Forensics Linguistics and Authorship Attribution using stylometry

by

Yousef Mohamed Abdullah - 184367

A dissertation submitted in partial fulfillment of the
requirements for the degree of
Bachelor of Computer Science(SE programme)

in the

Faculty of Computer Science
of the
October University for Modern Sciences and Arts (MSA), EGYPT

Graduation Projects advisor:
Dr. Wael Gomaa

(July 2021)

Abstract

Linguistic professionals have been manually analyzing and extracting sylometric features from texts for a long time, which is a tedious and time-consuming process. Previous software have attempted to aid linguists to search for said features, however most of them are either expensive or were counter-intuitive to use. This project aims to fix these issues by offering an easy to use interface with a simple but powerful set of tools(Regular Expression, NLP), using Obama's presidential speeches for validation, in order to achieve the desired objectives with high accuracy.

Acknowledgments

I am heartily thankful to Professor

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Introduction

In written text, each person has a unique style of writing similar to a fingerprint, and in order to identify that writing style linguists have devised methods and researched features that allows those styles to be differentiated and identified from one another.

1.1.1 Background

Stylometry is the statistical methods used to analyze and differentiate between different literary styles between one author and another. A stylometric analysis is be conducted using NLP techniques and libraries due to the large and extensive tool set and previous research and applications achieved.

1.1.2 Motivation

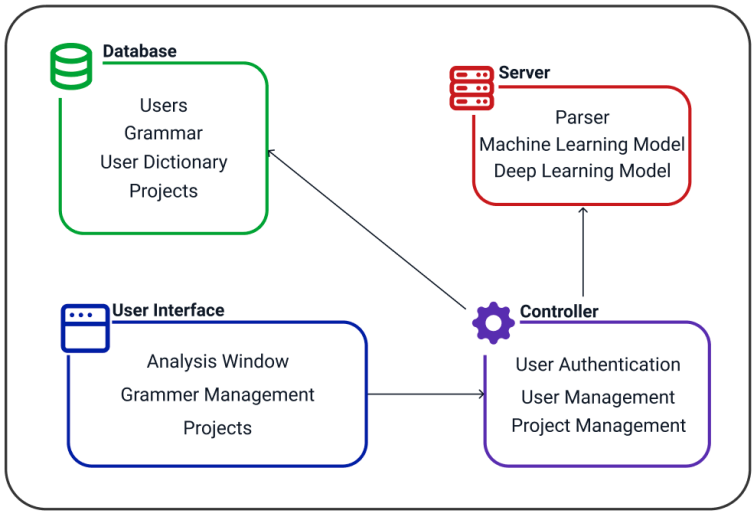
The process of identifying a writing's features is a difficult task that involves excessive manual labor and time analyzing each sentence and clause several times to extract all of the features.

1.1.3 Problem Definitions

The main problems are:

- Converting the stylometric features to machine readable form.
- Applying those features with NLP techniques for analysis.
- Providing a functional interface that allows for complete utilization of the system and its features.

1.2 Project Description



1.2.1 Scope

The scope of the project lies is limited to NLP, text analysis, and statistical operations.

1.2.2 Project Overview

- Document and project storage.
- Analysis server.
- User interface.

Chapter 2

Background

2.1 Project Context

The project is intended to be used by linguistic professionals to extract stylometric features and analyze written text to differentiate between authors.

- NLP: Natural Language Processing is a subfield of linguistics that is concerned with providing means for a machine to comprehend and analyze human language.
- Regular Expression: It is a simple tool used to find patterns in written text.
- Literary style: It is an approach to writing that is unique to each writer through the use of different vocabulary, grammar, techniques, etc..

2.2 Existing Solutions

2.3 General Guidelines

The purpose of the Background section is to provide the typical reader with information that they cannot be expected to know, but which they will need to know in order to fully understand and appreciate the rest of the report It should explain why the project is addressing the problem described in the report, indicate an awareness

of other work relevant to this problem and show clearly that the problem has not been solved by anyone else. This section may describe such things as:

- The wider context of the project;
- The problem that has been identified
- Likely stakeholders within the problem area
- Any theory associated with the problem area
- Any constraints on the approach to be adopted
- Existing solutions relevant to the problem area, and why these are unsuitable or Insufficient in this particular case Methods and tools that your solution may be based on or use to solve the problem and so on.

The wider context of the project includes such things as its non-computing aspects. So, for example, if you are producing software or any other products, including business recommendations, for a specific organization then you should describe aspects of that organizations business that are relevant to the project.

Chapter 3

Specification - (SRS)

3.1 General Guide Lines

3.2 Introduction

3.2.1 Purpose of this document

The purpose of this document is to define the requirements to be met by the system to be classified as successful and functional.

3.2.2 Scope of this document

The system is wholly engineered and developed by me, with the supervision of Dr. Wael Gomaa. The users are the professors of MSA University's Faculty of Languages.

3.2.3 Overview

The system will take a collection of texts, formatted and organized by author, to be analyzed by the system through a set of stylometric and linguistic features, and will return the extracted features embedded in the text with a statistical report of the analysis. The system will also require a selection of predefined grammar sets or new grammar sets defined by the user.

3.2.4 Business Context

The MSA University Faculty of Languages is the organization supporting and contributing to the project.

3.3 General Description

3.3.1 Product Functions

1. Create new project.
2. Upload texts.
3. Select from predefined grammar sets.
4. Create new grammar sets.
5. Extract stylistic and linguistic features.
6. Statistical analysis.
7. Highlight the extracted features.
8. Modify analysis results.
9. Login
10. Logout
11. Signup

3.3.2 User Characteristics

The users will be linguistic professionals that can verify the output of the system, and are able to evaluate and verify the results of the analysis.

3.3.3 User Problem Statement

The main problem the system is facing is the time consuming labor involved in extracting those linguistic features and their analysis.

3.3.4 User Objectives

- Identify features of the text.
- Modify the analysis results.
- Identify the disputed text's author.
- Create custom grammar for new features.

3.4 Functional Requirements

Table 3.1: Create New Project

Function Name	Create New Project
Description	The user will create a new project that will host the documents of the same context to be analyzed.
Critical	This requirement is the opening point of the system, so the system cannot work without it.
Technical issues	None.
Risks	None.
Dependencies with other requirements	None.
Precondition	Starting state.
Post-Condition	Awaiting documents to be uploaded.

Table 3.2: Upload Texts

Function Name	Upload Texts
Description	The user will upload a collection of texts with each group of texts put in a separate folder with the label of the folder being the name of the author, and the disputed text if available.
Critical	This requirement is the opening point of the project, so the system cannot work without it.
Technical issues	Detecting the user uploaded the texts in the proper format.
Risks	None.
Dependencies with other requirements	'Create New Project'
Precondition	A new project is created.
Post-Condition	Awaiting the features to be explored.

