Software Project Lab – III

Report

Bengali Braille to Text Translator

Course: Software Project Lab – III

Course No: SE – 801

Submitted by

Atiq Ahammed

BSSE0817

Session: 2015 – 2016

Supervised by

Dr. Ahmedul Kabir

Assistant Professor

Submitted to

Software Project Lab – III Coordinators

Institute of Information Technology

University of Dhaka

07 – November – 2019

Letter of Transmittal

07-11-2019

Software Project Lab – III Coordinators

Institute of Information Technology

University of Dhaka.

Dear Sir,

I was assigned to develop a desktop application that translate Bengali Braille code into the corresponding Bengali text. I am submitting my report with due respect. I have tried my best for the report.

So, may I therefore, hope that you would and oblige thereby.

Sincerely yours,

_____

Atiq Ahammed

BSSE 0817

Institute of Information Technology

University of Dhaka

Document Authentication

This project document has been approved by the following persons.

_____	_____
Prepared by	Approved by
Atiq Ahammed	Dr. Ahmedul Kabir
BSSE 0817	Assistant Professor
Institute of Information Technology	Institute of Information Technology
University of Dhaka	University of Dhaka

Acknowledgement

By the grace of Almighty Allah, I have completed my technical report on Bengali Braille to Text Translator.

I am grateful to my supervisor Dr. Ahmedul Kabir sir for his direction throughout the working time.

I am also thankful to the teachers and students of The Institute of Education and Research, University of Dhaka. They greatly helped me in collecting information among all business.

Abstract

This document contains the technical report for the Software Project Lab – III which is titled as "Bengali Braille to Text Translator." The aim of this project is to create a desktop application that translate Bengali text from scanned image of braille code typed on a paper. This document provides the overview of the scenario-based model, class-based model, and data flow model including the methodology for Bengali Braille to Text translation. Using this document as a guide, we are describing the requirements, necessary diagrams, procedures and working sequence of our project.

This is a project of a desktop application that will be develop to convert the braille code from a scanned image to corresponding Bengali text. Here I will discuss how I will identify the requirements, how to analyze them and how to present a recommended solution for the system.

This will help to make the software according to the demand of the stakeholders.

Contents

Tables

Figures

1 Introduction

This document contains functional, non-functional and support requirements and establishes a requirement baseline for the development of the system. The requirements contained in the SRS are independent, uniquely numbered, and organized by topic. The document serves as official means of communicating user requirements to the developer and provides a common reference point for both the developer team and stakeholder community. It will evolve over time as users and developers work together to validate, clarify and expand its contents.

1.1 Intended Audiences

Our Software Requirement Specification (SRS) is pinned for several audiences including students of IER as well as our project supervisor, SPL – 3 coordinators and me.

- Teachers and students of will use this SRS to verify that we have developed a product that the required
- My supervisor will use this SRS to plan milestones and ensure that we are on the right track when developing the system
- I will use this SRS as a basis for creating the system design. I will continually refer back to this SRS to ensure that the system we are designing, will fulfill the requirements of the teacher and students of IER.
- We will also use this SRS as a basis for developing the system functionality and link the requirements defined in this SRS to the software that we will create to ensure that we have created a software that will fulfill all the requirements

We wish, this analysis of the audience will help us to focus on the users who will be using our analysis. This document will help each and every person related to this project to perceive the subject matter of the project.

2 Background Studies

This part of this document contains necessary terms which be helpful to understand the next Usage Scenario and Methodology of this project.

2.1 Image

An image can be defined by a two-dimensional array specifically arranged in rows and columns. Digital Image is composed of a finite number of elements, each of which elements have a particular value at a particular location which is called pixel [1]. Each pixel has three values of RGB (Red, Green, Blue) in between 0-255 or we can say that colors here are of the 24-bit format, that means each color has 8 bits of red, 8 bits of green, 8 bits of blue, in it. Each color has three different portions.

2.2 Image Acquisition (RGB to Gray)

It is converting a digital image from a given one processing each pixel with some operation. On this project the RGB to Gray conversion will be used. Average method is the simplest one. You just have to take the average of three colors. Since it's an RGB image, so it means that you have add r with g with b and then divide it by 3 to get your desired grayscale image [2].

It's done in this way.

Grayscale = (R + G + B / 3)

2.3 Noise filtering in Digital Image Processing

Noise is always presenting in digital images during image acquisition, coding, transmission, and processing steps. Noise is abrupt change in pixel values in an image. So, when it comes to filtering of images, the first intuition that comes is to replace the value of each pixel with average of pixel around it.

This process smooths the image. To reduce the noise from the image, mean, median and/or Gaussian filter will be used based on image quality.

- In image processing, a Gaussian blur (also known as Gaussian smoothing) is the result of blurring an image by a Gaussian function (named after mathematician and scientist Carl Friedrich Gauss). It is a widely used effect in graphics software, typically to reduce image noise and reduce detail.
- Mean filter is a simple sliding window that replace the center value with the average of all pixel values in the window. The window or kernel is usually a square but it can be of any shape.
- Median filter is a simple sliding window that replace the center value with the Median of all pixel values in the window. The window or kernel is usually a square but it can be of any shape.

2.4 Otsu's Thresholding

Otsu's method is a global thresholding technique. It uses the histogram of the image for threshold searching process. It maximizes "between class variance" of the segmented classes. Otsu proves that Minimizing "within class variance" is same as maximizing "between class variance" of the segmented classes. And maximizing "between class variance" is computationally less expensive than minimizing "within class variance" [3].

Let 't' be the threshold.

This threshold subdivides the image into two classes: C0 and C1.

Now you calculate the function "between class variance" of the segmented classes C0 and C1.

BCV=var0+var1

Now what the method does is- search for the value of 't' so that the "between class variance", BCV is maximized.

2.5 Morphological Dilation and Erosion

The most basic morphological operations are dilation and erosion. Dilation adds pixels to the boundaries of objects in an image, while erosion removes

pixels on object boundaries. The number of pixels added or removed from the objects in an image depends on the size and shape of the structuring element used to process the image. In the morphological dilation and erosion operations, the state of any given pixel in the output image is determined by applying a rule to the corresponding pixel and its neighbors in the input image. The rule used to process the pixels defines the operation as a dilation or an erosion [4].

3 Project Description

The main concern of this document is to describe a project of Software Project Lab – 3 which will be a desktop application. This application basically translates Bengali braille code into corresponding Bengali text. It takes scanned image of Bengali braille code as input and provide translate it into Bengali text.

3.1 Quality Function Deployment

The technique which translates the needs of the customer into technical requirements for software is called Quality Function Deployment (QFD).

QFD concentrates on maximizing customer satisfaction from the Software Engineering process. With respect to our project the following requirements are identified by QFD-

3.1.1 Normal Requirements

Normal requirements refer to the objectives and the goals that are stated for the product during the meeting with the stakeholders. The presence of these requirements ensures the satisfaction of the customers. The normal requirements for the project are stated below.

- User can customize the work sequence, and store the configuration for future use.
- Will be able to run the tool and get output with only single 'run' button.
- User can view result from each level of Braille to text translation process.
- User will be able to run different types of image enhancement and other pre-processing algorithms to improve input image quality.
- There will be some default configuration templates available with the software.

3.1.2 Expected Requirements

The requirements that are implicit to the system might not be brought up during the meeting because of their fundamental nature. Despite being not explicitly mentioned their presence must be ensured. Otherwise, the product will leave customers dissatisfied. These requirements are called expected requirements and these are stated below.

- The system will be able to support all popular image formats like- JPG, JPEG, PNG.
- User will be able to process single image of Braille writing.
- The user interface of the system shall be easy to use. It will make use of drop-down boxes, radio buttons, and other selectable fields wherever possible instead of fields that require the user to type in data.

3.1.3 Exciting Features

The factors that go beyond the customer's expectations and prove to be satisfying when present are called exciting features. The exciting features are the so called 'wow factor' for our project.

- The user interface should provide appropriate error messages for invalid input as well as tool-tips and help.
- User will be able to view each processing step separately
- User will be able to save the output in a ".txt" file using this system if wishes
- There will an option for user to translate English braille code into text also.

3.2 Scenario

Bengali Braille to Text Translator will be a desktop application for Windows operating system. It will be a tool that will take a scanned image of Bengali Braille writing as input. The system will be able to support all popular image formats like- JPG, JPEG, PNG. The input will go through different types of image-preprocessing techniques, translate Braille to text through pattern recognition, and apply text correction procedures for final output. The methodology of whole process will be discussed later. Then it will provide the Bengali text that is written in the scanned input image. If the user wants to save the text file in the local directory of the computer our system will provide the user to save it.

