Software Design Document for Computer-aided Simultaneous Interpreting Software

Abdelrahman Ashraf
Ahmed Sameh
Mostafa Mahran
Wessam Ahmed

Supervised by

Dr. Ashraf Abdelraouf
Eng. Nada Shorim

March 2019

1 Introduction

1.1 Purpose

This software design document purpose is to fully describe the architecture of our Simultaneous Interpreting system. Our system depends on multiple glossaries provided by Al-Alsun concerning multiple domains from medicine to terrorism war. This document will explain in details, the components of the system represented in the block diagram, the flow of the project with sequence diagram, the data handling in the ER diagram also the implementation of the project and its development will be shown in the class diagram . This software design document (SDD) is, therefore, intended for the stakeholders and developers of the our system: Al-Alsun students and professors and the simultaneous interpreters as a whole. This document is also presented as a part of a graduation project at Misr International University (MIU).

1.2 Scope

The system described in this document builds on existing standards set by other language translation and interpretation software available on the market. It provides benefits for users in the Middle East and North Africa region through the use of advanced methods for context/domain specification in Arabic (and English) text. The accuracy of translation depends on the semantic understanding of words in context and not vocabulary translation alone through the use of a dictionary. Optimization benefits are gained by minimizing the number of terms the system has to look through in order to arrive to an accurate translation, resulting in cost savings. The system can be added to an existing network of components that perform tasks it is dependant on or exist independently, the limitations that drive this choice are based on implementation and network configuration, as it can exist solely to provide users with a display for contextual extraction or translation as well as domain specification.

1.3 Overview

Language interpreters rely on a number of methods for the easy look-up and translation of words, from glossary search engines to speech-to-text transcription software. The increasing complications that come with the use of these software makes their adoption harder to achieve on a wider scope, so their use is limited to diminishing groups within the language services industry. This document will explore the use of an artificially intelligent computer system for the transcription, semantic (natural) understanding, and translation of live speech using proven methods. The system will see usage within the Alsun department at the Misr International University, as such, its completion is subject to the the work of both the Computer Science and Alsun departments. The Alsun department will be responsible for gathering a glossary of terms in both English and Arabic that will be used by the system for translation as well as natural understanding. As such, this document will provide a road map for fulfilling the requirements laid out by the Alsun department at the Misr International University.

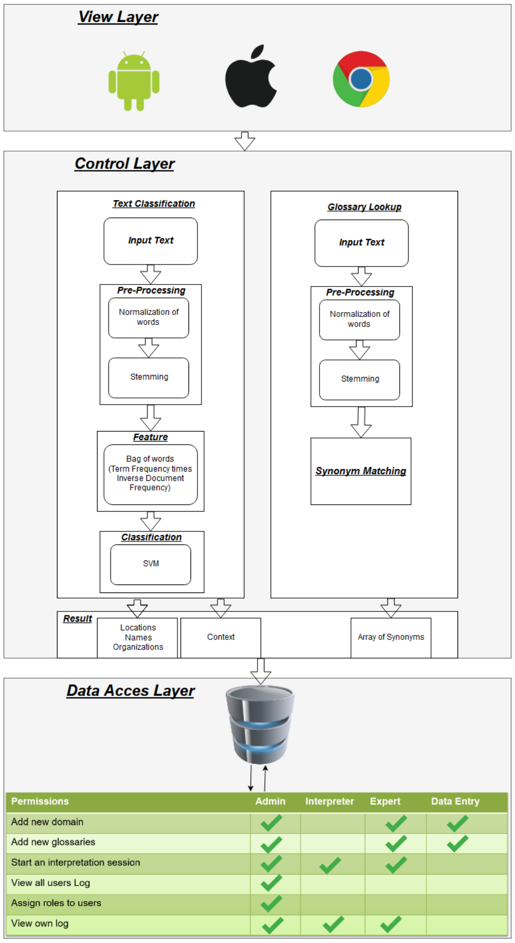
1.4 Definitions and Acronyms

Table 1: Definitions and Acronyms

Term	Definition
Software Design Document (SDD)	Used as the primary medium for communicating software design information.
NLU	Natural language understanding is a subtopic of natural-language processing in artificial intelligence that deals with machine reading comprehension. Natural-language understanding is considered an AI-hard problem.
NLTK	Natural Language Toolkit is a platform used for building Python programs that work with human language data for applying in statistical natural language processing (NLP
NLP	Natural language processing is a subfield of computer science, information engineering, and artificial intelligence concerned with the interactions between computers and human languages.

2 System Overview

In order to achieve the most accurate interpretation, the system has to do multiple tasks first from pre-processing and classification, also using the bag of words approach. The pre-processing part is very essential as it happens on two types of data. First we do the normalization and stemming on the input voice when converting it to text to achieve the most accurate speech to text conversion, specially in Arabic as it is a very difficult language to understand while listening to. After that the pre-processing part is done also on the glossaries on the database and the converted text from the speech to make the glossary look-up and the comparison of words much easier. Also there is a synonym matching methodology we are doing that we create an array of synonyms for each terms to achieve more accuracy and more easier lookup from the glossary.



Permissions	Admin	Interpreter	Expert	Data Entry
Add new domain	✓		✓	✓
Add new glossaries	✓		✓	✓
Start an interpretation session	✓	✓	✓	
View all users Log	✓			
Assign roles to users	✓			
View own log	✓	✓	✓	

Figure 1: Block Diagram

3 System Architecture

3.1 Architectural Design

For the Architectural Design, based on the analysis of the functional and non-functional requirement, the system was designed in a way that would insure the MVC system architecture as shown below:

3.1.1 View

It is responsible for the presentation of data and representing the User Interface(UI). We have two different interfaces one is responsible for Admin operations and the other one is responsible for representing the core part including

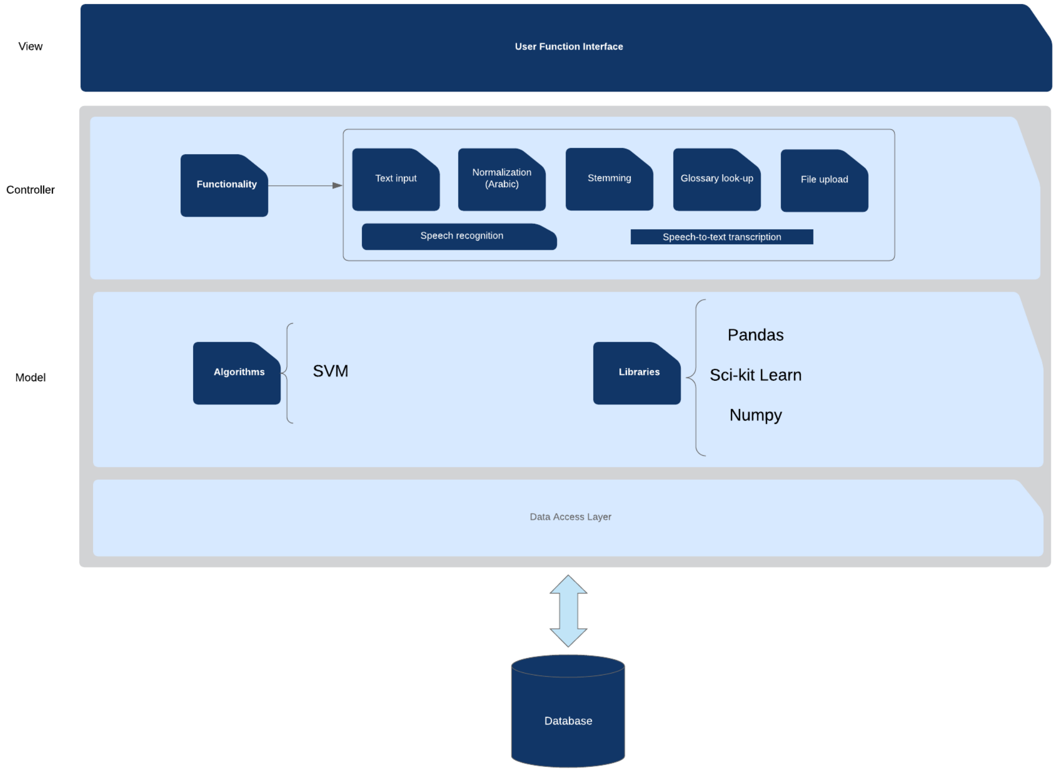


Figure 2: Software Architecture Diagram

speech to text and terms extraction from the text.

