Software Requirements Specification

for

Guftaar

version 1.0

Prepared by Group 04

Bakhtawar Ahtisham

Emaan Atique

Emaan Bilal

Harris Ahmad

Romessa Shahjahan

Saad Sher Alam

Date: 17-02-23

Effort: 20 hours

Instructor Maryam Abdul Ghafoor

SBASSE-LUMS

Contents

1 Introduction

1.1 Document Purpose

This document is meant to detail the complete functional and non-functional requirements as part of the Software Requirement Specifications for Project Guftaar, the semester-long project for the CS 360: Software Engineering course at the Lahore University of Management Sciences. Guftaar is a web application that provides virtual multi-feature treatment support, designed for PWS (people who stutter) in the English language. The document first covers the motivation for an mHealth solution that supports PWS, explaining the proposed product functionality as well as the scope of the project, then going on to explain the interface requirements and detailing key use case implementation details.

By clearly defining and documenting the functional and non-functional requirements of the application, this SRS document serves as a blueprint for the team to design and develop the application, as well as for the stakeholders (course staff) to understand the scope and goals of the project. All in all, this document solidifies the roadmap for the successful completion of this project, and the metrics to measure its fulfilment.

1.2 Product Scope

In light of the impact a communication deficit can create in the lives of those affected, this project is meant to serve as a technological intervention for young adults that supports speech and language therapy in the form of a web application. The platform is intended to empower those struggling with articulation with an opportunity to manage their fluency through an interactive, easy to use interface that provides resources in the form of exercises, progress tracking, live sessions as well as a way to connect with certified instructors for individual training, and course sign ups.

In the study of existing literature, and through PWS interaction, the need for a holistic technological support system was validated as a necessary shift from traditional speech therapy

in the country. Conventional support was quoted to be expensive, ranging from PKR 2000 to PKR 5000 per session, outdated, prone to relapse (80% occurrence) and laborious for all parties involved [1]. Resources also appeared scattered, progress unregulated and access to certified help difficult to reach. Subsequently, a tech-enabled platform where end users could find a host of resources to assist their personal speech improvement journey, and access existing support mechanisms in the form of experienced professionals is an essential layer of help currently absent in the local market. Guftaar is a project meant to accomplish just that.

1.3 Intended Audience and Document Overview

This document provides a detailed overview of Project Guftaar, including its goals, objectives, targets, functional and non-functional requirements, as well as a comprehensive use case view. As a result, it serves as a comprehensive guide for developers to follow, plan and monitor their progress using the specification details of each functionality and use case. The project lead and advisor can also use this document to fully understand the project's scope and requirements and manage resources efficiently, such as workers, time, and finances. Additionally, our client (the course staff) can refer to this document to assess the completeness of the project at the end of the semester and compare the final product with the requirements and functionalities outlined in this document.

To start with the development process of the Guftaar system, developers should begin by referring to Section 2, which provides a brief overview of the project and a summary of the product functions, as well as the Data Flow Diagram to help understand the relationship between the functions. Next, they should carefully review the detailed specifications given in Section 3 for each functional requirement, and refer to the use case diagram to understand how different users will interact with the system. In addition, Section 4 should be consulted to understand the non-functional requirements of the project and to review the implementation plan to ensure that the final product meets all the requirements specified in this document. For testers, course staff, and clients, it is recommended that they read this document carefully to fully understand the project goals and promised functionalities of the Guftaar system. Then they can compare these promises with the final product deployed by the team to make a fair

evaluation. To understand any technical terms used in this document and the way functionalities have been implemented based on major assumptions, Section 1.4 for definitions and Section 2.4 for assumptions can be referred to. For managers, leaders, and advisors, it is suggested that they read this document thoroughly in the given order to have a complete understanding of the project goals, requirements, and functionalities.

1.4 Definitions, Acronyms and Abbreviations

Term	Definition
Administrator	Person monitoring and managing the web application at the backend.
APIs	Application Programming Interface- It refers to a set of protocols, routines, tools, and standards that developers use to build software applications and interact with other software components or systems.
Calendly	An open-source calendar-based platform that allows you to schedule and book appointments
Client	In this document, reference to a client is specifically for the PWSs.
Coach	Person trained enough to assist a younger individual improve a specific skill. In our context, a coach is a Speech Language Pathologist (SLP).
Concurrently	This term refers to the ability of the website to handle multiple requests or actions at the same time. This means that multiple users can interact with the website simultaneously, without having to wait for their turn or experiencing delays.
D3.js Library	Data Driven Documents (D3.js) is a JavaScript library for creating dynamic, interactive data visualizations in web browsers.
Data Flow Diagram	A graphical representation of a system or a process that shows how data flows through different components or modules of the system. It is a structured analysis and design technique that helps to depict the flow

	of data, inputs and outputs of a system, and the processing and storage of data.
Database	A collection of data organized into rows and columns providing a backend storage and retrieval system for data used by the application.
Encrypted String	A string of characters that has been transformed using an encryption algorithm to make it unreadable by anyone who does not have the key to decrypt it.
Figma	A web-based collaborative design tool that allows users to create and share user interface designs, graphics, and prototypes.
Firebase	Cross-application platform to help build and deploy mobile and web applications, provided by Google.
Functional Requirements	A set of specifications that define what a system, software, or product must be able to do or achieve. They describe the specific features, functions, capabilities, and behaviors of the system that are required to meet the needs of the users or stakeholders.
Git/Github	A version control system and web-based hosting service that enables developers to collaborate on software projects and track changes to code over time.
GUI	Graphical User Interface is a type of user interface that allows users to interact with electronic devices or software applications using visual elements such as icons, menus, and windows
Heroku	Cloud platform used to host multiple web applications.
Integrated Commercial Components	A set of third-party software components that are integrated or used in a larger software system or application, typically for commercial or business purposes.
Javascript	A high-level programming language that is widely used for both

	front-end and back-end web development. It is a client-side scripting language that runs in a user's web browser and is used to create interactive web pages, web applications, and mobile apps.
Libraries	Collections of pre-written code that developers can use to add specific functionalities and features to their software applications without having to write everything from scratch.
Linux Operating System	A free and open-source operating system that manages a computer's hardware and provides a platform for running other software applications.
localStorage object	It allows JavaScript sites and applications to store key-value pairs in a web browser without an expiration date. As a result, the information stored in the browser will persist and remain intact even after the browser window is closed.
MediaStream Recording API	A web API that allows web developers to record media streams from a user's device, such as audio and video streams captured from a user's camera and microphone.
Metadata	Data that describes or provides information about other data. It provides context and additional information about a piece of data, such as the format, author, date created, and keywords associated with it.
mHealth	A term used for the practice of medicine and public health which is supported by mobile devices.
Modems	A device that converts digital data from a computer or other digital device into analog signals that can be transmitted over a telephone line, cable, or other communication channel, and vice versa.
MongoDB Atlas	A cloud-based database service that allows developers to store and manage data in a flexible and scalable manner.
Non-functional Requirements	A set of specifications that define how a system, software, or product

	should behave, perform, or operate, rather than what it should do. They are concerned with the overall quality, usability, reliability, and security of the system.
Pocketbase	Cross-application platform to help build and deploy mobile and web applications.
Premium Courses	Courses for which one has to pay to get access.
Prepared Statements	A feature of programming languages that are used to send SQL queries to a database. Prepared statements are used to execute SQL queries more efficiently and securely by separating the SQL query from the data values that are used in it.
PWS	Person who stutters.
ReactJS	It is an open-source JavaScript library for building user interfaces
Routers	A network device that forwards data packets between computer networks. It is commonly used in home and office networks to connect multiple devices to the internet and to each other.
Serverless	Applications that do not necessitate the use of any server-side code such as ExpressJS (JavaScript).
Speech Impediments	A wide range of speech disorders or difficulties that affect the ability to speak clearly and fluently. They can be caused by various factors, such as developmental issues, physical or neurological conditions, or injuries.
Speech Processing Techniques	A set of methods and algorithms used to analyze, understand, and manipulate speech signals. They involve the use of signal processing, machine learning, and natural language processing techniques to perform tasks such as speech recognition, speech synthesis, speaker identification, and language translation.