3.2.1 Choosing Image File

After launching the application, the user will have a user interface from which she/he will get a file choosing option. Using that option, the user can will be able to choose an image using Windows file chooser. The chosen image will be the input for the application.

3.2.2 Selecting Language

The user will get an option to choose a language between Bengali and English for translating the input image into text. By default, the application will provide Bengali language as chosen one. If the user wants to translate the image into English language, she/he have to choose the English option.

3.2.3 Image to Text Translation

The user will then click on the translate button. After some processing on the image the application will show the corresponding output in the user interface of the application.

3.2.4 Saving Output

The user will get an option to save the output in a file with ".txt" extension. After clicking on the save button on the user interface the user will get an file saving option to save the file in her/his chosen directory.

3.3 Methodology

Braille is a reading and writing system which can only be read with the sense of touch. It is used by blind and visually impaired people who cannot access print materials. Braille is not a language. Rather, it is a code by which many languages can be written and read. It uses raised dots to represent the letters of the print alphabet. It also includes symbols to represent punctuation. A Braille character is made using a combination of 6 dots which is arranged in two columns and three rows.

At first, the user will select scanned images of Braille writing. Then the following methodology will be applied for final output.

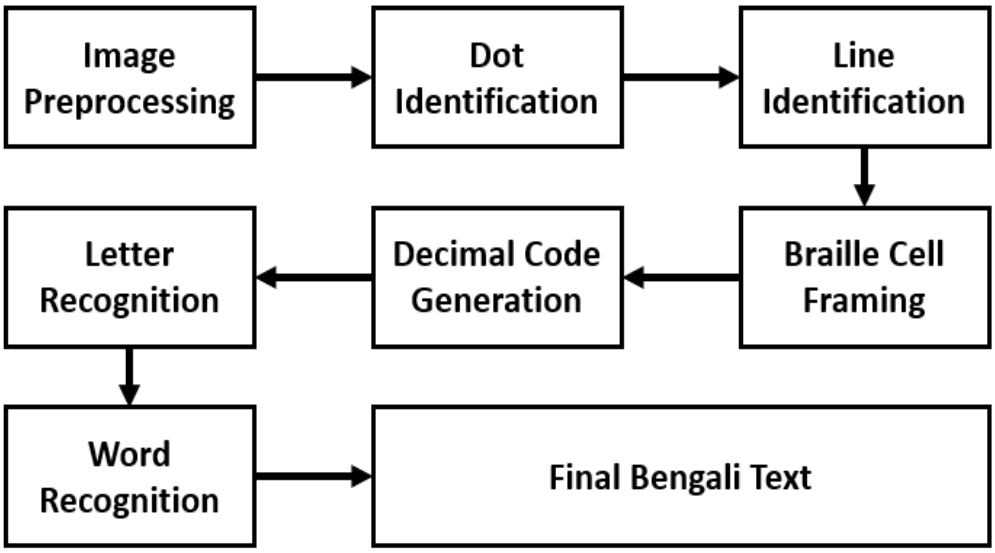


Figure 1:Methodology Step by Step

3.3.1 Image Preprocessing

Different types of image enhancement processes will be run. Then Image preprocessing is an essential step during which errors that occurred while the images were taken are eliminated. Errors include noise, deformation, bad illumination or blurring. Image preprocessing can be used for image enhancement by reducing noise [5].

1. Grayscale conversion: In any image, each of the pixels should have a variation of RGB values for three different colors Red, Green, and Blue. But for having the same value of each three colors for every pixel it requires to grayscale conversion for this image.
2. Noise reduction: For noise reduction median filter, gaussian elimination will be used.
3. Binary conversion: Then it will take a thresholded image consisting of couples of white and black spots, where each couple denotes a single Braille dot [6]. To do this Otsu thresholding will be used.
4. Filling: For filling the dots Morphological Dilation will be used based on dots quality.

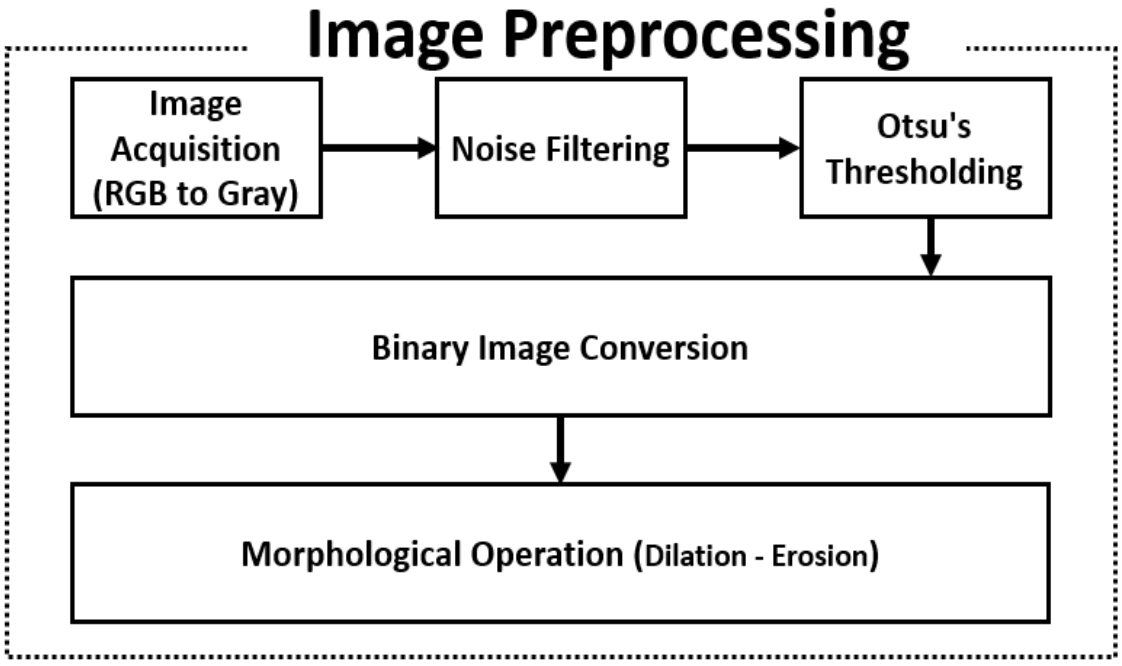


Figure 2: Preprocessing steps

3.3.2 Line Identification

After preprocessing the most important task is horizontal line identification. This will be done based on the dot frequency in the image. Each horizontal text line will consist of three-dot lines [7].

3.3.3 Braille Cells and Dots Framing

The framing process to determine the cells, words, and lines in the scanned image based on standard dimensions of Braille documents. Mainly statistics measures of central tendency will be used for performing this task.

3.3.4 Decimal Braille Code Generation

This stage is a core stage of the system and it was done by testing each dot in Braille cell, if it active the position of this dot take digit one and if inactivated the position of this dot takes digit zero. The recognizing process of active or inactive dots was depended on taking summation of dot frame, if the summation was been one's digits, that means this dot is activated, else that dot will be inactive.

3.3.5 Braille letter recognition and transcription

Braille is not a language. It is a code for mapping character sets of various languages to its 64 fixed permutations. Mapping of character sets of a language to Braille symbols is called character mapping. Different countries developed their own standard character mapping for their language. This character mapping is not necessarily one-to-one because some languages may have more than 64 letters in their alphabet. So possible character mapping can be one-to-one and many-to-one. For example

Bangla letter 'ক' ⟶ ⠅ (One-to-one)
Bangla letter 'হ' & digit '৮' ⟶ ⠓ (Many-to-one)

Each letter of Bangla alphabet, numeric and punctuation has corresponding Braille representation. Bangla letters and their corresponding Braille representations are given in Table I, II and III.

