# Consistency Detection in Software Requirement Specifications Using Natural Language Processing Techniques

Author
MUHAMMAD ASIF HASAN
FALL 2016-MS-16(CSE) 00000172238

MS-16 (CSE)
Supervisor
Dr. WASI HAIDER BUTT

DEPARTMENT OF COMPUTER & SOFTWARE ENGINEERING
COLLEGE OF ELECTRICAL & MECHANICAL ENGINEERING
NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY
ISLAMABAD

SEPTEMBER, 2020

# Consistency Detection in Software Requirement Specifications Using Natural Language Processing Techniques

Author

MUHAMMAD ASIF HASAN

FALL 2016-MS-16(CSE) 00000172238

A thesis submitted in partial fulfillment of the requirements for the degree of

MS Computer Software Engineering

Thesis Supervisor:

Dr. WASI HAIDER BUTT

Thesis Supervisor's

Signature:__________________________________________

DEPARTMENT OF COMPUTER & SOFTWARE ENGINEERING

COLLEGE OF ELECTRICAL & MECHANICAL ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY, ISLAMABAD

SEPTEMBER, 2020

# Declaration

I certify that this research work titled "*Consistency Detection in Software Requirement Specifications Using Natural Language Processing Techniques"* is my own work under the supervision of Dr. Wasi Haider Butt. The work has not been presented elsewhere for assessment. The material that has been used from other sources it has been properly acknowledged / referred.

Signature of Student

MUHAMMAD ASIF HASAN

FALL 2016-MS-16(CSE) 00000172238

# Language Correctness Certificate

This thesis has been read by an English expert and is free of typing, syntax, semantic, grammatical and spelling mistakes. Thesis is also according to the format given by the university.

Signature of Student

MUHAMMAD ASIF HASAN

FALL 2016-MS-16(CSE) 00000172238

Signature of Supervisor

## Copyright Statement

## Acknowledgements

I am thankful to my Creator Allah Subhana-Watala to have guided me throughout this work at every step and for every new thought which You setup in my mind to improve it. Indeed I could have done nothing without Your priceless help and guidance. Whosoever helped me throughout the course of my thesis, whether my parents or any other individual was Your will, so indeed none be worthy of praise but You.

I am profusely thankful to my beloved parents who raised me when I was not capable of walking and continued to support me throughout in every department of my life.

I would also like to express special thanks to my supervisor Dr. Wasi Haider Butt for his help throughout my thesis and also for Software Requirement Engineering and Software Development for Web courses which he has taught me. I can safely say that I haven't learned any other engineering subject in such depth than the ones which he has taught.

I would also like to pay special thanks to Mr. Abdullah Khalil for his tremendous support and cooperation. Each time I got stuck in something, he came up with the solution. Without his help I wouldn’t have been able to complete my thesis. I appreciate his patience and guidance throughout the whole thesis.

I would also like to thank Guidance Committee Members Dr. Arslan Shaukat and Dr. Muhammad Usman Akram for being on my thesis guidance and evaluation committee. I am also thankful to Babar, Faisal and Farhan for their support and cooperation.

Finally, I would like to express my gratitude to all the individuals who have rendered valuable assistance to my study.

*Dedicated to my exceptional parents and adored siblings whose tremendous support and cooperation led me to this wonderful accomplishment*

# Abstract

Understanding of human language text in written form by a computer tool is possible because of Natural language processing (NLP). NLP utilizes the principles of artificial intelligence (AI). The creation of NLP soft wares is a difficult task because computers only understand in some programming language which is precise, unambiguous and highly structured. Natural language text is obviously not precise. It can also be ambiguous. The linguistic structure of the sentences can also vary a lot due to slang, regional dialects and social context. Contradictory and inconsistent sentences in a set of requirements is known as conflicting requirements. In the Requirements Engineering phase of Software Development Life Cycle (SDLC) software requirements are gathered, analyzed, negotiated back and forth manually to come to a final requirements specification document that is free from a known problem – conflicting requirements. By automating conflict detection during requirements analysis phase, time, effort, and resources can be saved in going back and forth and checking for conflicts manually. Natural Language Processing is a way to pre-process software requirements contextually before a manual or automated model or algorithm can be applied on them. A fully automated tool was developed in python language using NLTK toolkit utilizing multiple NLP techniques to facilitate software requirement engineer in verifying the software requirements for consistency issues and conflicts. This tool generates a traceability matrix to identify which possible requirements have consistency issues.

**Key Words:** *Conflict detection, Inconsistency detection, Automated requirements analysis, Natural Language Processing (NLP), Software requirements, Cosine similarity, Traceability Matrix, NLTK toolkit*

# Table of Contents

# List of Figures

# List of Tables

# CHAPTER 1: INTRODUCTION

This chapter offers a detailed introduction of the research. **Section 1.1** discusses the background study, **Section 1.2** discusses the Inconsistency and Conflict Detection in Software Requirements, **Section 1.3** discusses about Software Development Life Cycle (SDLC), **Section 1.4** discusses about NLP Techniques for Inconsistency and Conflict Detection, **Section 1.5** discusses about Consistency Issues and Conflicts in Requirements, **Section 1.6** discusses about Traceability Matrix, **Section 1.7** gives problem statement and proposed solution and **Section 1.8** contains thesis organization.

The process of examine client needs and designing, constructing, and testing end user applications is part of software engineering. The requirements of the user needs to be satisfied at the end. As computers understand programming language, a choice of programming language over which the application has to be developed has to be decided first. Software engineering is a practical application of engineering principles for software development.

## 1.1 Background Study

Requirement Engineering includes the tasks to define, document and maintain the user requirements. This also includes the processes of gathering and defining privileges that is to be provided by the system. The main activities included in Requirements Engineering Process are as follows:

- ❖ Requirements elicitation
- ❖ Requirements specification
- ❖ Requirements verification and validation
- ❖ Requirements management

### 1.1.1 Requirements Elicitation

The different ways used to gain information about the project aspects and requirements are covered in this phase. Domain knowledge can be acquired from customers, business manuals, the existing software of same type, standards and other stakeholders of the project.

Interviews, brainstorming, task analysis, Delphi technique, prototyping, etc. are some of the practices used for requirements elicitation phase. However, this all does not create any formal models of the specifications understood. But it extends the knowledge of the engineer and thus helps in offering input in the latter stages.

### 1.1.2 Requirements Specification

In this phase formal software requirement models are created. Models list all the requirements which includes functional and also the non-functional. Any constraints are also highlighted. During this phase more information might be required which can require further requirement elicitation processes. Models used can include ER diagrams, data flow diagrams (DFDs), function decomposition diagrams (FDDs), data dictionaries, etc.

### 1.1.3 Requirements Verification and Validation

This phase will ensure that the software is working and performing the right functions. This work is referred to as Verification. However, to ensure that the software is working according to the customer needs is part of Validation. If these check are not placed, bugs and errors in the later stages of software development life cycle will result a lot of efforts and rework. Some of the methodologies to help includes Reviews, buddy checks, making test cases, etc.

The main steps for this stage are as follows:

- ❖ The specifications shall be consistent with all the other specifications i.e. no two specifications shall conflict with each other
- ❖ The specifications shall be complete in every aspect
- ❖ The specifications shall be achievable

### 1.1.4 Requirements Management

This phase includes the tasks to analyze, document, track, prioritize and approve the requirements with multiple stakeholders of software project. This all is covered in Requirement Management. This all ensures the measure is taken for changing that might happen in requirements. Software requirement specification is targeted to be as flexible and modifiable as possible, so that any changes can be facilitated by end users in next stages of development. Being able to modify the application as per specifications in a organized and controlled manner is very important aspect of requirement engineering.

## 1.2 Inconsistency and Conflict Detection in Software Requirements

For a software project to be a success, it is ensured a set of complete, consistent and clear requirements are specified. In the requirements engineering process people from multiple backgrounds and varying needs are included. So it gets challenging to fully satisfy the requirements specified by all the stakeholders. Keeping in mind the project resources available and limitations, the requirements engineer trade-off stakeholders needs. One of the core reasons of an increase in cost and scheduled time which will also result in project failure, is because of not understanding and managing requirements. Requirement conflicts can be fatal.

Gathering requirements can be a long and arduous task in the Software Development Life Cycle (SDLC), and this task can result in errors that can be a cause of worry later on. Understanding the root cause of false requirements is a necessity to avoid adverse effect of eliciting and incorporating wrong requirements that can ultimately hinder the progress of an efficient software project. The analysis and detection of inconsistency issues and conflicts in the specifications step are one of the very important steps in requirements engineering.

## 1.3 Software Development Life Cycle (SDLC)

Software development life cycle (SDLC) is basically a framework. It defines activities conducted at each step of the software development process. Development team of a software organization follows the structure defined by SDLC. A detailed plan to develop, maintain and replace specific tool is described in SDLC. A methodology to enhance the quality of software and the overall development process is also covered in SDLC. Another name for this cycle is knows as software development process. Implementing the best practices and stages of this cycle will ultimately make the project work in a smooth, efficient and productive way.

❖ **Identify the current problems**

"What don't we want?" In this phase of cycle includes to get input from all stakeholders, including customers, salespeople, industry experts, and programmers. It also covers to take lessons from the strengths and weaknesses of the current tool with an enhancement as the goal.

- **Plan**

"What do we want?" In this phase of cycle includes a detailed specification of the specifications of the new application and determines the cost and resources required by the software team. The risks involved and possible sub-plans for softening those risks are also enlisted. At the end of this phase, a Software Requirement Specification document is created.

- **Design**

"How will we get what we want?" In this phase of cycle involves the creation of a design plan from the software specifications. This is also called the Design Specifications. This plan is then checked by all the stakeholders' feedback and suggestions are collected. To incorporate and collect the stakeholder input into this document, things become little difficult, however, this is very important. If this phase is not focused properly and is not a success, this will cause in cost overruns or a total failure of the software project at worst.

- **Build**

"Let's create what we want." In this phase of cycle the development of the tool is done by writing all the actual code. This is not a very complicated step, however, only if the earlier stages have been followed with focus to detail.

- **Test**

"Did we get what we want?" In this phase of cycle testing for defects and deficiencies are involved. Until the product meets the original specifications, correction of problems are done.

- **Deploy**

"Let's start using what we got." In this phase of cycle the task happens in a restricted way at first. Based on the input from users, more changes are made.

- **Maintain**

"Let's get this closer to what we want." In reality the planning never fully turn out to perfection. Updating and advancing the software to match the conditions that change in real world are accommodated as well.

The SDLC has been changed a lot because of the DevOps movement. More and more stages of the entire development processes are becoming the responsibility of Developers. When the use of the same toolset to check performance and highlight defects from inception are done by development and Ops teams, this helps to bring a common language and quicker handoffs between both teams. During development, QA, and production multiple APM tools can be utilized. This will ensure that the same toolset across the entire development lifecycle is utilized.

The popular SDLC examples and models are listed below:-

- ❖ **Waterfall Model**

The oldest and simplest model of SDLC is Waterfall Model. The steps and method involves finishing one phase before the next is started. A mini-plan is available for every phase and each phase “waterfalls” into the next. But a disadvantage of this methodology is that if little of details are incomplete, it can pause up the entire process.

- ❖ **Agile Model**

In Agile Model the process is divided into cycles and provides a working software very quickly. A successive releases are produced using this model. As each version is tested, a feedback input is incorporated into later versions. Robert Half has pointed out an important disadvantage of this model. He tells that this model focuses heavily on stakeholder’s interaction and this can ultimately lead the project in the fatal direction in many cases.

- ❖ **Iterative Model**

Repetition is greatly involved in this model. A version can be developed by developers very quickly. This also results is relatively small cost and to test and improve it through rapid and successive versions is quite simple. However a drawback of this model is that it can consume up resources quickly if left uncontrolled.

- ❖ **V-Shaped Model**

This model is an enhanced version of waterfall model. In this model each stage of development is tested. However a drawback is that, it can go into a complete roadblock.

- ❖ **Big Bang Model**

This model is for small projects. This model focusses on deploying most of the resources at the development. But a detailed requirements specification stages of the other methods is not focused in this method.

- ❖ **Spiral Model**

This model is very flexible. This is quite similar to iterative model as it also focusses on repetition. The planning, design, build and test phases are gone through over and over again with enhancements being done side by side.

## 1.4 NLP Techniques for Inconsistency and Conflict Detection

If requirement conflicts are not managed properly, according to multiple studies this results in project failure. This is because by cost and lack of time issues. To prevent re-iterations of all stages of project life-cycle, it is important to detect and fix conflicts in early stages. Requirement elicitation and analysis is one of the main step in the development of a product. Usually this involves a list of reasons that can hinder proper and efficient requirements elicitation, analysis and specification. One of the issues faced is conflict emergence between requirements. If the requirements are not processed and issues not found and corrected, the base of a product starts as hollow. For software requirements there have been methods and techniques introduced and adapted over the years to weed out the conflicts, inconsistencies, and ambiguities amongst other issues. One of the most commonly used technique is Natural Language Processing (NLP) on a set of requirements. NLP is done on any set of requirements using a number of pre-processing techniques.

The communication that happen between natural language and computers is based on Natural Language Processing (NLP) techniques. Despite this technology has been around for a good long time, but it has been greatly taken for granted. An example of NLP technology that public use almost daily includes Spelling Checks.

Following are some types of commonly used natural language processing techniques:-

- **Sentence Tokenization**

Dividing paragraphs of written language into its subsequent sentences is part of Sentence Tokenization. This is also called sentence segmentation. The methodology is quite simple. As we know that we can break apart the sentences in English or any other language based on punctuation marks.

- **Word Tokenization**

Dividing individual sentences of written language into its composed words is part of Word Tokenization. This is also called Word Segmentation. As we know space is a good indication of a word token in English or any other language which uses Latin alphabet.

But this can however still cause some issue of only by space splitting is done to achieve results. As we know there are some English compound nouns that are written with spaces. To cater this issue a library is deployed to achieve wanted results.

- **Text Lemmatization and Stemming**

The text can often contain multiple forms of a word such as drive, drives, driving due to grammar. Related words with a similar meaning are also found such as nation, national, nationality. Reducing the inflectional forms and derivationally related forms of a token to a common core form is part of both stemming and lemmatization. Both are special cases of normalization. But they are not the same thing.

A crude heuristic activity that chops off the ends of tokens to is part of stemming process. This is based on the removal of derivational affixes.

However, the utilization of a vocabulary and morphological analysis of words to do this task is part of lemmatization process. The aim is to get rid of inflectional endings only and to return the base or dictionary form of a token. This form of word is called as the lemma.

So it can be seen that a stemmer works without learning of the context, and therefore cannot understand the difference between tokens which have different meaning. The meaning or semantics depend on part of speech. But they do have some pros. They are quite easy to implement and they often are also fast. This do cause a reduced "accuracy" but that may not matter for some environments.

- **Stop words**

The words which are removed out before or after processing of the content are included in Stop words. As they cause a lot of noise if input into machine learning, so they are omitted out of text. Commonly found words such as "and", "the", "a" in a natural language are included in stop words. However there is no one universal list of stop words which can be used as a standard. So they change according to the environment and application.

Stop words comprise of the most common words such as "and", "the", "a" in a language, but there is no even a once standard universal list of stop words. The list of the stop words can change depending on your software. The NLTK toolkit has a library of stop words that are also based on commonly found stop words.

- **Bag-of-words**

Machine learning algorithms require quantitative representation. A raw text cannot be directly processed. So it is required to transform text in to vectors of numbers and this process is called feature extraction. A simple feature extraction methodology used when we work with text utilizes the bag-of-words model. This is quite popular model. The occurrence of each word within a text is described in this model.

To use this model, we need to:

1. Design a vocabulary of known words (also called tokens)
2. Choose a measure of the presence of known words

The details about the order or structure of words is removed. This is the reason it is called a bag of words. In this model the learning is based on whether a known word occurs in a document. However, it doesn’t focus on where is that word in the document. As similar documents have similar contents, this is the motivation of this model. Furthermore from the contents, a learning of something about the meaning of the document can also be attained.

Machine learning algorithms to text and speech as applied using NLP techniques. A leading platform for building Python programs is NLTK (Natural Language Toolkit). These programs work with human language information. Breaking a paragraph of written language into its component sentences is part of Sentence tokenization. The issue of dividing a sentence of written language into its component words is part of Word tokenization. Both stemming and lemmatization bring down inflectional forms or derivationally related forms of these tokens in to core common token. The words which are removed out before or after processing of text are included in stop word list. A simple feature extraction technique is employed when we work with text utilizes bag-of-words methodology. This model details about the occurrence of each word within a content. A statistical measure can also be used based on TF-IDF. This analyzes the importance of a word to a document in a collection or corpus.

A popular tool for creating Python programs to work with natural language data is NLTK (Natural Language Toolkit). This toolkit has simple interfaces to many corpora and lexical resources. It has modules for text processing libraries that are used for classification, tokenization, stemming, tagging, parsing, and semantic reasoning. This is a free and open source toolkit. This is basically a community-driven project. This toolkit provides an environment where Python tools that work with natural language information can be build using the principles of Natural Language Processing techniques.

Natural Language Toolkit also includes graphical demonstrations and testing data sets. It also has a cook book. This book explains the fundamentals behind the language processing tasks that NLTK deploys. This was authentically written by Steven Bird, Edward Loper and Ewan Klein for utilization in development and education. Areas of computational linguistics as well as programming principles for Python are also guided in it. This makes it ideal for linguists who have no detailed concepts in programming. Also for engineers and researchers that require to go into computational linguistics it is very helpful. And for of course the students and educators utilize them for learning and educational purposes. 50 corpora and lexical sources are included in this toolkit. This includes the Penn Treebank Corpus, Open Multilingual WordNet, Problem Report Corpus, and Lin's Dependency Thesaurus.

## 1.5 Consistency Issues and Conflicts in Requirements

A cross check of requirements is done for consistency checking. It is made sure that no requirements should be contradictory to each other. So N X N operations are required to be performed, where N states for number of requirements. There are three types of inconsistencies which are mentioned as below:-

- **Terminology clash:** this refers to the same concept named differently in different requirements e.g. library management: "borrower" vs. "patron"
- **Designation clash:** this refers to the same name for different concepts in different requirements e.g. "user" for "library user" vs. "library software user"
- **Structure clash:** this refers to the same concept structured differently in different requirements e.g. "latest return date" as time point (e.g. Fri 5pm) vs. time interval (e.g. Friday)

A strong conflict is one in which requirements are not justifiable together i.e. logically inconsistent: S, **not** S. For example "participant constraints may not be disclosed to anyone else" vs. "the meeting initiator should know participant constraints".

A weak conflict is one in which requirements are not justifiable together under some boundary condition, i.e. strongly conflicting if B holds: potential conflict. This is much more frequent in requirement engineering. For example "patrons shall return borrowed copies within X weeks" vs "patrons shall keep borrowed copies as long as needed".