Speech Therapy	A type of therapy that helps individuals improve their speech, language, and communication abilities.
SQL	Structured Query Language is a programming language used for managing and manipulating relational databases. It is the standard language used for interacting with relational database management systems. SQL queries are commands written in SQL to retrieve, update or delete data in a database.
SQL Injection Attack	A type of web application security exploit that takes advantage of vulnerabilities in a web application's database layer. In this type of attack, a malicious user tries to insert SQL code into an input field or URL parameter of a web application, with the goal of accessing, modifying or deleting sensitive data stored in the application's database.
Streak	A prolonged period of success or failure in a particular activity for consecutive days.
Stuttering	A speech impediment characterized by disruptions in the fluency of speech, such as repetitions of sounds, syllables, or words, prolongations of sounds, and interruptions in speech known as blocks.
Supabase	Cross-application platform to help build and deploy mobile and web applications.
Switches	A network device that connects multiple devices on a computer network, allowing them to communicate with each other.
Traffic Spikes	Traffic spikes refer to sudden, unexpected surges in the amount of traffic to a website or web application.
Tutorials	Explanation of a subject or a task especially via interactive sessions or videos.
Use Case Diagram	A diagram that is used to depict the interactions between a system and its users or external systems. It provides a high-level view of the

	system's functionality from a user's perspective and shows how the system is used to achieve certain goals.
User Experience	This refers to the overall experience that a person has when using a product, system, or service. This experience includes a person's perceptions, feelings, and attitudes towards the product, as well as the ease of use and efficiency of the product in meeting their needs.
VS Code	A source code editor with features like syntax highlighting, debugging, and code completion. It can be used for various programming languages.
Web Browser	Software applications used to access and view web pages on the internet. They act as an interface between the user and the web, allowing users to navigate and interact with different websites by processing and displaying web content such as text, images, videos.
Windows Operating System	A Windows OS provides a graphical user interface (GUI) that allows users to interact with their computer through icons, menus, and other visual elements. It also provides a wide range of tools and features that allow users to perform various tasks, such as browsing the web, creating and editing documents, playing multimedia content, and running software applications.

Table 1: Definitions

1.5 References and Acknowledgments

[1] “In Pakistan, Lack of Trained Speech Therapists Feeding Quackery | the Express Tribune.” *The Express Tribune*, 21 Oct. 2019, tribune.com.pk/story/2083912/pakistan-lack-trained-speech-therapists-feeding-quackery.

2 Overall Description

2.1 Product Perspective

Guftaar is intended to be an English language m-Health web application catering to PWS (people who stutter), with virtual multi-feature treatment support, detailed in the following section. The application is meant to be an online platform providing at-home self management and professional assistance to those diagnosed, with instructional training sourced from established speech therapists, and peer-to-peer support via online and blog interactions. It is a new, self-contained product, whose complete feature details and requirement specifications are outlined in this document.

Guftaar hosts a breadth of features, extrapolated on the basis of these objectives.

- Self Management Assistance
- Sustained User Motivation
- Community Support
- Credible Resource Sharing

Based on our product functions (specified in the following subsection), we have divided our system into the following sub-systems:

1. **User Management**: This subsystem is responsible for managing user accounts, including registration, authentication, and authorization.
2. **Speech Therapy Content Management**: This subsystem is responsible for managing speech therapy content, such as daily activity tasks, speech techniques, and quick practice.
3. **Audio Management**: This subsystem is responsible for recording, analyzing and processing speech input from clients under the quick practice activities.
4. **Progress Tracking**: This subsystem is responsible for tracking user progress and providing feedback to users by generating reports on user performance.

5. **Scheduling**: This subsystem is responsible for facilitating communication between users and therapists such as managing coach availability, scheduling meetings and giving meeting reminders.
6. **Payment and Billing**: This subsystem is responsible for managing user payments and billing for the premium speech therapy courses.
7. **Coach Management:** This subsystem is responsible for maintaining coach details.
8. **Admin Logistics:** This subsystem is responsible for managing admin privileged tasks such as creating coach accounts, modifying the blog page and updating coach ratings.
9. **Client Sphere:** This subsystem is responsible for maintaining client facing interactions with the Guftaar application, like logging daily progress, displaying encouraging strength statements and blog content, along with facilitating the collection of client feedback on coaches.

The different subsystems are interconnected and work together to provide a seamless experience for the clients and coaches alike, from registration and authentication to progress tracking and billing. For example:

- The User Management subsystem interacts with almost all other subsystems as it is responsible for authenticating users and determining their level of access within the system.
- The Progress Tracking subsystem and the Speech Therapy Content Management subsystem interact when generating progress reports for clients. The Progress Tracking subsystem retrieves data on the client's performance from the Speech Therapy Content Management subsystem, and uses this data to generate a report for the client to view.
- The Schedule a Session subsystem interacts with the Coach Management subsystem to allow clients to select a coach, view their availability and schedule a meeting.
- The Speech Therapy Content Management subsystem interacts with the Audio Management subsystem to rate the user progress for the audio uploaded in the quick practice activity. The Speech Therapy Content Management subsystem selects the text

to be read by the client, and the Audio Management subsystem records the client's voice and analyzes their speech to provide feedback on their performance.

- The Coach Management subsystem may interact with the Progress Tracking subsystem to provide client progress reports to the therapists if allowed by the client.
- Admin Logistics subsystem interacts with the coach management subsystem to create coach profiles as well as the ClientSphere subsystem to view client coach evaluations and subsequently, update their ratings.
- ClientSphere Interactions between the admin logistics subsystem and the ClientSphere subsystem so a system is created where client feedback is valued with the thorough feedback review process.

These subsystem interactions are displayed in the following diagram[1]:

[1] While the user management subsystem plays an integral role in our system to authenticate and authorise user access, thus interacting with all other subsystems, it has not been shown in Figure 1 for simplicity's sake