TABLE I. ONE TO ONE CHARACTER MAPPING

Bangla	Braille	Bangla	Braille	Bangla	Braille
ক	⠅	ধ	⠹	ড়	⠳
খ	⠇	ন	⠝	ঢ়	⠻
ঘ	⠣	প	⠏	য়	⠪
ঙ	⠩	ম	⠍	ৎ	⠠⠞
ছ	⠡	য	⠽	ঁ	⠐
ঝ	⠵	র	⠗	ং	⠰
ট	⠾	ল	⠇	ঃ	⠆
ঠ	⠞	শ	⠩	ক্ষ	⠯
ড	⠙	ষ	⠱	জ্ঞ	⠚
ঢ	⠫	স	⠎	;	⠰
ত	⠾	।	⠲	!	⠖
থ	⠹	‘	⠠⠦	=	⠨⠅
দ	⠙	’	⠴⠄	*	⠔
-	⠤	[	⠐⠣	]	⠜⠂

TABLE II. TWO TO ONE CHARACTER MAPPING

Bangla		Braille	Bangla		Braille
অ	১	⠂	চ	৩	⠒
আ	া	⠆	জ	০	⠔
ঈ	ী	⠈	ঞ	:	⠐
উ	ু	⠢	ণ	Number prefix	⠼
ঊ	ূ	⠦	দ	৪	⠶
ঋ	ৃ	⠂⠦	ফ	৬	⠶
ও	ো	⠊	ব	২	⠃
ঔ	ৌ	⠌	হ	৮	⠓
গ	৭	⠶	্	"	⠤
(	)	⠷	?	"	⠴

TABLE III. THREE TO ONE CHARACTER MAPPING

Bangla			Braille
ই	ি	৯	⠊
এ	ে	৫	⠑
ঐ	ৈ	/	⠌
,	.	ʻ(lop)	⠄

In this stage of the project, the Braille letter was recognized using a matching algorithm to match each of the input decimal Braille code from an input processed image with codes of each Bengali letter. After the recognition process implemented the recognized letter transcript into equivalent text. The following represent the Bengali Braille Characters.

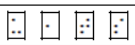
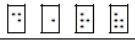
3.4 Braille word recognition and transcription

3.4.1 Grammatical conversion rules

In conversion of Bangla text to Braille, one needs to deal with conjunctions, consonants, dependent and independent vowels, punctuations and numbers. Here all grammatical conversion rules are discussed with examples.

1) Replacing rule: In replacing rule each Bangla character is replaced by its corresponding Braille cell. Some uses of consonants, vowels (dependent and independent) and punctuations are given in Table IV.

TABLE IV. USES OF CONSONANTS, VOWELS AND PUNCTUATIONS

Bangla Word	Distribution	Braille Representation
বল	ব ল	
উৎস	উ ৎ স	
কাঠ	ক া ঠ	
দৃঢ়	দ ৃ ঢ়	
কমা	,	
সেমি কোলন	;	

2) Inserting rule: In inserting rule a Braille cell is inserted as prefix. Other characters are replaced according to replace rule.

- If there is "i", "u" or "o" after consonant and if "a" is pronounced there then Bangla "a" equivalent Braille cell dot 1 is inserted after consonant in Braille [14]. Examples are shown in Table V.

TABLE V. INSERTION OF "অ"

Bangla Word	Distribution	Braille Representation
বই	ব ই	
রওনা	র ও না	

- Braille cell dot 4 is inserted before conjunctions having combination of two letters. Examples are shown in Table VI.

TABLE VI. CONJUNCTION OF 2 LETTERS

Bangla Word	Conjunct	Distribution	Braille Representation
গ্রাম	গ্র	গ ্ র	
পূর্ব	র্ব	র ্ ব	

- Braille cell dot 4, 6 is inserted before conjunctions having combination of three letters or four letters. Examples are shown in Table VII.

TABLE VII. CONJUNCTION OF 3AND 4 LETTERS

Bangla Word	Conjunct	Distribution	Braille Representation
রাষ্ট্র	ষ্ট্র	ষ ্ ট ্ র	
স্বাতন্ত্র্য	ন্ত্র্য	ন ্ ত ্ র ্ য	

- Two Bangla conjunctions have direct representation in Bangla Braille. So, there is no need to use Braille cell dot 4 before them. They are given in Table VIII.

TABLE VIII. TWO CONJUNCTIONS WITHOUT PREFIX

Bangla Word	Conjunct	Distribution	Braille Representation
কক্ষ	ক্ষ	ক ্ ষ	
জ্ঞান	জ্ঞ	জ ্ ঞ	

This is the last stage in implementation, the word recognition process flow through letter recognition and fill Braille the decimal array of the word, to apply matching process with stored addressed text and voices files of words, then to run equivalent files from the addressed word database.

4Scenario Based Modeling

For developing our software, we are giving the highest priority to user satisfaction. To identify the requirements to establish meaningful analysis and design model we determine how users want to interact with the system. Thus, our requirements modeling begins with scenario generation in the form of use cases, activity diagrams.

4.1 Use Case

Use case diagrams are usually referred to as behavior diagrams used to describe a set of actions that some system or sub-systems can perform in collaboration with one or more external users of the system.

The first step in writing a Use Case is to define that set of "actors" that will be involved in the story. Actors are the different people that use the system or product within the context of the function and behavior that is to be described. Actors represent the roles that people play as the system operators. Every user has one or more goals when using the system.

4.1.1 Primary Actor

Primary actors interact directly to achieve required system function and derive the intended benefit from the system. They work directly and frequently with the software. In our system both users the system both are primary actor.

4.2 Activity diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In this chapter we did try to provide each use case and its corresponding activity diagram together.

4.3 Use Case and Activity Diagram

4.3.1 Level 0 Use Case Diagram of Bengali Braille to Text Translator

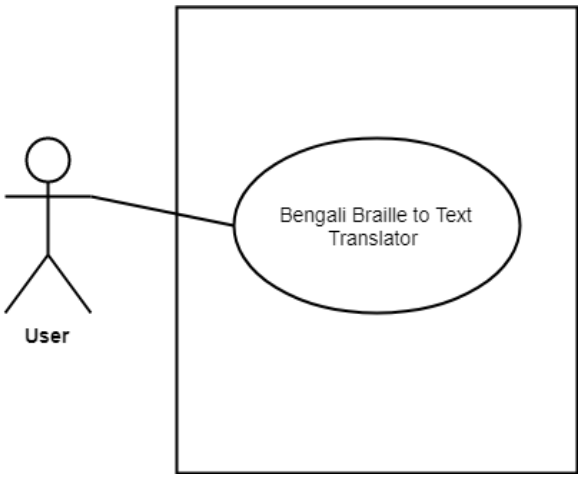


Figure 3: Level 0 use case diagram of Bengali Braille to Text Translator

Table 1: Information about level 0 use case diagram

Name:	Bengali Braille to Text Translator
ID:	L-0
Primary Actor:	User
Secondary Actor:	None

4.3.1.1 Description of Level 0 Use Case Diagram

After analyzing usage scenario, we found that user interact with our system as primary actor.

4.3.2 Level 1 Use Case Diagram of Bengali Braille to Text Translator

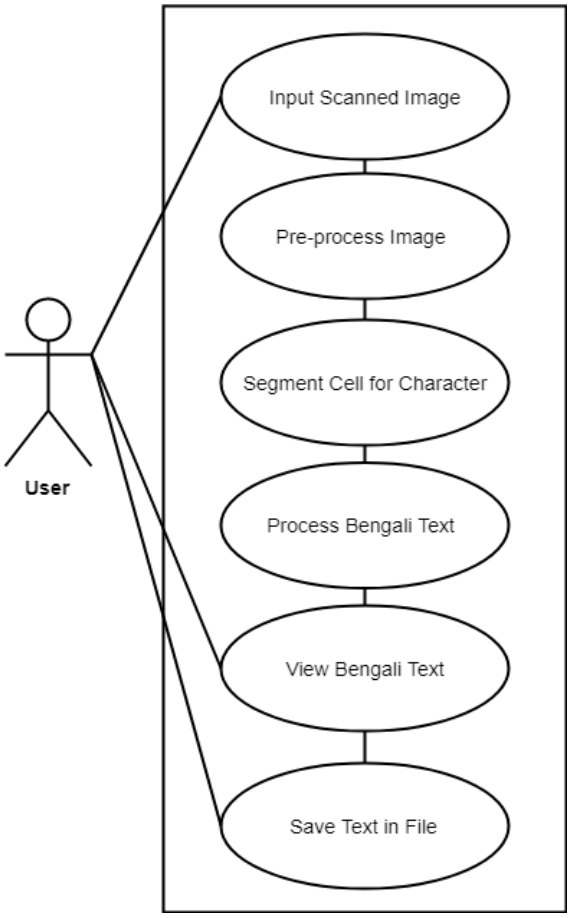


Figure 4: Level 1 use case diagram of Bengali Braille to Text Translator

Table 2: Information about level 1 use case diagram

Name:	Bengali Braille to Text Translator
ID:	L-1
Primary Actor:	User
Secondary Actor:	None

4.3.2.1 Description of Level 1 Use Case Diagram

In the usage scenario we separated our System into several modules. Here the user provides scanned image as input and get corresponding Bengali text.

