Software Requirements Specifications

Voice Coding

Project Code:

VC-WA01

Internal Advisor:

Professor Mudassir Ali Zaidi

Project Manager:

Professor Dr. Muhammad Ilyas

Project Team:

Alina Ahmed	BSCS51F22S044	Team Lead
Laiba Humayun	BSCS51F22S031	Team Member
Maryum Javed	BSCS51F22S009	Team Member

Submission Date:

October 20, 2025

_____

Project Manager's Signature

Document Information

Category	Information
Customer	University of Sargodha
Project	Voice Coding
Document	Requirement Specifications
Document Version	1.0
Identifier	
Status	Draft
Author(s)	Alina Ahmed Laiba Humayun Maryum Javed
Approver(s)	Professor Dr. Muhammad Ilyas
Issue Date	October 20, 2025
Document Location	
Distribution	1. Professor Mudassir Ali Zaidi 2. Professor Dr. Muhammad Ilyas

Definition of Terms, Acronyms and Abbreviations

Term/Acronym	Description
API	Application Programming Interface, a service for communication between systems
Web Speech API	Browser feature for speech recognition
Monaco Editor	Code editor component from VS Code
Paiza.io / JDoodle	Free online C++ compiler APIs
RSI	Repetitive Strain Injury
IDE	Integrated Development Environment
HTML / CSS / JS	Frontend web languages
SRS	Software Requirements Specification

Table of Contents

1.Introduction

1.1 Purpose of Document

*This Software Requirements Specification (SRS) describes the functional and non-functional requirements for the Voice Coder web application. It is structured according to the **IEEE Std 830-1998**[5] recommended practice. It serves as a guide for stakeholders, developers, and evaluators, defining the project's functionality, limitations, and design goals.*

1.2 Project Overview

Voice Coder is a web-based application that enables users to write C++ programs using voice commands. The system translates spoken instructions into syntactically correct C++ code, compiles it in real time, and displays the output. The objective is to improve accessibility for users with physical disabilities or RSI and promote hands-free programming.

1.3 Scope

The system will:

- *Convert voice commands into valid C++ code using the **Web Speech API**[1].*
- *Support basic to intermediate C++ constructs.*
- *Provide real-time cloud-based compilation via **Paiza.io** **or JDoodle APIs**[3], [4].*
- *Display program outputs and errors.*
- *Operate on modern browsers without installation.*

The system will not:

- *Support programming languages other than C++.*
- *Offer advanced IDE features such as debugging or version control.*
- *Work offline or in languages other than English.*
- *Handle complex C++ metaprogramming.*

2. Overall System Description

This section describes the environment in which the system will be developed and used, the anticipated users of the system, and the known constraints, assumptions, and dependencies.

2.1 User characteristics

Primary Users: *Students with disabilities, programmers with RSI, and computer science educators.*
Secondary Users: *Accessibility researchers and developers interested in voice-based programming.*

2.2 Operating environment

- ***Hardware:*** *PC, Laptop, or Tablet with a microphone.*
- ***Operating System:*** *Windows 10+, macOS 10.14+, Linux distributions.*
- ***Browser:*** *Chrome 70+, Firefox 75+, Edge 79+, Safari 13+ (supporting* ***Web Speech API****[1]).*
- ***Dependencies: Web Speech API****[1],* ***Paiza.io*** ***or JDoodle API****[3], [4],* ***Monaco Editor****[2].*

2.3 System constraints

- *Requires browser support for* ***Web Speech API****[1] and a stable internet connection.*
- *Limited to C++ language features supported by the chosen cloud compiler API* ***[3], [4]****.*
- *Needs a microphone, at least 2 GB RAM, a quiet environment, and clear English speech.*

3. External Interface Requirements

This section is intended to specify any requirements that ensure that our system will connect properly to external components.

3.1 Hardware Interfaces

The system is fully web-based and only requires a microphone and internet connection.