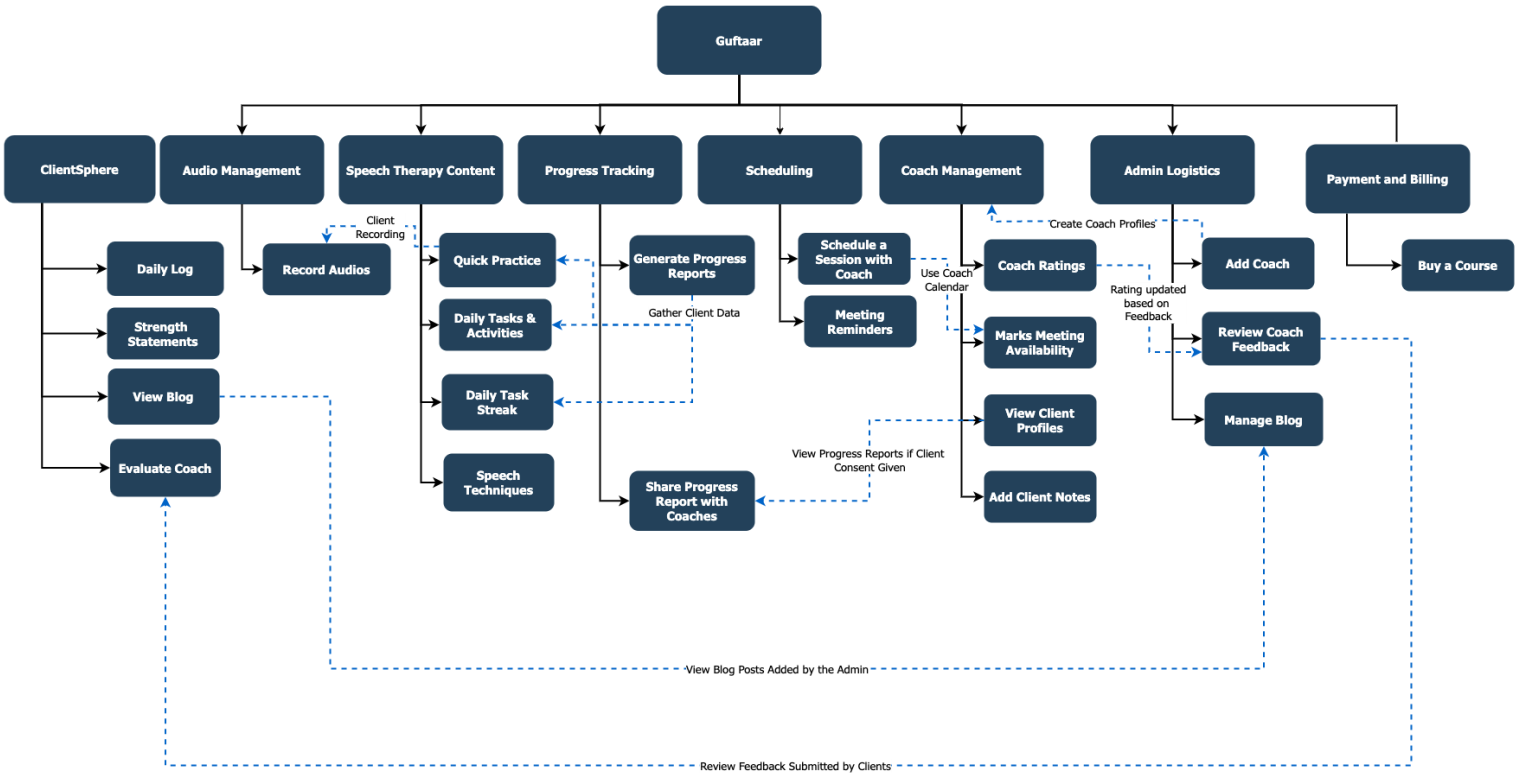

Figure 1: Subsystem Interactions[2]

Figure 2 outlines the various interactions between the different users of our system (detailed description of each user given in Table 3, Section 2.3). The dotted arrow for the 'Share Progress

[2] The solid black arrows show different functionalities that fall within each subsystem and the dotted blue arrows show interactions between different subsystems

Report' interaction is an indication of the functionality being limited to only those client-coach interactions where consent for progress sharing has been given by the client.

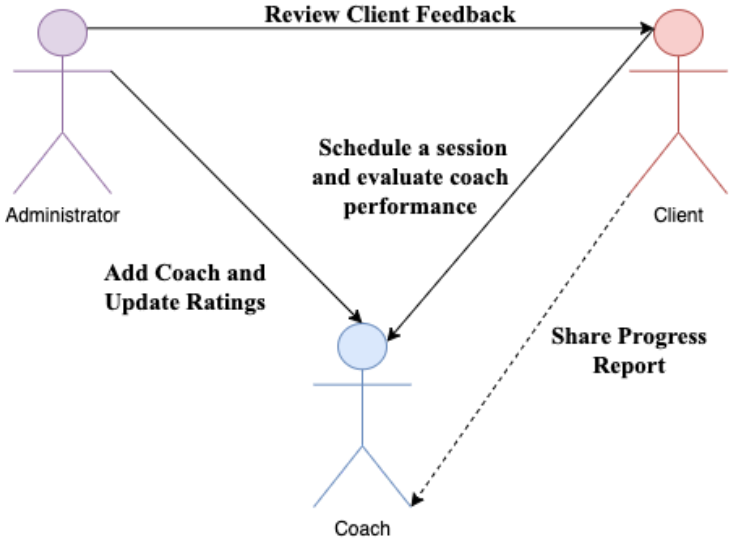

Figure 2: User Interactions

2.2 Product Functions

Our functionalities have been divided into 5 broad categories, represented in the table below:

Functionality Type	**Features**
Register/Log-in	• Client Sign Up • User Login • Update Password • Create an account for coaches and admins (*only administrators*)
Independent Speech Support	• Daily Tasks and Activities • Daily Task Streaks • Speech Techniques • Quick Practice
Guided Speech Support	• Schedule a Session with a Coach • Get Meeting Reminder • Buy a Course • Mark Meeting Availability (*only coaches*)
Feedback and Evaluation	• Generate client progress charts • Evaluate Coaches • Add Client Notes (*only coaches*) • Review Coach Feedback (*only administrators*)
SelfSphere	• Daily Log • Strength Statements

Table 2: Product Functions and Categories

The data-flow diagram below represents how these functionalities relate and interact with each other.

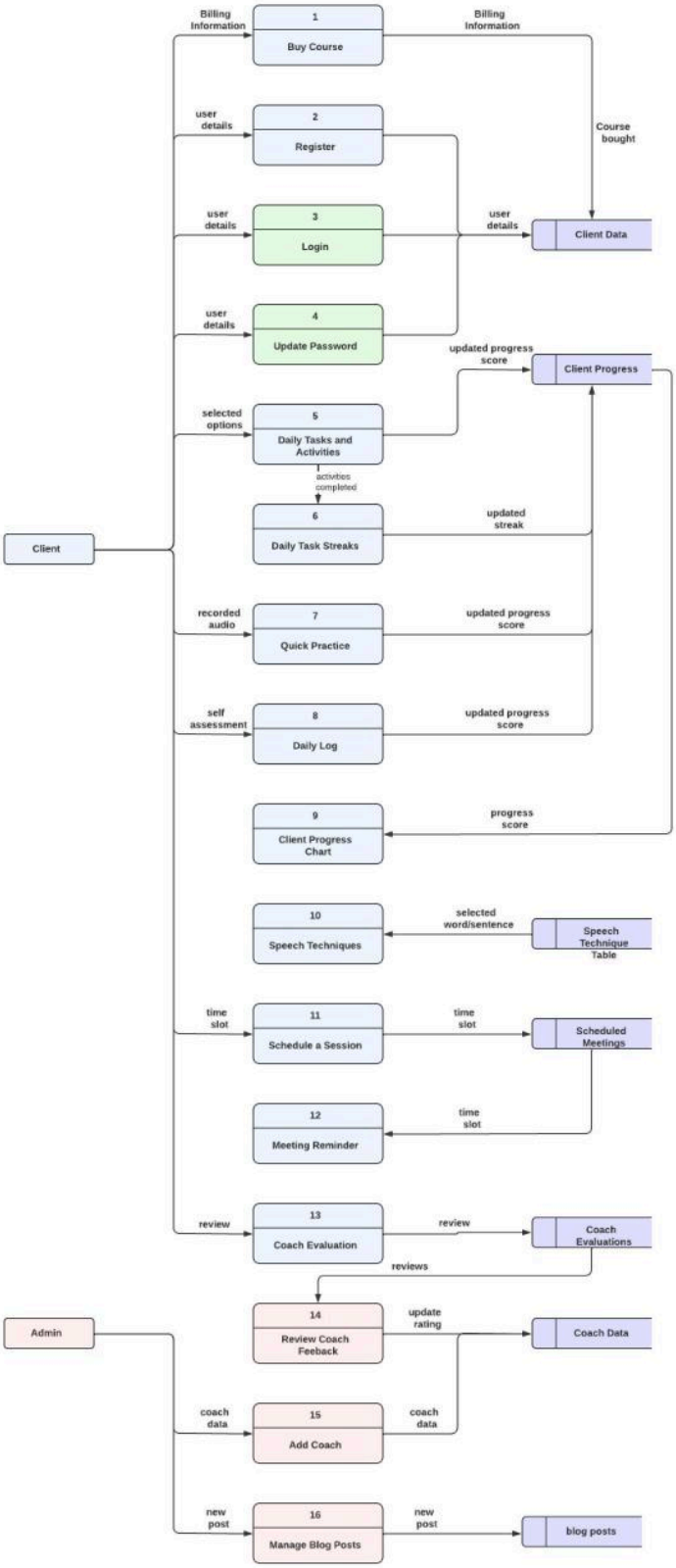

Figure 3: Data Flow Diagram

2.3 Users and Characteristics

Guftaar end users include both PWS and coaches along with the system administrators, whose roles and app involvement is explained in the table below (users listed in order of importance- based on frequency of use and level of interaction with the portal).