3.1.2 Controller

It is responsible for binding the view and model. The interactions and requests made within the view are taken and sent to the database to fetch data with the use of models then it forward data to the view again to be shown. Some of the controllers we have are: the user controller that deals with the interpreter and how they interacts with the system, also the admin controller which controls how the admin interacts with the system and how they give and allow access for other user. Moreover the controller controls multiple functionalities in the system like the input of the text and pre-processing done on it, the glossary look up to extract the relevant terms in the text and other similar tasks.

3.1.3 Model

The interpreter and expert models deals with simultaneous interpreters that will interact with the system either by using the main interpreting functionality or add and extract data from the database.
Algorithm:
We use the SVM classification algorithm and specially the linear kernel part of it as it is very efficient in working with text also it scales better when working with larger samples of data. Basically the SVM classifier takes the training data and plot depending on the number of the classes their features on the plane and draw a line to separate the classes from each other, then when a new text input comes it extracts its features and put them on the plane to find the best class or nearest one to this text.
Libraries:
We use the numpy library for handling of the array in python.
The Sci-Kit learn library which contains the classifier we are using in our system
And pandas library for data manipulation and analysis.

3.2 Decomposition Description

3.2.1 System Sequence Diagram

Each step that occurs across all subsystems to achieve text analysis is described in the figure below (Fig. 3).

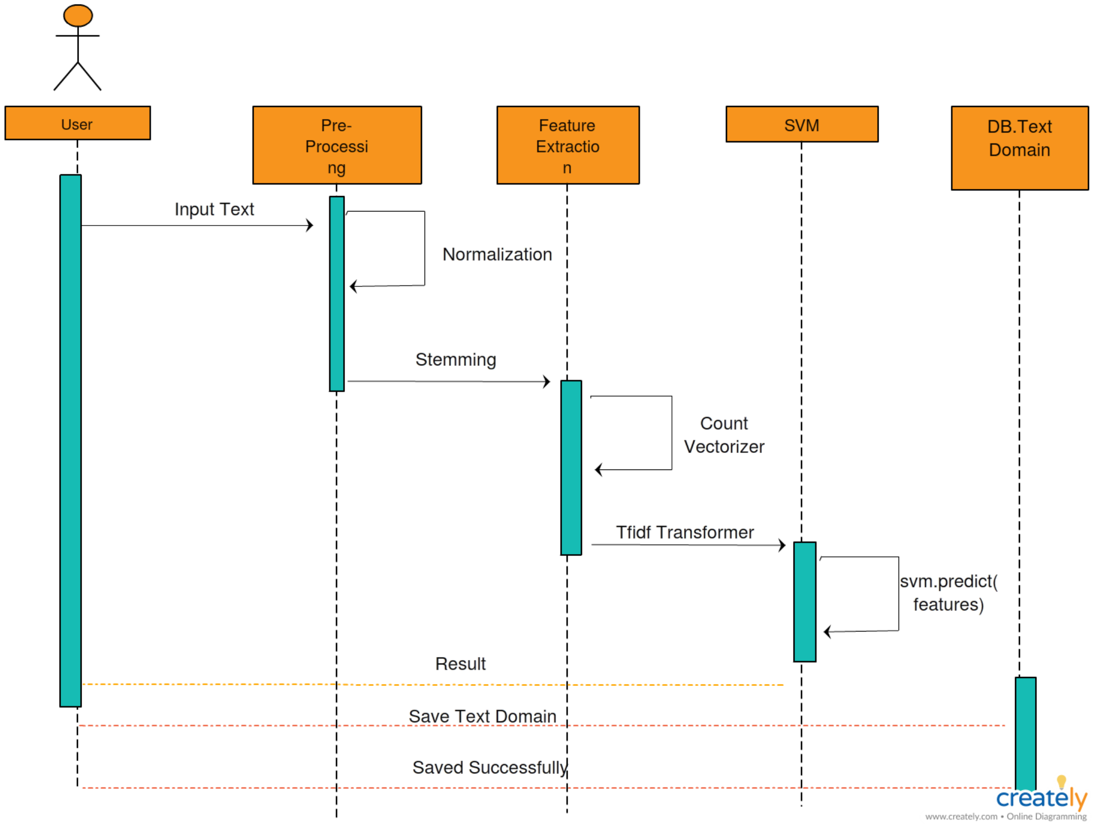


Figure 3: System Sequence Diagram

3.2.2 User Authentication Sequence

- Rationale: The user should register for an account before being able to use the application.
- Input: name, e-mail, password.
- Output: Validation response.
- Priority: Medium. Each user must have an account and become assigned a role before being able to use the application.