4.3.3 Level 1 Activity Diagram of Bengali Braille to Text Translator

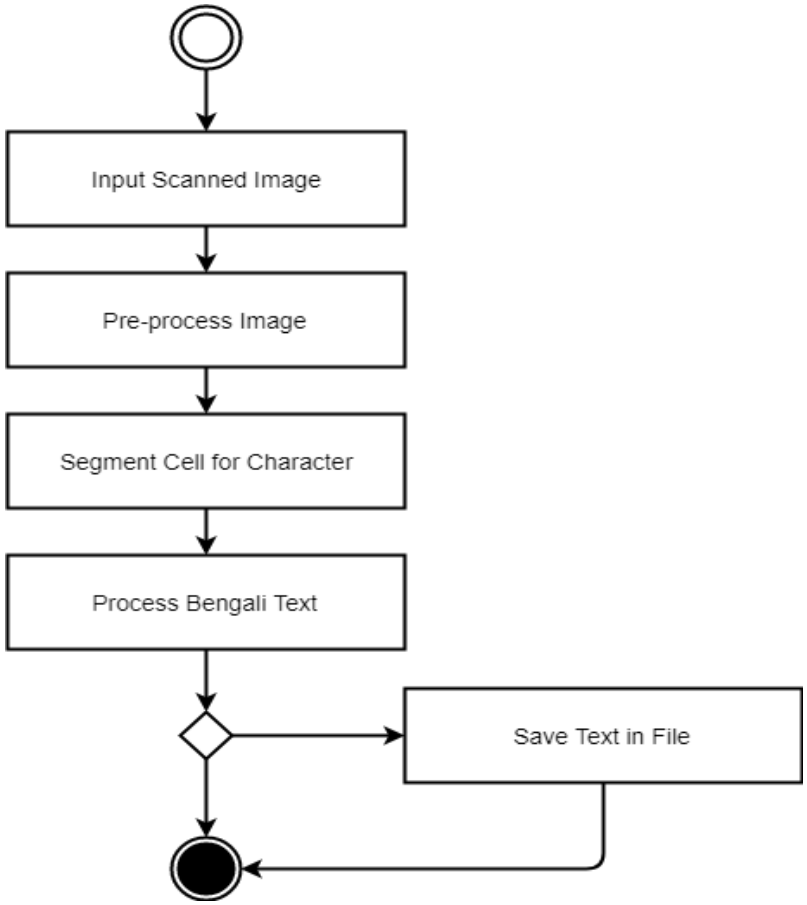


Figure 3: Level 1 activity diagram of Bengali Braille to Text Translator

5Class Based Modeling

We intended this chapter to describe class-based modeling for our "Bengali Braille to Text Translator".

In this chapter, our designed class-based model represents the objects that our "Bengali Braille to Text Translator" will manipulate, the operation that will applied to the objects, relationships between and the collaboration that occur between the classes that are defined.

5.1 Class Diagram

In this stage we designed class diagram in the Unified Modeling Language. This is a type of static diagram to describe the structure of our system. Here we also designed two individual design for our two subsystems.

5.1.1 Class Diagram of Bengali Braille to Text Translator

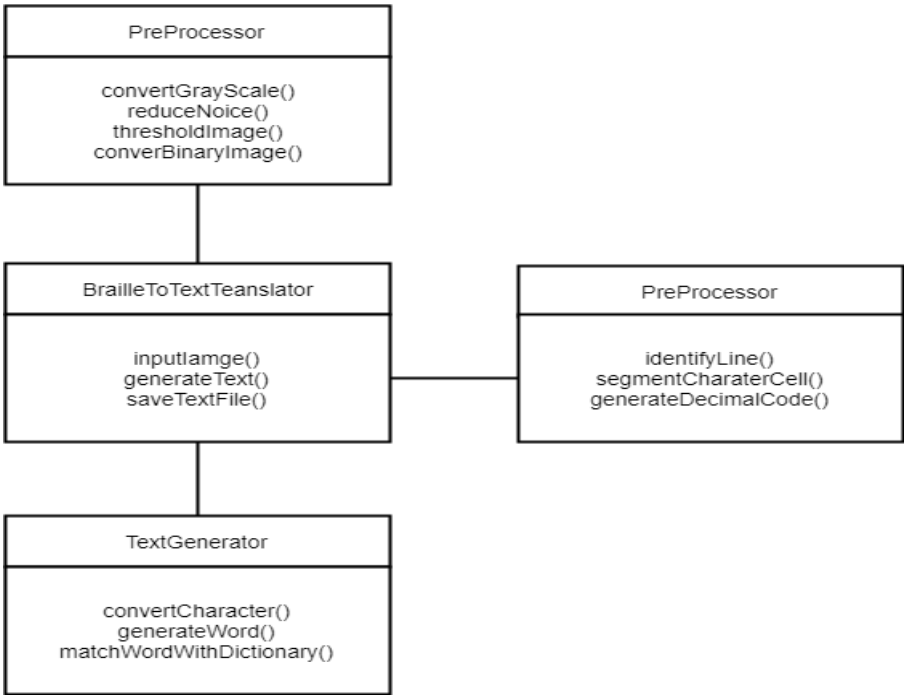


Figure 5: Class diagram of Bengali Braille to Text Translator

6 Data Flow Modeling

We intended this chapter to describe data flow modeling for our "Bengali Braille to Text Translator".

6.1 Introduction

A data flow diagram is a graphical representation of the flow of data through an information system. We use data flow diagram to diagrammatically represent the flow and exchange of information within our "Bengali Braille to Text Translator". As previous chapter, we modeled our data flow diagram based on our two main sub system.

6.2 Data Flow Diagram

We did try to go initial level to deep level in our system through our data flow diagram.

6.2.1 Level 0 Data Flow Diagram of Bengali Braille to Text Translator

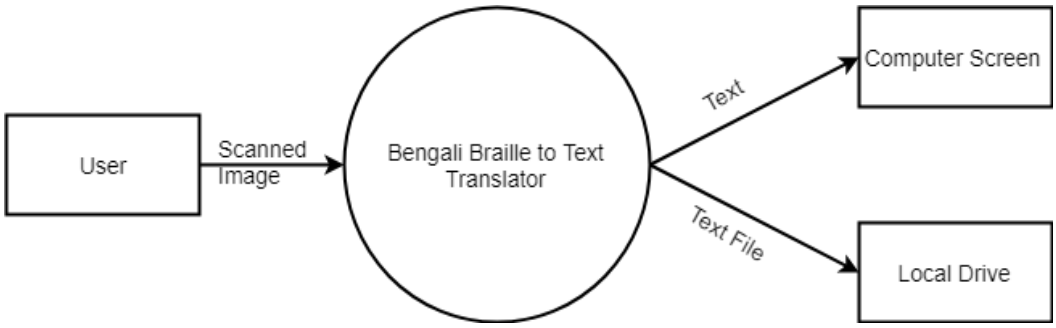


Figure 6: Level 0 data flow diagram of Bengali Braille to Text Translator

6.2.2 Level 1 Data Flow Diagram of Bengali Braille to Text Translator

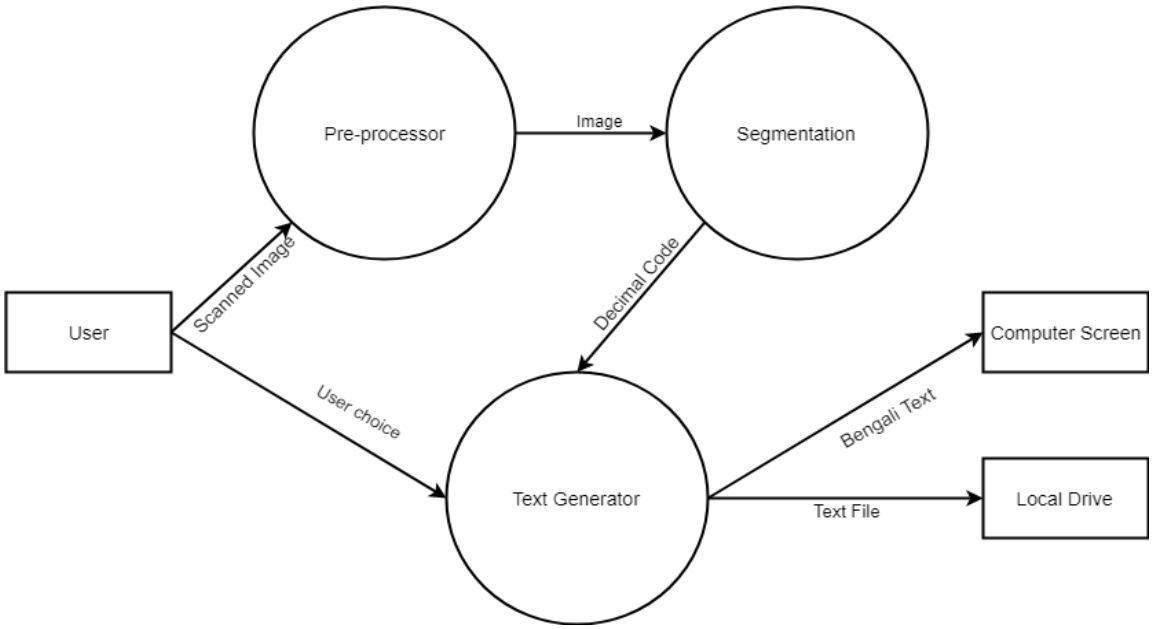


Figure 7: Level 1 data flow diagram of Bengali Braille to Text Translator

7 User Interface Design

The user interface (UI) is the point of human-computer interaction and communication in a device. This can include display screens, keyboards, a mouse and the appearance of a desktop. It is also the way through which a user interacts with an application or a website. As this application is a desktop application and both technical and non-technical user can use it. Concerning this graphical user interface (GUI) is chosen for its development.