End-User	Function
PWS	Access guided training, course material and auditory support with progress measures, community support via live sessions and certified medical help with in-app incentive mechanisms
Coaches	Connect with PWS according to availability to deliver virtual support via online sessions and facilitate speech therapy via feedback logging and progress monitoring
Administrator	Manage existing resources, profiles and content hosted on the application along with maintaining the application database and monitoring in-app activity

Table 3: System Users

Guftaar is intended for homogenous use across all age groups, however, a conscious focus remains on young adults, aged 13+, adept at independently managing their condition to help streamline, automate and guide the treatment process to be more informed, supportive and effective. This user group is assumed to have a strong grasp over the English language, and access to digital tools like a desktop computer to access the site. Previous experience with speech and language therapy is not required but familiarity with important terminology common across disfluency treatment is recommended.

Younger users like children and adolescents are encouraged to engage with the application in collaboration with guardians to extract maximum use and efficacy. Application features are intentionally kept simple, easy to follow and interactive for all age brackets to derive benefit.

Early stage intervention, i.e. for children and toddlers, is deemed to be specialized, nuanced and beyond the scope of this application which is meant to provide an all-encompassing and generalized m-Health solution given a base exposure to treatment options or preliminary experience in managing speech difficulty.

2.4 Assumptions and Dependencies

2.4.1 Assumptions

- Guftaar is designed to provide speech therapy for the English language, and therefore it is presumed that all users have fluency in English.
- Guftaar is a digital service, therefore it is assumed that clients have access to the Internet and an electronic device (mobile phone, laptop, desktop computer).
- Guftaar is a web-service, therefore it is assumed that navigation of the web and interaction with web pages will not be a hurdle for the client.
- To interact with the platform through the ‘Quick Practice’ activities we assume that all clients have the necessary hardware and software requirements needed to record audios through the device microphone.
- For the ‘Quick Practice’ activities we assume that clients always submit a legitimate attempt at recording the paragraph. For example, the client does not submit a short empty recording to receive a high score.
- Guftaar only allocates a single account per client.
- Since Guftaar supports live interactive sessions, it is assumed that all clients have agreed to a disciplinary and ethical code of conduct and that the integrity of the service is maintained at all times.
- Our design for this project is oriented towards providing speech therapy to our clients. We therefore assume that all other processes of the system, such as recruitment of the coaches, is done independently. And once a coach is selected

to join the team, a member of the administration is responsible for making their account using the Add Coach functionality provided.

- For the Daily Tasks and Activities, we assume that clients will attempt the three tasks only once per day.

2.4.2 Dependencies

Our platform will utilise publicly available APIs for features like scheduling and blog management, as well as audio and video content. As such, the external software that the Guftaar platform will be dependent on include:

1. Calendly *(scheduling)*
2. WordPress/Notion *(blog management)*
3. Google Drive *(media integration)*
4. Youtube *(media integration)*

2.5 Operating Environment

1. **Hardware Platforms:** The web application will work on all computing devices including mobile smartphones that have a web browser installed capable of running and rendering JavaScript code, support for connection to the Internet and a working microphone. Details in section 3.2.2
2. **Operating System:** The web application supports all operating systems listed in section 3.2.3
3. **Software/Application Components:** The software will successfully operate on all browsers that are compatible with JavaScript and able to retrieve integrated commercial components such as (Calandly, Wordpress, YouTube, Google Drive). This application will work on all computing devices including mobile smartphones that have a web browser installed capable of running and rendering JavaScript code.

3 Specific Requirements

3.1 Functional Requirements

Category 1: Register/Login

RQ 1.1: Register on the Guftaar platform

Description:

Clients will be able to make an account on the platform to avail services enlisted on our web application.

Input:

1. First name
2. Last name
3. Gender
4. Age
5. Email
6. Username
7. Password

Processing:

Create a new record in the database to save the client's information. Save the client's password as an encrypted string.

Output:

Client's account is created and they are automatically logged into the home page.

RQ 1.2: Log in to the Guftaar platform

Description:

Users will be able to log in on the platform to avail services enlisted on our web application and access their account.

Input:

1. Email
2. Password

Processing:

Check the password against the provided email in the database.

Output:

User enters the web application and is successfully logged in.

RQ 1.3: Update Password

Description:

Users will be able to change their password.

Input:

1. Old password
2. New password

Processing:

The system will check against the record in the database to verify the old password, and upon confirmation, update the password entry to the new one specified by the user.

Output:

The user's password is updated and the corresponding record in the database reflects this change.

RQ 1.4: Add Coach

Description:

System administrators can create account profiles for new coaches joining *Guftaar*. A unique username and password will be assigned to the coach at this stage which they can use to log in to the portal later. Coaches will have the option to update their password using the *update password (RQ 1.3)* feature.

Input:

1. First name
2. Last name
3. Gender
4. Age
5. Email

6. Username
7. Password
8. Years of Experience
9. Maximum Qualification

Processing:

Create a new record in the database to save the coach's information. Save the password as an encrypted string.

Output:

A message is printed on the screen reflecting the successful creation of an account for a coach.

The following functional requirements are predicated on the user being successfully logged in.

Category 2: Independent Speech Support

RQ 2.1: Daily Tasks and Activities

RQ 2.1.1: Syllable Counting Activity

Description:

Clients will be presented with three words, segmented by the difficulty level, and will be asked to identify the number of syllables in each in order to improve phonological awareness, articulation, and speech production skills.

Input:

Select the 'Syllable Counting Activity' button displayed on the dashboard. Once each question is displayed, the client will choose one option of those visible on the screen.

Processing:

The system will randomly choose three words from the easy, medium and hard categories respectively. Once the client selects an option, check if it's the correct answer. After completion of the task, the system will update the progress table in the database.

Output:

Each word will be displayed on the screen along with 4 multiple choice options for the client to select. On selection, the system will highlight the correct answer and a "Next" button will appear after the first two questions.

RQ 2.1.2: Breathing Exercise

Description:

Clients will follow the '7-7-7' breathing strategy on repeat till the specified time elapses.

Input:

Select the 'Breathing Exercise' button displayed on the dashboard. Then, select the desired time for the breathing activity from the provided options.

Processing:

On the basis of the option selected, the system will segment the total time into 7 second long intervals, each with a separate timer.

Output:

The system will display the timer along with the corresponding instructions.

RQ 2.1.3: Linklater Voice Progression

Description:

Clients will follow cheek and jaw relaxation exercises as specified on the screen.

Input:

Select the 'Linklater Voice Progression' button displayed on the dashboard.

Processing:

The system will run a timer for each facial movement, and transition to the next expression on each interval's completion.

Output:

The system will visually indicate the movement required along with the time for each movement.

RQ 2.2: Daily Task Streaks

Description:

The system will maintain a streak for all clients on the completion of their daily tasks. Users with a month's streak can access a premium course free of any cost.

Input:

N/A

Processing:

Whenever a client logs in, the system checks whether the last updated streak count was more than 24 hours ago, if so the streak value for that user is updated to 0 in the database and subsequently displayed on the dashboard. Otherwise, the streak value remains unchanged.

For each assigned task for the day, upon completion, a localStorage object is updated. If all three daily tasks are completed, the streak value for that user is incremented by one. This change is reflected in the database with the last updated field changed to the current date and time.

Output:

The streak value retrieved from the database for the user is displayed on the dashboard.

RQ 2.3: Speech Techniques

Description:

Clients will be provided tips and audio guided practice on how to articulate words and sentences clients generally struggle with.

Input:

Select the 'Speech Techniques' button displayed on the dashboard.

Next, select the word of choice.

Processing:

The system will fetch the requisite audio and play upon selection of a word.

Output:

Audio will be played illustrating the enunciation of the word chosen.

RQ 2.4: Quick Practice

Description:

Clients will be provided with a short paragraph which they will read out and record. System will grade the client's attempt through metrics such as the time taken and assign scores.

Input:

Select the 'Quick Practice' button displayed on the dashboard.

Select the start and stop recording button.

Processing:

The system will randomly choose a paragraph saved in the database and use the MediaStream Recording API to enable client recording. Based on the duration of the recorded audio the system will assign a score between 1-5 and update the client's stored average rating accordingly.

Output:

The chosen paragraph is displayed on the screen and upon completion and processing, the system displays the client's recorded score.

Category 3: Guided Speech Support

RQ 3.1: Schedule a Session

Description:

Clients will be able to view coach profiles and pick a coach of choice to schedule a one-on-one session. The clients will select an available time slot for the session based on the calendar of the coach.