## 1.6 Traceability Matrix

During software development a table which is called a traceability matrix is used. It assists to determine the connectivity of a relationship by correlating any two baselined requirements using a many-to-many relationship comparison. With the high-level requirements this is quite often used. It is also utilized with detailed requirements of the application to the matching parts of high-level design, detailed design, test plan, and test cases.

To check if the current project requirements are being met, a traceability matrix is a great help. It also helps in the development of a request for proposal, software requirements specification, various deliverable documents, and project plan activities. An identifier is used for every requirement of a SRS and placed in the left column. In the top row identifiers for the other requirements are placed. A mark is placed in the intersecting cell when a requirement in the left column is related to a requirement across the top. For each row and each column, the quantity of relationships are added up. This figure will give an idea of the mapping of the two items. If a value is null, it indicates that no relationship exists. The goal is to determine if a relationship happens. If the relationship is too complex, value will be large and hence, it should be simplified.

It is a good practice to add the relationships to the requirements for both backward traceability and forward traceability. This eases down the creation of traceability matrices. Traceability matrix can also be used in the form of requirements vs test cases. During the development of a new software, this is a very important tool. This will also ensure transparency and completeness of the product testing.

Testing matrix shows a great help when it comes to convince the stakeholders that all the needs have been met and there are no loopholes in the product at the time of software delivery and installation. It often involves columns such as requirement, document reference number, bug ID and test case. The columns involved makes things quite manageable as the tracking of any requirement is made by the user using defect ID.

Some of the usual entities that are involved in software testing matrix are requirement ID, risks involved, requirement type and description, unit test cases, integration test cases, user acceptance test cases and trace to test script. For the testing team, traceability matrix is very advantageous.

Some of these are as mentioned below:

- ❖ In each stage of the SDLC, the matrix helps the developer team to be sure about the inclusion of all the user needs
- ❖ This matrix also confirms that all the specifications have been traced in to the test cases

- ❖ This matrix also facilitates the user that the software is developed according to the specification conveyed by them
- ❖ Any missing functionalities are also quickly and easily identified

## 1.7 Problem Statement and Proposed Solution

Due to the growing complexity of products, it has become vital to find out issues in software requirements at an early stage. Detecting conflicts has become difficult in parallel. Moreover, hidden conflicts can cause issues in all the leading steps in a products development lifecycle. To avoid these problems, conflicts must be found at the requirements elicitation phase. Without a proper method to evaluate requirements or a set of rules to identify general categories of conflicts in requirements, the problem statement becomes moot.

The requirement engineer is supposed to find all the conflicts and inconsistency issues in software requirements in order to avoid bugs and errors at the later stages of software development life cycle. A bug and error at the later stage to project life cycle is going to increase the cost of software project exponentially. But doing this job as much as it is essential is far more complicated and cumbersome. Imagine if there is a software requirement specification document having 5 hundred requirements in it. The requirement engineer is supposed to check 124500 (NxN / 2 - 500) possible cases of checks. This a humanly very difficult task to do but it is done throughout the world and heavy project budgets being paid to requirement engineer teams to ensure a perfect job.

If the use to computing power can be utilized to help requirement engineers, this can greatly reduce project cost and most importantly time. This will also result in an efficient way of working and will require a smaller requirement engineer team. Hence, in order to detect inconsistency issues and conflicts in software requirements, the author and his supervisor decided to develop a fully automated tool, in python language, that will be using Natural Language Processing techniques and semantic similarity algorithms to facilitate the requirement engineer in this difficult task of software development life cycle.

## 1.8 Thesis Organization

Organization of the thesis is as follows:-

**Chapter 1: Introduction** offers a brief introduction containing the Background Study, Inconsistency and Conflict Detection In Software Requirements, Software Development Life

Cycle (SDLC), NLP Techniques for Inconsistency and Conflict Detection, Consistency Issues and Conflicts in Requirements, Traceability Matrix, Problem Statement and Proposed Methodology and Thesis Organization.

**Chapter 2: Literature Review** provides the detailed literature review highlighting the work done in the domain of conflict detection in software requirements. The systematic literature review is composed of three main sections. First section is review protocol which gives details on the methodology using which the literature review is carried out. Section two offers details on research works in this area. Whereas, section three highlights the research gaps that were encountered.

**Chapter 3: Proposed Methodology** covers the details of proposed methodology used for identification of problem.

**Chapter 4: Implementation** presents the detailed implementation regarding the proposed tool and all the modules that were created for it.

**Chapter 5: Validation** provides the validation performed for our proposed methodology using three important case studies. The three case studies selected for validation purposes are of different domains and different sizes to make sure that our proposed approach works on every case.

**Chapter 6: Discussion And Limitation** contains a brief discussion on the work done and also contains the limitations to our research.

**Chapter 7: Conclusion And Future Work** concludes the research and recommends future works for the research.

# CHAPTER 2: LITERATURE REVIEW

This chapter presents the literature review carried out for the research. **Section 2.1** discusses the review protocol, **Section 2.2** presents the results obtained from the review protocol and **Section 2.3** highlights the research gaps which form the foundation of our research.

The phase to define the expectations of the users for the software that is to be created or modified is covered in Requirements Analysis. All the steps that are done to identify the needs of different stakeholders are involved in this phase. To analyze, document, validate and manage application or system requirements is the entire requirement analysis process.

## 2.1 Review Protocol

A systematic literature review (SLR) is type of research literature review which will identify, select and critically appraise research in order to answer a clearly formulated questions. This review implements an explicitly defined protocol or plan, where the criteria is clearly stated before the review is started. A comprehensive and transparent search is conducted over different research repositories. This will include a plan for the strategy which will focus to answer the research questions lied down. The type of information searched, critiqued and reported within a specific timeframes is identified by this review process. This review includes the search terms, search strategies (including database names, platforms, dates of search) and limits.

Review protocol development for this study was based on already defined Systematic Literature Review by Kitchenham. This review protocol demonstrates the category definition, criteria of selection and rejection, assessment of quality, extraction of data and the mechanism used for data synthesis. The details of these elements are given in following subsections.

### 2.1.1. Categories Definition

Three categories to simplify the data extraction and synthesis process were defined. The details of each of these categories is given as below.

- ❖ **Natural Language Processing Techniques:**

  This category include the use of Natural Language Processing (NLP) techniques in detection of conflicts in software requirements irrespective of a specific industry. This includes all studies that make the use of one or more techniques of NLP on a set of requirements to focus on conflict detection.

- **Software Requirements:**

  This category includes software requirements and the issues found in requirements during elicitation, analysis or specifications phase of requirements in the development lifecycle of a product.
- **Conflict Detection:**

  This category deals with all the studies that focused on conflict detection in software requirements. This includes all the software requirements based studies that focused on finding conflicts. Conflict detection alone in software specific requirements were not quite enough, so any set of requirement for any product that is focused on conflict detection was included to start with a database of studies.

### 2.1.2. Research Questions

Research questions for which this Systematic Literature Review was focused to find answers for, are as follows:-

- What are significant researches reported from 2009-2019 where conflict detection in software requirements is focused?
- What are significant researches reported from 2009-2019 where conflict detection in software requirements using Natural Language Processing Techniques is focused?
- What are significant models/algorithms proposed or used by researchers to aid conflict detection in software requirements during 2009-2019?
- What are significant tools proposed or used by researchers to aid conflict detection in software requirements during 2009-2019?
- Is there any fully automated tool available in researches for conflict detection using Natural Language Processing techniques during 2009-2019?

To answer the above questions a Systematic Literature Review was performed. 30 research papers were selected from 4 scientific libraries ELSEVIER, IEEE, SPRINGER and ACM in the year bracket of 2009-2019.

### 2.1.3. Selection and Rejection Criteria

In order to achieve precision in literature selection, six constraints are defined. On basis of following constraints, research papers will either be selected or rejected. The studies that do not comply with and do not fulfill these six parameters are not considered. (1)

Nominated Research article will be selected only if it is relevant to our research perspective. Only those papers were selected that dealt with conflict detection in requirements and Natural Language Processing. Further selection was done on the basis of the responses of the research questions that were asked for. Furthermore, the unrelated research studies that did not include both conflict detection and NLP in them were rejected. (2) Research work published between 2009 and 2019 will be selected only. It was ensured that the collection of the latest studies were included by opting for those studies which lie in the years 2009 to 2019, and by not considering those researches that lie outside of our selected time range. (3) Primarily four popular scientific databases were used, which are ELSEVIER, IEEE, SPRINGER, and ACM; to ensure the selection of authentic and state of the art research works we opted for those articles which have been brought forward by the specified publishers. (4) Result-oriented studies were to be opted that are model/algorithm oriented. (5) Redundancy in research studies was to be evaluated and only most outstanding one of them were to be used. (6) Selected researches that proposed algorithms or used existing models for conflict detection were to be selected.

**Table 2-1:** Details of research works per publisher

| Sr.# | Database | Type | Selected Research | No. of Researches |
|---|---|---|---|---|
| **1** | **ELSEVIER** | **Conference**<br>**Journal** | **[6, 9]**<br>**[31,34]** | **6** |
| **2** | **IEEE** | **Conference**<br>**Journal** | **[10-18]**<br>**[37-39]** | **12** |
| **3** | **SPRINGER** | **Conference**<br>**Journal** | <br>**[20]** | **1** |
| **4** | **ACM** | **Conference**<br>**Journal** | **[21-28]**<br>**[41-43]** | **11** |

### 2.1.4. Search Process

According to the selection and rejection criteria “2009–2019” year-filter on all the search terms to get the searches put out during 2009–2019, merely. Four prime databases of publication (i.e. Springer, IEEE, Elsevier and ACM) were used to perform the systematic literature review process. “AND” operator was used to accomplish the possible investigation outcomes necessary for this study. Some of the search terms included (e.g. Conflict detection, software requirements, inconsistency detection, automated requirement analysis, requirement analysis, and NLP). The search flow process is as mentioned below:-

- ❖ **Identification:** Multiple search expressions were specified in four scientific databases and got about 2347 results.
- ❖ **Screening:** About 1561 studies were excluded in the screening process because their title of research did not comply with our criteria.
- ❖ **Eligibility:** About 693 researches were excluded by accessing and reading their abstracts because they did not match with our selection and rejection criteria.
- ❖ **Detailed Study:** A thorough qualitative and quantitative study of 93 researches was done by extracting their data and synthesizing it later for our research questions. After detailed examination of 93 papers we rejected 63 studies which did not fulfill our merit of quantitative and qualitative criteria.
- ❖ **Selected Researches:** Finally remaining 30 papers were included because they comply with our set criteria for selection and rejection.

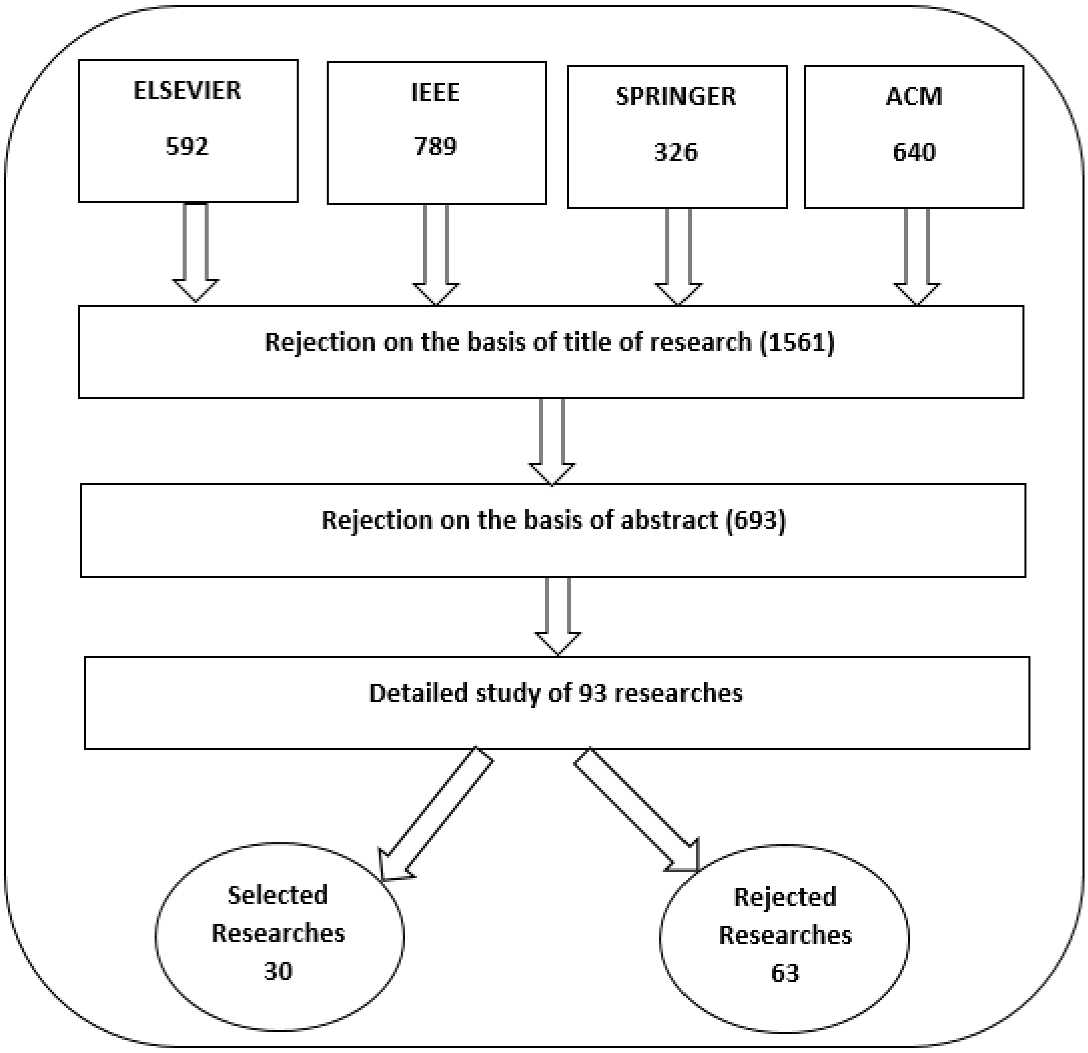

ELSEVIER
592
IEEE
789
SPRINGER
326
ACM
640
Rejection on the basis of title of research (1561)
Rejection on the basis of abstract (693)
Detailed study of 93 researches
Selected Researches 30
Rejected Researches 63

**Figure 2.1:** Search Process

### 2.1.5. Quality Assessment

Quality Assessment of research articles is very important because the result of this research depends on the crucial results of nominated research articles. Quality assessment principles were developed to ensure that high quality research is selected for this study. Journal papers were also selected as they are more authentic than conference proceedings. Only latest research articles were selected in the year limit from 2009-2019. Most reliable scientific databases to achieve quality results were selected which includes IEEE and Springer. The data evaluation of the researches is checked whether it is free from the ambiguous statements and relies on the solid facts and theoretical discerning. Uniqueness of the work is another critical feature. Therefore, only those research studies were included that are published in at least one of the following four well-known and internationally acknowledged scientific databases.

### 2.1.6. Data Extraction and Synthesis

Data extraction and synthesis is performed by comprehensive study of selected researches. **Table 2.2** shows the data extraction and synthesis performed for our chosen researches to find the answers of our research questions.

**Table 2-2:** Data Extraction and Synthesis

| Sr.# | Description | Details |
|---|---|---|
| **1** | **Bibliographic information** | **Author, Title, Publication Year, Publisher, Type of Research (Journal/Conference)** |
| | **Data Extraction** | |
| **2** | **Overview** | **Main objective of the selected paper** |
| **3** | **Results** | **Results acquired from the selected paper** |
| **4** | **Data Collection** | **Qualitative and quantitative method used** |
| **5** | **Assumptions** | **To validate the outcome** |
| **6** | **Validation** | **Manual and Automated testing comparison** |
| | **Data Synthesis** | |
| **7** | **Model/Algorithm selection** | **Models and Algorithms used for conflict detection** |
| **8** | **Tool Selection** | **Tools used for conflict detection** |

## 2.2. Research and Analysis

The literature from 30 research papers that were utilized for this work have been summarized in the **Table 2-3** and **Table 2-4**. This has been done to acknowledge the relevant research work.

**Table 2-3:** Conflict Detection Models and Algorithms

| Sr.# | Conflict Detection Models | Total References | References Identification |
|---|---|---|---|
| **1** | **AOP**<br><br>**UML Models, OMG Models (MDA), Theme/Doc Approach, KAOS** | **4** | **[21, 27, 31, 34]** |
| **2** | **Ontology-based**<br><br>**(OWL, Generalized Upper Model, Domain Ontologies, Ontology of Uncertainty)** | **4** | **[6, 14, 16, 33]** |
| **3** | **Machine Learning**<br><br>**(Regression Linear Model, Multi-Sentence Modelling of Requirements, K-means)** | **3** | **[15, 25, 32]** |
| **4** | **Data Models**<br><br>**(Semantic Data Models, Verb-centric General Semantic Model)** | **2** | **[16, 28]** |
| **5** | **Algebraic Models** | **1** | **[23]** |
| **6** | **Self-Proposed** | **5** | **[9, 13, 25, 26, 28]** |

**Table 2-4:** Conflict Detection Tools

| Sr.# | Conflict Detection Tools | Reference Identification |
|---|---|---|
| **1** | **ReVerb** | **[6]** |
| **2** | **SEMIOS** | **[15]** |
| **3** | **ReqWiki** | **[27]** |
| **4** | **cTAKES** | **[23]** |
| **5** | **Text Based Retrieval System** | **[9]** |
| **6** | **SAT-Analyzer** | **[18]** |
| **7** | **Drools Expert** | **[25]** |
| **8** | **Algebric Grammar Graph (AGG)** | **[21]** |
| **9** | **General Architecture for Text Engineering (GATE)** | **[25]** |

## 2.3. Gap Analysis

This section discusses the research gaps and limitations encountered in previous literature. Analysis was done on around 30 researches, after an extensive screening process to look for researches that provide automated conflict detection for software requirements. The studies also show that how few have focused on the use of automated tools altogether.

The gap found in our selected studies was that there's no research that focused on using Natural Language Processing techniques to detect conflicts and inconsistency issues in software requirements. There was no single fully automated tool available to facilitate requirement engineer to do this cumbersome job. Though partially implemented or semi-automated tools were available but that cannot help a requirement engineer in real life unless he/she gets workable tool right in his/her computer machine installed. Also there were models and algorithms available in literature but they are useful for research purposes only and cannot help a requirement engineer either.