7.1 Application First Page View

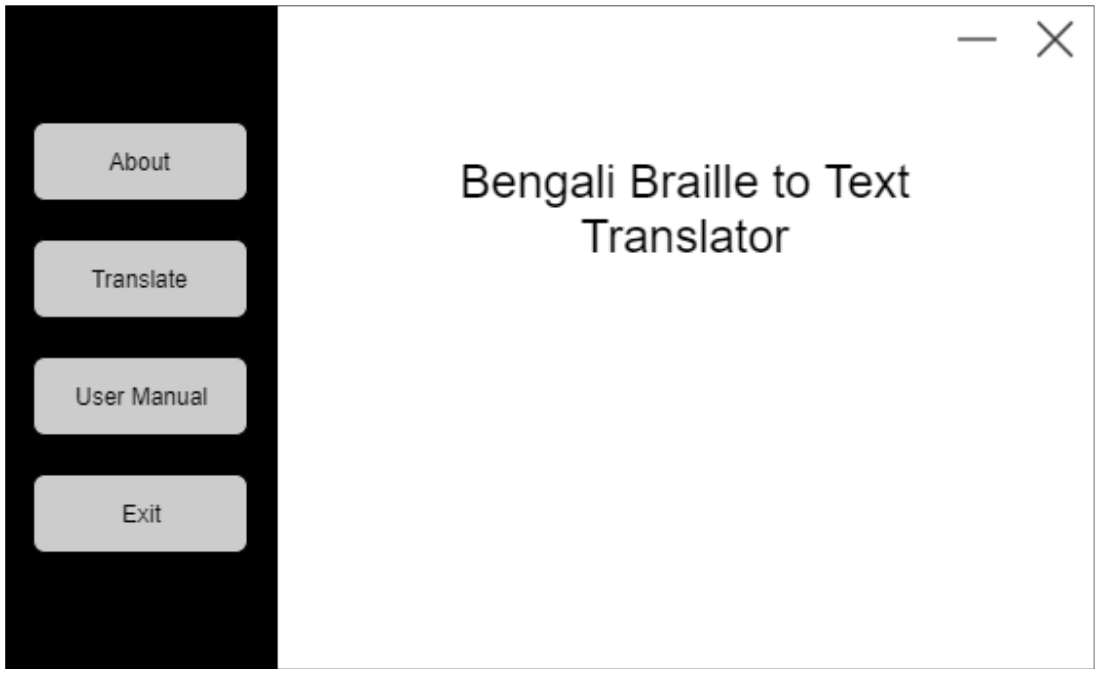


Figure 8: Application First Page View

7.2 Translation Page View

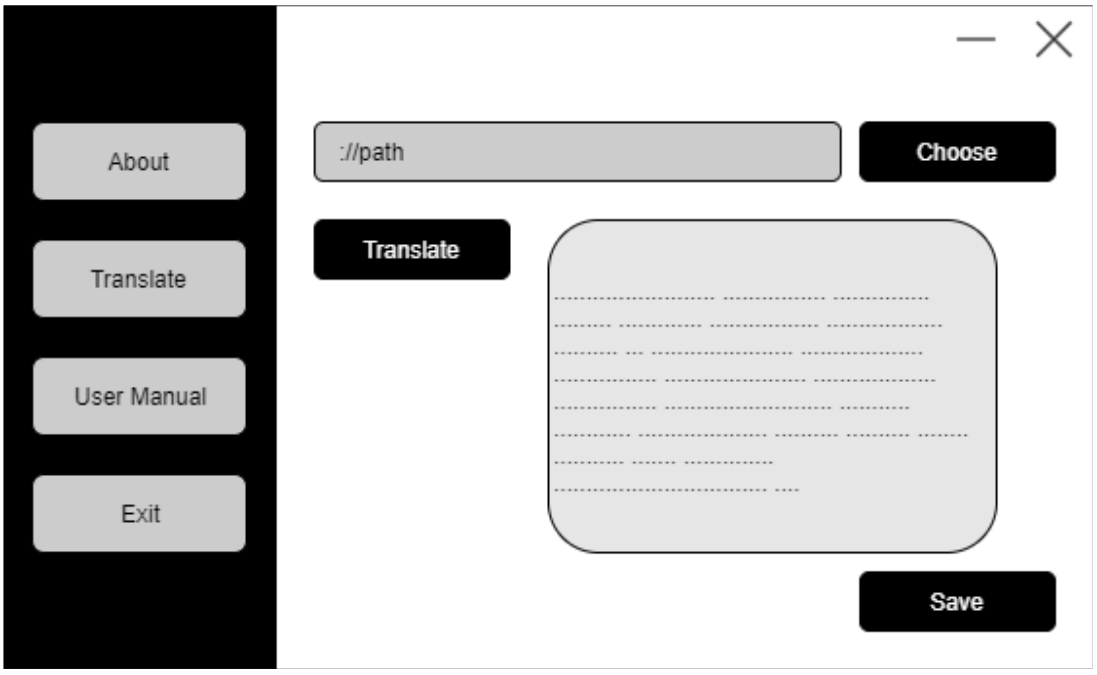


Figure 9: Application Translation Page View

7.3 Application User Manual Page View

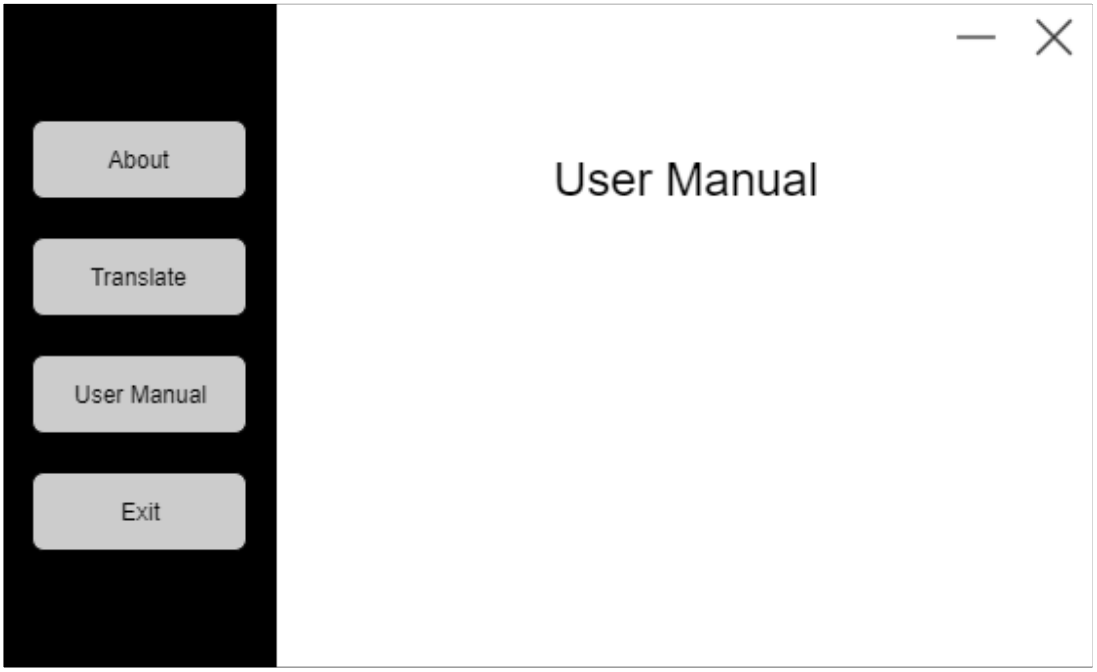


Figure 10: Application User Manual View

8Tools and Technologies

The "Bengali Braille to Text Translator" is a desktop application that can be installed in a computer with Windows 7, 8 and 10. This application is implemented with Java programming language.