Input:

Select the coaches button in the navbar to view profiles for all available coaches. Next, select the coach of choice and click the 'schedule a meeting' button. Pick an available time slot for the meeting from the displayed calendar.

Processing:

The system will update the coach calendar to include the newly scheduled meeting. The system will also store a record in the database reflecting the time for any upcoming meeting for each client so that a meeting reminder can be displayed.

Output:

The system will display the updated calendar of the coach with the newly scheduled meeting for the client.

RQ 3.2: Meeting Reminder

Description:

If a client has a one-on-one session scheduled with a coach, the system will display a countdown on the homepage indicating the time till the upcoming meeting. For example: *'Meeting with coach <John Doe> starts in XX:YY:ZZ.'*

Input:

N/A

Processing:

Upon login, the system will fetch the time for the next scheduled meeting from the database for the client. Based on the timestamp, the system will generate a countdown.

Output:

The system will display the countdown on the homepage.

RQ 3.3: Buy Courses

Description:

Clients will have access to premium courses which they can add to the shopping cart and buy.

Input:

Navigate to the 'Courses' tab through the navbar. Click on the 'View Details' button for the course of choice. Next, click on the 'Add to Cart' button. In case the client is interested in buying the course, click on the 'Proceed to Checkout'

button and enter billing information. Else, click on the select the course from the cart and click on the 'Remove from Cart' button.

Processing:

In case a client buys a course, the system will get billing information through the form displayed and save the information in a database.

Output:

The client will gain access to the newly bought course. The system will display the newly bought course under the 'My Courses' tab on the 'Courses' tab.

RQ 3.4: Meeting Availability

Description:

Coaches will mark available time slots on their respective calendars for one-on-one meetings with clients.

Input:

Coaches will select pick time slots on the calendar for which they are available.

Processing:

The system will update the coach calendar to incorporate the time slots that are available for one-on-one sessions.

Output:

The updated calendar will be displayed to the clients under the 'Coaches' tab after the select 'Schedule a Meeting' with a coach.

Category 4: Feedback and Evaluation

RQ 4.1: Client Progress Charts

Description:

Clients will be able to view performance feedback based on their quick practice scores, streak number, daily self assessment as well as feedback from instructors. They will also have the option to share their progress with their coaches.

Input:

Select the 'Progress Charts' button displayed on the dashboard.

Processing:

The system will fetch the user's progress data, as stored in the database, and visualise this information in graphical charts using the D3.js library.

Output:

The client's progress is displayed on the screen in chart and graph form. A button to share this data with their coach will also be visible.

RQ 4.2: Coach Evaluation

Description:

Clients can provide feedback on coaches they have had sessions with commenting on how satisfied they are with the help being provided.

Input:

Select the 'Add Coach Evaluation' button displayed on the dashboard.
Select the coach the client wishes to review from a dropdown menu.
Fill in the form displayed by the system.

Processing:

Upon submission of the form, save the client responses in the database for the administrator to review later.

Output

The system displays a dropdown menu with the names of coaches for the client to select. Once a coach is selected, the system displays the evaluation form. The system then displays a message on the successful submission of the evaluation form.

RQ 4.3: Add Client Notes

Description

Coaches will be able to add notes for their clients, to keep a record of client progress, visible only to the coach themselves.

Input

Select the 'Client Notes' button displayed on the dashboard.

Next, choose the client they wish to add notes for, and type as text in the box that appears. Once complete, press the “Save” button.

Processing

The system will create a record in the database with the note added, client and coach name

Output

The system will display the note added on the screen, along with previously added notes for that client.

RQ 4.4: Review Coach Feedback

Description

Administrators will have the ability to view coach evaluation notes submitted by clients and will also be able to update the rating for each coach in light of this feedback.

Input

Select the ‘Coach Feedback’ button displayed on the dashboard.

Next, select the coach whose feedback is to be viewed.

Optionally, press the “Update Rating” button to change the 5 scale rating of the chosen coach.

Processing

The system will fetch the reviews submitted by clients for that coach from the database. If rating is updated, the system will update the corresponding rating field for the coach in question in the database.

Output

The system will display all reviews submitted for the selected coach in order from latest to earliest, along with client names.

An “Update Rating” button will also be displayed, which when clicked, will display a field in which the administrator can enter the coach’s new rating, which can be a minimum of 0, and a maximum of 5.

Category 5: SelfSphere

RQ 5.1 Daily Log

Description:

Clients can log their daily personal review by assessing how their stuttering was throughout the day: No stuttering, extremely mild, moderate, frequent, or severe.

Input:

Select the 'Daily Assessment' button displayed on the dashboard.

Choose one of the displayed options and select the 'done' button to save the response.

Processing:

The system stores the client's response in the database, incrementing the count of the chosen category by 1, to be used for the progress report.

Output:

The daily assessment page is displayed asking the clients how their stuttering was today and displaying the following options:

1. No Stuttering
2. Extremely Mild
3. Moderate
4. Frequent
5. Severe

RQ 5.2 Strength Statements

Description:

Clients can view positive affirmations once they login which can help with negative thought patterns.

Input:

N/A

Processing:

Each time a client logs in the system randomly selects a quote from the database.

Output:

The chosen quote is displayed on the client's dashboard

RQ 5.3 Blog Page

Description:

All users can access informative articles, blog posts and videos categorised into different groups (stuttering, treatment, motivation, information etc)

Input:

Select the 'GuftaarBLOG' button displayed on the dashboard to view posts.

Processing:

Retrieve blog posts and contents from the database.

Output:

Display the content on the blog page.

RQ 5.4 Manage the Blog Page (TBD)

Description

Administrators will be able to add and remove blog entries that are part of the blog page on the Guftaar platform.

Input:

Select the 'GuftaarBLOG' button displayed on the dashboard.

Interact with the Wordpress API to create a new post or delete an existing post.

Processing

The system will load the external Wordpress API once the "GuftarBlog" button is clicked

Output

The system will display the blog posts once the button is clicked, and will display the updated feed once a new post is added or deleted.

3.2 External Interface Requirements

3.2.1 User Interfaces

Given that the primary aim and purpose of our project is to provide a platform where those looking to improve their speech articulation can self-reliantly access certified speech therapy services, a graphical user interface is a primary requirement for the ease of our users. Therefore, our webportal will be built on ReactJS (a framework of JavaScript), and will support GUI.

We will have a standard navigation bar at the top with all the navigation links to the main features of our web application. Whenever the user is required to input commands and information, our system will ensure to give them feedback in a timely manner upon incorrect commands. In addition, the application will be able to handle any errors and provide helpful instructions for resolving the issue. Our design goal is to aid all our user functionalities with interactive icons, graphics and multimedia content (audio and video support) to make speech therapy a seamless process with *Guftaar*.

3.2.2 Hardware Interfaces

The web application will work on all computing devices including mobile smartphones that have a web browser installed capable of running and rendering JavaScript code. Moreover, for smooth loading of the web page, it is advisable to access the portal on devices that have more than 2GB RAM. In addition, like every other website, our application requires a stable internet connection which necessitates the compulsion of having the relevant hardware components such as routers, switches and modems. A steady internet connection and such storage requirements are especially needed to support the graphical user interface requirements specified in section 3.2.1. To seamlessly interact with the services provided by our platform, a functioning microphone alongside a keyboard and mouse will also be necessary.

3.2.3 Software Interfaces

This product has connections to the following software components and databases:

- Windows operating system (7,8,10,11).

- Linux operating system (Ubuntu/16.04^, Fedora (Core/Linux) 1^, Elementary OS, MacOS 10.14^, etc).
- Libraries (ReactJS, ExpressJS, D3.js, etc).
- Tools (VS Code, Git/ GitHub, MongoDB Atlas, Figma, etc).
- Integrated commercial components (Calandly, Wordpress, YouTube, Google Drive)
- Browsers: Preferably Google Chrome or any other browser compatible with JavaScript

The web application runs in the web browser, mediating its interaction with the OS through the use of browser APIs, including the microphone. This is typically implemented using web technologies such as JavaScript, HTML, and CSS, which are displayed in the browser window. When the user interacts with the app, the browser generates events that are sent to the app for processing. The app then responds to these events by either updating the user interface or making requests to the browser's APIs in order to interact with the OS.