Hence, after a detailed meeting between the author and his supervisor, it was decided that this applied research gap shall be addressed. The goal was decided to use Natural Language Processing techniques along with semantic similarity algorithms to find solution to this gap. If the goal was achieved, a fully automated that is capable to produce a Traceability Matrix for the entire set of requirements shall be implemented in some high level language that is popular in this era.

# CHAPTER 3: PROPOSED METHODOLOGY

This chapter contains details of the proposed methodology. **Section 3.1** discusses the targeted problem and **Section 3.2** provides detailed proposed solution.

Requirements elicitation is a step in the Software Development Lifecycle (SDLC). In fact, it is the first phase of the software development lifecycle through which quality software can be developed according to the customer's need and handed over to them in the given time. Detecting inconsistency issues at this stage of SDLC is very important because if these issues are not addressed at this stage the cost to fix errors at the later stages of SDLC will be much greater.

## 3.1 Targeted Problem

The process of the production of application with the highest quality and lowest cost in the shortest time is part of the Software Development Life Cycle (SDLC). The detailed plan for how to develop, alter, maintain, and replace an application system is included in this plan.

Multiple steps are involved in this cycle which include plan, design, build, test and deployment. Waterfall model, spiral model, and Agile model are some the renowned models of SDLC.

The beauty of this cycle is that it lowers the cost of software development while side by side improve the quality and shorten the production time. These goals achievable because this cycle follows an organized plan that removes the typical pitfalls to software development projects. Firstly, the evaluation of existing application for deficiencies is conducted. Next, the specifications requirements of the new system are lied down. The application through the stages of design, development, testing, and deployment is then created. SLDC can possibly remove the redundant rework and after-the-fact fixes, by projecting costly mistakes like failing to ask the end client for feedbacks.

If this cycle is followed corrected, maximum level of management control and documentation is possible. The developer team know what they are building are why. All of the stakeholders and software development teams agree on the common goal up front and can visualize a clear plan.

SDLC implementation can move into a roadblock to development if there are pitfalls that are left unchecked. If the needs of customers and all users and stakeholders are not taken into proper consideration, this can cause a poor understanding of the project requirements at the outset. Benefits and advantages of SDLC can only be achieved if the plan is followed heart fully and faithfully.

The basic purpose of requirements elicitation is to extract requirements of every scope from the client and then to process those given requirements into a well-developed requirements specification document which is then passed off to the next step and becomes the basis of a baseline upon which all further phases are completed and the eventual software product developed. Usually, this phase requires the longest time to complete because of the requirement engineer's going back and forth with the client's or the requirement team from the client's end to sort out all requirements before they can be closed off and passed on to the next phase as a final requirements specification document. This is done to weed out problems that may arise due to human error such as ambiguity in meaning, conflict and inconsistency in needs, or incompatible priorities of multiples stakeholders. In this research the focus was on the inconsistent and conflicting requirements during the requirements analysis step in the requirements engineering phase.

In any product development, a concise and true requirements document is vital before the development phases can commence. In requirements engineering, one of the steps is the consistency checking and conflict detection of requirements to solve consistency and contradictory requirements issue which can later on impact the development of a software product. The quality of the requirements phase effects the overall quality of the subsequent phases and hence, the software product. Having a good software requirements specification (SRS) document is essential to a good final product.

To determine the needs or conditions to meet the new or altered software or project is focused in Requirements analysis phase. Conflicting specifications of the multiple stakeholders, and to analyze, document, validate and manage the software or system specifications is taken into consideration. For the success or failure of an application or software project, requirements analysis is very important part of it. The specifications should be documented, actionable, measurable, testable and traceable as well. They should be specified to a level of detail sufficient for application design.

## 3.2 Proposed Solution

An automated algorithm based on Natural Language Processing (NLP) techniques was proposed. NLP techniques include Parts of Speech (PoS) tagging, word tokenization, stemming, and stop word removal amongst many others. NLP techniques can be used by importing Natural Language Processing Toolkit (NLTK), which is the most generally used toolkit available and can be used on various platforms.

This work starts off with inputting a series of lines together in a joint thread of text, separated by full-stops. In order to make each requirement in to lower case, the first of the NLP technique that is used is text lowercase. Once this have been achieved, a lowercase version of requirements, then another NLP technique to split each requirement into individual sentences is applied. This technique is called sentence tokenization.

Once this have been achieved, individual list of requirements, a verification check in this tool has been put in place to determine the number of requirements successfully detected by software. This is displayed to user so that he/she can verify that software has correctly detected all the requirements or not.

Afterwards, another NLP technique to convert all the sentence tokens in to word tokens is used. This technique is called word tokenization. Once all requirements are in the form of word tokens, all the stop words present in the requirements were removed. To do this, another NLP technique which is called stop word removal was utilized.

Once the word tokens are free from stop words, common punctuation marks like full stops and commas from the tokens were removed. Once these were removed, another NLP technique to do stemming of the word tokens was utilized. This technique is called word stemming.

Afterwards, part of speech tags to each word token were allocated. To do this, another NLP technique called part of speech tagging was utilized. Now it was needed to create a training set called dictionary or vocabulary. So an algorithm was developed to produce this artificial intelligence in this tool.

Once this dictionary was created, all the requirements were converted into numerical vectors. Then these vectors were checked for cosine similarity to achieve similarity scores. Once all the scores were available they were organized into three types of traceability matrices.

The first traceability matrix shows result with score values. The second one shows in the form of PC and NC, representing possible conflict and no conflict respectively. The last one shows in the form of PC and --, representing possible conflict and blacks (no conflicts) respectively.

# CHAPTER 4: IMPLEMENTATION

In this chapter the use of NLP techniques along with cosine similarity to ultimately develop a fully automated tool for conflict detection has been shown. **Section 4.1** shows the development and detailed algorithms of modules used in this tool.

To develop this tool, the first decision that had to be taken was to choose which programming language should be opted for coding. It was decided to use Python language. The reason of choosing this language was because it has a very good compatibility with the NLP Toolkit which was going to be integrated into this tool. The version of Python language that was chosen was v3.7.4 which was at that time the latest version available to download.

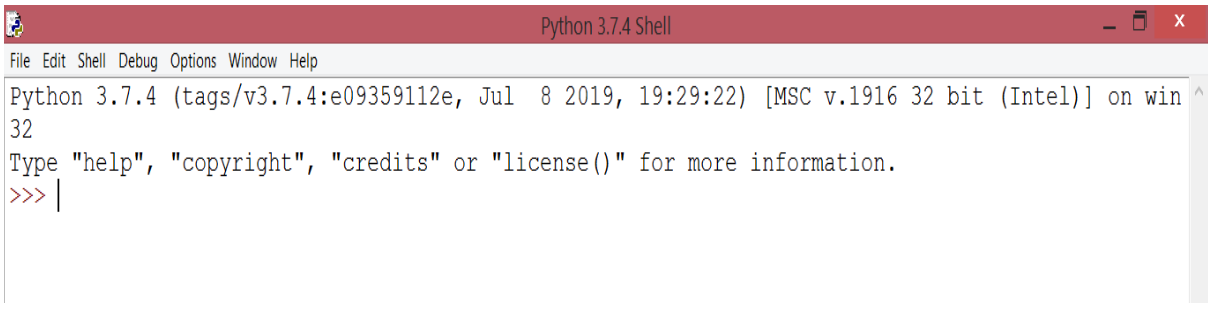
Once the programming language was all setup and ready to be utilized, the NLP Toolkit was downloaded that was intended to integrate into this tool. This NLP Toolkit was NLTK v3.4.5 which was the latest version available to be downloaded at that time.

Once NLP Toolkit was setup and ready to be utilized, a couple of Python Libraries which will be required to be used in this tool were further installed. These libraries included gensim v3.8.1, numpy v1.17.2, jieba v0.39, scipy v1.3.1 and prettytable v0.7.2.

Finally the software environment that will be required to develop this tool were all setup and ready to use. Windows 8.1 Professional was used over a laptop computer which was available.

## 4.1 Development

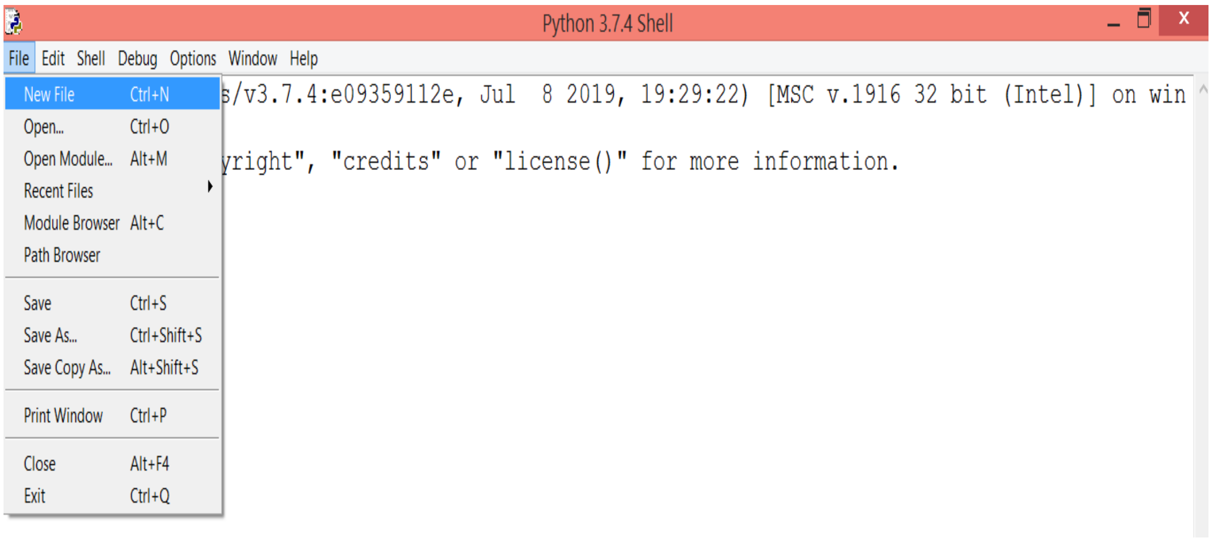
Once the Python is opened to start developing a software, below is an example snapshot of the Python Shell environment which will be noticed first.



```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 19:29:22) [MSC v.1916 32 bit (Intel)] on win
32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 
```

**Figure 4.1:** Python 3.7.4 Shell

Then a new file was created to write the code for this tool by doing as below.

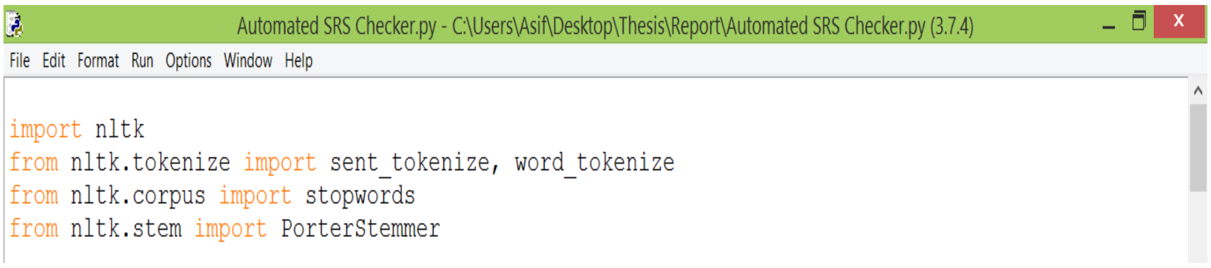


```
s/v3.7.4:e09359112e, Jul  8 2019, 19:29:22) [MSC v.1916 32 bit (Intel)] on win

yright", "credits" or "license()" for more information.
```

**Figure 4.2:** Creating New Editor File

Once the editor was launched, the writing of code started. It was needed to import a couple of modules which would be required, by writing the following lines of code.

| **Python Algorithm** |
|---|
| import nltk<br>from nltk.tokenize import sent_tokenize, word_tokenize<br>from nltk.corpus import stopwords<br>from nltk.stem import PorterStemmer |


Automated SRS Checker.py - C:\Users\Asif\Desktop\Thesis\Report\Automated SRS Checker.py (3.7.4)
File Edit Format Run Options Window Help

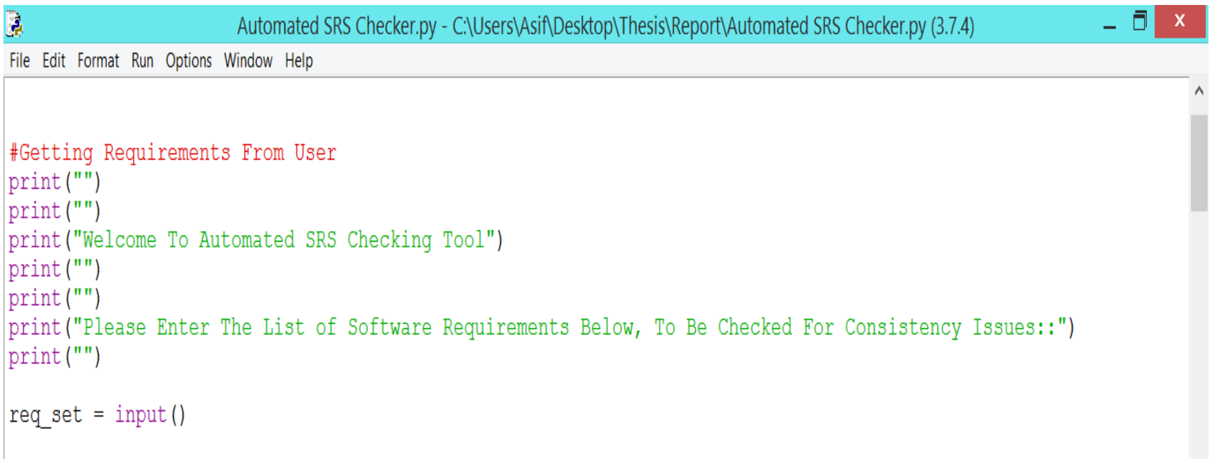
```
import nltk
from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
```

**Figure 4.3:** Importing Required Modules

The code for welcome screen and getting SRS requirements from user was written as below.

**Python Algorithm**

```
#Getting Requirements From User
print("")
print("")
print("Welcome To Automated Consistency Checking Tool")
print("")
print("")
print("Please Enter The List of Software Requirements Below, To Be Checked For Automated Consistency Checking::")
print("")

req_set = input()
```

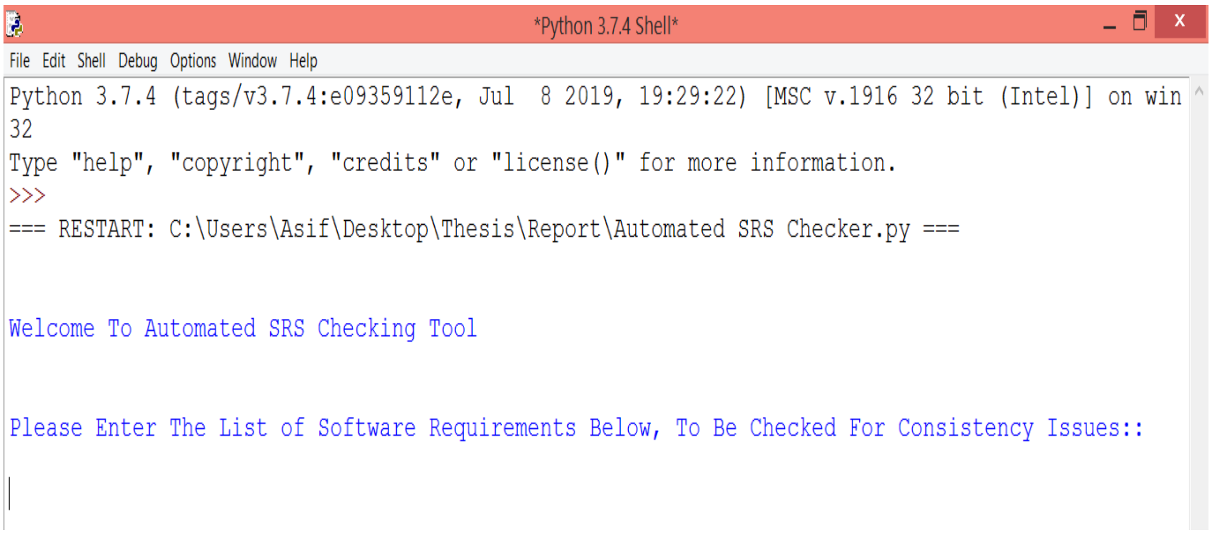

Automated SRS Checker.py - C:\Users\Asif\Desktop\Thesis\Report\Automated SRS Checker.py (3.7.4)
File Edit Format Run Options Window Help

```
#Getting Requirements From User
print("")
print("")
print("Welcome To Automated SRS Checking Tool")
print("")
print("")
print("Please Enter The List of Software Requirements Below, To Be Checked For Consistency Issues::")
print("")

req_set = input()
```

**Figure 4.4:** Code for welcome screen and input from user

**Output**

```
Welcome To Automated Consistency Checking Tool



Please Enter The List of Software Requirements Below, To Be Checked For Automated Consistency Checking::
```

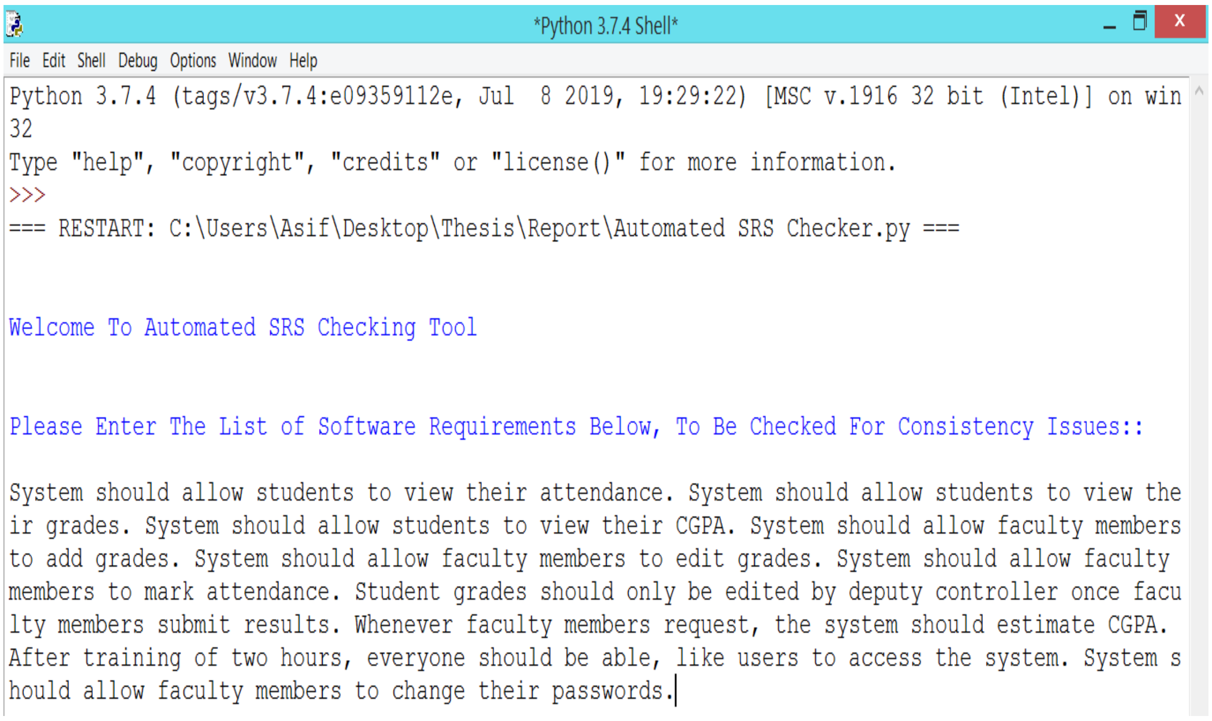


```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 19:29:22) [MSC v.1916 32 bit (Intel)] on win
32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
=== RESTART: C:\Users\Asif\Desktop\Thesis\Report\Automated SRS Checker.py ===


Welcome To Automated SRS Checking Tool


Please Enter The List of Software Requirements Below, To Be Checked For Consistency Issues::