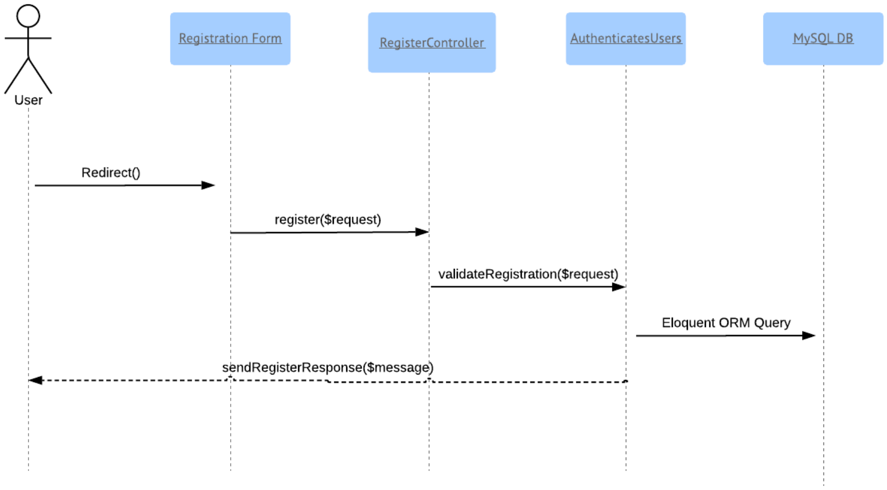


Figure 4: Registration Sequence Diagram

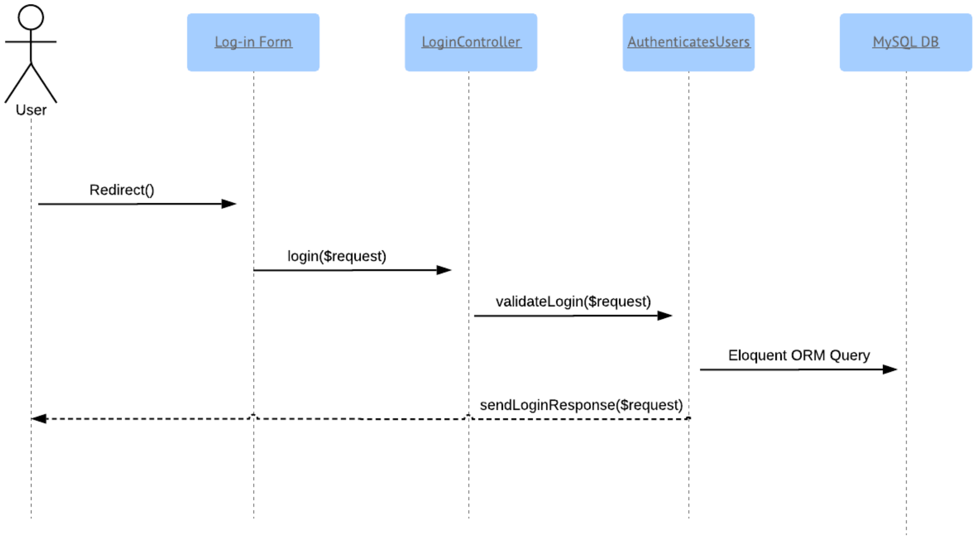


Figure 5: Login Sequence Diagram

3.2.3 Analyze Text Sequence

- Rationale: The user can input text to the system in the form of plain text or via a third-party-enabled speech-to-text module and receive a collection of terms in response, along with their corresponding translation.
- Input: Text, or audio speech.
- Output: Terms obtained via database look-up.
- Priority: High. This is an important functionality of the same.

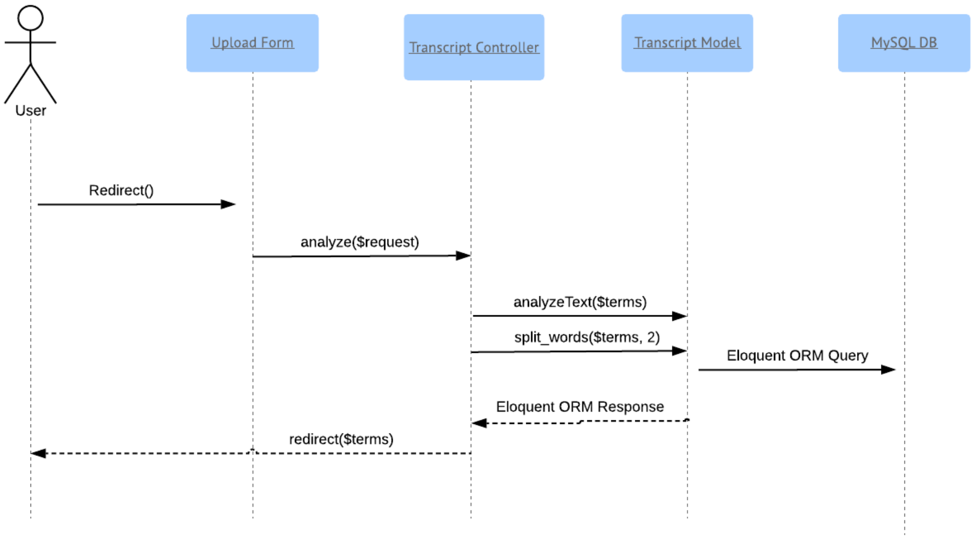


Figure 6: Analyze Transcript Sequence Diagram

3.2.4 Upload CSV Sequence

- Rationale: The user can upload terms in the form of CSV (comma-separated values) files in order to input terms into the system if they have the correct permission.
- Input: CSV file.
- Output: None.
- Priority: Medium. This is a functionality afforded to a select collection of users of the system.

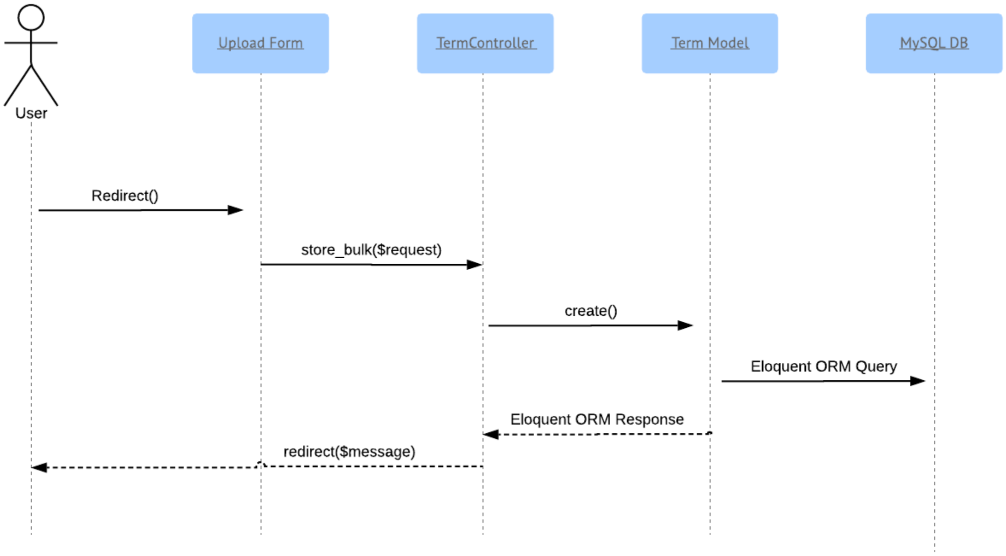


Figure 7: Upload CSV Sequence Diagram

3.3 Design Rationale

As mentioned previously, we have used Model-View-Controller (MVC) as our architecture as it helped us separate the functionality and data of our system from the presentation. So, we can easily make modifications, re-use and optimize functionality part as it is our core. Also the software we are developing efficiency and accuracy is a very important aspect of it so it will be very sensitive with data so this should be developed in a very accurate and efficient way.

3.3.1 Possible Algorithms

There were so many alternative algorithms that we could have used like decision trees and naive bayes that are generally very good for text classification.

Naive Bayes: It works on conditional probability. It calculates the probability of the input relative to some decisions that was previously taken. It is well suited when the input has large number of dimensions.

Decision Trees: It is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.

SVM: Linear SVM is given a set of train data which belong to a certain class to find an optimal separating line. It tries to maximize the distance between each class to avoid misclassification. Then a test data are given to be classified to one of the classes formed before[5].

We have chooses the linear SVM as our classifier as after several experiments it was found out that the SVM gave the best result and most accurate with the highest f-measure which is technically the mean between the precision and the recall score.

3.3.2 Frameworks

There are multiple frameworks that we have used like Django and Laravel frameworks for PHP to make the transition from the python to the web app easy to maintain and scalabale.

PHP Laravel: Laravel is a free, open-source PHP web framework, intended for the development of web applications following the modelviewcontroller architectural pattern. It was mainly used for the creation and maintaing of the web applicataion.

Django RESTful API: Django REST framework is a powerful and flexible toolkit for building Web APIs. Django is a high-level Python Web framework encouraging rapid development and pragmatic, clean design. A web application framework is a toolkit of components all web applications need. Meaning that we the help of Django we can integrate out python code which contains the machine learning part and text classification into the web app with ease.