Overall, Guftaar’s interface with the operating system is mediated by the web browser, which provides a simplified and secure way for the app to interact with the user and access the features of the underlying OS.

3.3 Use Case View

3.3.1 Use Case Table

Primary Actor	Associated Use Case
Clients	Registration Daily Tasks and Activities Daily Task Streaks Speech Techniques Quick Practice Client Progress Charts Coach Evaluation Daily Log Strength Statements Schedule a Session Meeting Reminder Blog Page Buy Courses
Coaches	Meeting Availability Add Client Notes View Client Profiles
Admin	Add Coach Manage the Blog Page Review Coach Feedback

Table 4: Use Case Table

3.3.2 Use Case Diagram

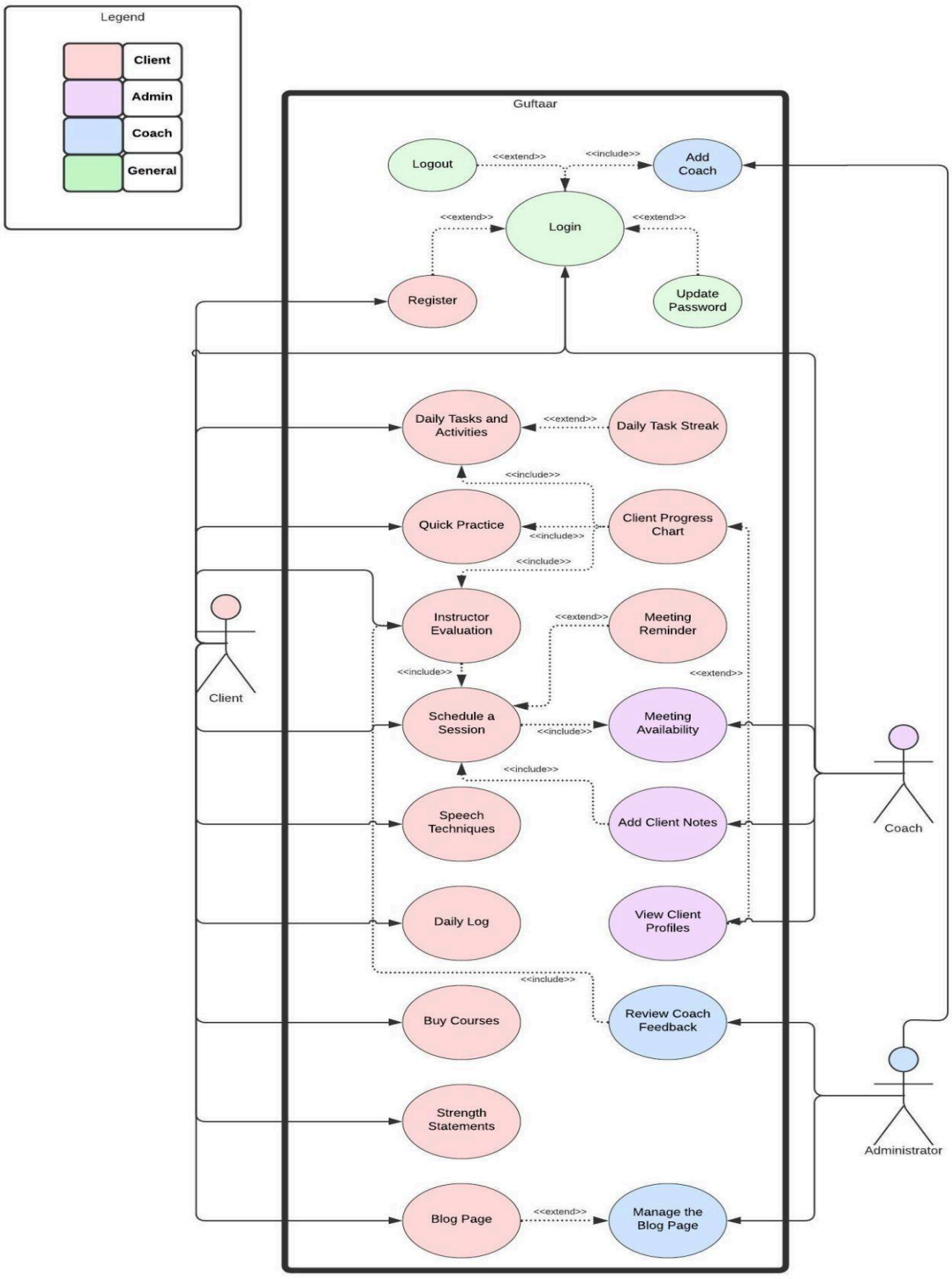

Figure 4: Use Case Diagram

3.3.3 Use Case Description

<table>
<tr><td colspan="2">Use Case: Daily Tasks and Exercises</td></tr>
<tr><td colspan="2">Scenario: Syllable Counting Activity</td></tr>
<tr><td colspan="2">Description: Clients will be presented with three words, segmented by the difficulty level, and will be asked to identify the number of syllables in each in order to improve phonological awareness, articulation, and speech production skills.
Actors: Clients
Precondition: Clients should be registered, logged in to their accounts, and should not have completed the activity in the current login session.
Main Scenario:</td></tr>
<tr><td>User</td><td>System</td></tr>
<tr><td>1. The client selects the ‘Syllable Counting’ button under the daily exercises tab on the dashboard.

2. For each question presented, the client selects one option.</td><td>1. The system will choose three words from the easy, medium, and hard category respectively.

2. For every question, the system validates the answer selected by the client.

3. The system updates the progress table in the database.</td></tr>
<tr><td colspan="2">Alternative:
The client does not select any options, and navigates to some other page. In that case, the system does not update the progress table in the database.

Exceptions: None</td></tr>
</table>

Table 5.1: Use Case Description- Syllable Counting Activity

Use Case: Daily Tasks and Exercises	
Scenario: Breathing Exercise	
Description: Clients will follow the '7-7-7' breathing strategy on repeat till the specified time elapses. **Actors:** Clients **Precondition:** Clients should be registered and logged in. **Main Scenario:**	
User	**System**
1. The client selects the 'Breathing Exercise' button under the daily exercises tab on the dashboard. 2. From the provided options, select the desired time for the activity.	1. Based on the time chosen, the system will divide time into segments of 7 seconds. 2. The system will display a timer along with corresponding instructions.
Alternative: The client does not select any timer option, and navigates to some other page. System navigates to the other page. **Exceptions:** None	

Table 5.2: Use Case Description-Breathing Exercise

Use Case: Daily Tasks and Exercises	
Scenario: Linklater Voice Progression	
Description: Clients will follow cheek and jaw relaxation exercises as specified on the screen. **Actors:** Clients **Precondition:** Clients should be registered and logged in. **Main Scenario:**	
User	**System**
1. The client selects the 'Linklater Voice Progression' button under the daily exercises tab on the dashboard.	1. The system will display a timer for a facial movement, and transition to the next exercise after each interval's completion.
Alternative: None **Exceptions:** None	

Table 5.3: Use Case Description-Linklater Voice Progression

Use Case: Speech Techniques	
Description: Clients will be provided tips and audio guided practice on how to articulate words and sentences clients generally struggle with. **Actors:** Clients **Precondition:** Clients should be registered and logged in. **Main Scenario:**	
User	**System**
1. Select the 'Speech Techniques' button from the dashboard. 2. Select the word of choice.	1. The system will fetch the requisite audio from the database and play it for the client.
Alternative: None **Exceptional:** None	

Table 6: Use Case Description- Speech Techniques

Use Case: Quick Practice	
Description: Clients will be provided with a short paragraph which they will read out and record. System will grade the client's attempt through metrics such as the time taken and assign scores. **Actors:** Clients **Precondition:** Clients should be registered and logged in. **Main Scenario:**	
User	**System**
1. Clients will select the 'Quick Practice' button from the dashboard. 3. Clients will start and stop the recording to submit their attempt.	1. The system will randomly choose a paragraph from the database. 2. The system will use the MediaStreamRecording API to enable client recording. 3. The system will assign a score between 1-5, based on recording length, and update the client's stored average rating.
Alternative: The client does not submit a recording and navigates to some other page on the Guftaar portal. In that case, the system does not update the client's rating and redirects them to the other page. **Exceptional:** None	