```

**Figure 4.5:** Output on shell

Now for example, lets input a couple of SRS requirements in to this tool.

| **Requirements** |
|---|
| System should allow students to view their attendance. System should allow students to view their grades. System should allow students to view their CGPA. System should allow faculty members to add grades. System should allow faculty members to edit grades. System should allow faculty members to mark attendance. Student grades should only be edited by deputy controller once faculty members submit results. Whenever faculty members request, the system should estimate CGPA. After training of two hours, everyone should be able, like users to access the system. System should allow faculty members to change their passwords. |

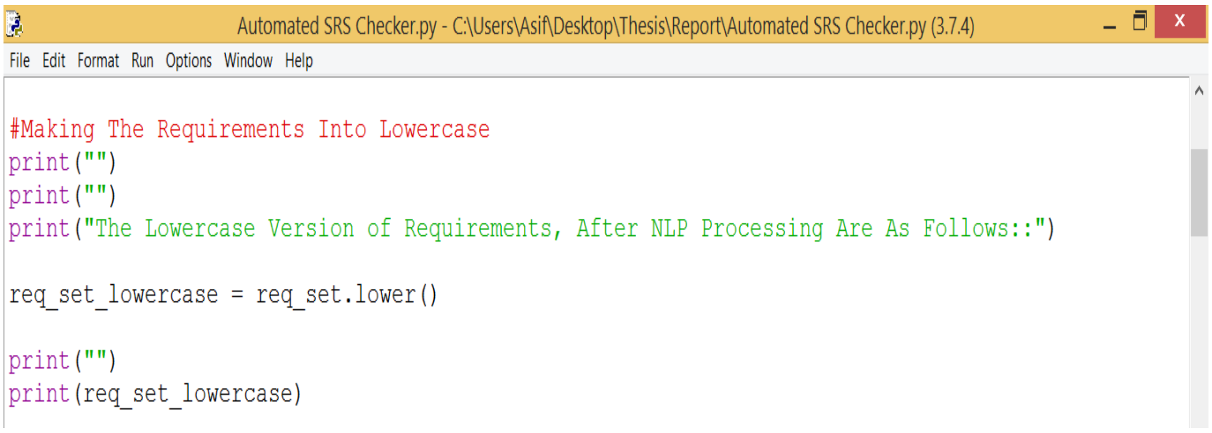


```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 19:29:22) [MSC v.1916 32 bit (Intel)] on win
32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 
=== RESTART: C:\Users\Asif\Desktop\Thesis\Report\Automated SRS Checker.py ===


Welcome To Automated SRS Checking Tool


Please Enter The List of Software Requirements Below, To Be Checked For Consistency Issues::

System should allow students to view their attendance. System should allow students to view the
ir grades. System should allow students to view their CGPA. System should allow faculty members
to add grades. System should allow faculty members to edit grades. System should allow faculty
members to mark attendance. Student grades should only be edited by deputy controller once facu
lty members submit results. Whenever faculty members request, the system should estimate CGPA.
After training of two hours, everyone should be able, like users to access the system. System s
hould allow faculty members to change their passwords.
```

**Figure 4.6:** Example SRS requirements

The code to apply NLP technique to convert all the user requirements in to lower case was written as below.

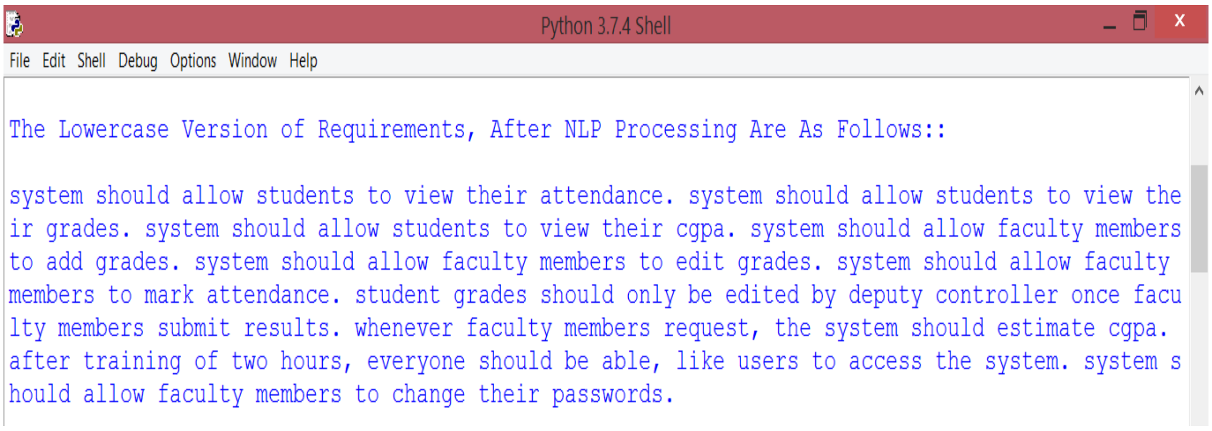
| Python Algorithm |
|---|
| #Making The Requirements Into Lowercase<br>print("")<br>print("")<br>print("The Lowercase Version of Requirements, After NLP Processing Are As Follows::")<br><br>req_set_lowercase = req_set.lower()<br><br>print("")<br>print(req_set_lowercase) |


Automated SRS Checker.py - C:\Users\Asif\Desktop\Thesis\Report\Automated SRS Checker.py (3.7.4)
File Edit Format Run Options Window Help

```
#Making The Requirements Into Lowercase
print("")
print("")
print("The Lowercase Version of Requirements, After NLP Processing Are As Follows::")

req_set_lowercase = req_set.lower()

print("")
print(req_set_lowercase)
```

**Figure 4.7:** Code for NLP lower case technique

| **Output** |
|---|
| The Lowercase Version of Requirements, After NLP Processing Are As Follows::<br><br>system should allow students to view their attendance. system should allow students to view their grades. system should allow students to view their cgpa. system should allow faculty members to add grades. system should allow faculty members to edit grades. system should allow faculty members to mark attendance. student grades should only be edited by deputy controller once faculty members submit results. whenever faculty members request, the system should estimate cgpa. after training of two hours, everyone should be able, like users to access the system. system should allow faculty members to change their passwords. |



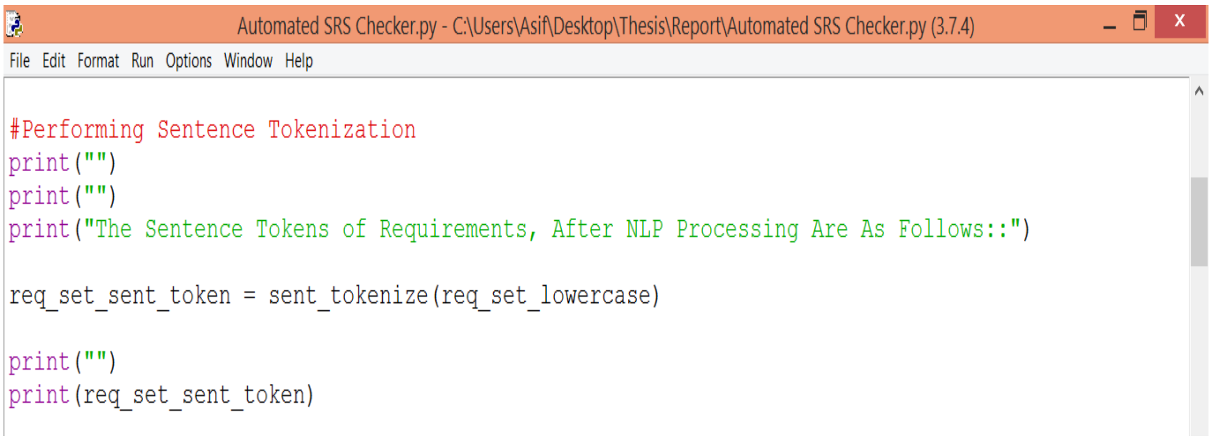
```
The Lowercase Version of Requirements, After NLP Processing Are As Follows::

system should allow students to view their attendance. system should allow students to view the
ir grades. system should allow students to view their cgpa. system should allow faculty members
to add grades. system should allow faculty members to edit grades. system should allow faculty
members to mark attendance. student grades should only be edited by deputy controller once facu
lty members submit results. whenever faculty members request, the system should estimate cgpa.
after training of two hours, everyone should be able, like users to access the system. system s
hould allow faculty members to change their passwords.
```

**Figure 4.8:** Output on shell

The code to apply NLP technique to sentence tokenize all the user requirements was written as below.

| **Python Algorithm** |
|---|
| #Performing Sentence Tokenization |

```
print("")
print("")
print("The Sentence Tokens of Requirements, After NLP Processing Are As Follows::")

req_set_sent_token = sent_tokenize(req_set_lowercase)

print("")
print(req_set_sent_token)
```



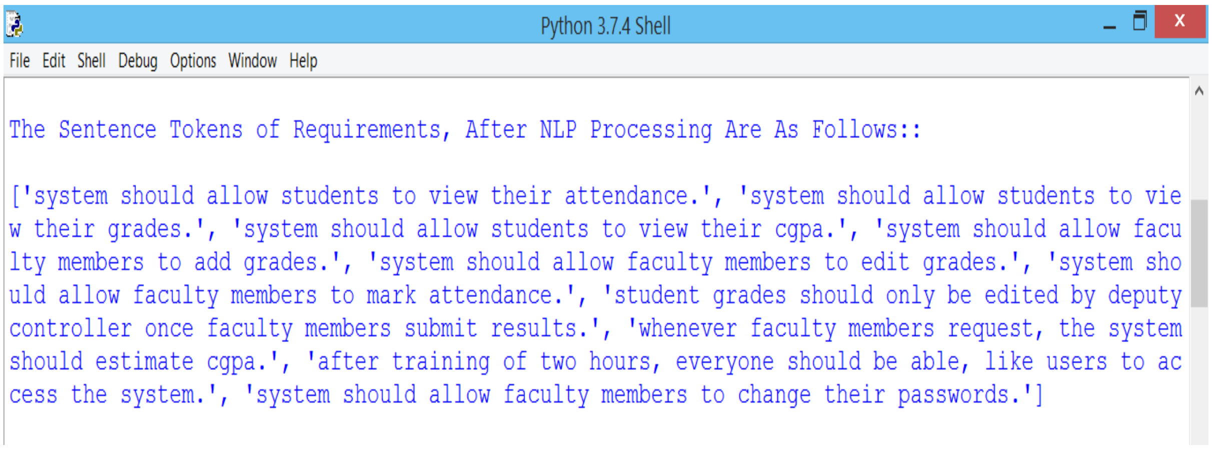
Automated SRS Checker.py - C:\Users\Asif\Desktop\Thesis\Report\Automated SRS Checker.py (3.7.4)
File Edit Format Run Options Window Help

```
#Performing Sentence Tokenization
print("")
print("")
print("The Sentence Tokens of Requirements, After NLP Processing Are As Follows::")

req_set_sent_token = sent_tokenize(req_set_lowercase)

print("")
print(req_set_sent_token)
```

**Figure 4.9:** Code for NLP sentence tokenization technique

**Output**

```
The Sentence Tokens of Requirements, After NLP Processing Are As Follows::

['system should allow students to view their attendance.', 'system should allow students to view their grades.', 'system should allow students to view their cgpa.', 'system should allow faculty members to add grades.', 'system should allow faculty members to edit grades.', 'system should allow faculty members to mark attendance.', 'student grades should only be edited by deputy controller once faculty members submit results.', 'whenever faculty members request, the system should estimate cgpa.', 'after training of two hours, everyone should be able, like users to access the system.', 'system should allow faculty members to change their passwords.']
```

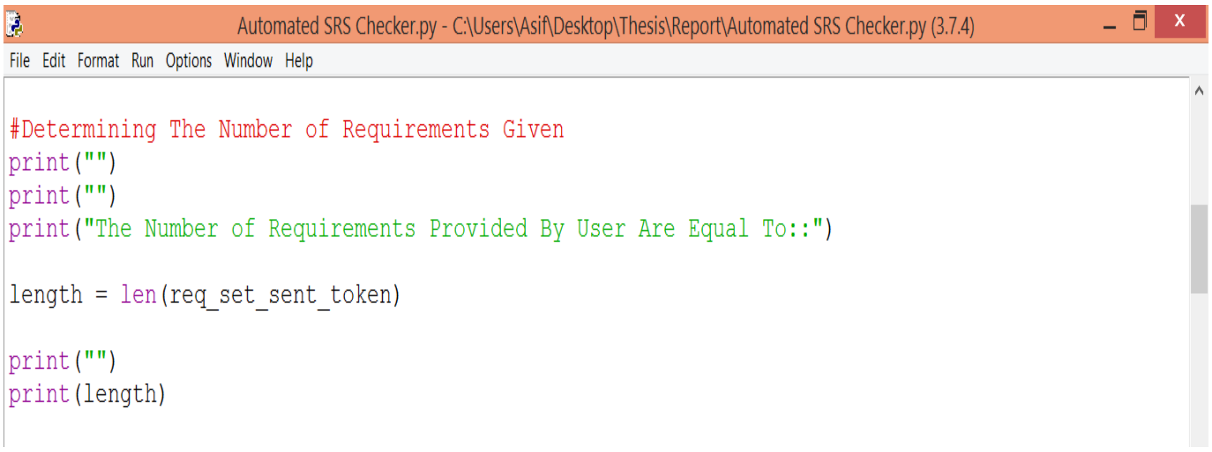


```
The Sentence Tokens of Requirements, After NLP Processing Are As Follows::

['system should allow students to view their attendance.', 'system should allow students to vie
w their grades.', 'system should allow students to view their cgpa.', 'system should allow facu
lty members to add grades.', 'system should allow faculty members to edit grades.', 'system sho
uld allow faculty members to mark attendance.', 'student grades should only be edited by deputy
controller once faculty members submit results.', 'whenever faculty members request, the system
should estimate cgpa.', 'after training of two hours, everyone should be able, like users to ac
cess the system.', 'system should allow faculty members to change their passwords.']
```

**Figure 4.10:** Output on shell

The code to verify the total number of user requirements entered was written as below.

| **Python Algorithm** |
|---|
| #Determining The Number of Requirements Given<br>print("")<br>print("")<br>print("The Number of Requirements Provided By User Are Equal To::")<br><br>length = len(req_set_sent_token)<br><br>print("")<br>print(length) |

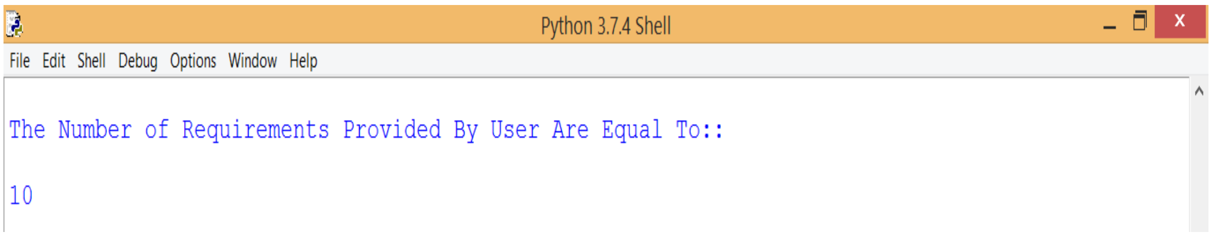

Automated SRS Checker.py - C:\Users\Asif\Desktop\Thesis\Report\Automated SRS Checker.py (3.7.4)
File Edit Format Run Options Window Help

```
#Determining The Number of Requirements Given
print("")
print("")
print("The Number of Requirements Provided By User Are Equal To::")

length = len(req_set_sent_token)

print("")
print(length)
```

**Figure 4.11:** Code for verification

| **Output** |
|---|
| The Number of Requirements Provided By User Are Equal To:: |

```
10
```



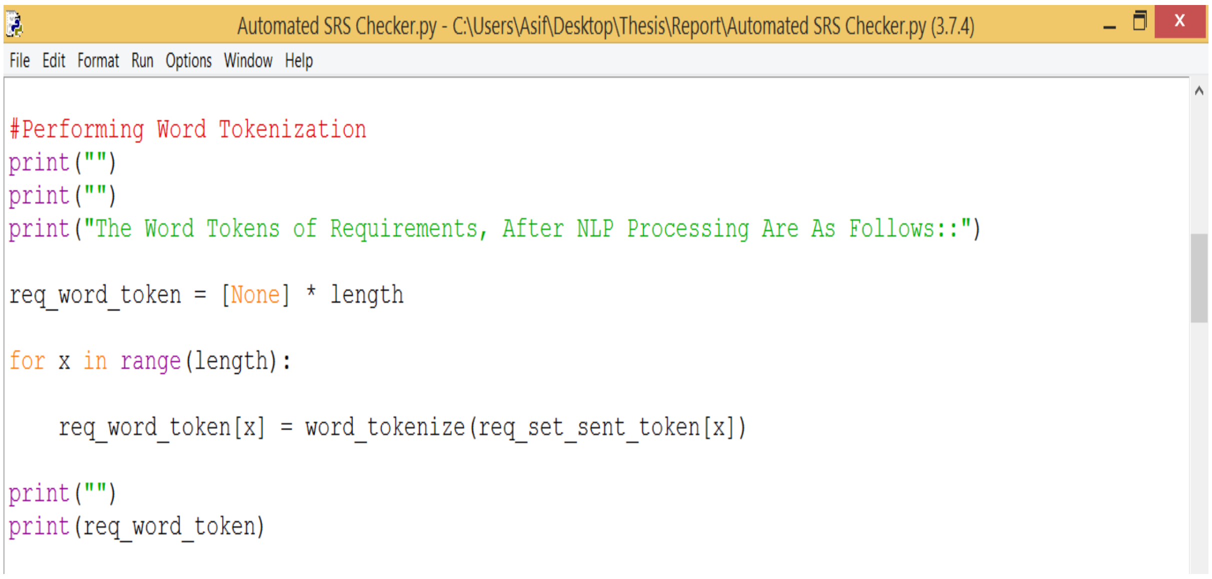
```
The Number of Requirements Provided By User Are Equal To::

10
```

**Figure 4.12:** Output on shell

The code to apply NLP technique to word tokenize all the user requirements was written as below.