Table 3.3: Select from predefined grammar sets

Function Name	Select from predefined grammar sets
Description	The user will select a number of grammar sets that the text will be analyzed for.
Critical	System can't advance without it.
Technical issues	None.
Risks	None.
Dependencies with other requirements	None.
Precondition	Texts have been uploaded.
Post-Condition	Awaiting analysis results.

Table 3.4: Create new grammar sets

Function Name	Create new grammar sets
Description	The user will be presented with an interface that will allow them to define their own grammar to apply on the corpus. The generated grammar will also be saved to the user's account.
Critical	Optional.
Technical issues	Ensuring that the grammar is of proper and valid format.
Risks	None.
Dependencies with other requirements	None.
Precondition	Grammar sets have been selected.
Post-Condition	Awaiting analysis of the corpus.

Table 3.5: Extract stylistic and linguistic features

Function Name	Extract stylistic and linguistic features
Description	The system will receive the user's input and proceed to make the required analysis and return the result.
Critical	Critical, system can't function without it.
Technical issues	None.
Risks	None.
Dependencies with other requirements	'Create New Project', and 'Upload texts', 'Select from predefined grammar sets'
Precondition	Grammar is selected.
Post-Condition	Feature extraction is complete and returned to the user.

Table 3.6: Statistical analysis

Function Name	Statistical analysis.
Description	The system will apply statistical analysis to the extracted features received from the parser.
Critical	Critical, system can't function without it.
Technical issues	None.
Risks	None.
Dependencies with other requirements	'Extract stylistic and linguistic features'
Precondition	Features are extracted.
Post-Condition	Statistical analysis is applied and ready for modification.

Table 3.7: Highlight the extracted features

Function Name	Highlight the extracted features.
Description	The system will highlight the extracted features in the work space section of the project and offer methods of editing of the results.
Critical	Not critical.
Technical issues	None.
Risks	None.
Dependencies with other requirements	'Extract stylistic and linguistic features'
Precondition	Features are extracted.
Post-Condition	System awaiting features modification.

Table 3.8: Modify analysis results

Function Name	Modify analysis results.
Description	The user can hover over a word in the work space (highlighted or not) and edit its feature association.
Critical	Not critical.
Technical issues	None.
Risks	None.
Dependencies with other requirements	'Highlight extracted features'
Precondition	Features are highlighted in the work space.
Post-Condition	Modifications are applied to the work space and the statistical analysis.

Table 3.9: Login

Function Name	Login.
Description	User is logged in to system using their credentials.
Critical	Critical.
Technical issues	None.
Risks	None.
Dependencies with other requirements	None.
Precondition	User has no access to the system.
Post-Condition	User is greeted with the home screen.

Table 3.10: Logout

Function Name	Logout.
Description	User is logged out of the system and is greeted with the homepage.
Critical	Critical.
Technical issues	None.
Risks	None.
Dependencies with other requirements	None.
Precondition	User has access to the system.
Post-Condition	User is greeted with the Login screen.

Table 3.11: Sign-up

Function Name	Sign-up.
Description	User creates a new account using their credentials.
Critical	Critical.
Technical issues	None.
Risks	None.
Dependencies with other requirements	None.
Precondition	User has no account registered in the system.
Post-Condition	User account is registered and they are greeted with the system's homepage.

