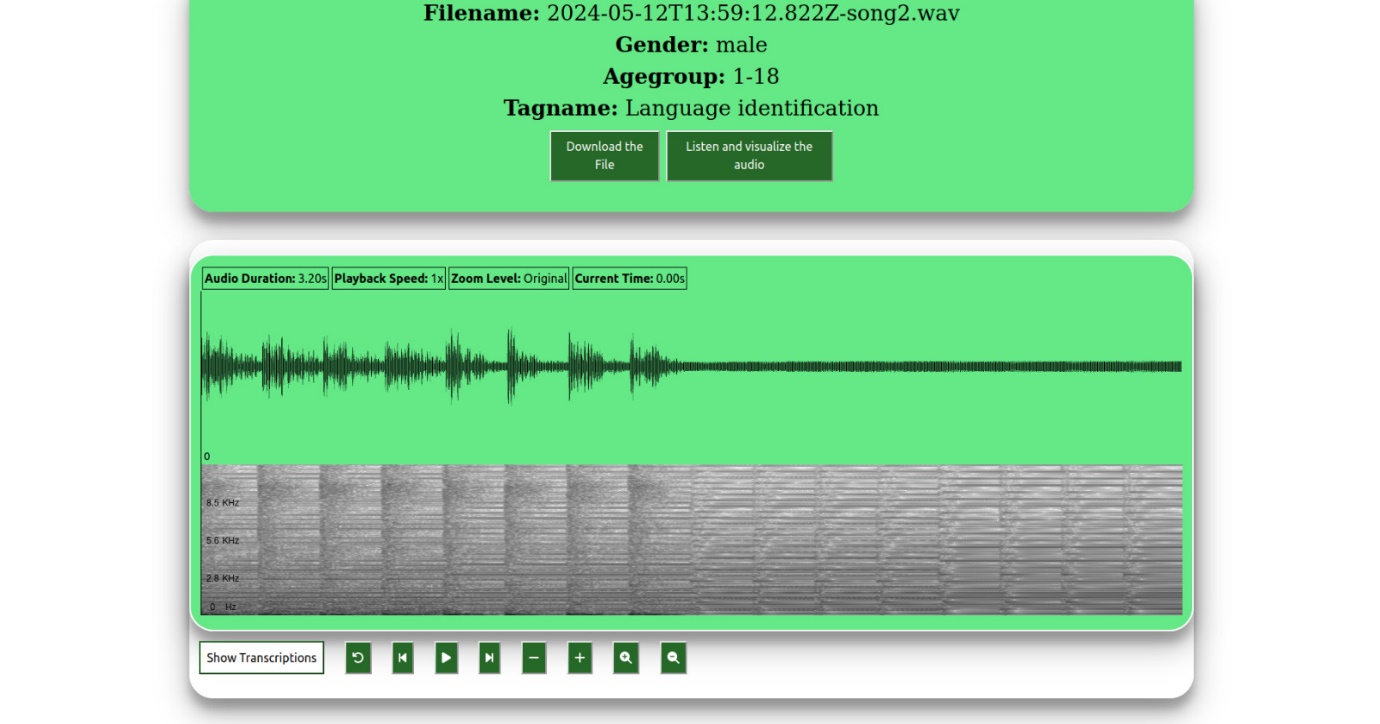


Figure-9: visualization of audio files

**Upload button for contributers:-**

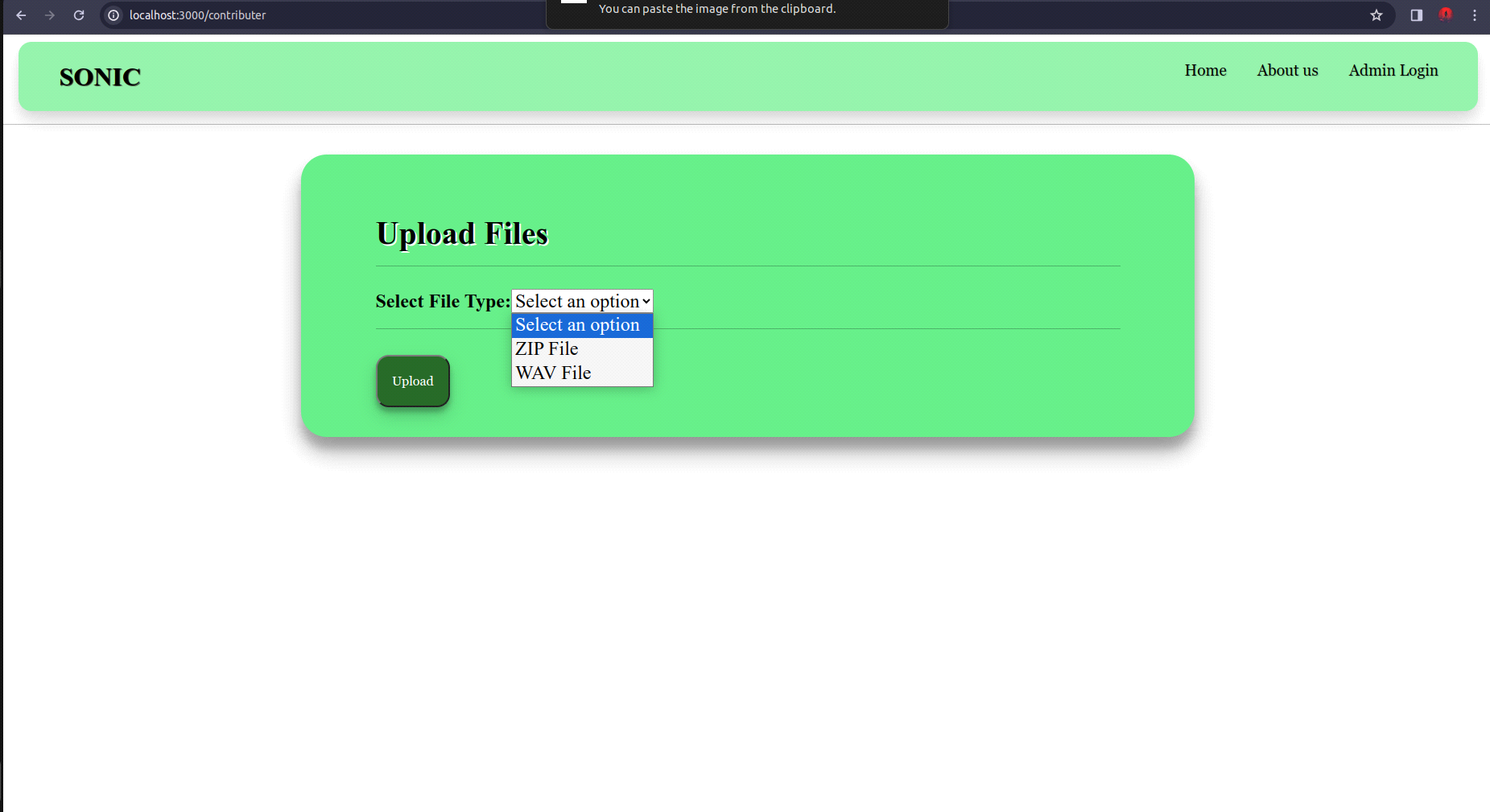


Figure-10: Interface for contributers

**For zip file upload:-**

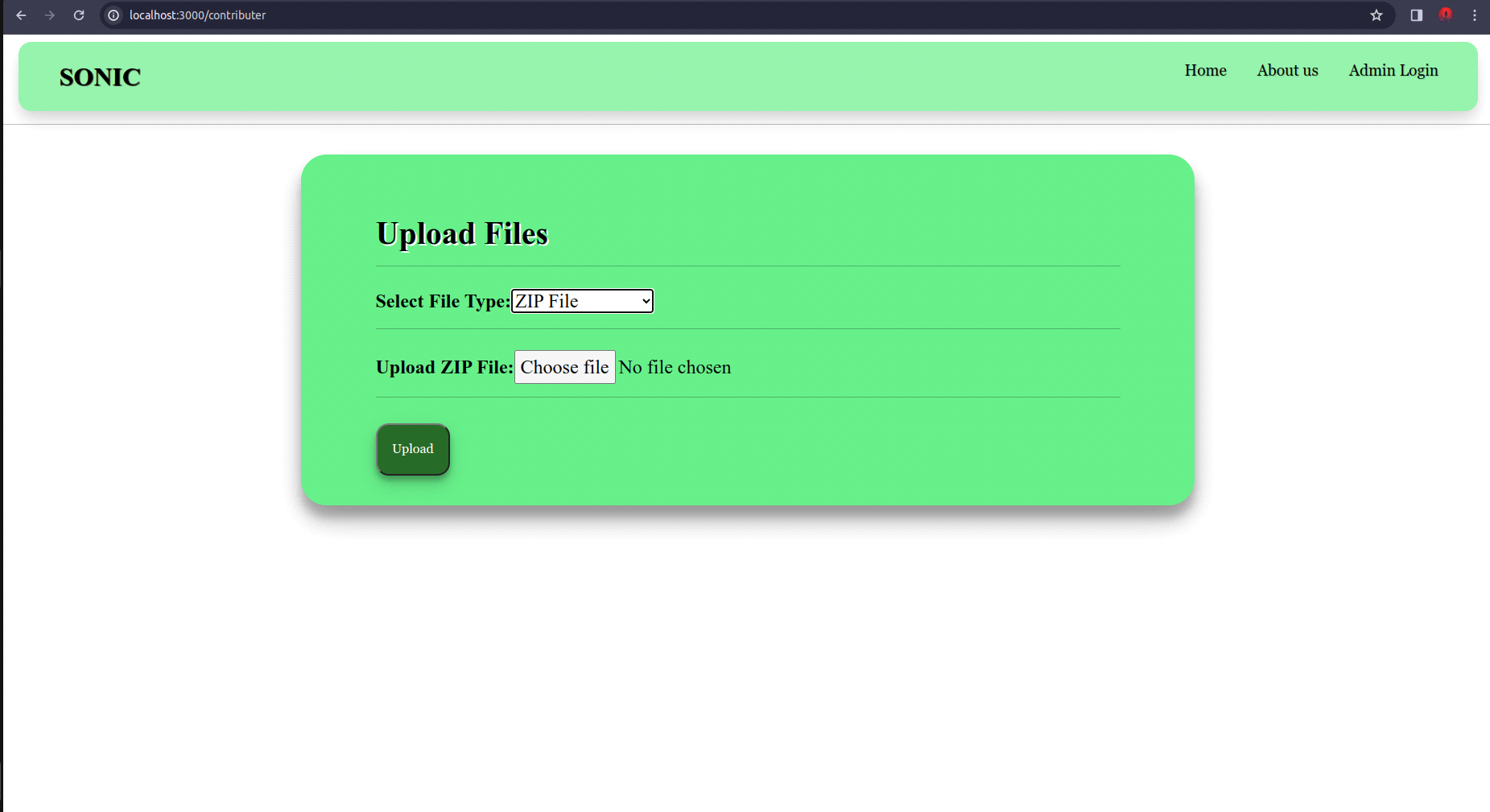


Figure-11 : Zip file uploading option

**For WAV file upload:-**

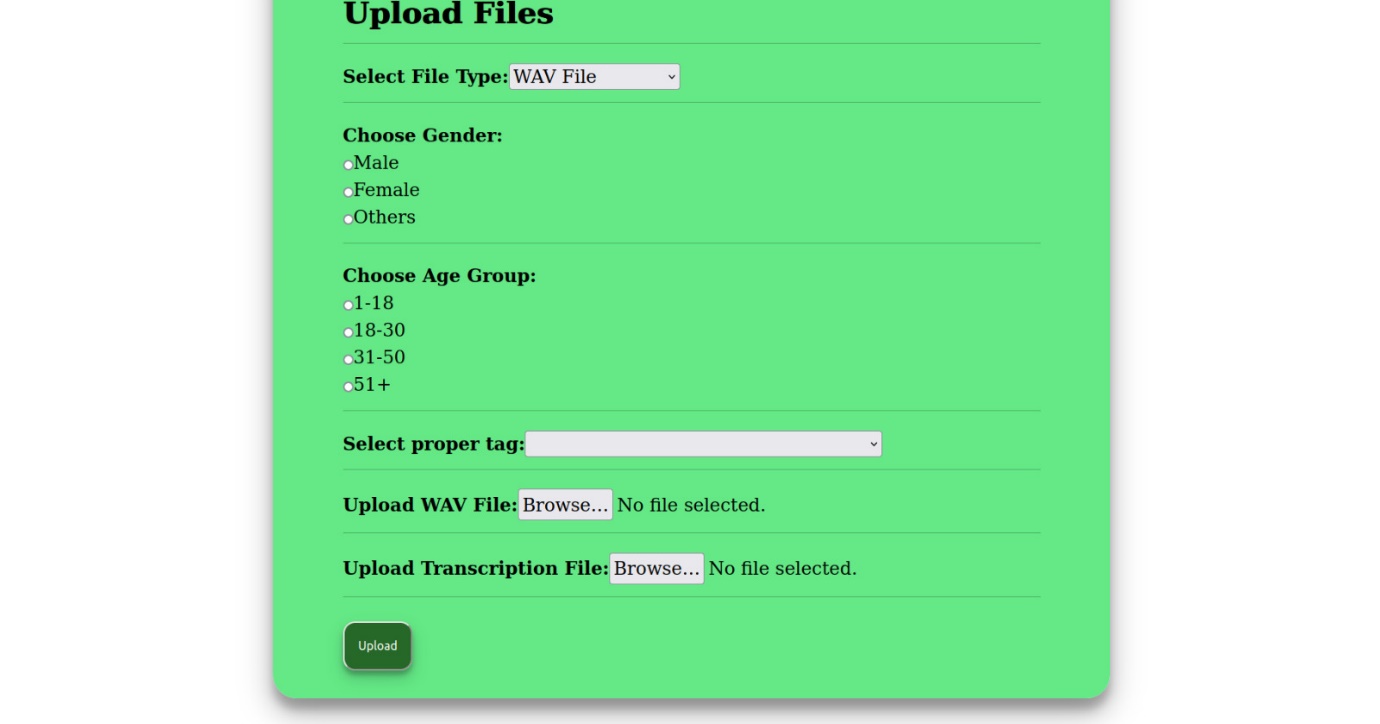
****

Figure-12: form for WAV file uploading