4 Data Design

4.1 Data Description

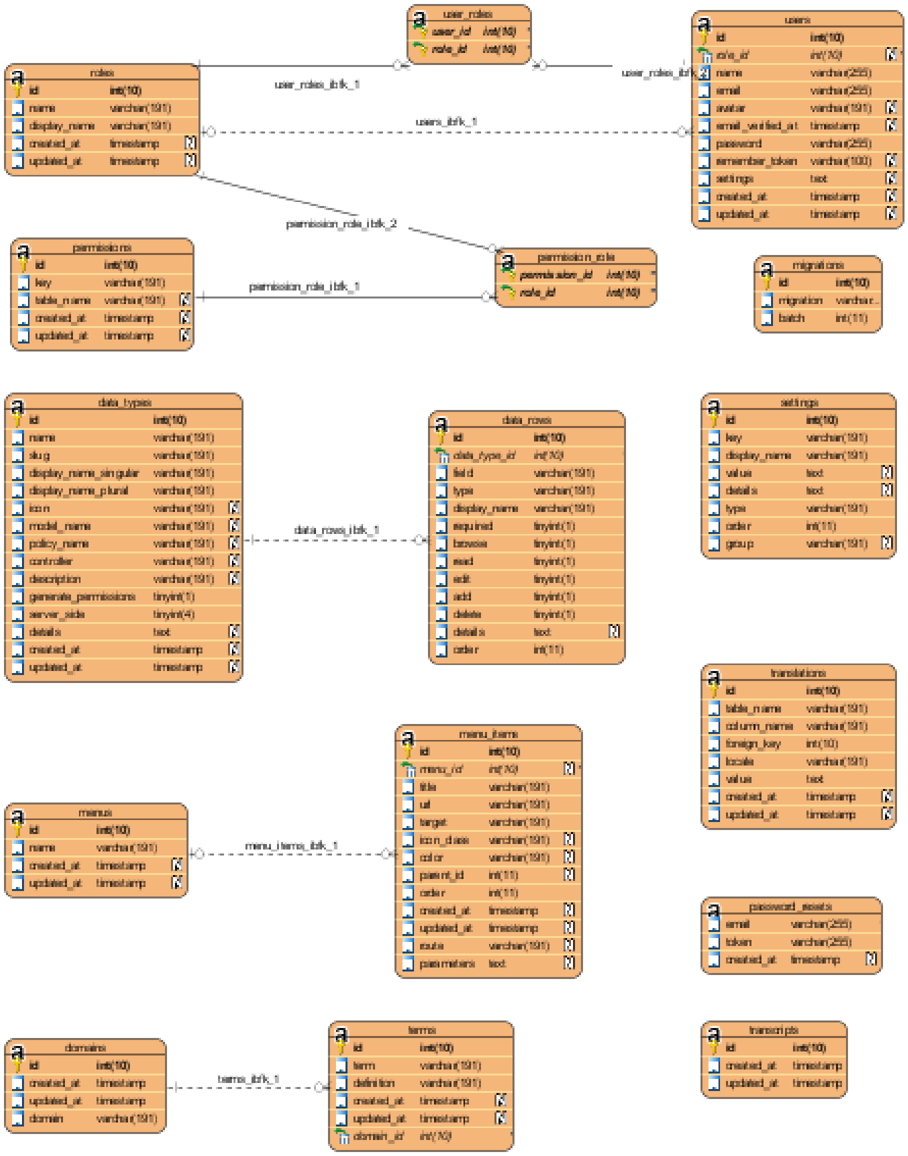


Figure 8: System ERD

4.2 Data Dictionary

Table 2: Data Dictionary

Domains	Attributes: ID, domain. This table holds the information that specifies what domain a term belongs to.
Permission Roles	Attributes: role_id, permission_id. This is a pivot table that ties each permission to its corresponding role.
Permissions	Attributes: ID, key. This table holds all user permissions.
Roles	Attributes: ID, name, display_name. This table holds all user roles.
Terms	Attributes: ID, term, definition, domain_id, normalized, stem. This table holds all terms, their corresponding domain ID, their Arabic/English definition, stemmed and normalized form.
Transcripts	Attributes: ID, transcript, user_id. This table holds a user's transcript history/
Users	Attributes: ID, name, email, avatar, remember_token, settings, email_verified_at, password. This table holds all users and their information.
User Roles	Attributes: user_id, role_id. This is a pivot table that ties each user to their role.

5 Component Design

5.1 Class Diagram

5.1.1 Pre-Processing Class Diagram

Figure 8 showcases all classes responsible for the pre-processing phase of text analysis.

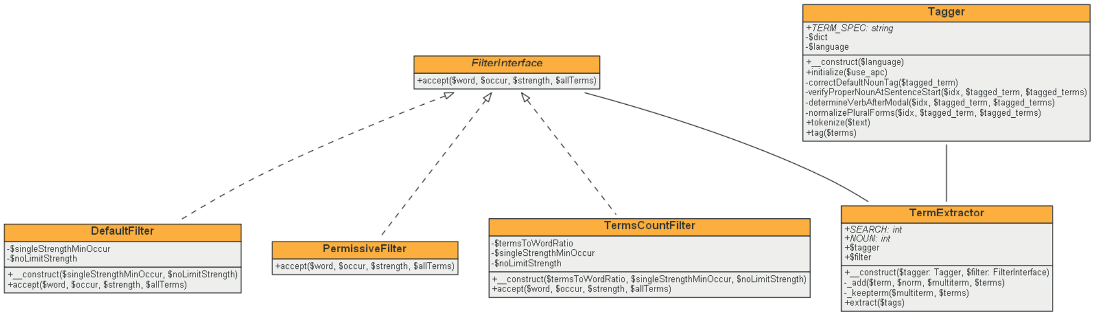

Tagger
+TERM_SPEC: string
-$dict
-$language
+__construct($language)
+initialize($use_apc)
-correctDefaultNounTag($tagged_term)
-verifyProperNounAtSentenceStart($idx, $tagged_term, $tagged_terms)
-determineVerbAfterModal($idx, $tagged_term, $tagged_terms)
-normalizePluralForms($idx, $tagged_term, $tagged_terms)
+tokenize($text)
+tag($terms)
FilterInterface
+accept($word, $occur, $strength, $allTerms)
DefaultFilter
-$singleStrengthMinOccur
-$noLimitStrength
+__construct($singleStrengthMinOccur, $noLimitStrength)
+accept($word, $occur, $strength, $allTerms)
PermissiveFilter
+accept($word, $occur, $strength, $allTerms)
TermsCountFilter
-$termsToWordRatio
-$singleStrengthMinOccur
-$noLimitStrength
+__construct($termsToWordRatio, $singleStrengthMinOccur, $noLimitStrength)
+accept($word, $occur, $strength, $allTerms)
TermExtractor
+SEARCH: int
+NOUN: int
+$tagger
+$filter
+__construct($tagger: Tagger, $filter: FilterInterface)
-_add($term, $norm, $multiterm, $terms)
-_keepterm($multiterm, $terms)
+extract($tags)

Figure 9: Pre-Processing Class Diagram

5.1.2 Model(s) Class Diagram

Figure 9 showcases the core models associated with the system.