3.5 Interface Requirements

3.5.1 User Interfaces

3.5.1.1 GUI

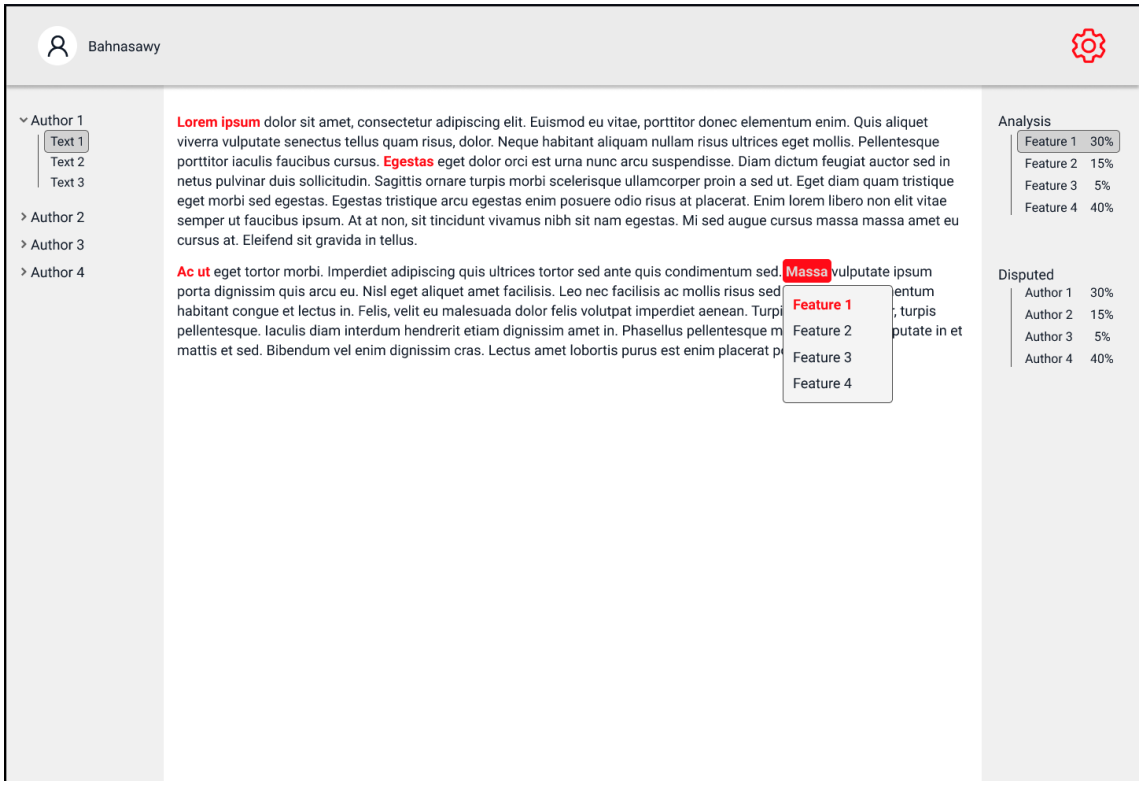


Figure 3.1: Screen 1

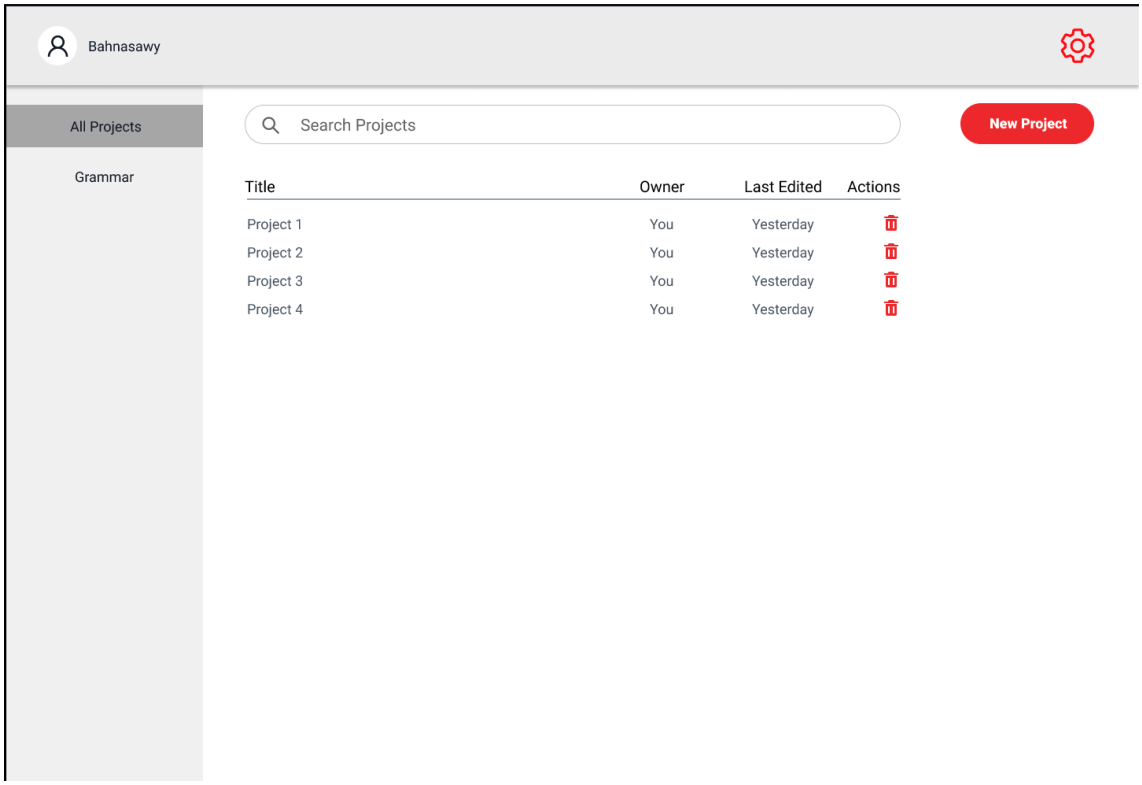


Figure 3.2: Screen 2

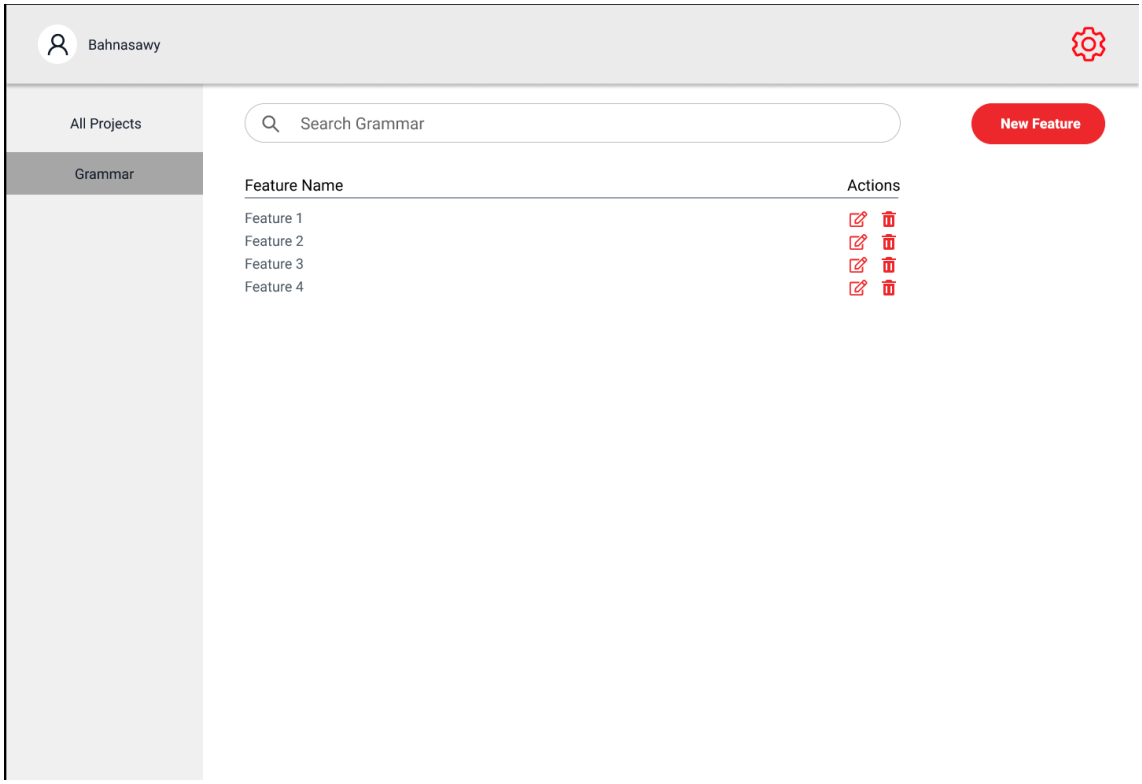


Figure 3.3: Screen 3

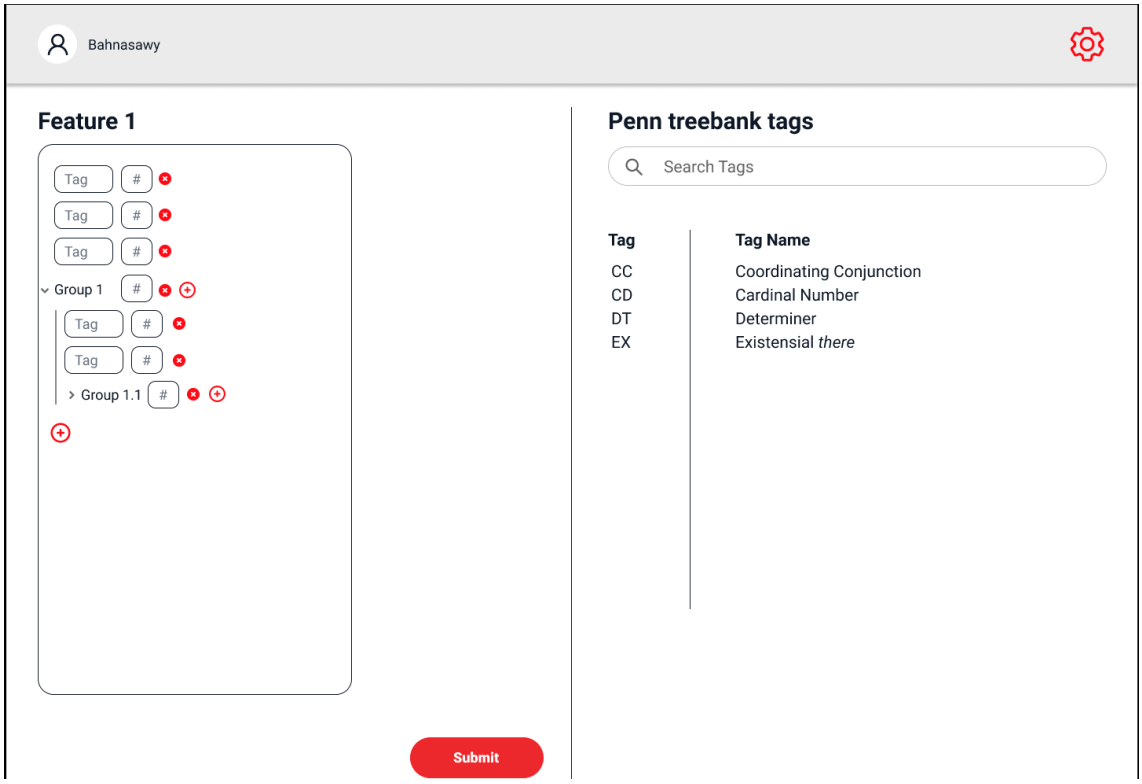


Figure 3.4: Screen 4

3.6 Design Constraints

No constraints provided.

3.7 Other non-functional attributes

3.7.1 Security

The system will use JWT (JSON Web Token) for user authentication.

3.7.2 Reliability

The system will be deployed as web services on AWS which offers a 99.99

3.7.3 Maintainability

The system will be utilizing the micro-services architecture ensuring smaller and more maintainable code.

3.7.4 Portability

The system will be web based, so it is absolutely portable.

3.7.5 Extensible

The system is using the micro-services architecture, so it is easily extensible.

3.7.6 Re-usability

The system is using the micro-service architecture, so any bit of code can be reused.