**About Us page:-**

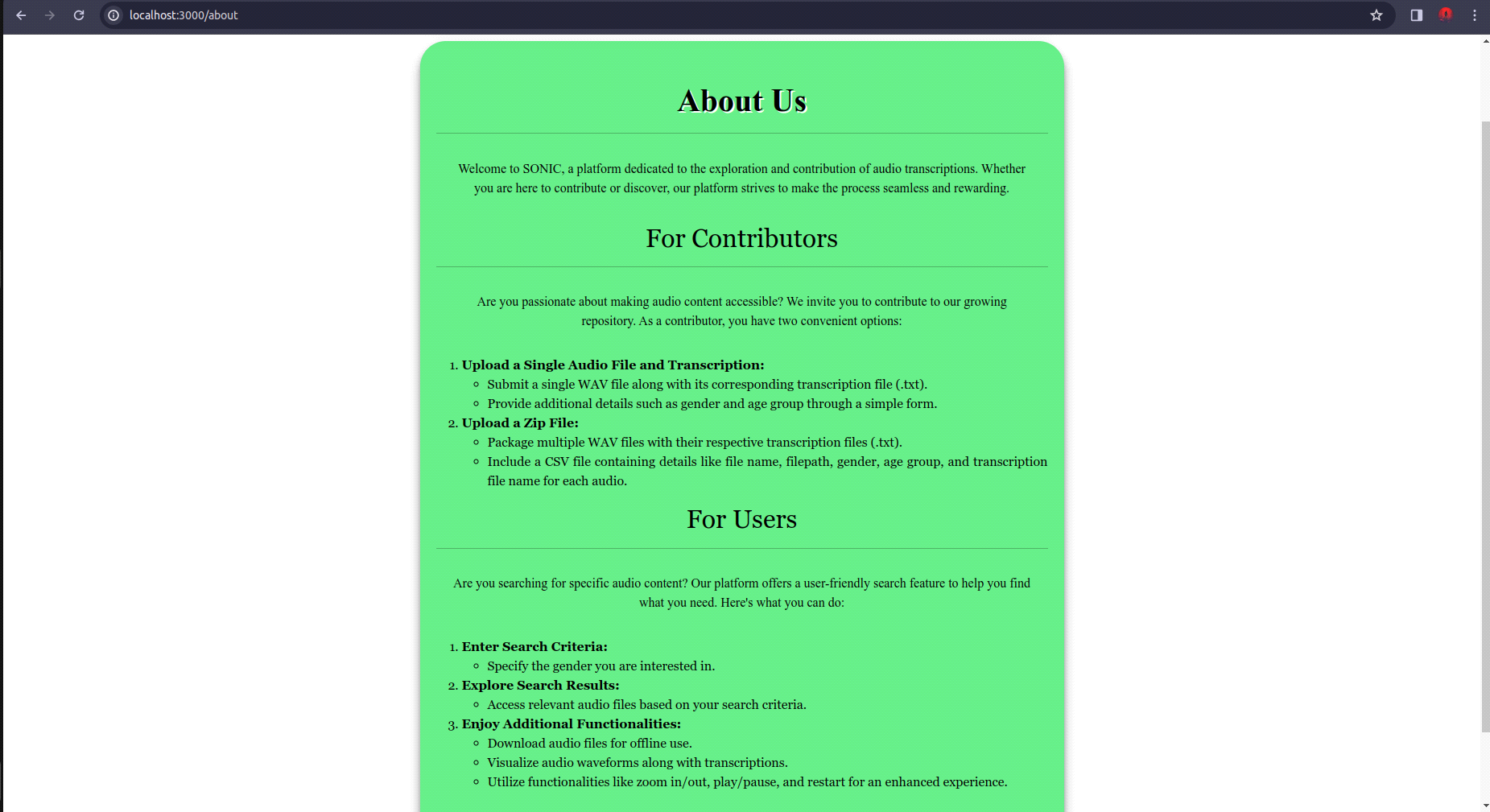


Figure-13: About page

**CONCLUSION**

The SONIC project, a collaborative effort of Jhantu Samui, Suchana Singh, and Ronit Singh under the guidance of Mr. Madhab Pal at CDAC Kolkata, represents a significant milestone in the domain of audio database management. This section concludes the project report by summarizing key achievements, challenges faced, and the project's impact.

**Key Achievements**

* **Robust Data Contribution Mechanism:**
* The implementation of a user-friendly interface for contributors ensures a seamless data contribution process. Contributors can upload single audio files with transcriptions or batch upload using a zip file, enhancing the diversity and volume of available audio data.
* **Efficient Data Retrieval for Users:**
* The search functionality empowers users to find relevant audio files based on specific criteria, facilitating efficient retrieval. Users can explore and download audio files tailored to their needs, enhancing the overall user experience.
* **Interactive Audio Visualization:**
* The integration of WaveSurfer.js enables users to visualize audio waveforms interactively. Playback controls, zoom functionalities, and visual representations enhance the user's ability to analyze and comprehend audio data.
* **Scalable and Maintainable Architecture:**
* The adoption of a robust architecture, including React for the frontend, Node.js and Express.js for the backend, and MySQL for the database, ensures scalability and maintainability. Continuous integration practices further contribute to a reliable and up-to-date system.
* **Role-Based User Managemen:**
* **User CRUD Operations:**
* **Super Admin Transfer and Profile Management:**
* **Database Synchronization:**
* **RESTful API for User Management**