**Python Algorithm**

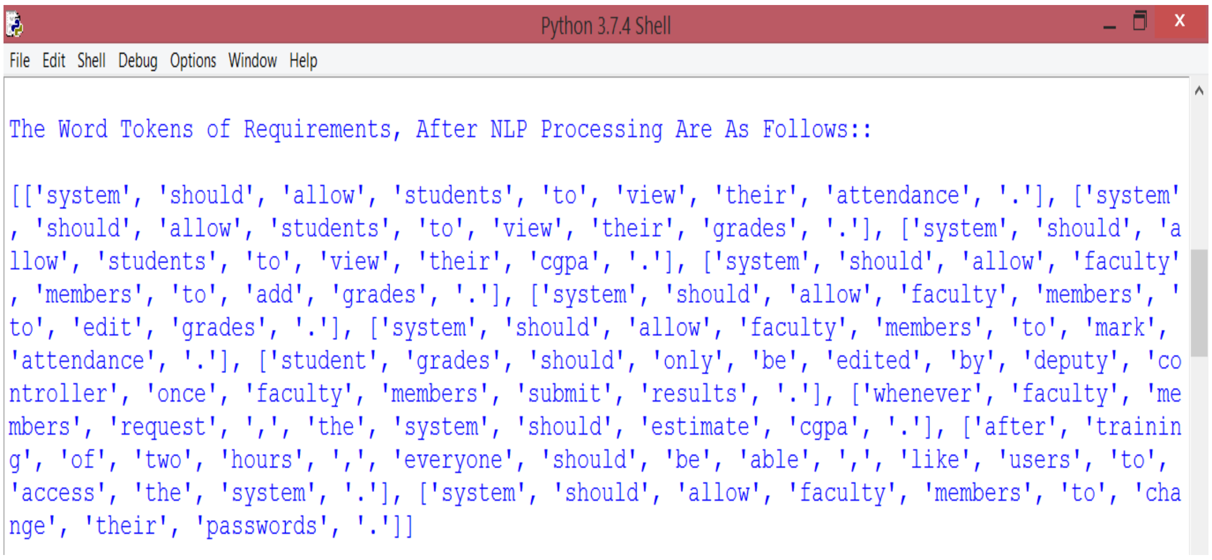
```
#Performing Word Tokenization
print("")
print("")
print("The Word Tokens of Requirements, After NLP Processing Are As Follows::")

req_word_token = [None] * length

for x in range(length):

    req_word_token[x] = word_tokenize(req_set_sent_token[x])

print("")
print(req_word_token)
```


Automated SRS Checker.py - C:\Users\Asif\Desktop\Thesis\Report\Automated SRS Checker.py (3.7.4)
File Edit Format Run Options Window Help

```
#Performing Word Tokenization
print("")
print("")
print("The Word Tokens of Requirements, After NLP Processing Are As Follows::")

req_word_token = [None] * length

for x in range(length):

    req_word_token[x] = word_tokenize(req_set_sent_token[x])

print("")
print(req_word_token)
```

**Figure 4.13:** Code for NLP word tokenization technique

| **Output** |
|---|
| The Word Tokens of Requirements, After NLP Processing Are As Follows::<br><br>[['system', 'should', 'allow', 'students', 'to', 'view', 'their', 'attendance', '.'], ['system', 'should', 'allow', 'students', 'to', 'view', 'their', 'grades', '.'], ['system', 'should', 'allow', 'students', 'to', 'view', 'their', 'cgpa', '.'], ['system', 'should', 'allow', 'faculty', 'members', 'to', 'add', 'grades', '.'], ['system', 'should', 'allow', 'faculty', 'members', 'to', 'edit', 'grades', '.'], ['system', 'should', 'allow', 'faculty', 'members', 'to', 'mark', 'attendance', '.'], ['student', 'grades', 'should', 'only', 'be', 'edited', 'by', 'deputy', 'controller', 'once', 'faculty', 'members', 'submit', 'results', '.'], ['whenever', 'faculty', 'members', 'request', ',', 'the', 'system', 'should', 'estimate', 'cgpa', '.'], ['after', 'training', 'of', 'two', 'hours', ',', 'everyone', 'should', 'be', 'able', ',', 'like', 'users', 'to', 'access', 'the', 'system', '.'], ['system', 'should', 'allow', 'faculty', 'members', 'to', 'change', 'their', 'passwords', '.']] |



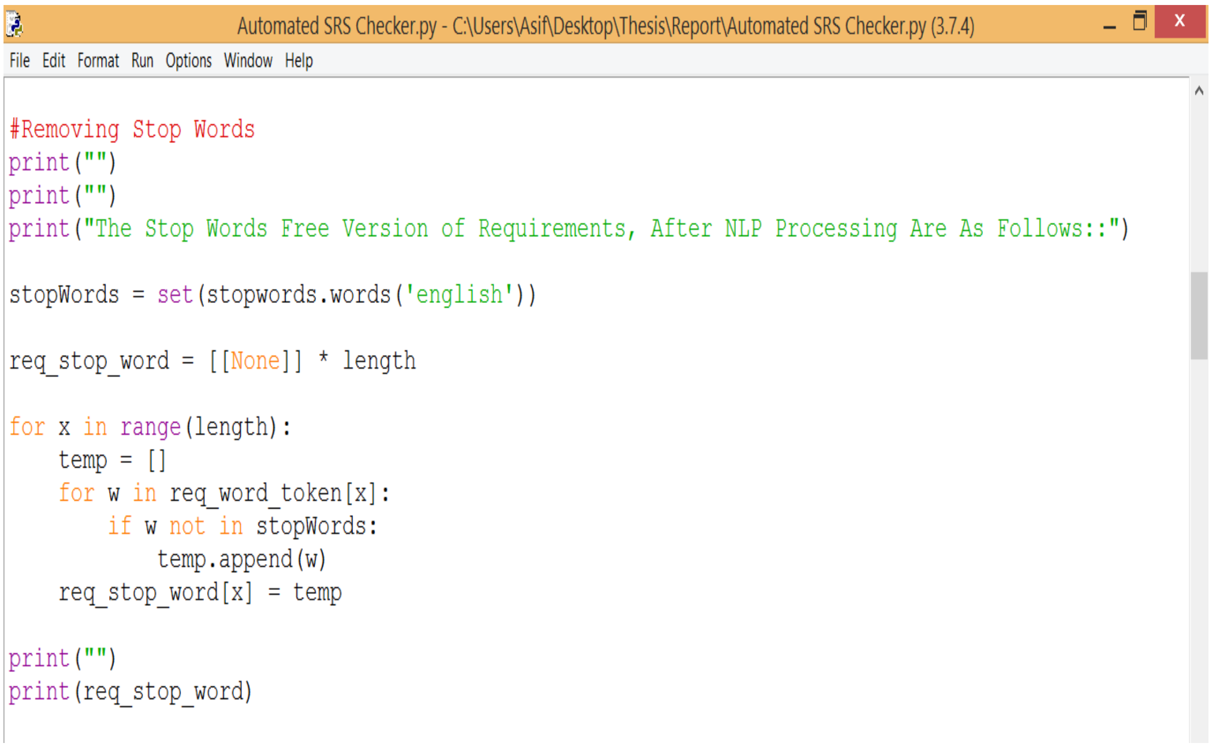
```
The Word Tokens of Requirements, After NLP Processing Are As Follows::

[['system', 'should', 'allow', 'students', 'to', 'view', 'their', 'attendance', '.'], ['system'
, 'should', 'allow', 'students', 'to', 'view', 'their', 'grades', '.'], ['system', 'should', 'a
llow', 'students', 'to', 'view', 'their', 'cgpa', '.'], ['system', 'should', 'allow', 'faculty'
, 'members', 'to', 'add', 'grades', '.'], ['system', 'should', 'allow', 'faculty', 'members', '
to', 'edit', 'grades', '.'], ['system', 'should', 'allow', 'faculty', 'members', 'to', 'mark',
'attendance', '.'], ['student', 'grades', 'should', 'only', 'be', 'edited', 'by', 'deputy', 'co
ntroller', 'once', 'faculty', 'members', 'submit', 'results', '.'], ['whenever', 'faculty', 'me
mbers', 'request', ',', 'the', 'system', 'should', 'estimate', 'cgpa', '.'], ['after', 'trainin
g', 'of', 'two', 'hours', ',', 'everyone', 'should', 'be', 'able', ',', 'like', 'users', 'to',
'access', 'the', 'system', '.'], ['system', 'should', 'allow', 'faculty', 'members', 'to', 'cha
nge', 'their', 'passwords', '.']]
```

**Figure 4.14:** Output on shell

The code to apply NLP technique to remove stop words from all the user requirements was written as below.

**Python Algorithm**

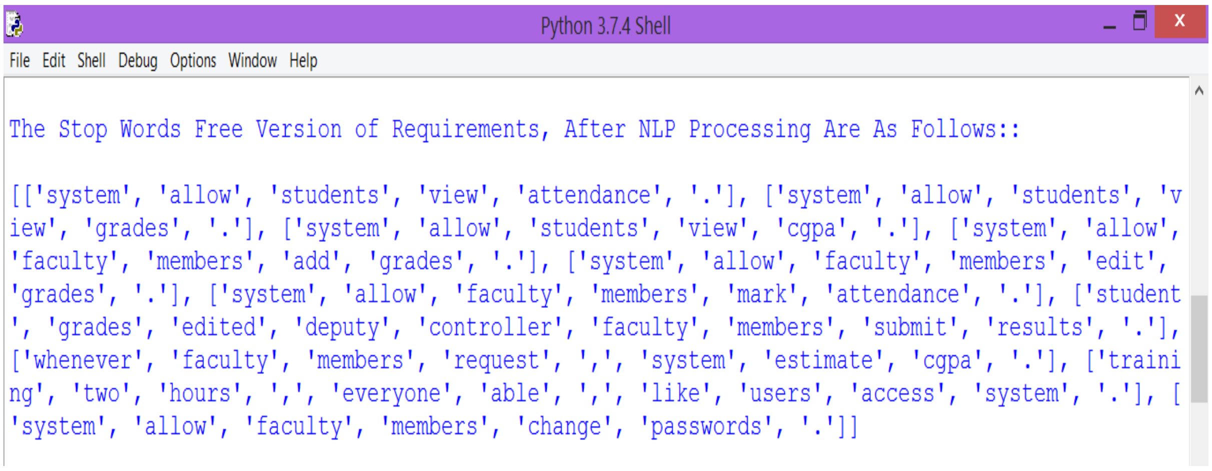
```
#Removing Stop Words
print("")
print("")
print("The Stop Words Free Version of Requirements, After NLP Processing Are As Follows::")

stopWords = set(stopwords.words('english'))

req_stop_word = [[None]] * length

for x in range(length):
    temp = []
    for w in req_word_token[x]:
        if w not in stopWords:
            temp.append(w)
    req_stop_word[x] = temp

print("")
print(req_stop_word)
```


Automated SRS Checker.py - C:\Users\Asif\Desktop\Thesis\Report\Automated SRS Checker.py (3.7.4)
File Edit Format Run Options Window Help

```
#Removing Stop Words
print("")
print("")
print("The Stop Words Free Version of Requirements, After NLP Processing Are As Follows::")

stopWords = set(stopwords.words('english'))

req_stop_word = [[None]] * length

for x in range(length):
    temp = []
    for w in req_word_token[x]:
        if w not in stopWords:
            temp.append(w)
    req_stop_word[x] = temp

print("")
print(req_stop_word)
```

**Figure 4.15:** Code for NLP stop words removal technique

| Output |
|---|
| The Stop Words Free Version of Requirements, After NLP Processing Are As Follows::<br><br>[['system', 'allow', 'students', 'view', 'attendance', '.'], ['system', 'allow', 'students', 'view', 'grades', '.'], ['system', 'allow', 'students', 'view', 'cgpa', '.'], ['system', 'allow', 'faculty', 'members', 'add', 'grades', '.'], ['system', 'allow', 'faculty', 'members', 'edit', 'grades', '.'], ['system', 'allow', 'faculty', 'members', 'mark', 'attendance', '.'], ['student', 'grades', 'edited', 'deputy', 'controller', 'faculty', 'members', 'submit', 'results', '.'], ['whenever', 'faculty', 'members', 'request', ',', 'system', 'estimate', 'cgpa', '.'], ['training', 'two', 'hours', ',', 'everyone', 'able', ',', 'like', 'users', 'access', 'system', '.'], ['system', 'allow', 'faculty', 'members', 'change', 'passwords', '.']] |



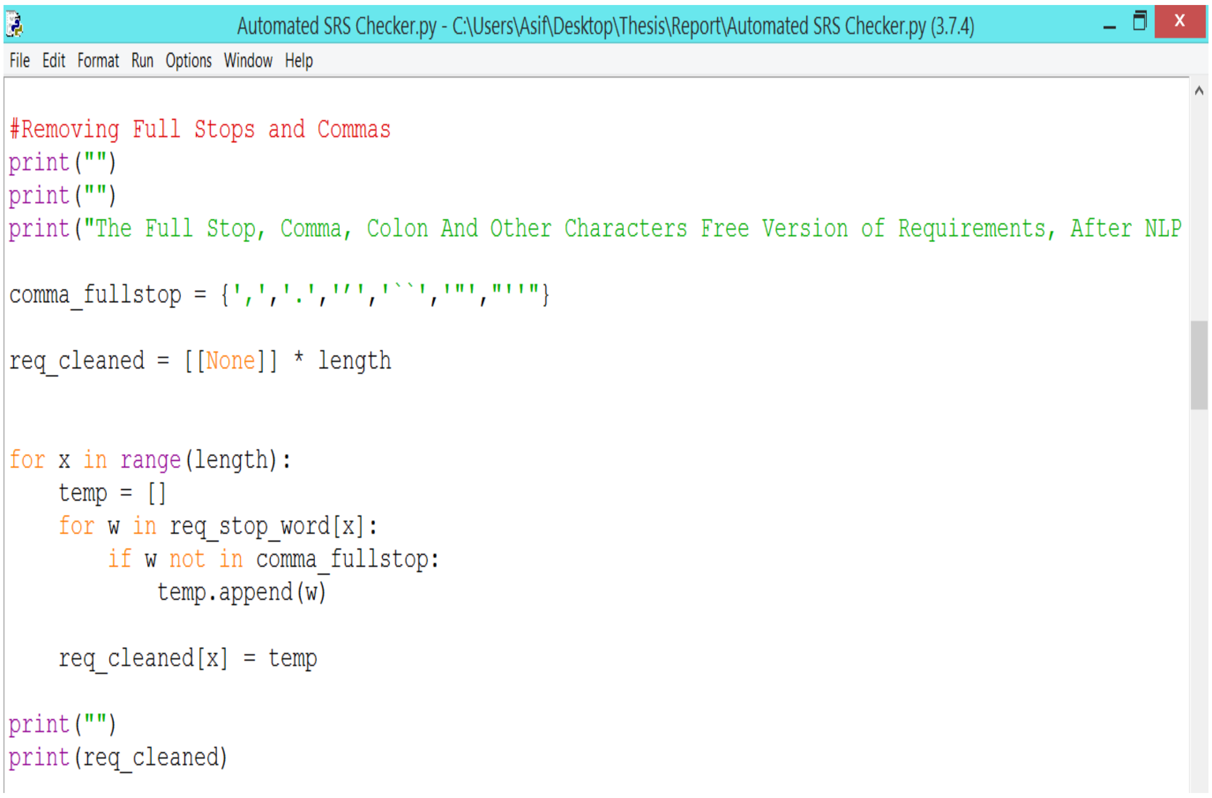
```
The Stop Words Free Version of Requirements, After NLP Processing Are As Follows::

[['system', 'allow', 'students', 'view', 'attendance', '.'], ['system', 'allow', 'students', 'v
iew', 'grades', '.'], ['system', 'allow', 'students', 'view', 'cgpa', '.'], ['system', 'allow',
'faculty', 'members', 'add', 'grades', '.'], ['system', 'allow', 'faculty', 'members', 'edit',
'grades', '.'], ['system', 'allow', 'faculty', 'members', 'mark', 'attendance', '.'], ['student
', 'grades', 'edited', 'deputy', 'controller', 'faculty', 'members', 'submit', 'results', '.'],
['whenever', 'faculty', 'members', 'request', ',', 'system', 'estimate', 'cgpa', '.'], ['traini
ng', 'two', 'hours', ',', 'everyone', 'able', ',', 'like', 'users', 'access', 'system', '.'], [
'system', 'allow', 'faculty', 'members', 'change', 'passwords', '.']]
```

**Figure 4.16:** Output on shell

The code to remove common punctuation marks from all the user requirements was written as below.

| **Python Algorithm** |
|---|

```
#Removing Full Stops and Commas
print("")
print("")
print("The Full Stop, Comma, Colon And Other Characters Free Version of Requirements, After NLP Processing Are As Follows::")

comma_fullstop = {',','.','’','``','''','''''}

req_cleaned = [[None]] * length



for x in range(length):
    temp = []
    for w in req_stop_word[x]:
        if w not in comma_fullstop:
            temp.append(w)

    req_cleaned[x] = temp

print("")
print(req_cleaned)
```

