Solution Deployment

Audio Collection Application with Docker Deployment
S4-DACS-01-1

---

1 Introduction

This document specifies the requirements for the development of a web-based application designed to collect audio recordings from users reading predefined sentences. The application must facilitate the acquisition of speech data in a controlled and user-friendly manner while ensuring data integrity and user privacy. The solution must be containerized using Docker and orchestrated via Docker Compose to ensure consistency across different environments and ease of deployment. The source code and documentation will be managed via GitHub, with an optional extension to deploy the application on a cloud platform.

2 Functional Requirements

The application shall present a homepage serving as the entry point for users. Following this, users will be required to provide demographic data consisting exclusively of their age and gender, alongside explicit consent confirmation. No personally identifiable information such as names shall be collected, thereby preserving user anonymity.

Once this information is submitted, the application will sequentially display sentences for the user to read aloud. The interface must provide controls enabling the user to record their voice, re-record as necessary, save the current recording, and proceed to the next sentence. The number of sentences presented should be configurable by the user at the start of the recording session.

The system must accommodate early termination of the session by the user. In such cases, all audio recordings completed up to the point of termination must be reliably saved without data loss. Upon completion or early exit, the application should redirect the user back to the homepage, ready for a subsequent session.

3 Technical Specifications

The entire application must be containerized using Docker to encapsulate the runtime environment, dependencies, and configuration, ensuring platform independence and reproducibility. If the architecture involves multiple services—for example, separate backend and frontend components—Docker Compose shall be utilized to manage and orchestrate these containers effectively.

Version control shall be maintained using GitHub. The repository must include all source code, Docker configuration files, and a comprehensive README document. The README should provide clear instructions on building, running, and testing the application both in a local environment and within the Docker containers.

4 Optional Extension: Cloud Deployment

An optional enhancement involves deploying the containerized application on a cloud infrastructure of choice, such as Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, Heroku, or any equivalent platform. This deployment should demonstrate the capability to scale and manage the application in a production-like environment. Detailed deployment instructions and any platform-specific configurations must be documented thoroughly in the GitHub repository.

5 Deliverables

The final deliverable is a fully functional web application meeting the functional and technical requirements outlined above. The project will be evaluated based on application functionality, code quality, completeness and clarity of documentation, and adherence to containerization best practices. Demonstration of cloud deployment and associated documentation will be considered for bonus recognition.