Table 7: Use Case Description- Quick Practice

Use Case: Meeting Availability	
Description: Coaches will mark available time slots on their respective calendars for one-on-one meetings with clients. **Actors:** Coaches **Precondition:** Coaches should be added to the portal by the administrator and should be logged into their account. **Main Scenario:**	
User	**System**
1. Coaches will pick time slots on the calendar for which they are available.	1. The system will update the coach calendar, through the calendly API, based on the available time slots highlighted by the coach.
Alternative: None **Exceptional:** None	

Table 8: Use Case Description- Meeting Availability

Use Case: Add Coach	
Description: System administrators can create account profiles for new coaches joining *Guftaar*. A unique username and password will be assigned to the coach at this stage which they can use to log in to the portal later. Coaches will have the option to update their password using the *update password (RQ 1.3)* feature. **Actors:** Administrators. **Precondition:** The coach should not be an already registered coach on the Guftaar platform. **Main Scenario:**	
User	**System**
1. The admin will enter the following credentials for the coach: First name, last name, gender, age, email, username, password, years of experience, maximum qualification.	1. The system will create a new record in the database to save the details of the newly registered coach. 2. The system will display a message on the screen to reflect the successful creation of an account.
Alternative: None **Exceptional:** None	

Table 9: Use Case Description- Add Coach

4 Other Non-functional Requirements

4.1 Performance Requirements

1. The web application should respond to the user input within 0.5 seconds. This is necessary to provide a smooth user experience.
2. The web application should be able to process up to 50 sign up requests concurrently. This is to ensure that the application can handle traffic spikes without causing significant delays.
3. The web application should use encryption to protect user data and isolate every client's activity from their respective assigned coaches unless they provide their consent to share their data.
4. The web application should be able to identify at least 10 different speech impediments. This will allow the application to provide users with more accurate results.
5. The web application should be able to recognize speech in real-time with an accuracy of at least 95%. This is to ensure that the application can provide accurate results in a timely manner.

4.2 Safety and Security Requirements

1. The web application must have a system in place to detect, track, and report any suspicious or malicious activities that occur while the web application is being used.
2. The web application must be designed in such a way that users can easily recognize and report any potential risks or dangers of using the web application, including the ability to report any suspicious activity.
3. All users must be required to provide complete and accurate personal information (name, age, gender, contact information, etc.) before using the web application via HTML forms and all of our forms on the web app are resistant to SQL injection attacks.
4. No third-party source would be able to track what programs a particular user is enrolled in or who the assigned coach is since the security of our application motivates the use of session hosting instead of cookies so there is no way of tracking the activity of a given

user. Also, everything is stored in the secure clusters of MongoDB Atlas on their respective servers.

5. Some metadata will be collected since our application uses third-party client APIs such as Wordpress, Calendly, and Google Drive – albeit majority of which don't incorporate any advertisements except for Wordpress.
6. Passwords are going to be encrypted at the time of signing up, and via parameterized SQL or prepared statements, hashed passwords would get stored to the database keeping the service sanitized from malicious intent.
7. Understanding the sensitivity of speech therapy, the app should have role-based access control to ensure that users can only access the features and data that they are authorized to. It will be ensured that a client's progress reports can only be viewed by those granted access to be the client.
8. Permission to use the device microphone will be taken from the clients for activities in the *Quick Practice* section.

4.3 Software Quality Attributes

This web application must meet certain quality characteristics in order to serve its purpose for users with speech impediment. The following subsections provide requirements related to the different software quality attributes.

4.3.1 Reliability

The web application must be highly reliable and able to perform consistently and accurately with minimal errors. It should have a high availability rate and be able to withstand system failures. We will include unit testing, integration testing, and user acceptance testing to make sure that the application meets all the requirements and works as expected. Additionally, we will monitor the application for any potential issues that may arise in production. Monitoring can be done using tools such as application performance management (APM) and log analysis. Finally, we as developers should be prepared to quickly respond to any issues that arise and have a plan for addressing and resolving them. By taking these steps, web applications can be made more reliable and less prone to unexpected issues.

4.3.2 Portability

The web application must be portable and able to run on any operating system with minimal modifications. It should be designed in a way that requires minimal effort to deploy it in different environments. By making a web application more portable, developers can create a product that can meet a wide range of customer needs, and make the application more accessible to a greater number of users. Here are a few tips on how to make a web application more portable:

1. Cross-platform technologies, such as HTML5 and JavaScript, can help to make web applications more portable. By utilizing these technologies, we can create applications that are compatible with multiple platforms, such as Windows, Mac OS X, iOS, and Android.
2. Heroku is a cloud-based deploying service which means that by integrating cloud computing, we will be able to make our web applications more portable by making them accessible from any device regardless of the platform. By hosting our service on the cloud, we can ensure that our users can access the application from any device at any time.
3. By using responsive web design and relevant frameworks that enable us to employ this strategy such as TailwindCSS and Bootstrap, we can ensure that their web application looks great and is fully functional at any time.

4.3.3 Adaptability

The web application must be able to adjust and adapt to different user environments and changing requirements. It should be able to adjust to new user requirements and preferences with minimal effort. To make our web application more adaptable, we need to ensure that our service is able to respond quickly and effectively to changes in the environment. First, we will make sure that our code is as modular and extensible as possible. This will give us flexibility to quickly and easily add new features, or make changes to existing ones, without having to rewrite larger portions of code. Second, we will be working on designing and implementing a

unit-testing framework that will ensure that any changes to the code do not cause unexpected problems, which will help us identify and address any issues that may arise in a timely manner.

4.3.4 Availability

The web application must be available 24/7 with minimal downtime. It should be designed to be resilient to outages and provide uninterrupted service. First, we will ensure that our web application is optimized for speed and performance which includes a fast web server, effective caching and ensuring the codebase is optimized for performance. Since we will be using pocketbase or firebase to host our server-side, it will take care of all these factors. Second, we will be designing the web app with scalability in mind, i.e., it should be able to handle increased usage and traffic without significant performance degradation. This involves monitoring usage metrics and trends and scaling appropriately. All of these tasks could be performed by the deploying service that we shall be using to host our application.

4.3.5 Flexibility

The web application must be flexible and able to be easily changed or modified. It should be easily extendable and customizable to meet user needs. First, we will consider improving the user experience by adding customization options. This could include giving users the ability to change the font size, color scheme, or layout of the application according to their preferences. Second, we could consider adding more options for integration with other web services. This could include giving users the ability to easily connect their account with other services and allowing them to sync data between them.

4.3.6 Interoperability

The web application must be interoperable with other applications and systems. It should be designed to be compatible with existing systems to ensure seamless integration. To make our web application more interoperable, we will make sure that we use open standards and industry-recognized technologies that have been trial and tested extensively ensuring our application is compatible with as many systems as possible. Second, we will focus on web APIs that will allow for automated integration with third-party systems such as WordPress, YouTube, Google Drive, etc. Lastly, we will test our application on multiple operating systems and

browsers ranging from flagship smartphones to PCs and Laptops. By taking these steps, we can ensure that our web application is interoperable with a wide range of systems and technologies.

4.3.7 Maintainability

The web application must be easy to maintain and update with minimal effort. It should be designed with maintainability and scalability in mind to reduce the effort required for maintenance and upgrade. There are a couple of ways through which we could make our code more maintainable. First, we will create an intuitive design in the form of functional prototypes on Figma which will make the web app easy to understand and navigate. Second, we will use best practices and write code that is easy to read and understand. Third, we will create comprehensive documentation that explains our code and how it works. Fourth, we will set up automated tests to ensure our code is working correctly. Finally, we will use a reliable deployment process that allows us to quickly and easily deploy new versions of our application.

4.3.8 Robustness

The web application must be robust and able to handle various conditions and errors. It should be designed to be resilient to errors and be able to handle unexpected inputs and conditions.We can make our web application more robust by ensuring that our system is fault-tolerant and can handle unexpected changes in the user input and/ or system requirements. Second, we will take parallelism into account in a way that ensures that our application can handle large increases in user demand without requiring major changes into the infrastructure.