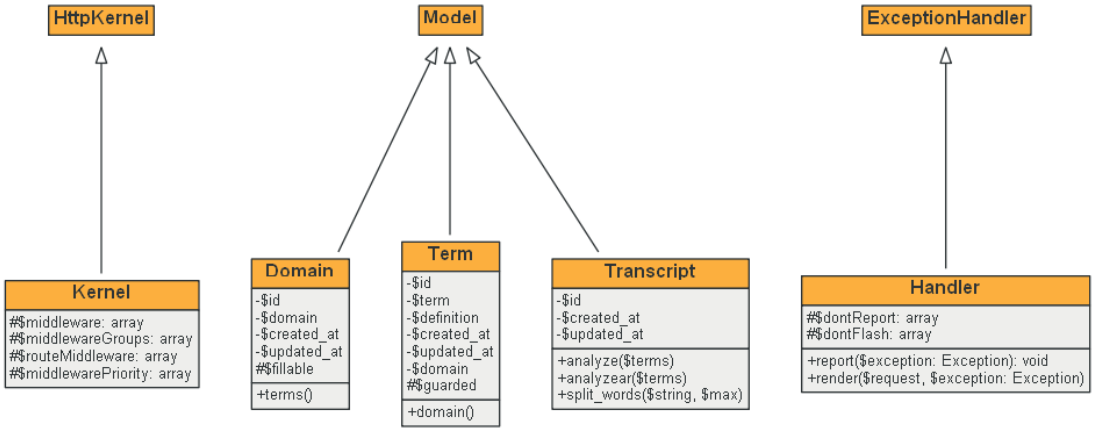
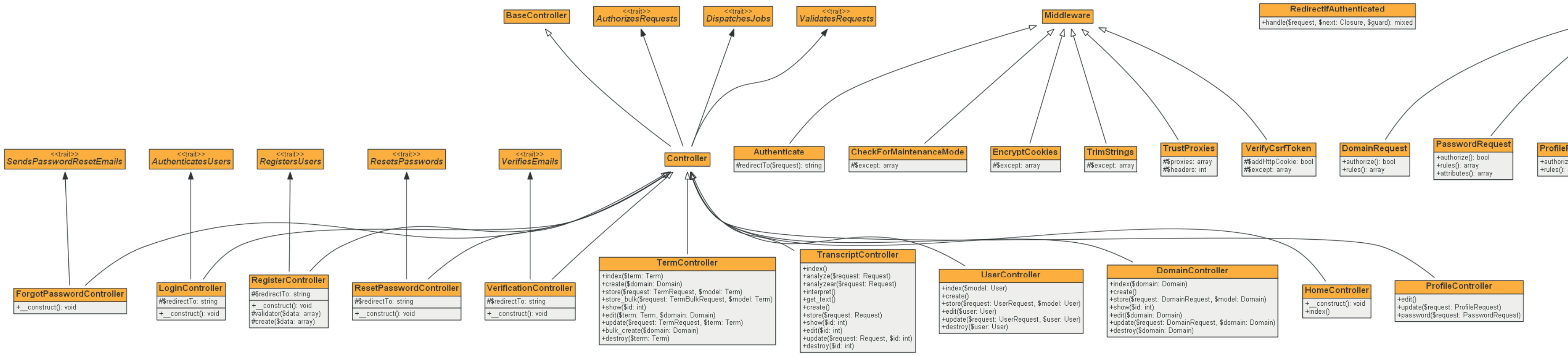


Figure 10: Model Class Diagram

5.1.3 Controller(s) Class Diagram

Figure 10 showcases the main controllers associated with the system.


BaseController
<<trait>>
AuthorizesRequests
<<trait>>
DispatchesJobs
<<trait>>
ValidatesRequests
Middleware
RedirectIfAuthenticated
+handle($request, $next: Closure, $guard): mixed
<<trait>>
SendsPasswordResetEmails
<<trait>>
AuthenticatesUsers
<<trait>>
RegistersUsers
<<trait>>
ResetsPasswords
<<trait>>
VerifiesEmails
Controller
Authenticate
#redirectTo($request): string
CheckForMaintenanceMode
#$except: array
EncryptCookies
#$except: array
TrimStrings
#$except: array
TrustProxies
#$proxies: array
#$headers: int
VerifyCsrfToken
#$addHttpCookie: bool
#$except: array
DomainRequest
+authorize(): bool
+rules(): array
PasswordRequest
+authorize(): bool
+rules(): array
+attributes(): array
ProfileR
+authoriz
+rules():
ForgotPasswordController
+__construct(): void
LoginController
#$redirectTo: string
+__construct(): void
RegisterController
#$redirectTo: string
+__construct(): void
#validator($data: array)
#create($data: array)
ResetPasswordController
#$redirectTo: string
+__construct(): void
VerificationController
#$redirectTo: string
+__construct(): void
TermController
+index($term: Term)
+create($domain: Domain)
+store($request: TermRequest, $model: Term)
+store_bulk($request: TermBulkRequest, $model: Term)
+show($id: int)
+edit($term: Term, $domain: Domain)
+update($request: TermRequest, $term: Term)
+bulk_create($domain: Domain)
+destroy($term: Term)
TranscriptController
+index()
+analyze($request: Request)
+analyzear($request: Request)
+interpret()
+get_text()
+create()
+store($request: Request)
+show($id: int)
+edit($id: int)
+update($request: Request, $id: int)
+destroy($id: int)
UserController
+index($model: User)
+create()
+store($request: UserRequest, $model: User)
+edit($user: User)
+update($request: UserRequest, $user: User)
+destroy($user: User)
DomainController
+index($domain: Domain)
+create()
+store($request: DomainRequest, $model: Domain)
+show($id: int)
+edit($domain: Domain)
+update($request: DomainRequest, $domain: Domain)
+destroy($domain: Domain)
HomeController
+__construct(): void
+index()
ProfileController
+edit()
+update($request: ProfileRequest)
+password($request: PasswordRequest)

5.1.4 Service Provider Class Diagram

Figure 11 showcases all HTTP service providers within the application.

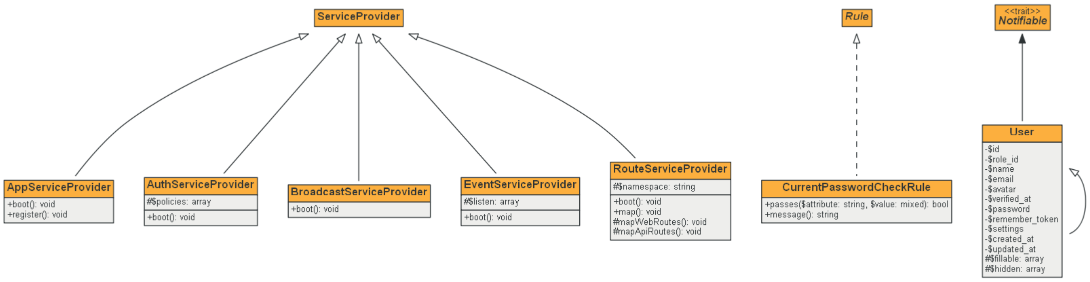


Figure 12: Service Provider Class Diagram

5.2 Pre-Processing

5.2.1 Normalization

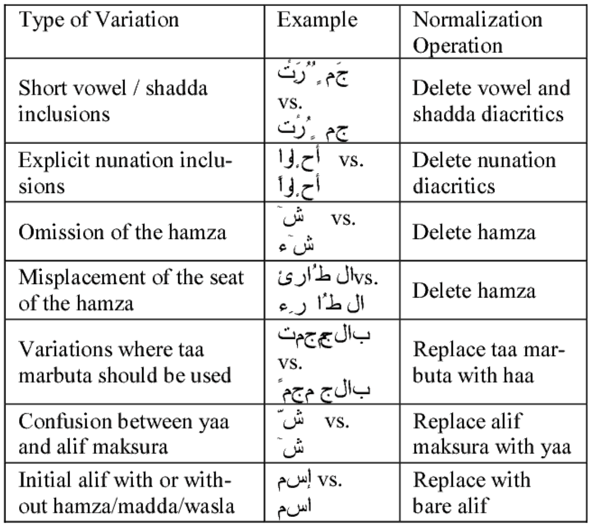
Normalization to remove some characters. We remove some characters that can be detected in the Arabic language like the hamza, taa' marboota and yaa' layena.