8.1 Technologies

For developing this project java programming language is used. For backend development java was used. For making this application more user friendly is was required to developed a user interface. Javafx is used for this purpose.

8.1.1 Backend

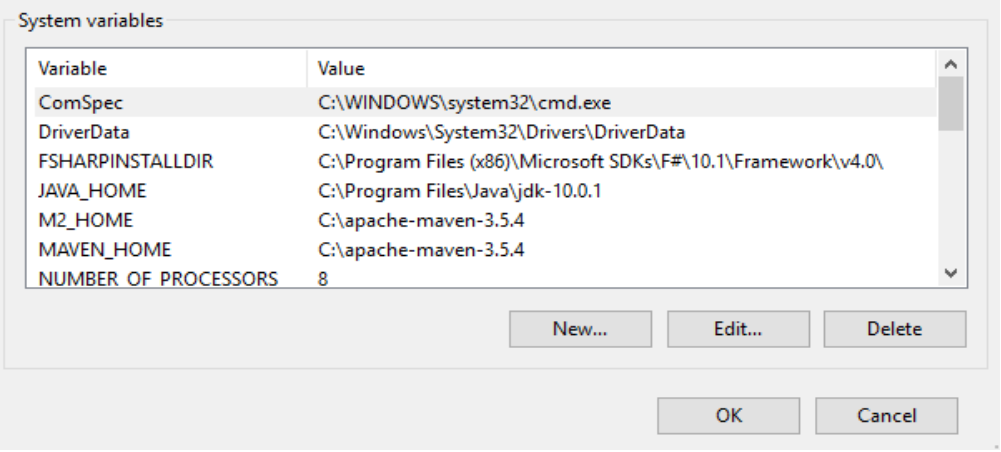
The JDK is a development environment for building applications using the Java programming language. For this project JDK 10 has been used. This application will be workable for the upper version of the JDK.

8.1.2 Frontend

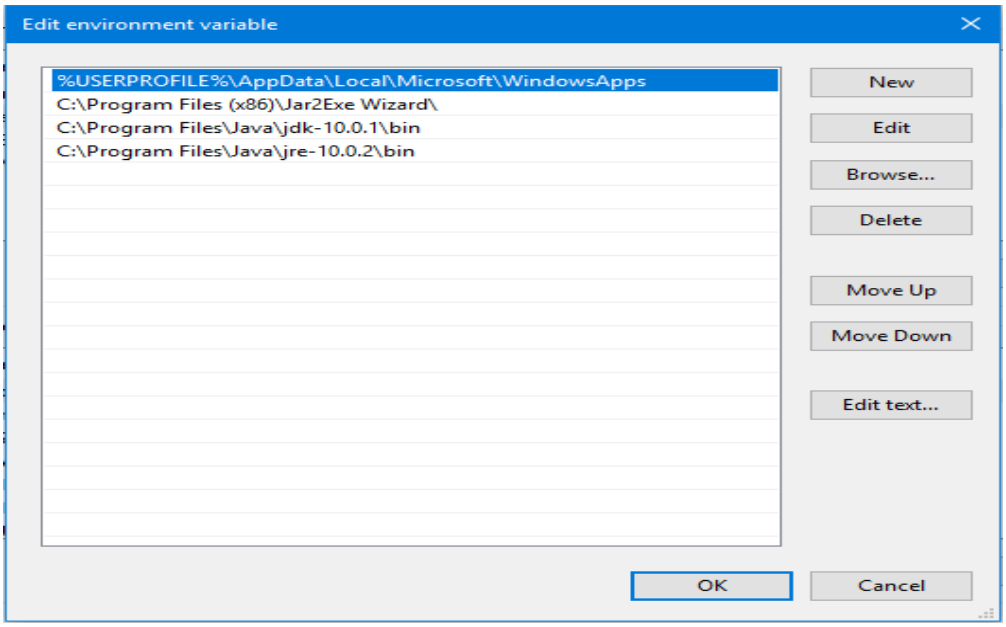
The user interface of this desktop is developed in JavaFx. As it's a standard GUI library for Java SE, I prefer it most for the development of this project.

8.2 System Requirements

- This is application will work on windows 7, 8 and 10.
- As the system was developed in Java 10. JDK 10 or upper version should be installed in the system.
- Check the system variable has "JAVA_HOME" with the path of the Java JDK directory. It it not exists then add it to the system variable.



- Check the environment variable has the jdk and jre director path. If it not exists then add those in the environment variable path.



- Install openvc version opencv-4.1.1 and copy the opencv_java411.dll file and add it "C:\Program Files (x86)\Common Files\Oracle\Java\javapath" directory.

9Testing

In this project, a desktop application. In this section, we report the expected output and the produced output after smoke testing.

9.1 Plan Identifier

Desktop Application Test Plan 1.

9.2 Introduction

This test plan has been developed for "Bengali Braille to text translator". The whole system is divided into only one portion.

9.3 Test-Item to be Tested

I tested the desktop application named Bengali Braille to Text Translator. Software Requirement Analysis and Specification Document of the system will be used for this purpose.

9.4 Features to be Tested

The following features are tested:

- File choosing from local disk
- Translation of the image file into text
- Saving the translated text into a txt file

9.5 Approach

I used end to end smoke testing technique to test the system.

9.5.1 Item Pass/Fail Criteria

If actual output of a test case does not match with expected output of the test case, the test case is considered as failed. 100% of all test cases should pass. No failed case should be crucial to the end-user's ability to use the application.

9.6 Test Deliverables

I will deliver test plan document, test case and test report.

9.7 Scheduling

Scheduling is given below with different part of Quality Assurance (QA) and duration

QA	Duration
Test plan	3 days
Testing	4 days

9.8 Planning Risks and Contingencies

The following scenarios are considered as risks for the project:

- Delay in requirements engineering.
- Delay in developing.
- Modification in development technology.

9.9 Test Cases

Test Case ID	Scenario	Steps	Input	Expected output	Actual Output	Result
1	Translate an invalid file	Go to translate page, select an invalid file using the file chooser	A file that is not with extension .png, .jpg, .jpeg	An error message will show and not text translation will occur	Same as expected output	Passed
2	Translate a valid file	Go to translate page, select a valid file using the file chooser	A file that is with extension .png, .jpg, .jpeg	Translated text will be appeared in the output textbox	Same as expected output	Passed

			A file that is not with extension .png, .jpg, .jpeg			
3	Translated text will appear in output field	After selecting the valid file click on the translate button	N/A	The corresponding output will appear in the output field and that may be correct or incurrent. But there will be some visible output.	Some text was visible in the text field	Passed
4	Save the text in a .txt file	Click on the save button after translating text	N/A	The text should be in a file with .txt extension int the chosen directory	Same as expected output	Passed

10 User Manual

Braille to Text Translator is a desktop application. The use of this application is given step by step.

10.1 Step 1: Launch the application

After launching the application a window will come. The screenshot is given below.

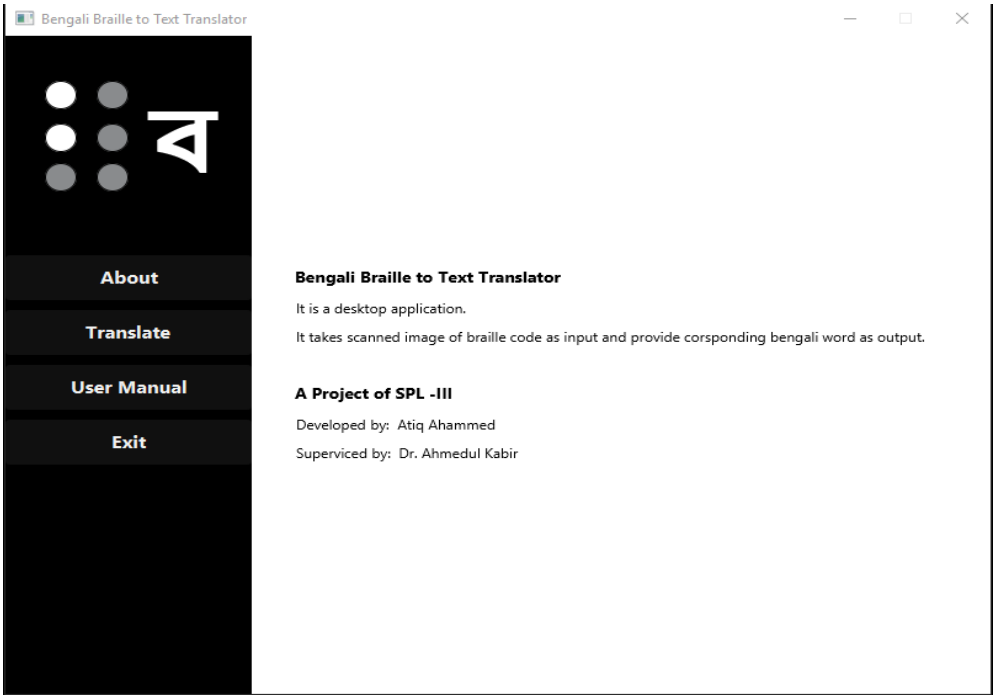


Figure 11: Application landing page.