3.8 Operational Scenarios

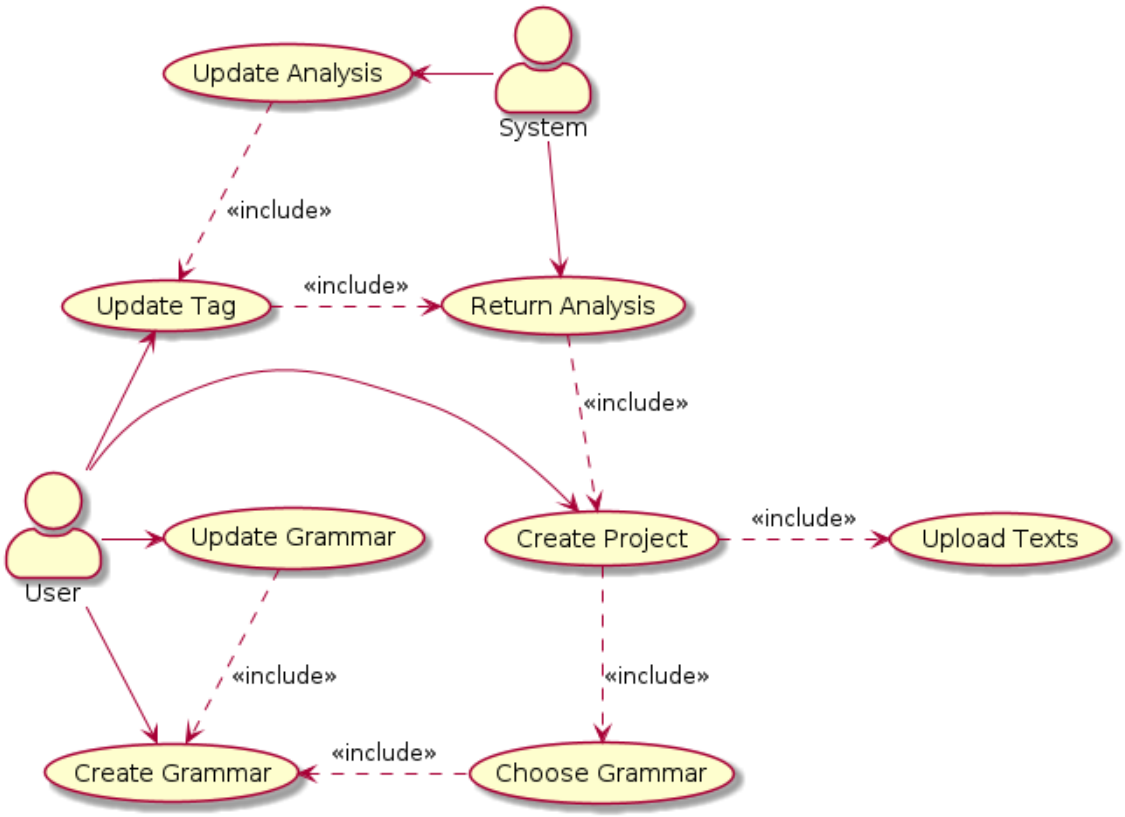


Figure 3.5: Use case Diagram

3.9 Preliminary Schedule Adjusted

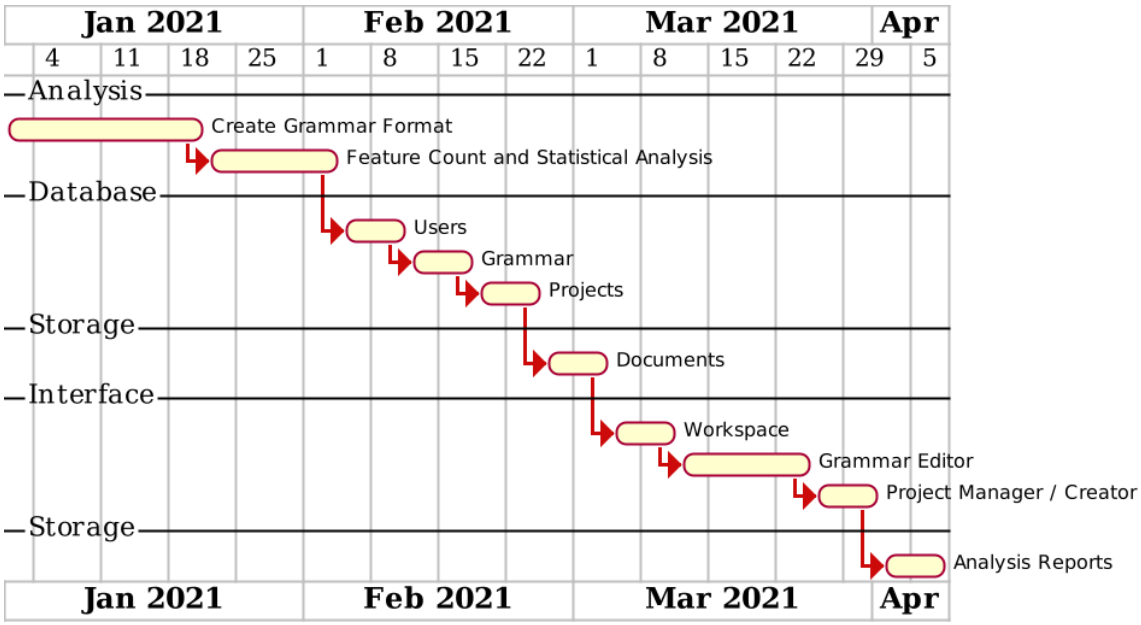


Figure 3.6: Gantt Chart

Chapter 4

Design

4.1 Introduction

4.1.1 Purpose

This software design document describes the architecture and system design of Forensics Linguistics and Authorship Attribution using Stylometry.

4.1.2 Scope

The project is intended to create a system for parsing and analyzing text for the extraction of linguistic and stylometric features to be used in authorship attribution performed by linguistic analysts.

4.1.2.1 Goals

- Reduce analysis time.
- Ease exploration of new features.
- Authorship attribution.

4.1.2.2 Objectives

- Intuitive interface.

- Bulk analysis.
- Features management.
- Analysis modification.
- Statistical and machine learning methods for authorship attribution.

4.1.3 Overview

The system is meant to be used by linguistic professionals to extract features from text and receive a preliminary analysis on the whole corpus' author.

4.2 System Overview

The system will receive the corpus from the user in the form of a compressed file, and the features required for extraction, which will then be processed for the required output and analyzed for a disputed text if necessary.