5.2.2 Stemming

The second part of our Arabic approach is to stemming of a given word. We take every word and implement the algorithm on it to return the root of the word. We used the ISRI stemmer in the NLTK library. If the function cannot find a root for a specific word it returns the normalized form of the word rather that returning it without any change. The ISRI stemmer added some modifications to the word as they added more than 60 stopwords.

5.3 Text Classification

After that we use the bag of words approach to do text classification on the input text to extract the domain of it. This was done by using the CountVectorizer function in the sklearn library. Simply what this function does is to build a dictionary of features and transforms documents to feature vectors. Once fitted, the vectorizer has built a dictionary of feature indices and the index value of a



Type of Variation	Example	Normalization Operation
Short vowel / shadda inclusions	جَم ٍ ٌرَتْ vs. جم ٍ رُتْ	Delete vowel and shadda diacritics
Explicit nunation inclusions	أح.لوا vs. أح.لوأ	Delete nunation diacritics
Omission of the hamza	شَ vs. شَ ء	Delete hamza
Misplacement of the seat of the hamza	ال طُارئvs. ال طًا رِء	Delete hamza
Variations where taa marbuta should be used	بالجمجمت vs. بالج مجمً	Replace taa marbuta with haa
Confusion between yaa and alif maksura	شّ vs. شَ	Replace alif maksura with yaa
Initial alif with or without hamza/madda/wasla	إسم vs. اسم	Replace with bare alif

Figure 13: Normalization of Arabic Language

word in the vocabulary is linked to its frequency in the whole training corpus. It supports counts of N-grams of words or consecutive characters.

This is a very good start do feature extraction on text, but it has some limitations. Longer documents will have more occurrences of a word even they talk about the same subject as a shorter one. So a good solution for this is to divide the number of occurrences of each word in a document by the total number of words in the document. In addition to that we decrease the weight of words that occur in different or many documents in the corpus as they are less informative than those who occurred less, this is called Term Frequency times Inverse Document Frequency. We do that by using the TfidfTransformer function also from the sklearn library.

To complete the process we trained the liner kernel of the SVM, it was used as it is the best kernel available working with text. The idea mainly is to create a model that can predict the category of the input text and to make the task of the glossary look-up to extract the revelant terms according to this domain easier.

set	description	examples
D	diacritics-vowelizations	سَ سِ سُ سْ سّ سً سٍ سٌ
$P3$	prefixes of length three	ولل، وال، كال، بال
$P2$	length two prefixes	ال، لل
$P1$	length one prefixes	ل، ب، ف، س، و ي، ت، ن، ا
$S3$	length three suffixes	تمل، همل، تان، تين، گمل
$S2$	length two suffixes	ون، ات، ان، ين تن، گم، هن، نا يا، ها، تم، گن ني، وا، ما، هم
$S1$	length one suffixes	ة،ه،ي،گ،ت،ا،ن

Figure 14: Stemming of Arabic Language

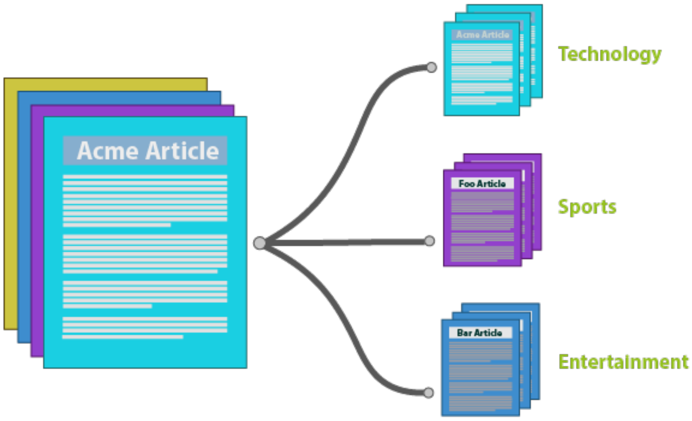


Figure 15: SVM classification of text

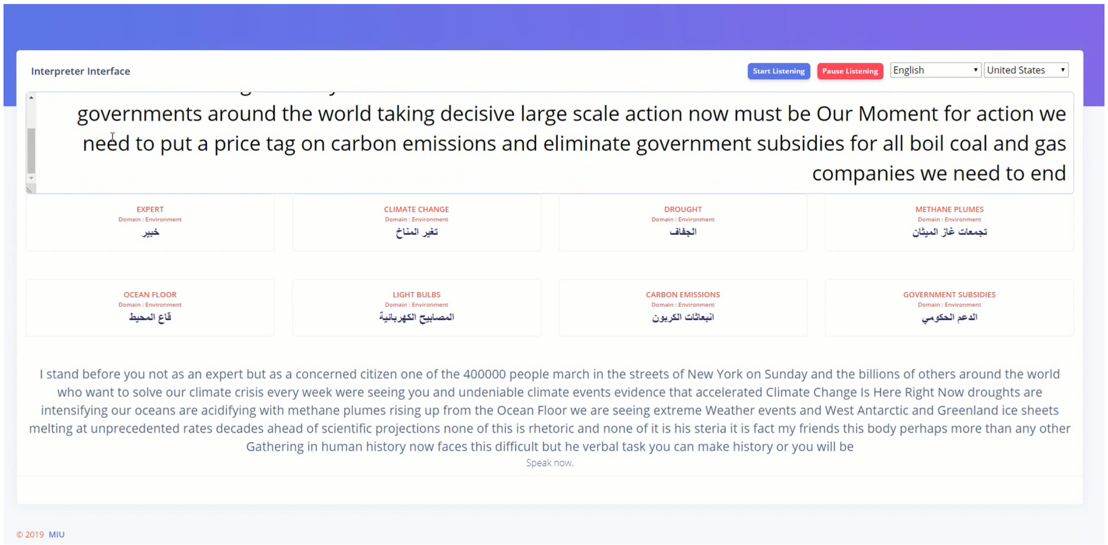
6 Humnan Interface Design

6.1 Overview of User Interface

Describe the functionality of the system from the user s perspective. Explain how the user wil be able to use your system to complete al the expected features and the feedback information that wil be displayed for the user.