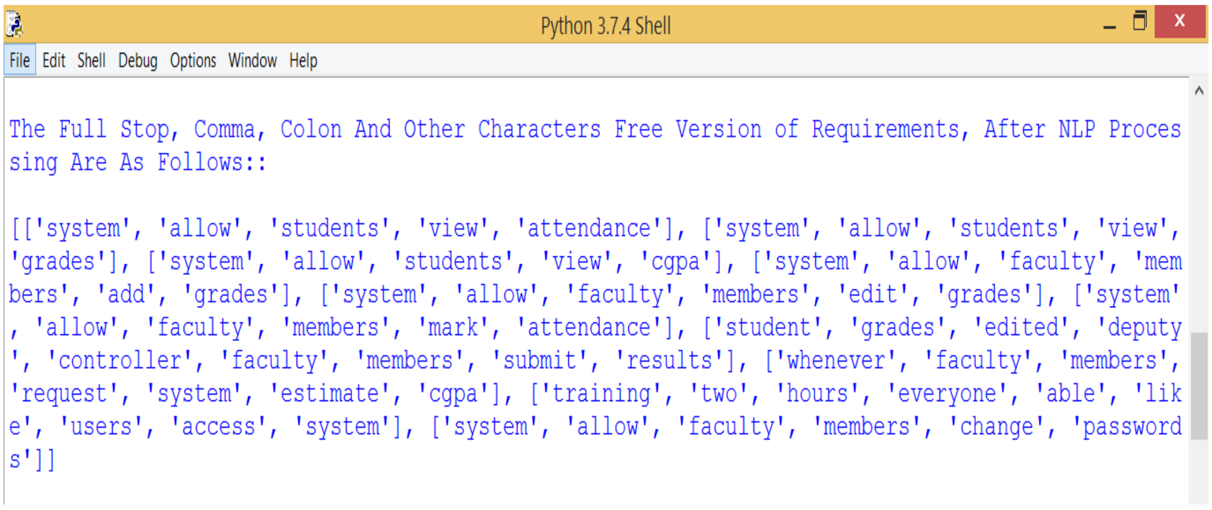


```
#Removing Full Stops and Commas
print("")
print("")
print("The Full Stop, Comma, Colon And Other Characters Free Version of Requirements, After NLP

comma_fullstop = {',','.','’','``','"',"''"}

req_cleaned = [[None]] * length


for x in range(length):
    temp = []
    for w in req_stop_word[x]:
        if w not in comma_fullstop:
            temp.append(w)

    req_cleaned[x] = temp

print("")
print(req_cleaned)
```

**Figure 4.17:** Code for removing common punctuation marks

**Output**

```
The Full Stop, Comma, Colon And Other Characters Free Version of Requirements, After NLP Processing Are As Follows::


[['system', 'allow', 'students', 'view', 'attendance'], ['system', 'allow', 'students', 'view', 'grades'], ['system', 'allow', 'students', 'view', 'cgpa'], ['system', 'allow', 'faculty', 'members', 'add', 'grades'], ['system', 'allow', 'faculty', 'members', 'edit', 'grades'], ['system', 'allow', 'faculty', 'members', 'mark', 'attendance'], ['student', 'grades', 'edited', 'deputy', 'controller', 'faculty', 'members', 'submit', 'results'], ['whenever', 'faculty', 'members', 'request', 'system', 'estimate', 'cgpa'], ['training', 'two', 'hours', 'everyone', 'able', 'like', 'users', 'access', 'system'], ['system', 'allow', 'faculty', 'members', 'change', 'passwords']]
```



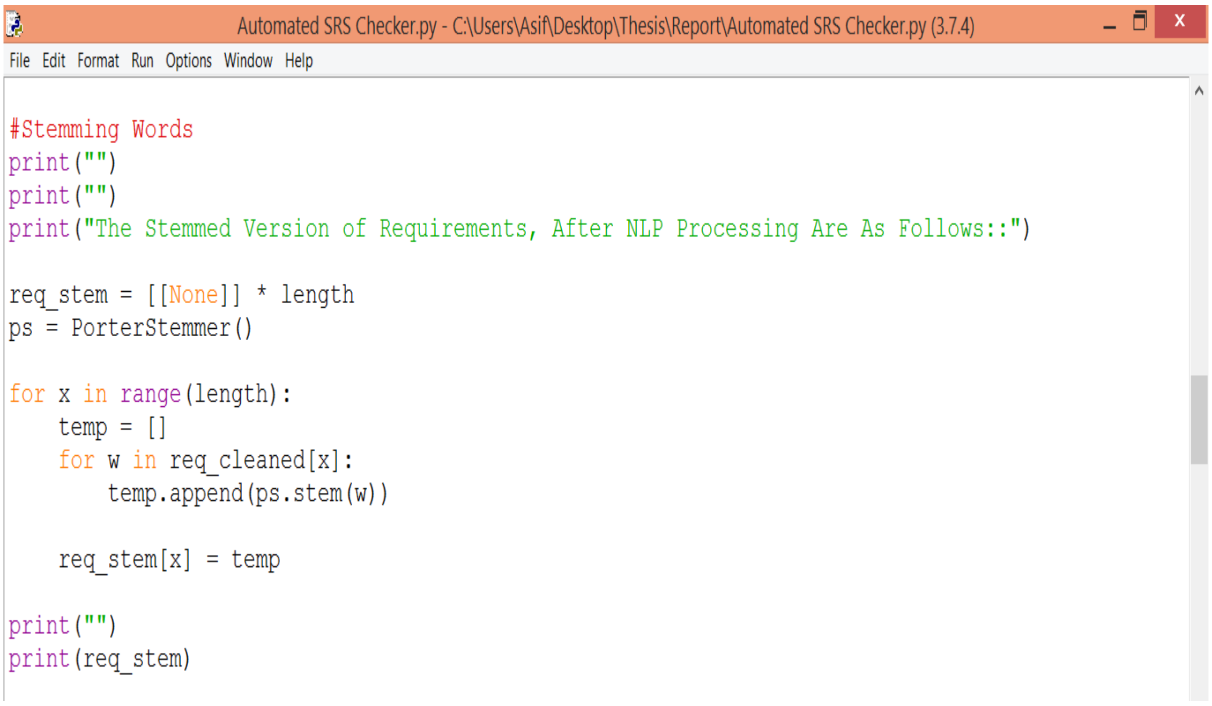
```
The Full Stop, Comma, Colon And Other Characters Free Version of Requirements, After NLP Proces
sing Are As Follows::

[['system', 'allow', 'students', 'view', 'attendance'], ['system', 'allow', 'students', 'view',
'grades'], ['system', 'allow', 'students', 'view', 'cgpa'], ['system', 'allow', 'faculty', 'mem
bers', 'add', 'grades'], ['system', 'allow', 'faculty', 'members', 'edit', 'grades'], ['system'
, 'allow', 'faculty', 'members', 'mark', 'attendance'], ['student', 'grades', 'edited', 'deputy
', 'controller', 'faculty', 'members', 'submit', 'results'], ['whenever', 'faculty', 'members',
'request', 'system', 'estimate', 'cgpa'], ['training', 'two', 'hours', 'everyone', 'able', 'lik
e', 'users', 'access', 'system'], ['system', 'allow', 'faculty', 'members', 'change', 'password
s']]
```

**Figure 4.18:** Output on shell

The code to apply NLP technique of stemming to all the user requirements was written as below.

**Python Algorithm**

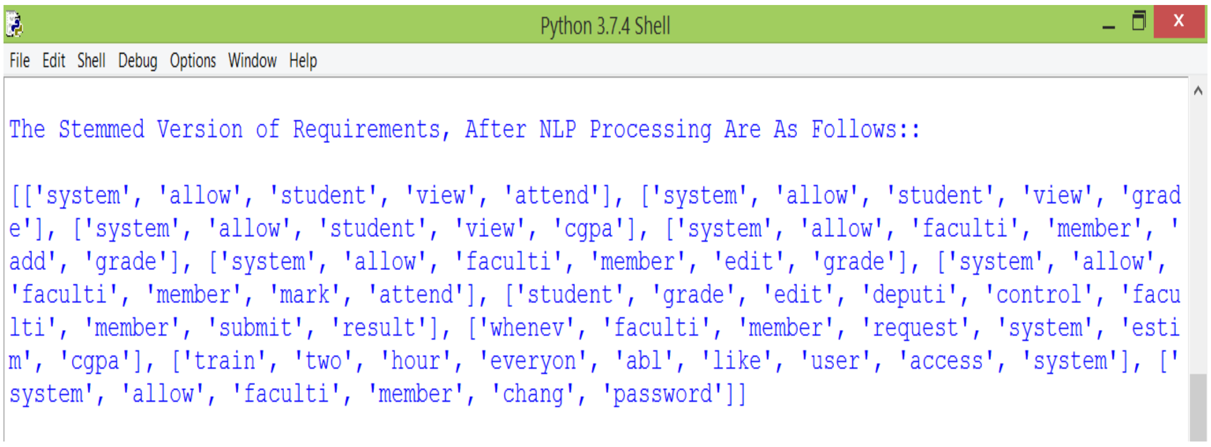
```
#Stemming Words
print("")
print("")
print("The Stemmed Version of Requirements, After NLP Processing Are As Follows::")

req_stem = [[None]] * length
ps = PorterStemmer()

for x in range(length):
    temp = []
    for w in req_cleaned[x]:
        temp.append(ps.stem(w))

    req_stem[x] = temp

print("")
print(req_stem)
```


Automated SRS Checker.py - C:\Users\Asif\Desktop\Thesis\Report\Automated SRS Checker.py (3.7.4)
File Edit Format Run Options Window Help

```
#Stemming Words
print("")
print("")
print("The Stemmed Version of Requirements, After NLP Processing Are As Follows::")

req_stem = [[None]] * length
ps = PorterStemmer()

for x in range(length):
    temp = []
    for w in req_cleaned[x]:
        temp.append(ps.stem(w))

    req_stem[x] = temp

print("")
print(req_stem)
```

**Figure 4.19:** Code for NLP stemming technique

| Output |
|---|
| The Stemmed Version of Requirements, After NLP Processing Are As Follows::<br><br>[['system', 'allow', 'student', 'view', 'attend'], ['system', 'allow', 'student', 'view', 'grade'], ['system', 'allow', 'student', 'view', 'cgpa'], ['system', 'allow', 'faculti', 'member', 'add', 'grade'], ['system', 'allow', 'faculti', 'member', 'edit', 'grade'], ['system', 'allow', 'faculti', 'member', 'mark', 'attend'], ['student', 'grade', 'edit', 'deputi', 'control', 'faculti', 'member', 'submit', 'result'], ['whenev', 'faculti', 'member', 'request', 'system', 'estim', 'cgpa'], ['train', 'two', 'hour', 'everyon', 'abl', 'like', 'user', 'access', 'system'], ['system', 'allow', 'faculti', 'member', 'chang', 'password']] |

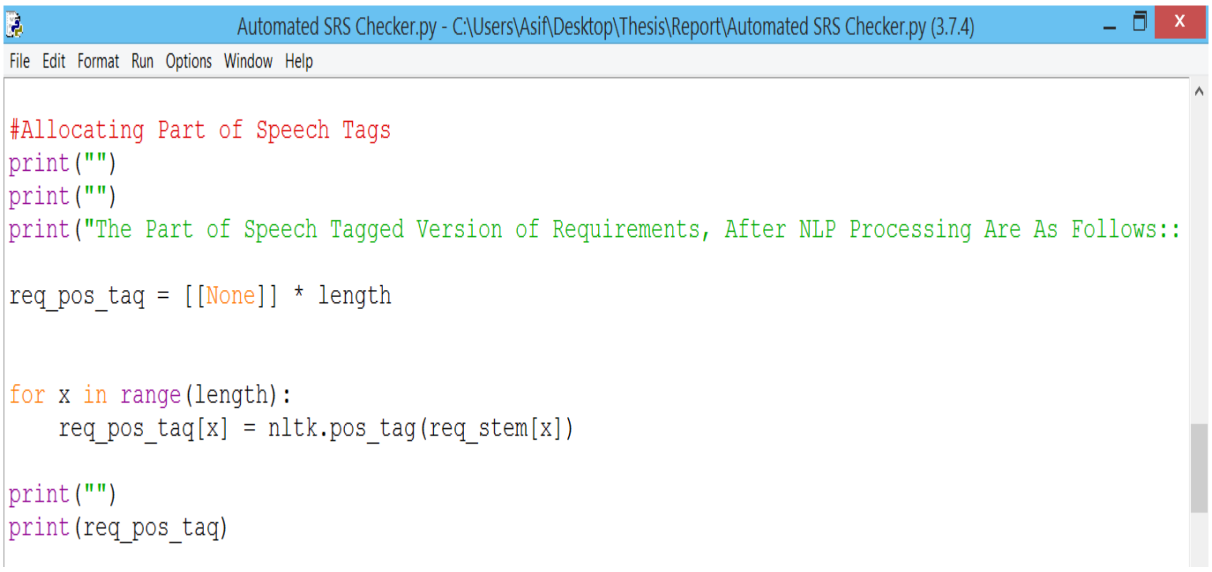


```
The Stemmed Version of Requirements, After NLP Processing Are As Follows::

[['system', 'allow', 'student', 'view', 'attend'], ['system', 'allow', 'student', 'view', 'grad
e'], ['system', 'allow', 'student', 'view', 'cgpa'], ['system', 'allow', 'faculti', 'member', '
add', 'grade'], ['system', 'allow', 'faculti', 'member', 'edit', 'grade'], ['system', 'allow',
'faculti', 'member', 'mark', 'attend'], ['student', 'grade', 'edit', 'deputi', 'control', 'facu
lti', 'member', 'submit', 'result'], ['whenev', 'faculti', 'member', 'request', 'system', 'esti
m', 'cgpa'], ['train', 'two', 'hour', 'everyon', 'abl', 'like', 'user', 'access', 'system'], ['
system', 'allow', 'faculti', 'member', 'chang', 'password']]
```

**Figure 4.20:** Output on shell

The code to apply NLP technique to allocate part of speech tags to all the user requirements was written as below.

| Python Algorithm |
|---|
| #Allocating Part of Speech Tags<br>print("")<br>print("")<br>print("The Part of Speech Tagged Version of Requirements, After NLP Processing Are As Follows::")<br><br>req_pos_taq = [[None]] * length<br><br><br>for x in range(length):<br>    req_pos_taq[x] = nltk.pos_tag(req_stem[x])<br><br>print("")<br>print(req_pos_taq) |

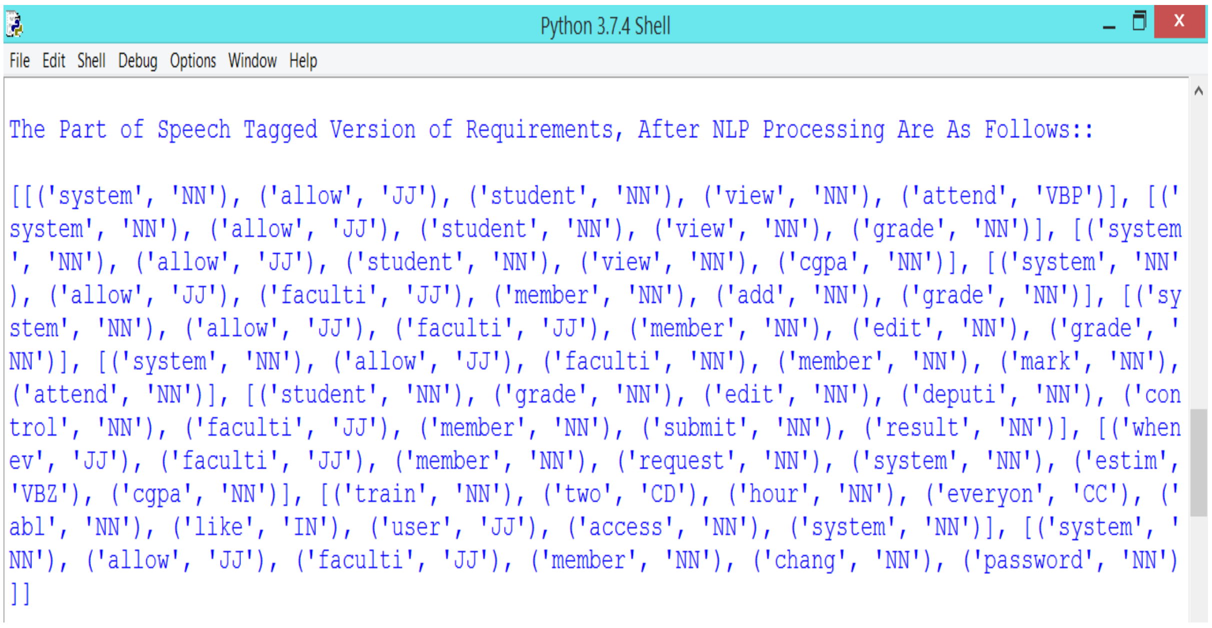

Automated SRS Checker.py - C:\Users\Asif\Desktop\Thesis\Report\Automated SRS Checker.py (3.7.4)
File Edit Format Run Options Window Help

```
#Allocating Part of Speech Tags
print("")
print("")
print("The Part of Speech Tagged Version of Requirements, After NLP Processing Are As Follows::

req_pos_taq = [[None]] * length


for x in range(length):
    req_pos_taq[x] = nltk.pos_tag(req_stem[x])

print("")
print(req_pos_taq)
```

**Figure 4.21:** Code for NLP part of speech tagging technique

| Output |
|---|
| The Part of Speech Tagged Version of Requirements, After NLP Processing Are As Follows:: |

```
[[('system', 'NN'), ('allow', 'JJ'), ('student', 'NN'), ('view', 'NN'), ('attend', 'VBP')], [('system', 'NN'), ('allow', 'JJ'), ('student', 'NN'), ('view', 'NN'), ('grade', 'NN')], [('system', 'NN'), ('allow', 'JJ'), ('student', 'NN'), ('view', 'NN'), ('cgpa', 'NN')], [('system', 'NN'), ('allow', 'JJ'), ('faculti', 'JJ'), ('member', 'NN'), ('add', 'NN'), ('grade', 'NN')], [('system', 'NN'), ('allow', 'JJ'), ('faculti', 'JJ'), ('member', 'NN'), ('edit', 'NN'), ('grade', 'NN')], [('system', 'NN'), ('allow', 'JJ'), ('faculti', 'NN'), ('member', 'NN'), ('mark', 'NN'), ('attend', 'NN')], [('student', 'NN'), ('grade', 'NN'), ('edit', 'NN'), ('deputi', 'NN'), ('control', 'NN'), ('faculti', 'JJ'), ('member', 'NN'), ('submit', 'NN'), ('result', 'NN')], [('whenev', 'JJ'), ('faculti', 'JJ'), ('member', 'NN'), ('request', 'NN'), ('system', 'NN'), ('estim', 'VBZ'), ('cgpa', 'NN')], [('train', 'NN'), ('two', 'CD'), ('hour', 'NN'), ('everyon', 'CC'), ('abl', 'NN'), ('like', 'IN'), ('user', 'JJ'), ('access', 'NN'), ('system', 'NN')], [('system', 'NN'), ('allow', 'JJ'), ('faculti', 'JJ'), ('member', 'NN'), ('chang', 'NN'), ('password', 'NN')]]
```



```
The Part of Speech Tagged Version of Requirements, After NLP Processing Are As Follows::

[[('system', 'NN'), ('allow', 'JJ'), ('student', 'NN'), ('view', 'NN'), ('attend', 'VBP')], [('
system', 'NN'), ('allow', 'JJ'), ('student', 'NN'), ('view', 'NN'), ('grade', 'NN')], [('system
', 'NN'), ('allow', 'JJ'), ('student', 'NN'), ('view', 'NN'), ('cgpa', 'NN')], [('system', 'NN'
), ('allow', 'JJ'), ('faculti', 'JJ'), ('member', 'NN'), ('add', 'NN'), ('grade', 'NN')], [('sy
stem', 'NN'), ('allow', 'JJ'), ('faculti', 'JJ'), ('member', 'NN'), ('edit', 'NN'), ('grade', '
NN')], [('system', 'NN'), ('allow', 'JJ'), ('faculti', 'NN'), ('member', 'NN'), ('mark', 'NN'),
('attend', 'NN')], [('student', 'NN'), ('grade', 'NN'), ('edit', 'NN'), ('deputi', 'NN'), ('con
trol', 'NN'), ('faculti', 'JJ'), ('member', 'NN'), ('submit', 'NN'), ('result', 'NN')], [('when
ev', 'JJ'), ('faculti', 'JJ'), ('member', 'NN'), ('request', 'NN'), ('system', 'NN'), ('estim',
'VBZ'), ('cgpa', 'NN')], [('train', 'NN'), ('two', 'CD'), ('hour', 'NN'), ('everyon', 'CC'), ('
abl', 'NN'), ('like', 'IN'), ('user', 'JJ'), ('access', 'NN'), ('system', 'NN')], [('system', '
NN'), ('allow', 'JJ'), ('faculti', 'JJ'), ('member', 'NN'), ('chang', 'NN'), ('password', 'NN')
]]
```