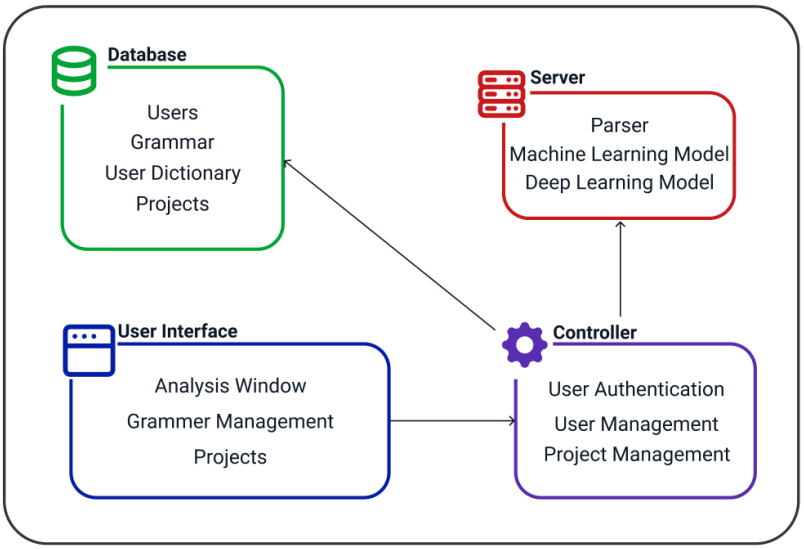


Figure 4.1: Architectural Design

4.3 System Architecture

4.3.1 Architectural Design

4.3.2 Decomposition Description

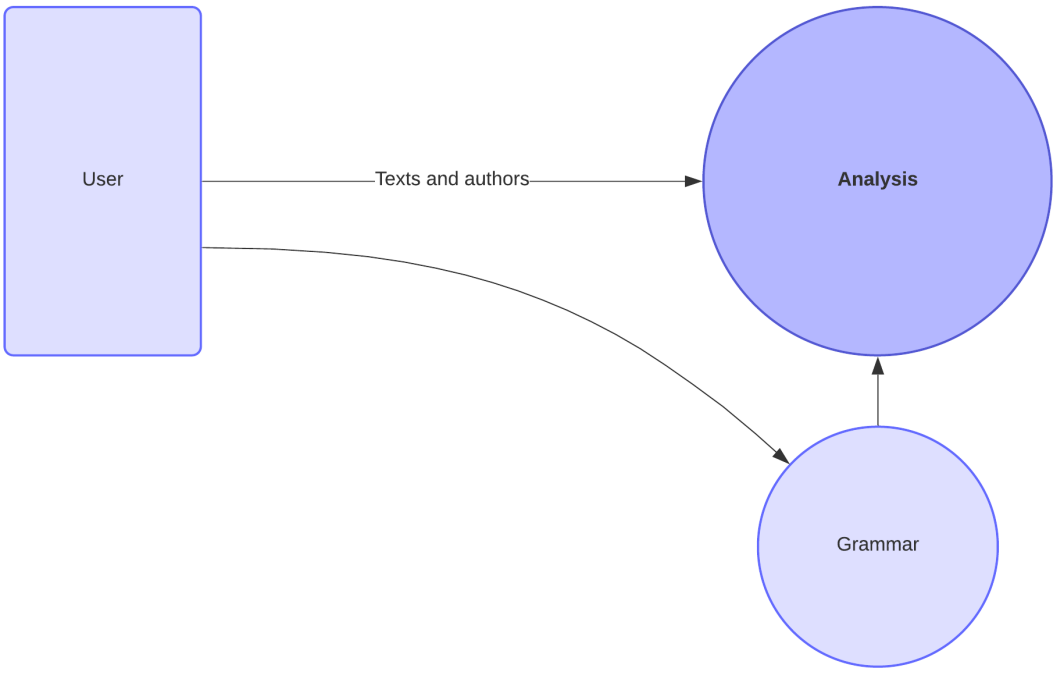


Figure 4.2: Data Flow Diagram

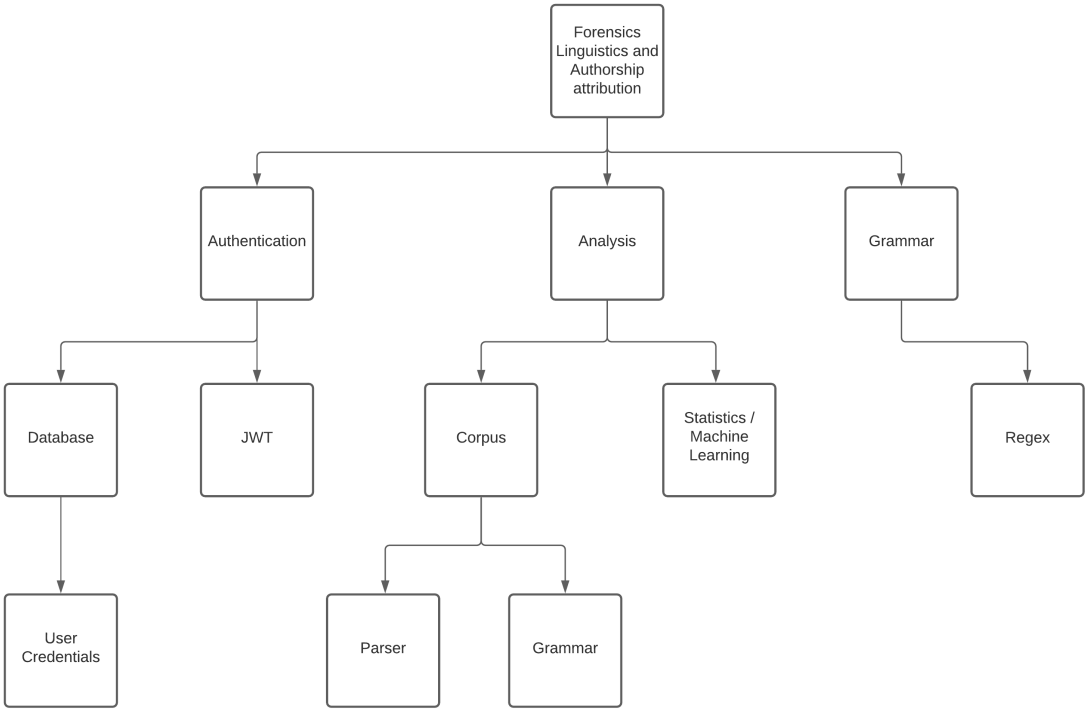


Figure 4.3: Functional Decomposition Diagram

4.3.3 Design Rationale

The principle behind the architecture used is separation of concerns, the main components of the system beside the user interface and the database are in the server and the controller. The server and the controller are two different entities and are deployed independently from each other. The reason for the separation is that they use different technologies with different architectures and different deployment needs. The server hosts the parser and the machine learning model of the system, which run on python and have a very specific task that may be modified but no extra features may be added. The controller on the other hand handles all the logistics of the system, by managing the users, grammar and projects by providing a graphQL API that is completely expandable.

4.4 Data Design

4.4.1 Data Description

1. Corpus
    - The corpus is uploaded as a compressed file containing folders of the texts, with the authors being the title of each folder. The file is extracted and turned into JSON format before parsing, then each element of the JSON object is parsed and returned as a parsed array that is saved in a noSQL database completely dedicated to storing the corpus and their analysis results.
2. Grammar
    - Grammar data is saved in a SQL database as a REGEX string that is imported during parsing of text, and editing of features. The grammar is created using a tool specifically made for this system.
3. Users
    - User data is saved in a SQL database. The user data includes their credentials, projects, and grammar.
4. Projects
    - Project data is split into multiple tables in a SQL database. The data includes collaborators, corpus id, and used features.

4.4.2 Data Dictionary

- createGrammar(featureName: string, regex: string)
- formatText(file: zip)
- parse(corpus: {[authorName]: string[][],
  disputed: string[][]}, grammar: string[])
- updateAnalysis(wordIdx: number, featureId: number)
- updateGrammar(featureId: number, regex: string)
- uploadText(file: zip)

4.5 Component Design

4.5.1 Authentication

```
username = input(username)
password = input(password)
userId = Select userId from database
              where username = username & password = password
if userId
    response.send(userId, username, JWT)
              // JWT Token is auto generated
```

4.5.2 Analysis

```
corpus = input(corpus)
grammar = input(grammar)
formattedCorpus = format(corpus)
parsedCorpus = parse(formattedCorpus, grammar)
response.send(parsedCorpus)
```

4.5.3 Grammar

```
featureName = input(featureName)
regex = input(regex)
res.send(Insert Into database (featureName, regex),
 Values (featureName, regex))
```

4.6 Human Interface Design

4.6.1 Overview of User Interface

- Creating new grammar: The user will open the grammar tab and press on the 'New Feature' button, they will then be redirected to a page where they are presented with a tool that allows them to write the grammar which will be converted into regex for the parser.
- Editing grammar: On the grammar tab, the user will be presented with a list of all the features they created, which they can either delete or edit. If the user

chooses to edit, they will be redirected to the grammar creation tool page, but with the existing feature characteristics already set for them to change.

- Creating a project: The user will be presented with a wizard involving a number of steps, they will first upload the corpus in a specified format which will be demonstrated on the wizard, then they will choose the needed features from the grammar they created, then all the data will be sent to the server and the user will be redirected to the workspace page where they can see the result of the analysis.
- Editing the analysis: The user can hover on any word in the workspace, set as a feature or not, and change its feature association as they please by selecting from the list of features they chose at the beginning of the project. The edit will also reflect on the statistical and the authorship attribution results accordingly.