3.2 Software Interfaces

- ***Frontend:*** *HTML, CSS, JavaScript.*
- ***Speech Recognition: Web Speech API****[1].*
- ***Compilation:*** ***Paiza.io*** ***or JDoodle API****[3], [4].*
- ***Code Editor: Monaco Editor****[2].*

3.3 Communications Interfaces

The system will communicate securely with compiler APIs via HTTPS, ensuring that voice and code data are processed temporarily and not stored permanently, following standard web security practices ***[6]****.*

4. Functional Requirements

4.1 Voice Command Recognition

*The system must capture voice input and convert it to text using the **Web Speech API**[1].*

4.2 Command Parsing and Code Generation

The application will interpret recognized text into predefined C++ templates.
Example:

"Include Iostream" → #include <iostream>
"For loop from i to 10" → for(int i=0; i<10; i++){}

4.3 Code Display and Editing

*The recognized code must appear in the **Monaco Editor**[2], allowing manual edits or corrections before execution.*

4.4 Code Compilation and Execution

*The system must send the final code from the editor to a cloud compiler API (**Paiza.io** **or JDoodle**[3], [4]) for execution and return the output or errors.*

4.5 Error Handling

The system must display clear messages for invalid or unrecognized commands, network timeouts, or compilation errors from the external API.

4.6 User Interaction and Accessibility

The system must have accessible buttons labeled "Start Recording," "Stop," and "Run Code."

5. Non-functional Requirements

5.1 Performance Requirements

- *The web-based system must process spoken commands and generate corresponding C++ code within 2–4 seconds of input under standard internet conditions.*
- *The speech recognition module (**Web Speech API**[1]) should maintain a minimum accuracy rate of 90–95% in transcribing programming-related keywords.*
- *The integration with cloud compiler APIs (**JDoodle/Paiza.io**[3], [4]) should allow code execution and output retrieval within 5 seconds per request.*

5.2 Safety Requirements

- *The platform will execute all user-generated C++ code within a secure cloud sandbox environment provided by **JDoodle or Paiza.io**[3], [4], preventing unauthorized access to the host system.*
- *No locally executed code or external file access will be permitted, reducing the risk of data loss or malware execution.*

5.3 Security Requirements

- *All communication between the client browser and external APIs must use secure HTTPS/TLS encryption.*
- *The system should conform to **OWASP Top 10 web security guidelines**[7] to prevent common vulnerabilities such as Cross-Site Scripting (XSS) and injection attacks.*
- *User data — including speech transcripts and code — must not be stored or shared with third parties without consent.*
- *Any authentication tokens or API keys used to access cloud compilers must be secured and hidden from the client side.*

5.4 User Documentation

*Comprehensive user documentation will be provided, including a web-based user manual describing all features. Documentation will be updated iteratively, following principles of **agile software engineering**[6].*

6. Assumptions and Dependencies

- *The system assumes the user has access to a stable internet connection for cloud-based speech recognition (**Web Speech API**[1]) and compilation (**Paiza.io/JDoodle APIs**[3], [4]).*
- *It is assumed that the user's browser supports **Web Speech API**[1] (Google Chrome, Microsoft Edge, or Firefox latest version).*
- *The system depends on the availability and reliability of the external APIs; any downtime may affect performance.*
- *The parser logic is designed according to C++17 standards.*
- *The development process relies on iterative testing, following a **practitioner's approach to software engineering**[6].*

7. References

[1] Mozilla Developer Network (MDN): Web Speech API. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API

[2] Monaco Editor – Microsoft Open Source Project. [Online]. Available: https://microsoft.github.io/monaco-editor

[3] Paiza.io API Documentation. [Online]. Available: https://paiza.io/en

[4] JDoodle API Documentation. [Online]. Available: https://www.jdoodle.com/compiler-api

[5] IEEE Recommended Practice for Software Requirements Specifications, IEEE Std 830-1998, 1998.

[6] R. S. Pressman, Software Engineering: A Practitioner's Approach, 9th ed. New York, NY, USA: McGraw-Hill, 2019.

[7] OWASP Foundation. "OWASP Top Ten Web Application Security Risks." [Online]. Available: https://owasp.org/www-project-top-ten/

[8] World Wide Web Consortium (W3C). "Web Content Accessibility Guidelines (WCAG) 2.2." [Online]. Available: https://www.w3.org/TR/WCAG22/