**Challenges Faced**

* **Zip File Parsing:**
* The parsing of CSV files within zip archives presented a challenge during implementation. However, thorough testing and debugging resulted in a robust solution, maintaining the integrity of metadata.
* **User Authentication Complexity:**
* Implementing a secure and user-friendly authentication system required meticulous planning. Ensuring a balance between contributor and user roles was essential for providing distinct functionalities to different user types.

**FUTURE SCOPE**

The SONIC project, with its current capabilities and achievements, opens the door to numerous possibilities for future enhancements and expansions. This section explores potential avenues for growth and improvement, outlining the future scope of the project.

**1. Machine Learning Integration:**

* Explore the integration of machine learning models to automate transcription processes. Implementing speech-to-text algorithms could significantly reduce manual transcription efforts and enhance the accuracy of transcriptions.

**2. Advanced Audio Analysis:**

* Investigate the incorporation of advanced audio analysis techniques. This could include the implementation of signal processing algorithms, pattern recognition, and feature extraction methods to extract more detailed insights from audio data.

**3. Collaboration with Research Communities:**

* Foster collaborations with academic and research communities in the field of audio processing. Engaging with experts can provide valuable insights, foster innovation, and open avenues for incorporating cutting-edge technologies into the SONIC platform.

**4. User Feedback and Iterative Improvements:**

* Establish a robust feedback mechanism to gather user input continually. Regularly update the platform based on user feedback, ensuring that SONIC remains responsive to evolving user needs and technological advancements.

**5. Enhanced Security Measures:**

* Implement additional security measures to safeguard user data and maintain user privacy. Regular security audits and updates will be essential to address emerging cybersecurity challenges and ensure the platform's integrity.

**6. Community Building:**

* Build a community around SONIC by engaging with contributors, users, and researchers. This can be achieved through forums, webinars, and collaborative initiatives. A vibrant community can contribute to the project's sustainability and foster a culture of knowledge sharing.

**7. Expand Dataset Diversity:**

* Encourage contributors to upload diverse audio datasets representing various languages, accents, and domains. This will contribute to a more inclusive and comprehensive audio database, catering to a broader audience.

**8. Integration with External APIs:**

* Explore the possibility of integrating SONIC with external APIs related to audio processing, transcription services, or machine learning frameworks. This integration can bring in additional functionalities and keep the platform aligned with industry standards.

**9. Mobile Application Development:**

* Develop a mobile application for SONIC, making the platform more accessible to users on different devices. A mobile app can enhance the user experience and provide on-the-go access to audio resources.

**10. Gamification Elements:**

* Introduce gamification elements to encourage user engagement and contributions. Implementing badges, achievements, or collaborative challenges can create a more interactive and rewarding experience for users.

**11.Enhanced Security Measures:**

Implement advanced password hashing mechanisms, such as bcrypt, for storing and verifying passwords securely.

Introduce multi-factor authentication (MFA) for an additional layer of security.

**12.Real-Time Notifications:**

Implement real-time notifications for important events, such as user additions, updates, and deletions.

Utilize WebSocket or other real-time communication technologies to push notifications to administrators.

**13. Data Encryption:**

Implement data encryption at rest and in transit to enhance the overall security of stored and transmitted data.

Use SSL/TLS for secure communication between the admin panel and the database.

The future scope outlined above reflects the potential for SONIC to evolve into a dynamic and feature-rich platform. By embracing these opportunities, the project can stay at the forefront of audio database management, meeting the evolving needs of contributors and users alike.