10.2 Step 2: Go to Translation Page

After clicking on the **Translate** button the following window will open from where the user can translate a scanned image of braille code.

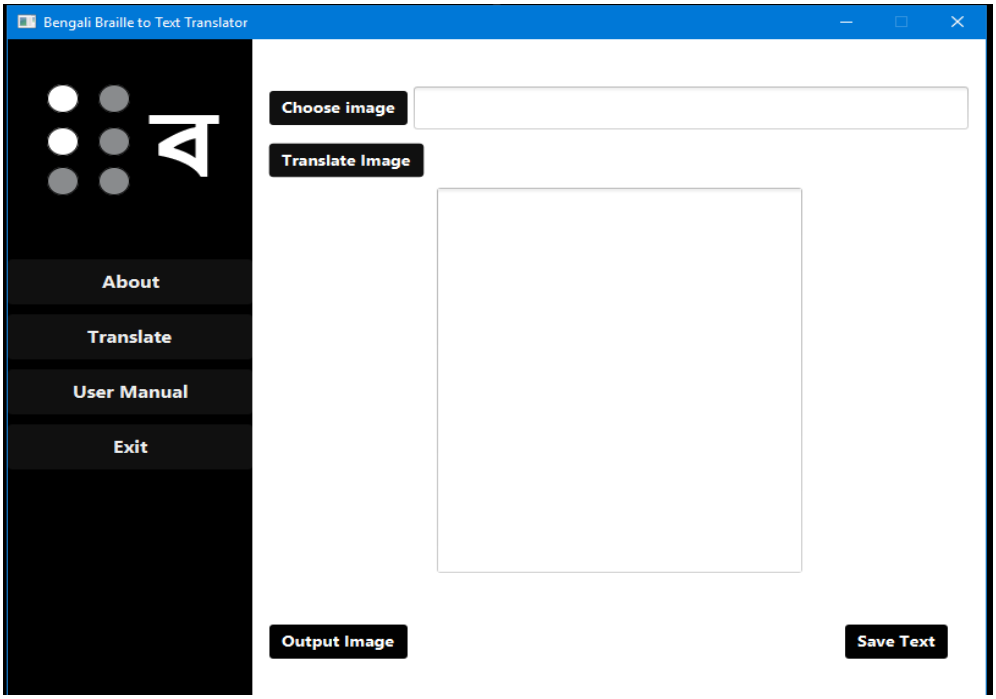


Figure 12: Translation page

10.3 Step 3: Choosing Image File

Clicking on the **Choose image** button there a file chooser window will open. Using the file chooser user can choose any image file for translation. If the user wish to provide the file path directly she/he can type the file path in the text box just right to the **Choose Image** button.

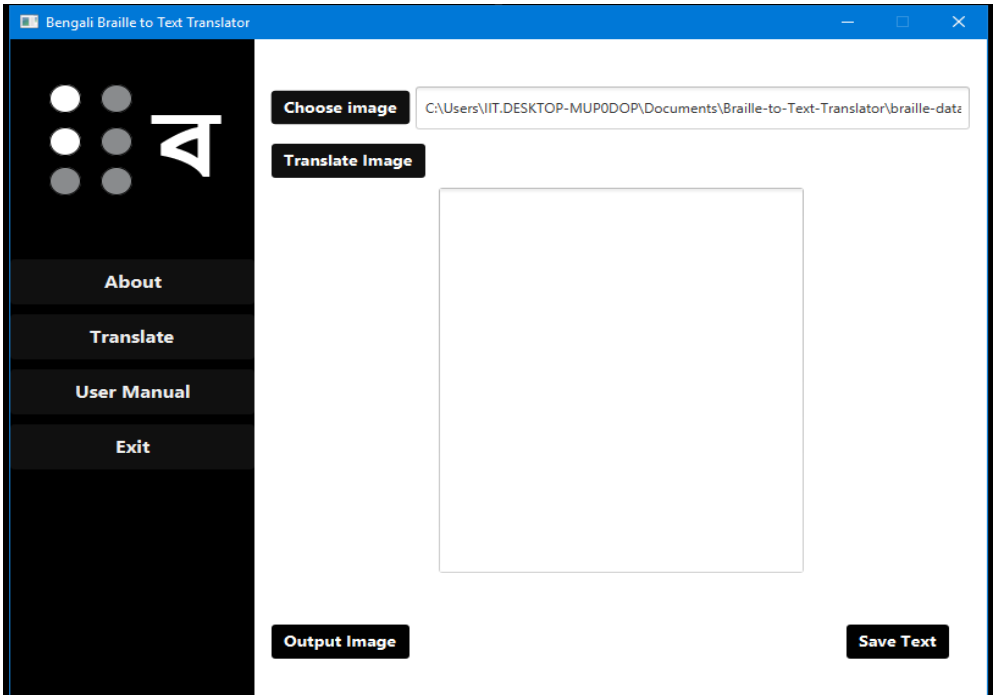


Figure 13: File choosing

10.4 Step 4: Image to Text Translation

After selecting user will click on the **Translate image** button. The corresponding Bengali text will be in the text field below the Translate image button as output.

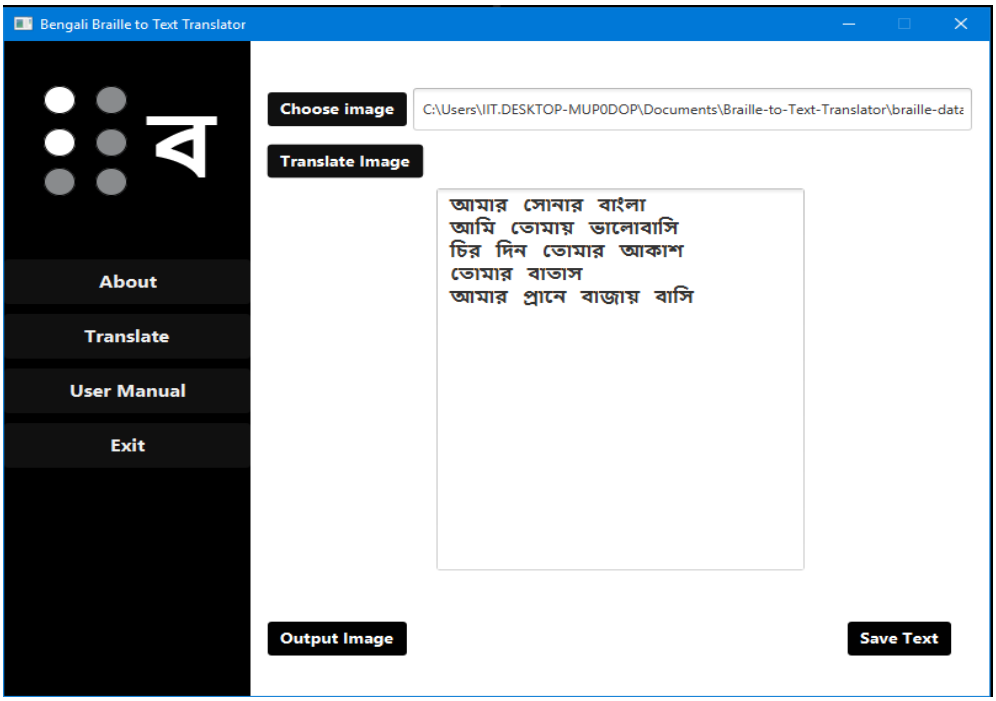


Figure 14: Translating image into text

10.5 Step 5: Saving the Output

To save the output in file the user will click on the **Save Text** button. A file chooser will open. The user will get option to select directory and file name to save the output file.