4.6.2 Screen Images

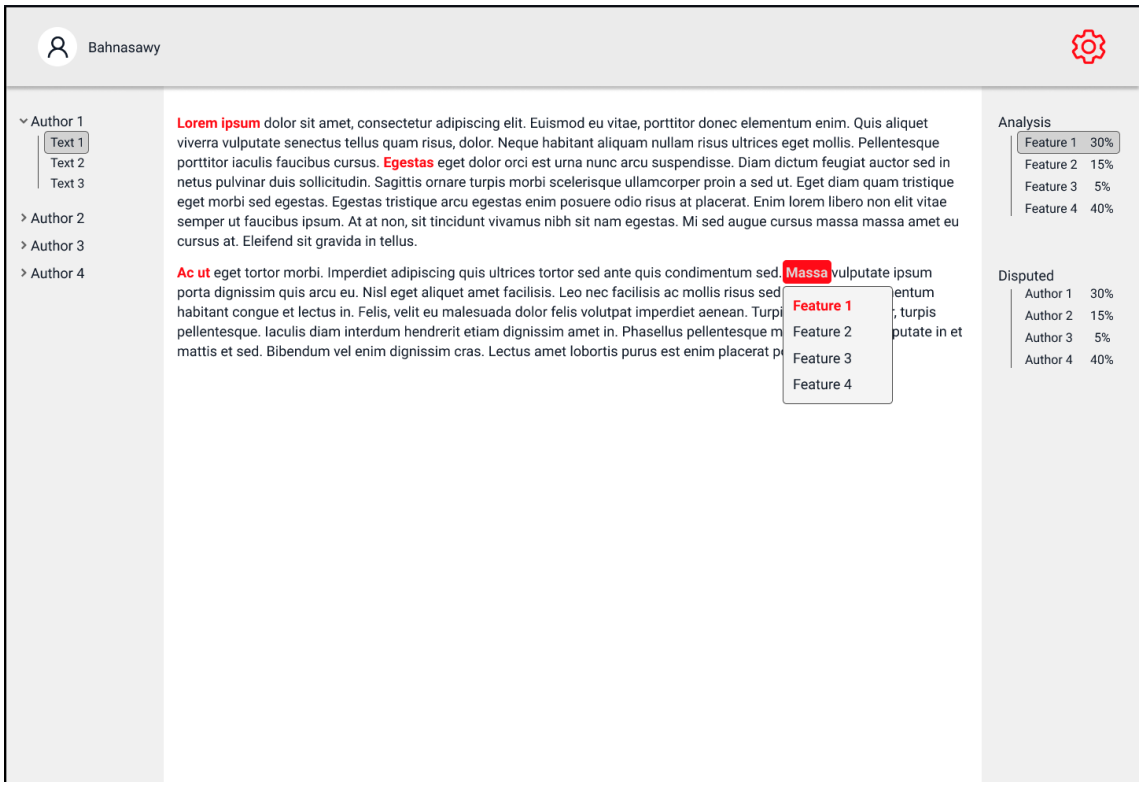


Figure 4.4: Screen 1

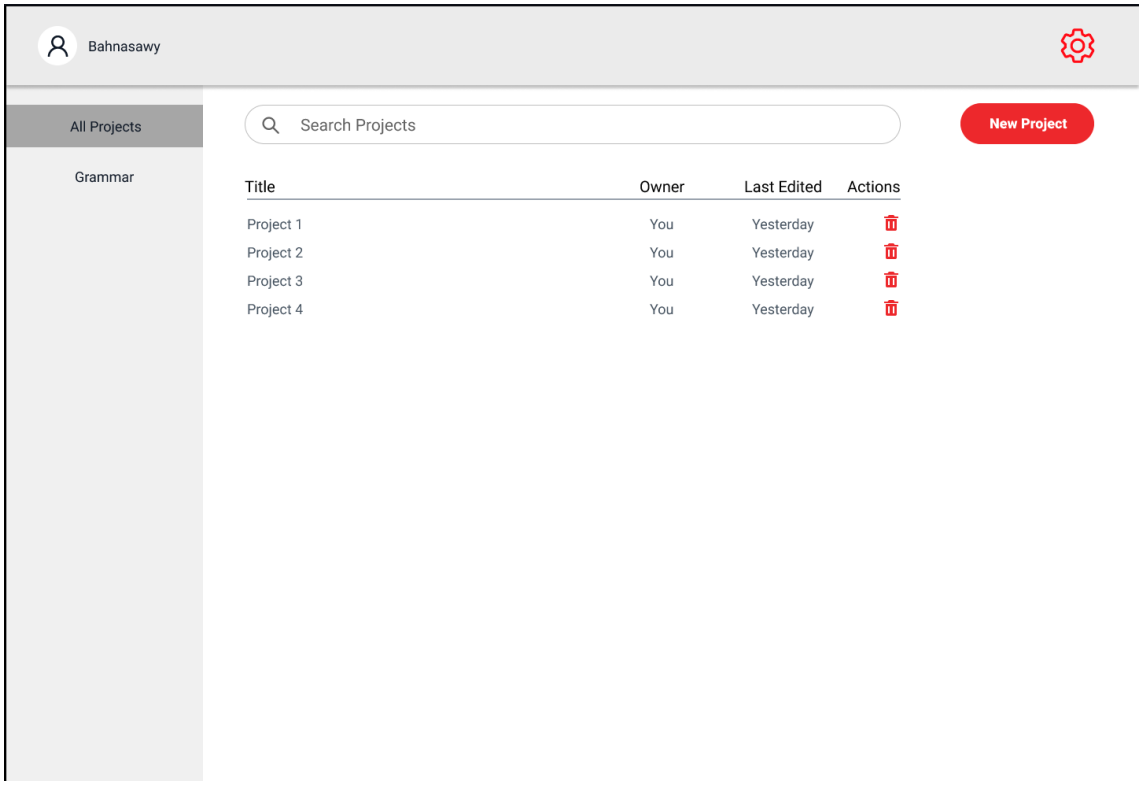


Figure 4.5: Screen 2

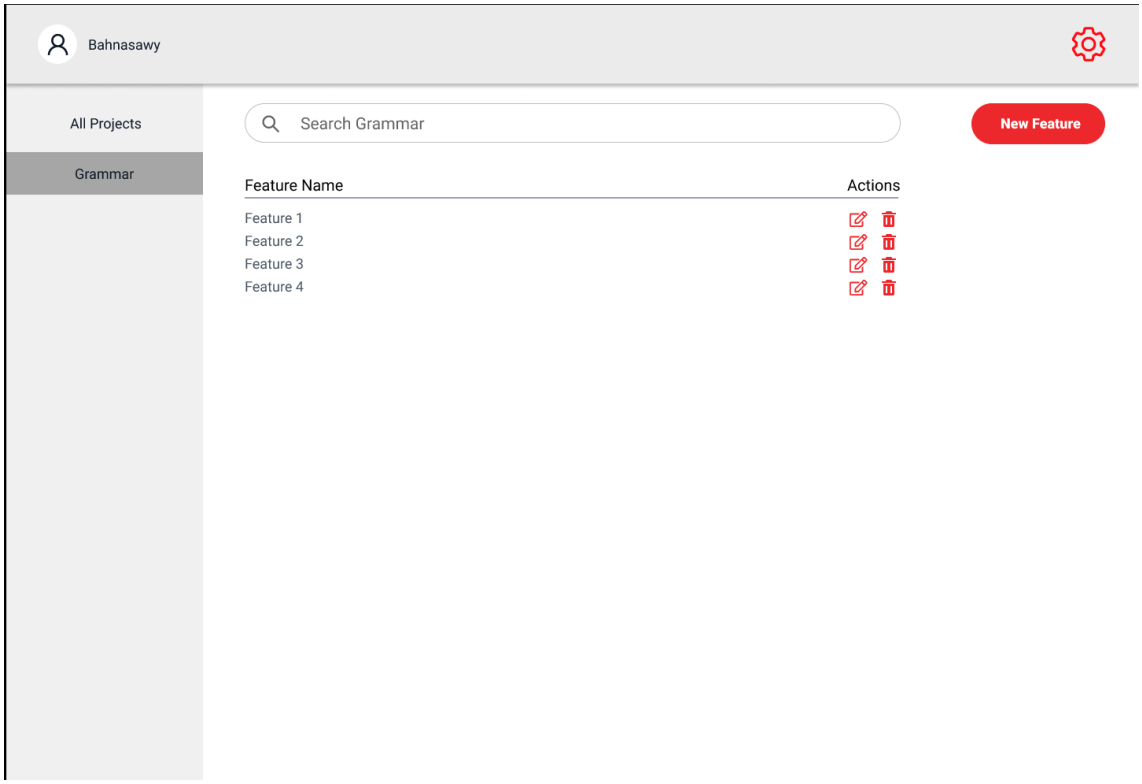


Figure 4.6: Screen 3

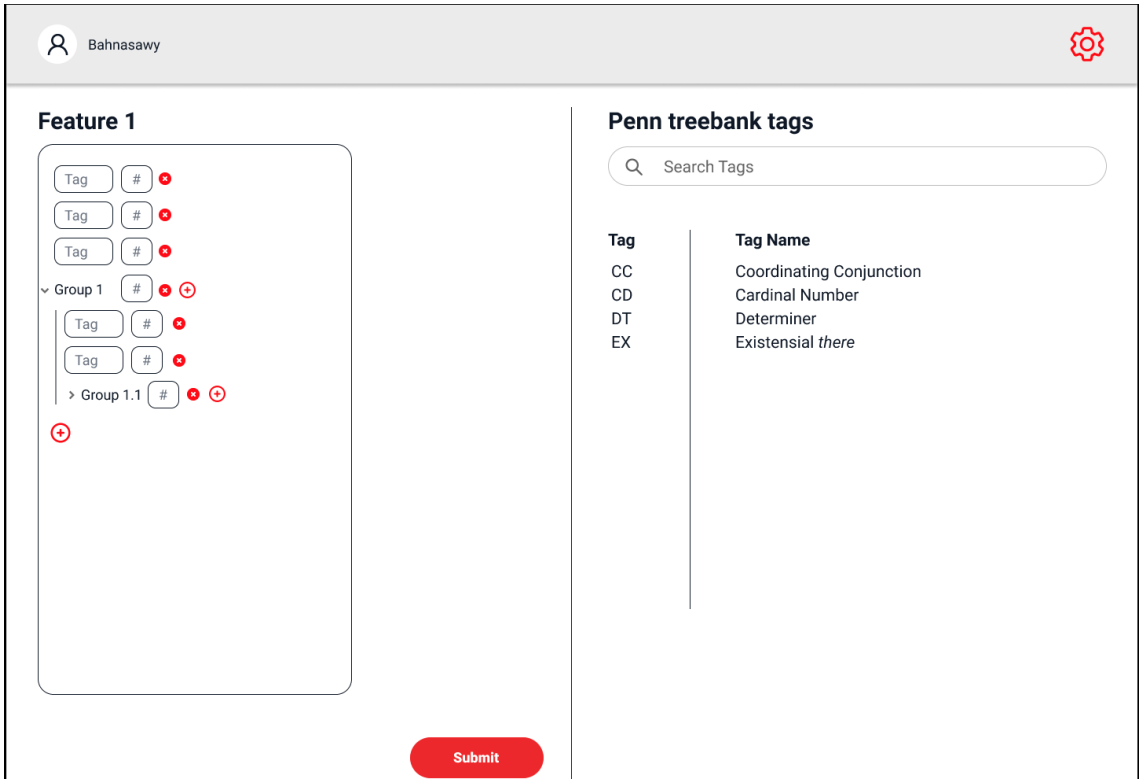


Figure 4.7: Screen 4

4.7 Requirements Matrix

System Component	Functional Requirement
Grammar	3, 4
Analysis	1, 2, 5, 6, 7, 8
Authentication	9, 10, 11

Chapter 5

Implementation

5.1 General Guidelines

Students can choose between

1. Illustrate your implementation parts
2. (Application oriented) "Deliverable" from Selected Approach
3. (Comparing Algorithm) Experiment Design

The Implementation section is similar to the Specification and Design section in that it describes the system, but it does so at a finer level of detail, down to the code level. This section is about the realization of the concepts and ideas developed earlier. It can also describe any problems that may have arisen during implementation and how you dealt with them. Do not attempt to describe all the code in the system, and do not include large pieces of code in this section. Complete source code should be provided separately. Instead pick out and describe just the pieces of code which, for example:

- Are especially critical to the operation of the system;
- You feel might be of particular interest to the reader for some reason;
- Illustrate a non-standard or innovative way of implementing an algorithm, data structure, etc..

- You should also mention any unforeseen problems you encountered when implementing the system and how and to what extent you overcame them. Common problems are:
- Difficulties involving existing software, because of, e.g., its complexity, lack of documentation; lack of suitable supporting software;

A seemingly disproportionate amount of project time can be taken up in dealing with such problems. The Implementation section gives you the opportunity to show where that time has gone

Chapter 6

Results and Evaluation

6.1 General Rules

In this section you should describe to what extent you achieved your goals. You should describe how you demonstrated that the system works as intended (or not, as the case may be). Include comprehensible summaries of the results of all critical tests that were carried out. You might not have had the time to carry out any full rigorous tests – you may not even got as far as producing a testable system. However, you should try to indicate how confident you are about whatever you have produced, and