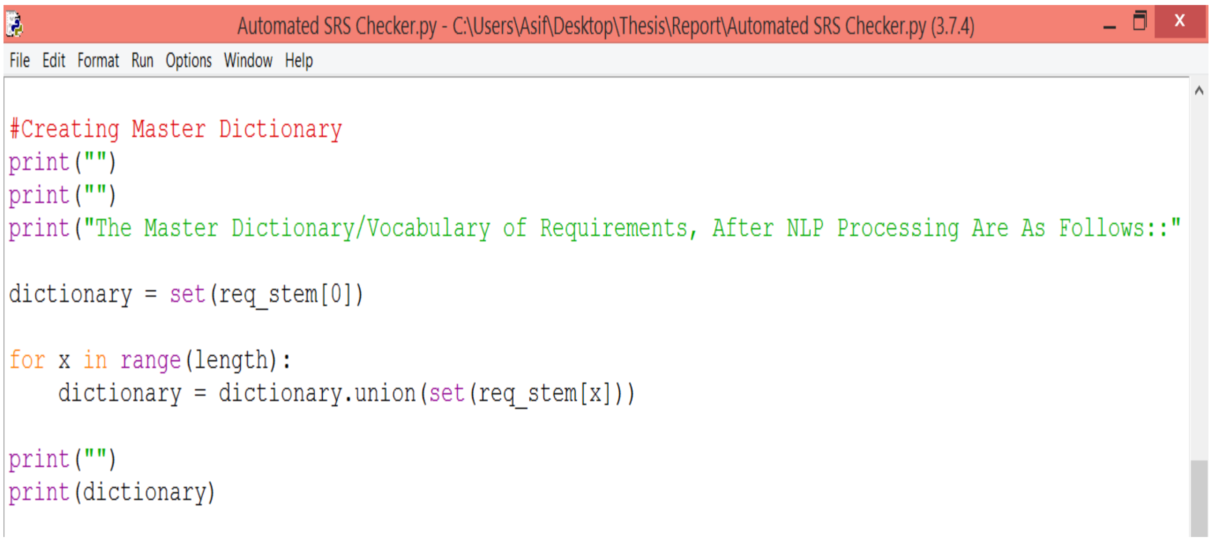
**Figure 4.22:** Output on shell

The code to create a training set was written as below.

**Python Algorithm**

```
#Creating Master Dictionary
print("")
print("")
print("The Master Dictionary/Vocabulary of Requirements, After NLP Processing Are As Follows::")

dictionary = set(req_stem[0])
```

```
for x in range(length):
    dictionary = dictionary.union(set(req_stem[x]))


print("")
print(dictionary)
```

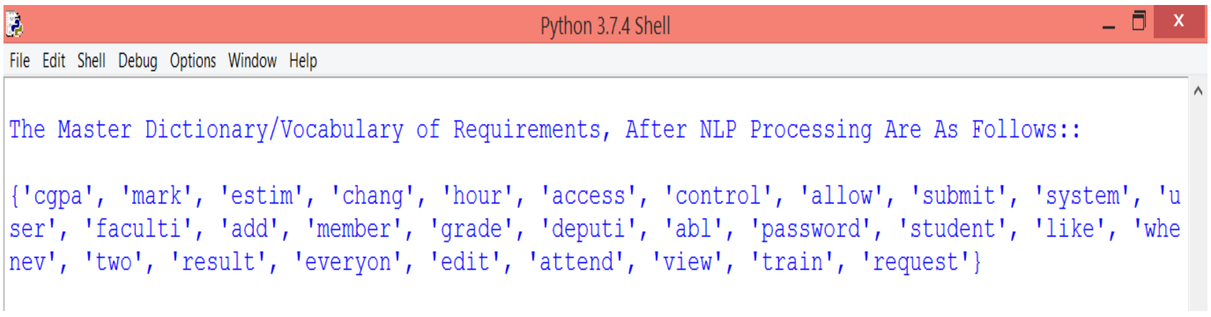

Automated SRS Checker.py - C:\Users\Asif\Desktop\Thesis\Report\Automated SRS Checker.py (3.7.4)
File Edit Format Run Options Window Help

```
#Creating Master Dictionary
print("")
print("")
print("The Master Dictionary/Vocabulary of Requirements, After NLP Processing Are As Follows::"

dictionary = set(req_stem[0])

for x in range(length):
    dictionary = dictionary.union(set(req_stem[x]))

print("")
print(dictionary)
```

**Figure 4.23:** Code for creating training set

**Output**

```
The Master Dictionary/Vocabulary of Requirements, After NLP Processing Are As Follows::


{'faculti', 'add', 'view', 'mark', 'member', 'allow', 'estim', 'hour', 'like', 'submit', 'chang', 'system', 'edit', 'attend', 'access', 'train', 'password', 'abl', 'two', 'whenev', 'result', 'student', 'request', 'user', 'everyon', 'control', 'grade', 'cgpa', 'deputi'}
```



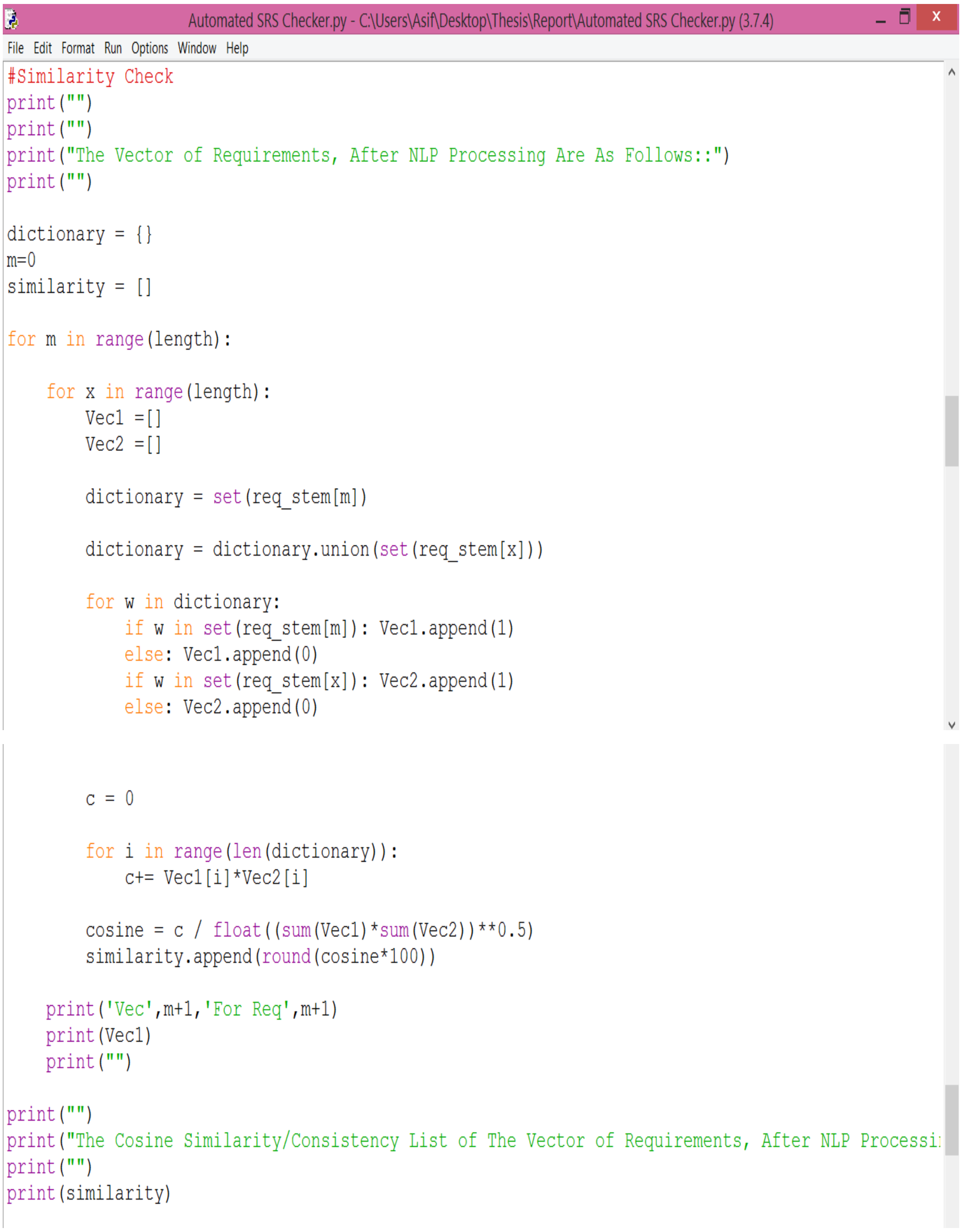
```
The Master Dictionary/Vocabulary of Requirements, After NLP Processing Are As Follows::

{'cgpa', 'mark', 'estim', 'chang', 'hour', 'access', 'control', 'allow', 'submit', 'system', 'u
ser', 'faculti', 'add', 'member', 'grade', 'deputi', 'abl', 'password', 'student', 'like', 'whe
nev', 'two', 'result', 'everyon', 'edit', 'attend', 'view', 'train', 'request'}
```

**Figure 4.24:** Output on shell

The code to create vectors of all the user requirements and checking them for semantic similarities was written as below.

**Python Algorithm**

```
#Similarity Check
print("")
print("")
print("The Vector of Requirements, After NLP Processing Are As Follows::")
print("")

dictionary = {}
m=0
similarity = []

for m in range(length):

    for x in range(length):
        Vec1 =[]
        Vec2 =[]

        dictionary = set(req_stem[m])

        dictionary = dictionary.union(set(req_stem[x]))

        for w in dictionary:
            if w in set(req_stem[m]): Vec1.append(1)
            else: Vec1.append(0)
            if w in set(req_stem[x]): Vec2.append(1)
            else: Vec2.append(0)



        c = 0


        for i in range(len(dictionary)):
            c+= Vec1[i]*Vec2[i]


        cosine = c / float((sum(Vec1)*sum(Vec2))**0.5)
        similarity.append(round(cosine*100))


    print('Vec',m+1,'For Req',m+1)
    print(Vec1)
    print("")


print("")
print("The Cosine Similarity/Consistency List of The Vector of Requirements, After NLP
Processing Are As Follows::")
print("")
print(similarity)
```


Automated SRS Checker.py - C:\Users\Asif\Desktop\Thesis\Report\Automated SRS Checker.py (3.7.4)
File Edit Format Run Options Window Help

```
#Similarity Check
print("")
print("")
print("The Vector of Requirements, After NLP Processing Are As Follows::")
print("")

dictionary = {}
m=0
similarity = []

for m in range(length):

    for x in range(length):
        Vec1 =[]
        Vec2 =[]

        dictionary = set(req_stem[m])

        dictionary = dictionary.union(set(req_stem[x]))

        for w in dictionary:
            if w in set(req_stem[m]): Vec1.append(1)
            else: Vec1.append(0)
            if w in set(req_stem[x]): Vec2.append(1)
            else: Vec2.append(0)


        c = 0

        for i in range(len(dictionary)):
            c+= Vec1[i]*Vec2[i]

        cosine = c / float((sum(Vec1)*sum(Vec2))**0.5)
        similarity.append(round(cosine*100))

    print('Vec',m+1,'For Req',m+1)
    print(Vec1)
    print("")

print("")
print("The Cosine Similarity/Consistency List of The Vector of Requirements, After NLP Processi
print("")
print(similarity)
```

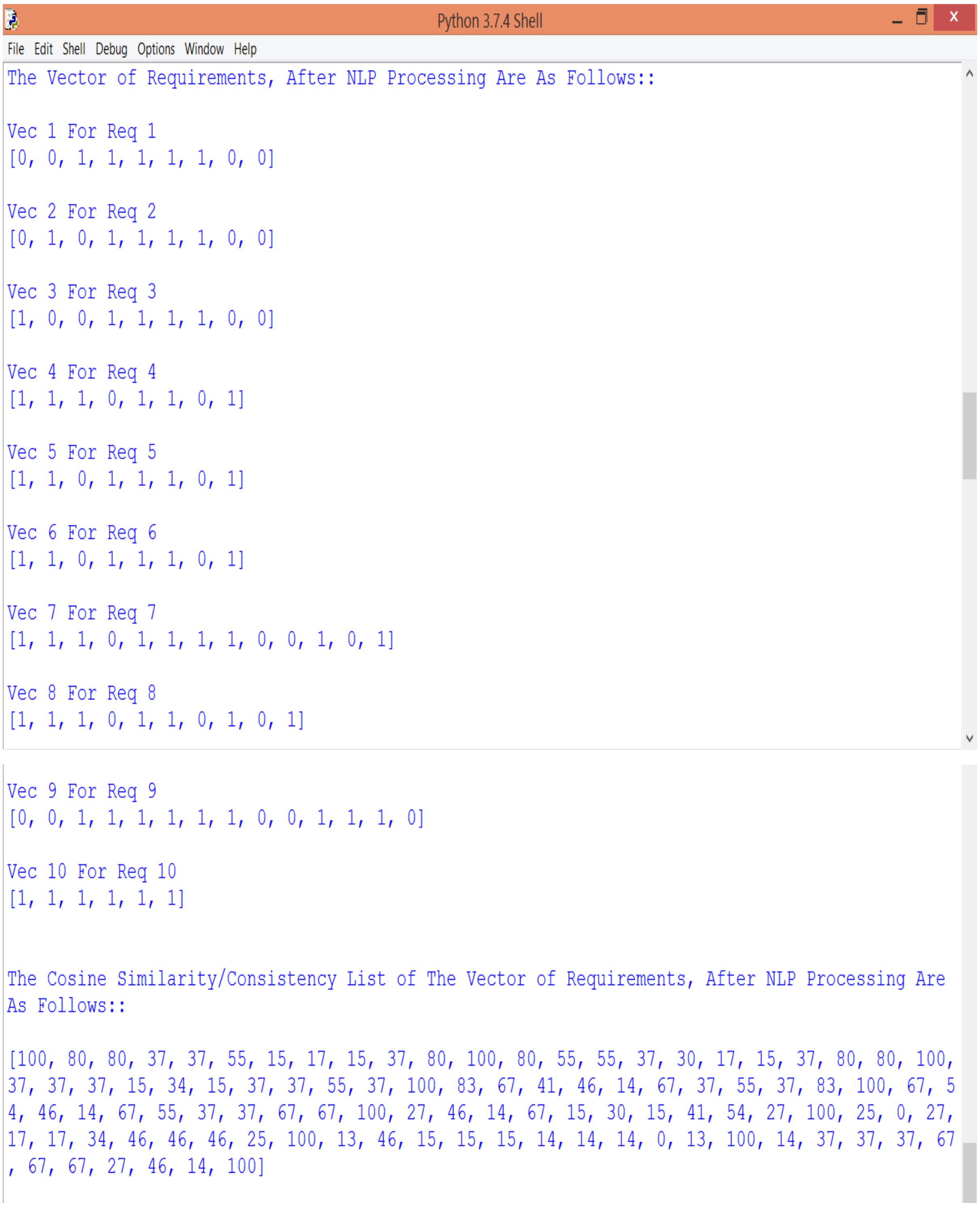
**Figure 4.25:** Code for creating vectors and check for semantic similarities

| **Output** |
|---|
| The Vector of Requirements, After NLP Processing Are As Follows::<br><br>Vec 1 For Req 1 |

```
[0, 0, 1, 0, 1, 1, 0, 1, 1]

Vec 2 For Req 2
[0, 0, 1, 0, 1, 1, 0, 1, 1]

Vec 3 For Req 3
[0, 0, 1, 0, 1, 1, 0, 1, 1]

Vec 4 For Req 4
[1, 1, 0, 1, 1, 0, 1, 1]

Vec 5 For Req 5
[1, 0, 1, 1, 0, 1, 1, 1]

Vec 6 For Req 6
[1, 0, 1, 1, 1, 0, 1, 1]

Vec 7 For Req 7
[1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1]

Vec 8 For Req 8
[1, 0, 1, 1, 0, 1, 1, 0, 1, 1]

Vec 9 For Req 9
[1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1]

Vec 10 For Req 10
[1, 1, 1, 1, 1, 1]
```

The Cosine Similarity/Consistency List of The Vector of Requirements, After NLP Processing Are As Follows::

```
[100, 80, 80, 37, 37, 55, 15, 17, 15, 37, 80, 100, 80, 55, 55, 37, 30, 17, 15, 37, 80, 80, 100, 37,
37, 37, 15, 34, 15, 37, 37, 55, 37, 100, 83, 67, 41, 46, 14, 67, 37, 55, 37, 83, 100, 67, 54, 46,
14, 67, 55, 37, 37, 67, 67, 100, 27, 46, 14, 67, 15, 30, 15, 41, 54, 27, 100, 25, 0, 27, 17, 17,
34, 46, 46, 46, 25, 100, 13, 46, 15, 15, 15, 14, 14, 14, 0, 13, 100, 14, 37, 37, 37, 67, 67, 67,
27, 46, 14, 100]
```