6.2 Screen Images

6.2.1 Main Interface



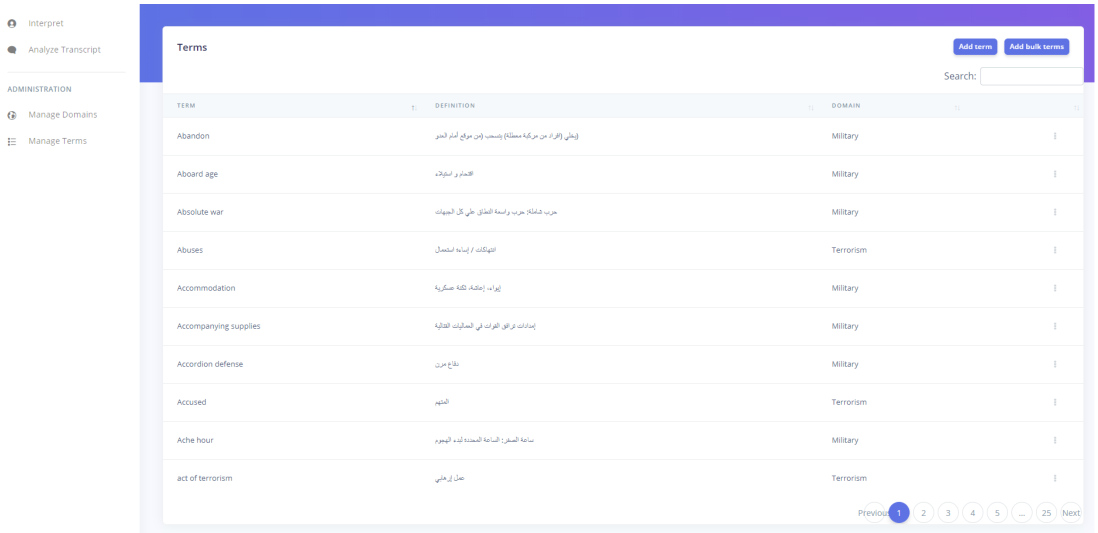
governments around the world taking decisive large scale action now must be Our Moment for action we need to put a price tag on carbon emissions and eliminate government subsidies for all boil coal and gas companies we need to end

EXPERT Domain : Environment خبير	CLIMATE CHANGE Domain : Environment تغير المناخ	DROUGHT Domain : Environment الجفاف	METHANE PLUMES Domain : Environment تجمعات غاز الميثان
OCEAN FLOOR Domain : Environment قاع المحيط	LIGHT BULBS Domain : Environment المصابيح الكهربائية	CARBON EMISSIONS Domain : Environment انبعاثات الكربون	GOVERNMENT SUBSIDIES Domain : Environment الدعم الحكومي

I stand before you not as an expert but as a concerned citizen one of the 400000 people march in the streets of New York on Sunday and the billions of others around the world who want to solve our climate crisis every week were seeing you and undeniable climate events evidence that accelerated Climate Change Is Here Right Now droughts are intensifying our oceans are acidifying with methane plumes rising up from the Ocean Floor we are seeing extreme Weather events and West Antarctic and Greenland ice sheets melting at unprecedented rates decades ahead of scientific projections none of this is rhetoric and none of it is his steria it is fact my friends this body perhaps more than any other Gathering in human history now faces this difficult but he verbal task you can make history or you will be

Figure 16: Interpreter Interface

6.2.2 Manage Terms Interface



TERM	DEFINITION	DOMAIN
Abandon	يخلي (افراد من مركبة معطلة) ينسحب (من موقع أمام العدو)	Military
Aboard age	اقتحام و استيلاء	Military
Absolute war	حرب شاملة: حرب واسعة النطاق على كل الجبهات	Military
Abuses	انتهاكات / إساءه استعمال	Terrorism
Accommodation	إيواء، إعاشة، تكنة عسكرية	Military
Accompanying supplies	إمدادات ترافق القوات في العمليات القتالية	Military
Accordion defense	دفاع مرن	Military
Accused	المتهم	Terrorism
Ache hour	ساعة الصفر: الساعة المحدده لبدء الهجوم	Military
act of terrorism	عمل إرهابي	Terrorism

Figure 17: Manage Terms

6.2.3 Add Individual Term

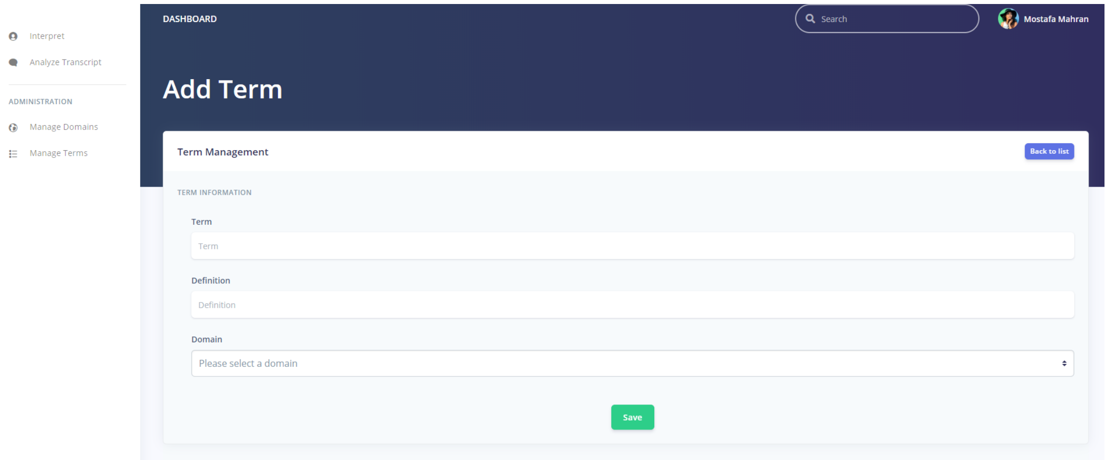


Figure 18: Add Individual Term

6.2.4 Add Bulk Terms

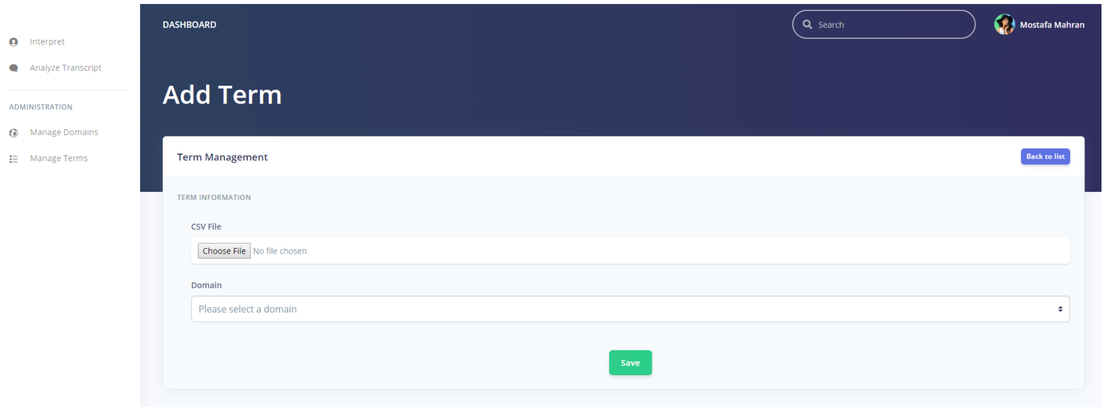
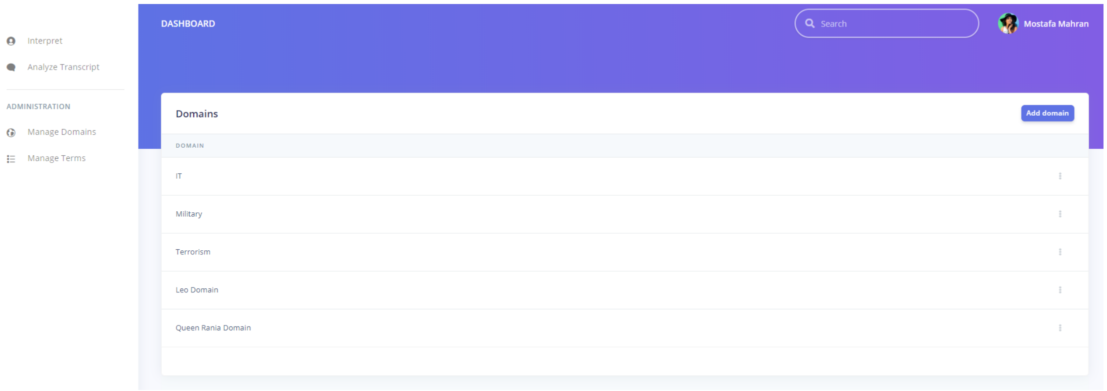


