Requirement & Function Size

of

Sign Language Detection System

Course Title: Software Metrics Lab

Course Code: SE 3206

Team Members :

Shoriful Habib MUH2025003M Year-3 Term-2 shoriful2515@student.nstu.edu.bd	Sanjida Akter Samanta BFH2025010F Year-3 Term-2 samanta2515@student.nstu.edu.bd
Sumaiya Begum BKH2025015F Year-3 Term-2 sumaiyabegum2515@student.nstu.edu.bd	Mehedi Hasan MUH205032M Year-3 Term-2 mehedi2515@student.nstu.edu.bd

Submitted to:

Md. Hasan Imam
Lecturer
Institute of Information Technology
Noakhali Science and Technology university

Date of Submission: 26/12/2023

Table of Contents

List of Tables

1. Project Information

Project Name	Sign Language Detection System
Supervised By	**Dipanita Saha** Assistant Professor Intitute of Information Technology(IIT) Noakhali Science and Technology University
Team Members	**Shoriful Habib** (MUH2025003M)
	Sanjida Akter Samanta (BFH2025010F)
	Sumaiya Begum (BKH2025015F)
	Mehedi Hasan (MUH205032M)
GitHub Link	Sign Language Detection System
SRS Document	Software Requirement & Specification

2. Requirement Size

Definition: Requirements and specification documents incorporate a combination of text, graphical representations, and specialized mathematical diagrams and symbols. The manner of presentation varies based on the chosen style, method, or notation. While measuring the size of code or design involves identifying atomic entities such as lines, statements, bytes, classes, and methods, requirements or specification documents present a challenge due to their mixed nature of text and diagrams. For instance, a use case analysis might include a UML use case diagram accompanied by a set of use case scenarios expressed in either textual or UML activity diagram form. The multifaceted nature of requirements analysis makes it challenging to generate a single, unified size measure.

Process

- There are obvious atomic elements like number of pages that can be counted.
- **Use case diagrams**: Number of use cases, actors, and relationships of various types.

- **Use case**: Number of scenarios, size of scenarios in terms of steps, or activity diagram model elements.
- **Domain model** (expressed as a UML class diagram): Number of classes, abstract classes, interfaces, roles, operations, and attributes.

2.1 Stakeholders

Table 2 : Stakeholders

Metric Type	Metric	Quantity
Stakeholders	Number of stakeholders	5

2.2 Functional & Non-functional

Table 3 : Requirements and use case measurements

Metric Type	Metric	Measurement
Functional & non-functional requirements	Number of functional requirements	11
	Number of non-functional Requirements	20
Requirements gathering	Number of persons interacted	4
	Number of interviews taken	5
	Number of questionnaire responded	60
Use case diagram	Number of use cases	16
	Number of actors	2
	Number of relations between use cases	8
	Number of relations between use case and actors	14
	Number of relations per use cases	1.375
Use case description	Number of use case descriptions	16
	Number of actors per use case (Average)	0.125

2.3 Activity Diagram

Table 4 : Activity Diagram Metrics

Metric Type	Properties	Metric	
Activity diagram	Title	Number of steps	Number of branching actions
	Registration	6	2
	Login	4	1
	Forgot Password	6	1
	Change Password	5	0
	Edit Profile	5	2
	Start Webcam	1	2
	Enter Plain Text	7	2
	Convert Text to Speech	4	0
	Translate Gesture	4	1
	Show Gesture	7	1
	Match Feature	5	1
	Capture Feature Point	7	3
	Capture Video	8	3
	Capture Gesture	6	3
	Display Result	8	2

2.4 Total Information

Table 5 : Total information of activity diagram

Metric Type	Metric	Measurement
Activity diagram	Total number of activity diagrams	15
	Total number of Steps	83
	Total number of branching actions	24
	Average Steps per activity diagram	5.53
	Average branching actions per activity diagram	1.6

3. Function Size

Albrecht's effort estimation method was largely based on the notion of FPs. As their name suggests, FPs are intended to measure the amount of functionality in a system as described by a specification. We can compute FPs without forcing the specification to conform to the prescripts of a particular specification model or technique.

To compute the number of FPs we first compute an unadjusted function point count (UFC).

- **External inputs**: Those items provided by the user that describe distinct application- oriented data (such as file names and menu selections). These items do not include inquiries,which are counted separately.
- **External outputs**: Those items provided to the user that generate distinct application-oriented data (such as reports and messages, rather than the individual components of these).
- **External inquiries**: Interactive inputs requiring a response.
- **External files**: Machine-readable interfaces to other systems.
- **Internal files**: Logical master files in the system.

Next, each item is assigned a subjective "complexity" rating on a three point ordinal scale: simple, average, or complex. Then, a weight is assigned to the item, In theory, there are 15 different varieties of items (three levels of complexity for each of the five types), so we can compute the UFC by multiplying the number of items in a variety by the weight of the variety and summing over all 15.

3.1 Function Points

Table 6 : Function Points

Function Points	Quantity	Name	Complexity	Complexity Weight
External Inputs	3	User gestures for sign language input	Average	4
		User account creation	Complex	6
		Text input for translation	Simple	3
External Outputs	5	Display recognized sign	Complex	7
		Convert text to sign	Complex	7
		Text-to-speech output	Complex	7
		User account creation confirmation	Average	5
		Confirmation message for profile update	Average	5
External Inquiries	2	Sign meaning inquiry	Average	4
		User profile inquiry	Average	4
External Files	0			
Internal Files	1	A database storing user profiles and signs for letters.	Complex	10
Count Total				**62**

To complete our computation of FPs, we calculate an adjusted function point count, FP, by multiplying Count Total by a technical complexity factor, TCF. Each component or sub factor is rated from0 to 5, where 0 means the sub factor is irrelevant, 3 means it is average, and 5 means it is essentialto the system being built.

3.2 Technical Complexity Factor

Table 7 : Complexity Factors

No.	Complexity Factors	Rating (0-5)
F1	Reliable backup and recovery	4
F2	Data communications	4
F3	Distributed functions	3
F4	Performance	5
F5	Heavily used configuration	3
F6	Online data entry	4
F7	Operational ease	5
F8	Online update	3
F9	Complex interface	5
F10	Complex processing	5
F11	Reusability	5
F12	Installation ease	5
F13	Multiple sites	1
F14	Facilitate change	3
	Total ($\sum f_i$)	**55**

Technical Complexity Factor (TCF) = $0.65 + 0.01 * \sum f_i =$ **1.20**

Function Point (FP) = Total Count * Technical Complexity Factor (TCF)

$$= 62 * 1.20 = 74.4$$

Here if we assign 1 day to implement 1 function points that the project will take more than 2 months.