11 Result Discussion

In this application the main focus was translate image into corresponding Bengali text. Predicting Bengali characters and the predicting the corresponding word from braille code was the main concern.

For developing this project 11 dataset of Bengali braille code has been used. All the dataset is collected from a visual impaired person who is a former student of University of Dhaka. At time of collecting the dataset we consider to cover all the characters to be appeared in the dataset. We also try to cover the complex letter and the Bengali character those takes two braille cells to represent a single Bengali character. Dataset 1 has 12 lines with 188 characters and 35 words. Dataset 2 has 5 lines with 73 characters and 16 words. In total there exists 1745 letters and 353 words in all datasets.

The average accuracy of predicting correct letter is 94.7%. From the predicting letter the average accuracy of predicting word is 83%. All results are represented in the table given below.

Data File	Line Number	Number of Letter	Correctly Identified Letter	Accuracy (%)	Number of Words	Number of Words Where All Characters are Correctly Identified	Word Accuracy (%)
data_01	1	21	20	95.238095	3	1	33.33333333
data_01	2	19	18	94.736842	4	3	75
data_01	3	7	6	85.714286	1	0	0
data_01	4	17	16	94.117647	4	2	50
data_01	5	17	17	100	4	4	100
data_01	6	22	21	95.454545	3	2	66.66666667
data_01	7	15	15	100	3	3	100
data_01	8	9	9	100	2	2	100
data_01	9	18	17	94.444444	3	2	66.66666667
data_01	10	16	15	93.75	4	3	75
data_01	11	18	17	94.444444	3	2	66.66666667
data_01	12	9	9	100	1	1	100
data_01	All Lines	188	180	95.744681	35	25	71.42857143
data_02	1	14	14	100	3	3	100
data_02	2	16	16	100	3	3	100

data_02	3	15	15	100	4	4	100
data_02	4	10	10	100	2	2	100
data_02	5	18	18	100	4	4	100
data_02	All Lines	73	73	100	16	16	100
data_03	1	20	20	100	3	3	100
data_03	2	15	14	93.333333	2	2	100
data_03	3	18	17	94.444444	4	3	75
data_03	4	17	16	94.117647	4	3	75
data_03	5	19	19	100	3	3	100
data_03	6	6	6	100	2	2	100
data_03	7	18	18	100	4	4	100
data_03	8	10	9	90	2	1	50
data_03	9	15	14	93.333333	4	4	100
data_03	10	22	19	86.363636	3	1	33.33333333
data_03	11	15	13	86.666667	5	5	100
data_03	All Lines	175	165	94.285714	36	31	86.11111111
data_04	1	13	13	100	3	3	100
data_04	2	4	4	100	1	1	100
data_04	3	12	12	100	3	3	100
data_04	4	14	14	100	4	4	100
data_04	5	18	18	100	4	4	100
data_04	6	4	4	100	1	1	100
data_04	7	13	13	100	3	3	100
data_04	8	16	16	100	4	4	100
data_04	9	10	9	90	2	1	50
data_04	All Lines	104	103	99.038462	25	24	96
data_05	1	15	15	100	4	4	100
data_05	2	18	16	88.888889	4	3	75
data_05	3	16	15	93.75	3	2	66.66666667
data_05	4	21	19	90.47619	4	3	75
data_05	5	17	16	94.117647	4	3	75
data_05	6	14	12	85.714286	3	1	33.33333333
data_05	7	14	14	100	4	4	100
data_05	8	5	5	100	1	1	100
data_05	9	15	13	86.666667	3	2	66.66666667
data_05	10	15	14	93.333333	4	2	50
data_05	11	13	12	92.307692	3	2	66.66666667
data_05	All Lines	163	151	92.638037	37	27	72.97297297

data_06	1	11	10	90.909091	4	3	75
data_06	2	19	18	94.736842	4	4	100
data_06	3	10	10	100	2	2	100
data_06	4	18	18	100	4	4	100
data_06	5	19	18	94.736842	4	3	75
data_06	6	16	16	100	*3*	3	100
data_06	All Lines	93	90	96.774194	21	19	90.47619048
data_07	1	24	23	95.833333	4	2	50
data_07	2	24	22	91.666667	4	2	50
data_07	3	24	23	95.833333	5	4	80
data_07	4	23	22	95.652174	5	4	80
data_07	5	25	23	92	5	3	60
data_07	6	14	14	100	3	3	100
data_07	7	25	19	76	5	3	60
data_07	8	25	25	100	5	5	100
data_07	9	9	6	66.666667	2	1	50
data_07	All Lines	193	177	91.709845	38	27	71.05263158
data_08	1	26	20	76.923077	4	2	50
data_08	2	10	9	90	2	2	100
data_08	3	23	21	91.304348	6	5	83.33333333
data_08	4	24	24	100	4	4	100
data_08	5	23	23	100	5	5	100
data_08	6	22	21	95.454545	5	4	80
data_08	7	20	19	95	4	3	75
data_08	8	6	6	100	1	1	100
data_08	9	20	20	100	5	5	100
data_08	10	23	21	91.304348	5	4	80
data_08	11	7	7	100	1	1	100
data_08	All Lines	204	191	93.627451	36	32	88.88888889
data_09	1	26	26	100	6	6	100
data_09	2	22	19	86.363636	6	5	83.33333333
data_09	3	10	10	100	2	2	100
data_09	4	21	19	90.47619	4	3	75
data_09	5	25	21	84	5	3	60
data_09	6	21	19	90.47619	5	3	60
data_09	7	25	24	96	4	3	75
data_09	8	6	5	83.333333	1	1	100
data_09	9	20	17	85	5	3	60
data_09	10	19	17	89.473684	2	1	50

data_09	All Lines	195	177	90.769231	40	30	75
data_10	1	23	23	100	5	5	100
data_10	2	23	22	95.652174	4	3	75
data_10	3	22	21	95.454545	4	3	75
data_10	4	25	25	100	5	3	60
data_10	5	11	11	100	2	2	100
data_10	6	21	21	100	6	6	100
data_10	7	11	11	100	3	3	100
data_10	8	21	20	95.238095	4	2	50
data_10	9	19	19	100	3	3	100
data_10	All Lines	176	173	98.295455	36	30	83.33333333
data_11	1	21	20	95.238095	6	5	83.33333333
data_11	2	24	23	95.833333	4	3	75
data_11	3	12	10	83.333333	2	1	50
data_11	4	21	21	100	4	4	100
data_11	5	12	12	100	2	2	100
data_11	6	25	25	100	4	6	66.66666667
data_11	7	21	21	100	6	6	100
data_11	8	24	22	91.666667	4	2	50
data_11	9	4	4	100	1	1	100
data_11	10	17	16	94.117647	3	2	66.66666667
data_11	All Lines	181	174	96.132597	33	32	96.96969697
Total		1745	1654	94.7851	353	293	83.00283286

12 Conclusion

I am pleased to submit the technical report on Bengali Braille to Text Translator. From this, the readers will get a clear and easy view of tactile writing system. They will also get a good understanding of the translation process.

This document can be used effectively to maintain the software development cycle for the project. I have presented a detailed description of the total system. It will be much easy to conduct the whole project using this SRS. It will also help me to determine the pitfalls that may come ahead. Hopefully, this document can also help other software engineering students as well as practitioners.

I have tried my best to make effective and fully focused all over the project in a short time. I wish, the readers will find it in order.

13 Bibliography

[1] "Digital Image Processing Basics," 30 August 2019. [Online]. Available: https://www.geeksforgeeks.org/digital-image-processing-basics/.

[2] "Grayscale to RGB Conversion," tutorialspoint, [Online]. Available: https://www.tutorialspoint.com/dip/grayscale_to_rgb_conversion.htm. [Accessed 30 August 2019].

[3] [Online]. Available: http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html. [Accessed 30 August 2019].

[4] "MathWOrks," [Online]. Available: https://www.mathworks.com/help/images/morphological-dilation-and-erosion.html. [Accessed 31 August 2019].

[5] S. D. Al-Shamma, "Arabic Braille Recognition and Transcription into Text and Voice," *Cairo International Biomedical Engineering Conference,* p. 228, 2010.

[6] M. Gadag, "Generation of English Text from Scanned Braille Document," *International Journal of Engineering Science and Computing,* vol. 6, no. 4, p. 4447, 2016.

[7] S.Padmavathi, "CONVERSION OF BRAILLE TO TEXT IN ENGLISH, HINDI AND TAMIL LANGUAGES," *International Journal of Computer Science, Engineering and Applications,* vol. 3, no. 3, p. 25, 2013.