4.3.9 Testability

The web application must be easily testable and have automated tests. It should be designed in a way that allows for automated tests to be created and run with minimal effort. Making our web application more testable is important or ensuring that our product is reliable and of high quality. We will be taking several steps to guarantee the testability element of our app. We will start by setting up automated test cases that run on a regular basis. Second, we will use tools such as unit tests and integration tests to test our application. Lastly, we will employ debugging tools to identify and fix any issues we may encounter.

4.3.10 Reusability

The web application must be reusable and able to be used in various contexts. It should be designed to be easily reusable in different scenarios and be modifiable to meet different requirements. First, we will ensure that the code is well-structured and easy to read. This will make it easier to reuse components in the future. Second, we will use a modular approach when designing the application. This will make it easier to replace and update components without having to recreate the entire application. Finally, we will create a library of components that can be used in multiple web sub-applications within our main system.

4.3.11 Usability

The web application must be user-friendly and intuitive, with a focus on ease of use rather than ease of learning. It should be designed to be intuitive and easy to use without requiring extensive knowledge or training. To leverage the usability factor, we will make sure all navigation links are clearly visible and easy to understand. We will add interactive elements to make the user experience more engaging while ensuring that the web page loading time is fast. We will keep the test simple and concise and instead of using text to represent clickable objects, icons would be preferred.

4.3.12 Correctness

The web application must be correct and produce correct results. It should be designed to produce accurate and reliable results in the expected output format. First, we will check our source code for any possible errors and ensure that all the logic is correct and reflects the goals we aim to achieve using our service. We will use static analysis tools to identify any potential errors or issues with the code. Second, we will ensure that our application is properly designed to handle any potential traffic or load.

Appendix A – Top 10 User Stories

1. As a client, I want to practice my speech on a daily basis in a structured and engaging way, with activities that help me regulate my breathing, identify syllables and exercise my jaw and cheek muscles, so that I can improve my communication skills everyday.

2. As a client, I want to give a rating to my assigned coach as a form of feedback to show my satisfaction with the help being provided.

3. As a client, I want to be able to track my progress and effort by maintaining a daily streak of completing therapy tasks so that I can stay motivated and encouraged to keep using Guftaar, and to make speech therapy a daily habit in order to achieve my communication goals.

4. As a client, I want to browse and buy a course according to my preference so that I can gain access to the content of the course, and be guided about speech therapy efforts with professional instruction.

5. As a client, I want to access informative articles and encouraging blog posts on speech therapy, so that I can stay informed and motivated in my therapy journey and learn new tips and techniques to improve my speech and communication skills.

6. As a client, I want to log in a daily self assessment of the severity and frequency of my stuttering throughout the day, so that I am able to track progress, identify patterns in my stuttering, and focus my efforts.

7. As a coach, I want to add and store private notes during the sessions with my clients which are only visible to me, so that I can provide tailored and effective therapy based on their individual needs and goals. Notes are in the form of comments which can be viewed later on to keep a track of the client's progress.

8. As a coach, I want to be able to set and manage my availability for client meetings at my convenience, so that I can efficiently schedule and conduct therapy sessions at times that work for both myself and my clients.

9. As an administrator, I want to be able to add and manage speech therapy coaches, so that I can expand the services offered by the app and provide clients with a range of coaching options.

10. As an administrator, I want to be able to review client feedback and update coach ratings on the basis of client reviews, so that I can ensure that the therapy provided is of high quality and clients are receiving effective treatment from highly rated coaches

Appendix B – Architectural Spike

One of Guftaar's functional requirements that requires proof-of-concept, owing to uncertainty around the technical area of audio processing, is the **Quick Practice** feature (RQ 2.4), that allows users to record their voice and receive feedback on their speech patterns. Given that no one on the team currently has any background in HTML audio integration, and the use of the MediaStream Recording API, and that this is a technically challenging area that involves dealing with the user's device and browser limitations, an architectural spike is developed around it.

The goal of the spike is to explore different approaches for capturing and processing audio in a web app and determine the best strategy for the Guftaar platform.

1. To initiate the spike, the team will research the available browser APIs for capturing audio, including popular options like the Web Audio API and the MediaStream Recording API. Both APIs are prototyped as part of the Quick Practice feature to compare their performance and compatibility with different browsers.

2. Next, the team will experiment with different ways of processing the recorded audio to extract meaningful information that can be used in scoring. This will include exploration of techniques such as speech recognition, audio length evaluation, voice activity detection, and pitch analysis as measures to evaluate the accuracy and performance of each recording.

3. After conducting the spike, the team will finalise the API technologies that have the best compatibility and performance on different devices and browsers along with the most appropriate, and practically feasible method for evaluating each audio recording to assign a corresponding speech efficacy score. This is done using both research and prototyping results, alongside knowledge of team bandwidth and capabilities.

Appendix C - Group Log

1. **Meeting with Assigned TA, February 11 2023**

   Time: 30 Minutes

   Meeting Details:

   - The group was given feedback on the project proposal:
     - To include in text citations.
     - To further specify the intended user group.
     - To format the document better (remove extra spaces).
     - To specify the assumption that coaches will be hired by the administrators in an independent process.
   - The group was also given advice on the SRS document:
     - To be thorough in explanations
     - To utilise a comprehensive format for the Functional Requirement specifications so the team can derive maximum use from them in subsequent phases.

2. **In-person Group Meeting, February 13 2023**

   Time: 240 Minutes

   Meeting Details:

   - The group thoroughly went over the SRS Template, brainstorming and discussing what they will include in each sub-section.
   - The group revisited the proposal document submitted earlier to divide the proposed functional requirements in 5 general functional areas.
   - Next, each functional requirement was discussed in detail to identify and outline its description, inputs required, processes to be done and the projected outputs. These functional requirements were completed for section 3.1 of this SRS document during this meeting.
   - While discussing the processing required for each functionality, the group also planned a tentative design and structure for the system's database.

- The group also explored different APIs and libraries that can be used to successfully implement functionalities such as recording audios and exporting multimedia content.

3. **Meeting with Assigned TA, February 18 2023**

   Time: 35 Minutes

   Meeting Details:

   - Got clarifications regarding the two associations in use cases (include and extend) in the use case diagrams as well as the confusion the group had regarding the system boundary.
   - Discussed the issues we might face when using third party softwares such as Google Drive, Calendly, or WordPress.
   - Discussed the use of APIs in our website such as the MediaStream Recording API for getting recorded audios from the client.
   - Got clarification regarding the requirements of Section 2.1 (specifically the diagram showing subsystem interactions and what else can we include in it)
   - Went over the Data Flow Diagram and the TA suggested some changes to it that would make it look more concise and elaborative
   - Discussed the implementation specifications of RQ5.3 and RQ5.4 and the confusion the group had regarding hosting and managing the blog locally or taking assistance from an external API.

4. **Online Group Meeting, February 18 2023**

   Time: 180 Minutes

   Meeting Details:

   - The group reviewed feedback given by the assigned TA and revised the following things:
     - Figure 1 was improved so the structure of the system can be visualised in a more systematic manner.
     - An additional diagram (Figure 2) was added in section 2.1 to show interactions between different users.

- The data flow diagram (Figure 3) was changed to show interconnection of inputs and outputs within the features.
- A legend was added in the use case diagram (Figure 4), all use cases were grouped together within one system boundary, and the association between different use cases (include or extend) was decided.

Appendix D – Contribution Statement

Name	Contributions in this phase	Approx. Number of Hours
Bakhtawar Ahtisham	Section 2.5, Section 3.2, Section 4.3, Appendix A	15-20
Emaan Atique	Section 1.3, Section 2.1, Section 2.4, Section 3.1	15-20
Emaan Bilal	Section 1.1, Section 1.2, Section 2.3, Section 3.1, Appendix A + B	15-20
Harris Ahmad	Section 3.2, Section 4.1, Section 4.2, Section 4.3	15-20
Romessa Shahjahan	Section 1.4, Section 3.1, Section 2.2, Appendix C	15-20
Saad Sher Alam	Data Flow Diagram, Section 3.1, Section 3.3	15-20

Table 10: Contribution Statement