Figure 19: Add Bulk Terms (Upload CSV)

6.2.5 Manage Domains Interface



DOMAIN
IT
Military
Terrorism
Leo Domain
Queen Rania Domain

Figure 20: Manage Domains

6.2.6 Add Domain Interface

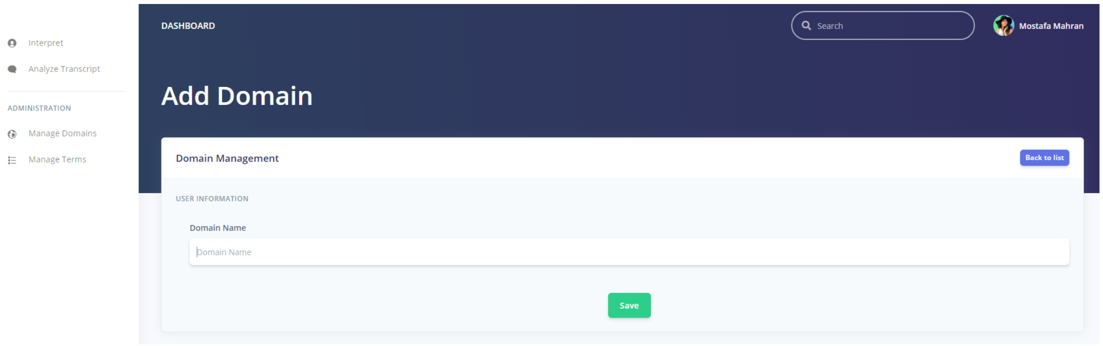


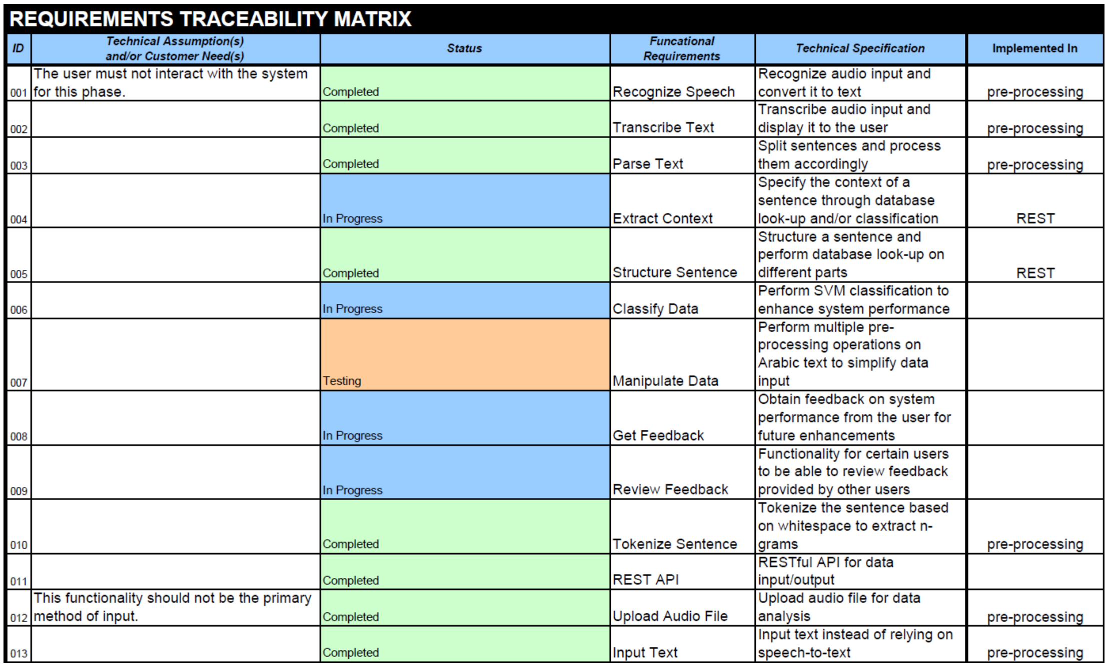
Figure 21: Add Domains

6.3 Screen Objects and Actions

The main interpreter interface (Fig. 16) is afforded to users of all types. The user is prompted to begin an interpretation session and trigger the speech-to-text module using the ”Start Listening” button. Audio input is received and dissected for database look-up, the equivalent terms are returned to the interface and displayed in the small cards located below the transcript text field. As soon as the speaker pauses, the transcript is delegated to the static text field at the bottom of the page.
Certain users are allowed to look at an overview of terms as shown in Fig. 17, and users with the appropriate permissions can add new terms using the interface in Fig.18, or upload a .csv file as shown in Fig. 19.

Accordingly, certain users can also look at the available domains that each term would be associated with in Fig. 20, or add a new domain as shown in Fig. 21.

7 Requirements Matrix



REQUIREMENTS TRACEABILITY MATRIX

ID	Technical Assumption(s) and/or Customer Need(s)	Status	Funcational Requirements	Technical Specification	Implemented In
001	The user must not interact with the system for this phase.	Completed	Recognize Speech	Recognize audio input and convert it to text	pre-processing
002		Completed	Transcribe Text	Transcribe audio input and display it to the user	pre-processing
003		Completed	Parse Text	Split sentences and process them accordingly	pre-processing
004		In Progress	Extract Context	Specify the context of a sentence through database look-up and/or classification	REST
005		Completed	Structure Sentence	Structure a sentence and perform database look-up on different parts	REST
006		In Progress	Classify Data	Perform SVM classification to enhance system performance	
007		Testing	Manipulate Data	Perform multiple pre-processing operations on Arabic text to simplify data input	
008		In Progress	Get Feedback	Obtain feedback on system performance from the user for future enhancements	
009		In Progress	Review Feedback	Functionality for certain users to be able to review feedback provided by other users	
010		Completed	Tokenize Sentence	Tokenize the sentence based on whitespace to extract n-grams	pre-processing
011		Completed	REST API	RESTful API for data input/output	
012	This functionality should not be the primary method of input.	Completed	Upload Audio File	Upload audio file for data analysis	pre-processing
013		Completed	Input Text	Input text instead of relying on speech-to-text	pre-processing

Figure 22: Requirements Matrix

8 References