also suggest what tests would be required to gain further confidence. This is also the place to describe the reasoning behind the tests to evaluate your results, what tests to execute, what the results show and why to execute these tests. It may also contain a discussion of how you are designing your experiments to verify the hypothesis of a more scientifically oriented project. E.g., describe how you compare the performance of your algorithm to other algorithms to indicate better performance and why this is a sound approach. Then summarize the results of the tests or experiments.

You must also critically evaluate your results in the light of these tests, describing its strengths and weaknesses. Ideas for improving it can be carried over into the Future Work section. Remember: no project is perfect, and even a project that has failed to deliver what was intended can achieve a good pass mark, if it is clear that you have learned from the mistakes and difficulties. This section also gives you

an opportunity to present a critical appraisal of the project as a whole. This could include, for example, whether the methodology you have chosen and the programming language used were appropriate

Chapter 7

Conclusions and Future work

7.1 General Rules

The Conclusions section should be a summary of the aims of project and a re-statement of its main results, i.e. what has been learned and what it has achieved. An effective set of conclusions should not introduce new material. Instead it should briefly draw out, summarize, combine and reiterate the main points that have been made in the body of the project report and present opinions based on them. The Conclusions section marks the end of the project report proper. Be honest and objective in your conclusions.

7.2 Future Work

7.3 General Rules

It is quite likely that by the end of your project you will not have achieved all that you planned at the start; and in any case, your ideas will have grown during the course of the project beyond what you could hope to do within the available time. The Future Work section is for expressing your unrealized ideas. It is a way of recording that I have thought about this, and it is also a way of stating what you would like to have done if only you had not run out of time1. A good Future Work

section should provide a starting point for someone else to continue the work which you have begun.

Please be sure to put here any materials you have collected during your graduation project.

Bibliography