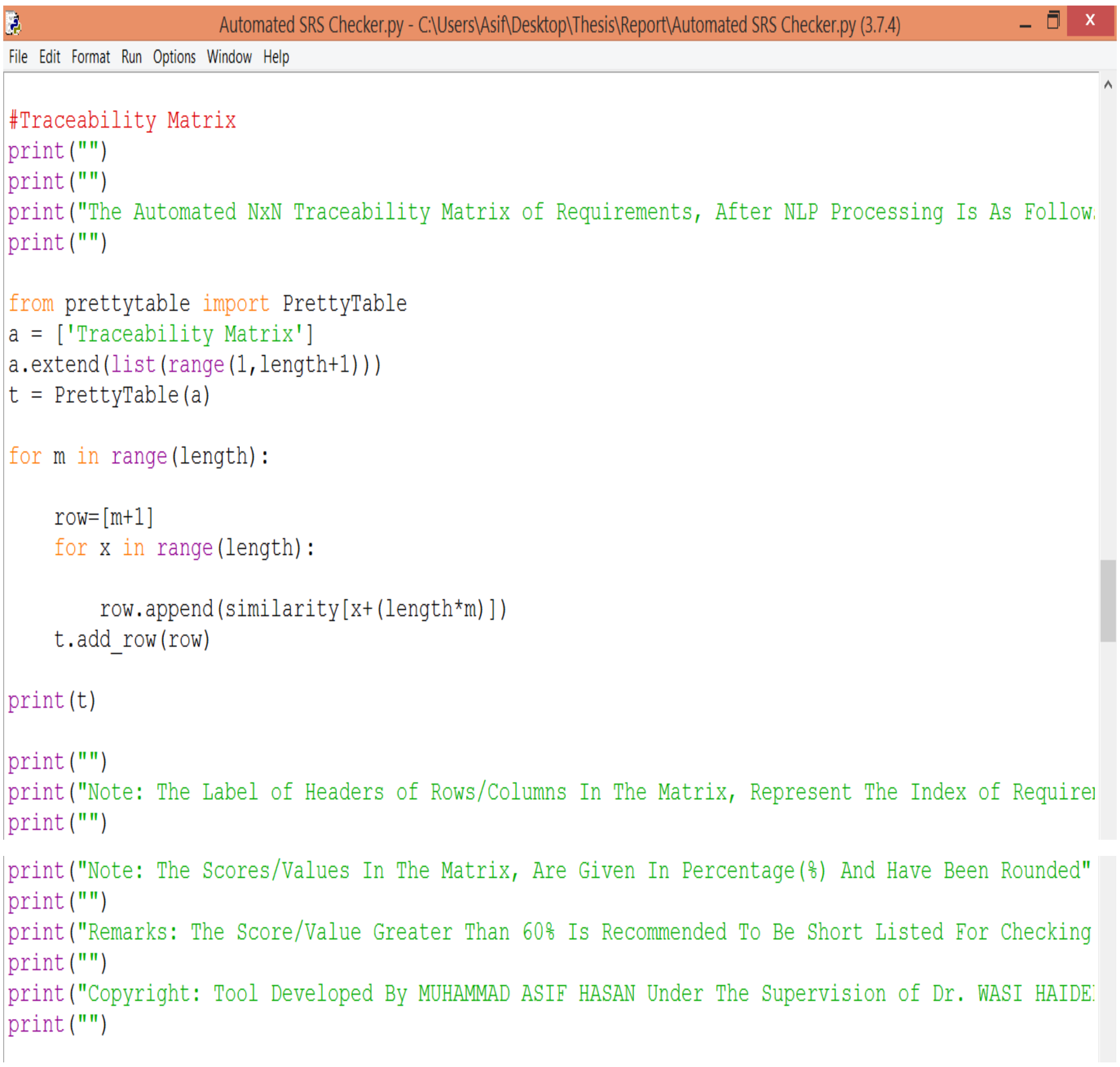
```
The Vector of Requirements, After NLP Processing Are As Follows::

Vec 1 For Req 1
[0, 0, 1, 1, 1, 1, 1, 0, 0]

Vec 2 For Req 2
[0, 1, 0, 1, 1, 1, 1, 0, 0]

Vec 3 For Req 3
[1, 0, 0, 1, 1, 1, 1, 0, 0]

Vec 4 For Req 4
[1, 1, 1, 0, 1, 1, 0, 1]

Vec 5 For Req 5
[1, 1, 0, 1, 1, 1, 0, 1]

Vec 6 For Req 6
[1, 1, 0, 1, 1, 1, 0, 1]

Vec 7 For Req 7
[1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1]

Vec 8 For Req 8
[1, 1, 1, 0, 1, 1, 0, 1, 0, 1]

Vec 9 For Req 9
[0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0]

Vec 10 For Req 10
[1, 1, 1, 1, 1, 1]


The Cosine Similarity/Consistency List of The Vector of Requirements, After NLP Processing Are
As Follows::

[100, 80, 80, 37, 37, 55, 15, 17, 15, 37, 80, 100, 80, 55, 55, 37, 30, 17, 15, 37, 80, 80, 100,
37, 37, 37, 15, 34, 15, 37, 37, 55, 37, 100, 83, 67, 41, 46, 14, 67, 37, 55, 37, 83, 100, 67, 5
4, 46, 14, 67, 55, 37, 37, 67, 67, 100, 27, 46, 14, 67, 15, 30, 15, 41, 54, 27, 100, 25, 0, 27,
17, 17, 34, 46, 46, 46, 25, 100, 13, 46, 15, 15, 15, 14, 14, 14, 0, 13, 100, 14, 37, 37, 37, 67
, 67, 67, 27, 46, 14, 100]
```

**Figure 4.26:** Output on shell

The code to create a traceability matrix was written as below.

| Python Algorithm |
|---|
| #Traceability Matrix |
| print("") |
| print("") |

```
print("The Automated NxN Traceability Matrix of Requirements, After NLP Processing Is As Follows::")
print("")

from prettytable import PrettyTable
a = ['Traceability Matrix']
a.extend(list(range(1,length+1)))
t = PrettyTable(a)

for m in range(length):

    row=[m+1]
    for x in range(length):

        row.append(similarity[x+(length*m)])
    t.add_row(row)

print(t)

print("")
print("Note: The Label of Headers of Rows/Columns In The Matrix, Represent The Index of Requirement")
print("")
print("Note: The Scores/Values In The Matrix, Are Given In Percentage(%) And Have Been Rounded")
print("")
print("Remarks: The Score/Value Greater Than 60% Is Recommended To Be Short Listed For Checking")
print("")
print("Copyright: Tool Developed By MUHAMMAD ASIF HASAN Under The Supervision of Dr. WASI HAIDER BUTT At College of Electrical & Mechanical Engineering, NUST University, Pakistan")
print("")
```


Automated SRS Checker.py - C:\Users\Asif\Desktop\Thesis\Report\Automated SRS Checker.py (3.7.4)
File Edit Format Run Options Window Help

```python
#Traceability Matrix
print("")
print("")
print("The Automated NxN Traceability Matrix of Requirements, After NLP Processing Is As Follow
print("")

from prettytable import PrettyTable
a = ['Traceability Matrix']
a.extend(list(range(1,length+1)))
t = PrettyTable(a)

for m in range(length):

    row=[m+1]
    for x in range(length):

        row.append(similarity[x+(length*m)])
    t.add_row(row)

print(t)

print("")
print("Note: The Label of Headers of Rows/Columns In The Matrix, Represent The Index of Require
print("")

print("Note: The Scores/Values In The Matrix, Are Given In Percentage(%) And Have Been Rounded"
print("")
print("Remarks: The Score/Value Greater Than 60% Is Recommended To Be Short Listed For Checking
print("")
print("Copyright: Tool Developed By MUHAMMAD ASIF HASAN Under The Supervision of Dr. WASI HAIDE
print("")
```

**Figure 4.27:** Code for creating a traceability matrix

**Output**

The Automated NxN Traceability Matrix of Requirements, After NLP Processing Is As Follows::

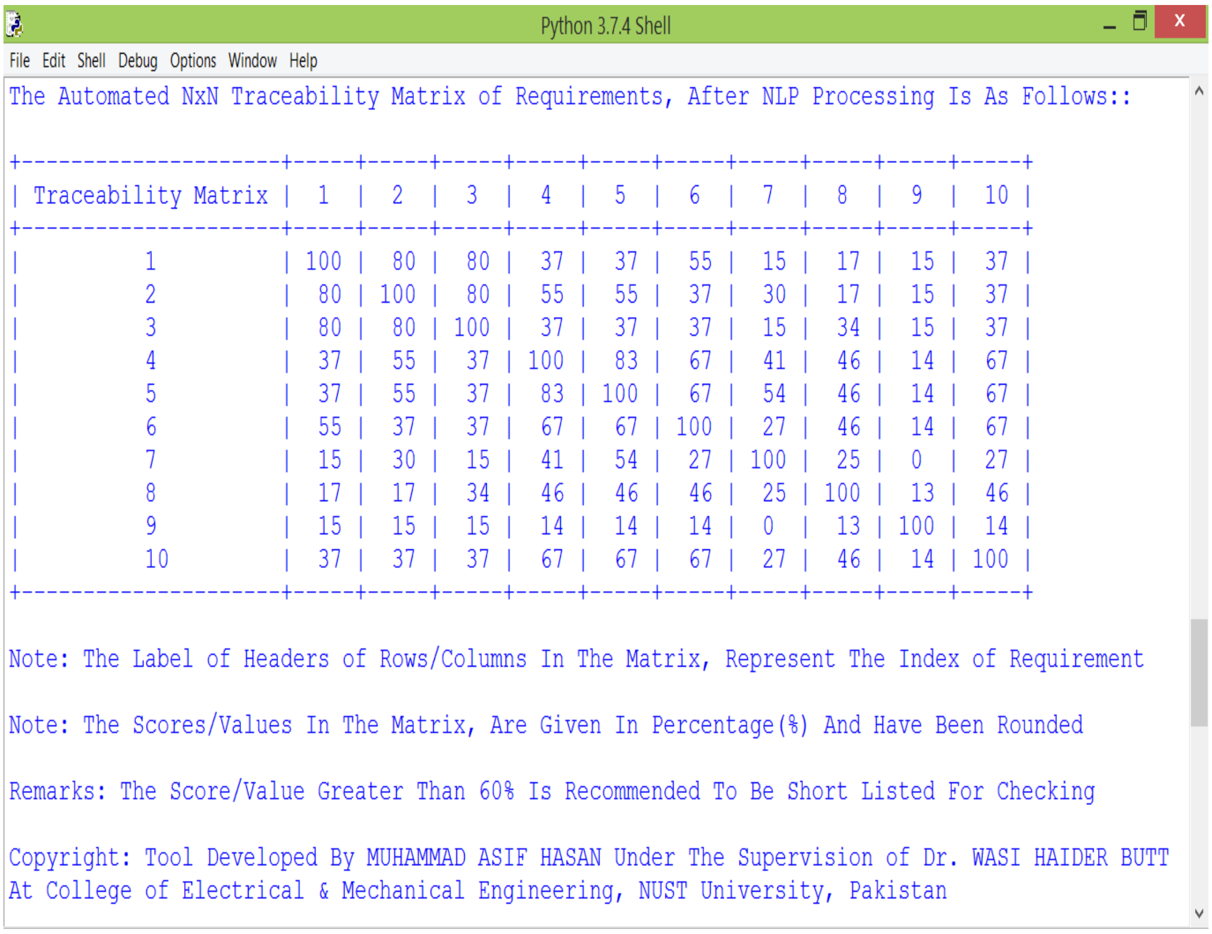
```
+---------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| Traceability Matrix |  1  |  2  |  3  |  4  |  5  |  6  |  7  |  8  |  9  |  10 |
+---------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|          1          | 100 |  80 |  80 |  37 |  37 |  55 |  15 |  17 |  15 |  37 |
|          2          |  80 | 100 |  80 |  55 |  55 |  37 |  30 |  17 |  15 |  37 |
|          3          |  80 |  80 | 100 |  37 |  37 |  37 |  15 |  34 |  15 |  37 |
|          4          |  37 |  55 |  37 | 100 |  83 |  67 |  41 |  46 |  14 |  67 |
|         5          |  37 |  55 |  37 |  83 | 100 |  67 |  54 |  46 |  14 |  67 |
|         6          |  55 |  37 |  37 |  67 |  67 | 100 |  27 |  46 |  14 |  67 |
|         7          |  15 |  30 |  15 |  41 |  54 |  27 | 100 |  25 |  0  |  27 |
|         8          |  17 |  17 |  34 |  46 |  46 |  46 |  25 | 100 |  13 |  46 |
|         9          |  15 |  15 |  15 |  14 |  14 |  14 |  0  |  13 | 100 |  14 |
|         10         |  37 |  37 |  37 |  67 |  67 |  67 |  27 |  46 |  14 | 100 |
+--------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
```

Note: The Label of Headers of Rows/Columns In The Matrix, Represent The Index of Requirement

Note: The Scores/Values In The Matrix, Are Given In Percentage(%) And Have Been Rounded

Remarks: The Score/Value Greater Than 60% Is Recommended To Be Short Listed For Checking

Copyright: Tool Developed By MUHAMMAD ASIF HASAN Under The Supervision of Dr. WASI HAIDER BUTT At College of Electrical & Mechanical Engineering, NUST University, Pakistan



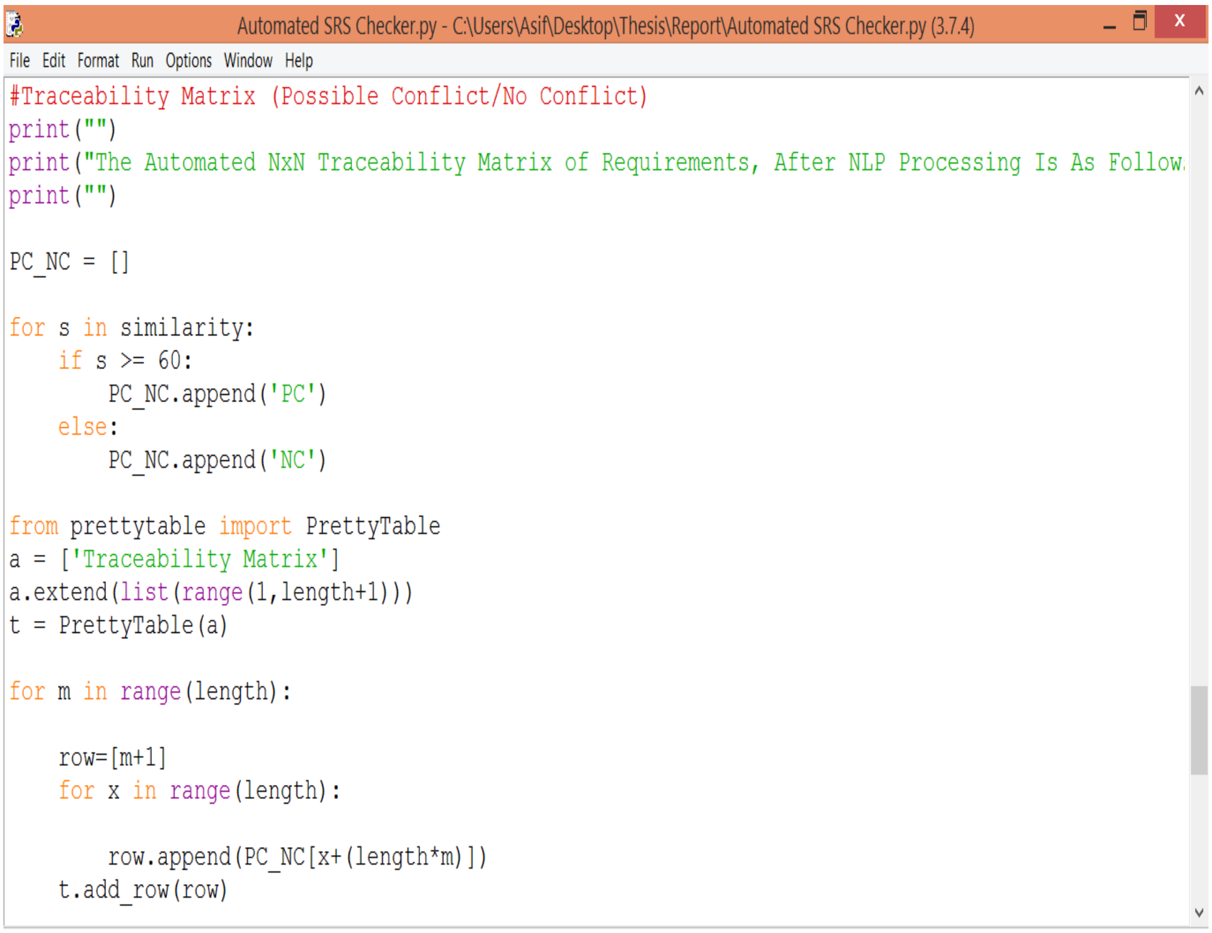
```
The Automated NxN Traceability Matrix of Requirements, After NLP Processing Is As Follows::

+---------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| Traceability Matrix |  1  |  2  |  3  |  4  |  5  |  6  |  7  |  8  |  9  |  10 |
+---------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|          1          | 100 |  80 |  80 |  37 |  37 |  55 |  15 |  17 |  15 |  37 |
|          2          |  80 | 100 |  80 |  55 |  55 |  37 |  30 |  17 |  15 |  37 |
|          3          |  80 |  80 | 100 |  37 |  37 |  37 |  15 |  34 |  15 |  37 |
|          4          |  37 |  55 |  37 | 100 |  83 |  67 |  41 |  46 |  14 |  67 |
|          5          |  37 |  55 |  37 |  83 | 100 |  67 |  54 |  46 |  14 |  67 |
|          6          |  55 |  37 |  37 |  67 |  67 | 100 |  27 |  46 |  14 |  67 |
|          7          |  15 |  30 |  15 |  41 |  54 |  27 | 100 |  25 |  0  |  27 |
|          8          |  17 |  17 |  34 |  46 |  46 |  46 |  25 | 100 |  13 |  46 |
|          9          |  15 |  15 |  15 |  14 |  14 |  14 |  0  |  13 | 100 |  14 |
|          10         |  37 |  37 |  37 |  67 |  67 |  67 |  27 |  46 |  14 | 100 |
+---------------------+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+

Note: The Label of Headers of Rows/Columns In The Matrix, Represent The Index of Requirement

Note: The Scores/Values In The Matrix, Are Given In Percentage(%) And Have Been Rounded

Remarks: The Score/Value Greater Than 60% Is Recommended To Be Short Listed For Checking

Copyright: Tool Developed By MUHAMMAD ASIF HASAN Under The Supervision of Dr. WASI HAIDER BUTT
At College of Electrical & Mechanical Engineering, NUST University, Pakistan
```

**Figure 4.28:** Output on shell

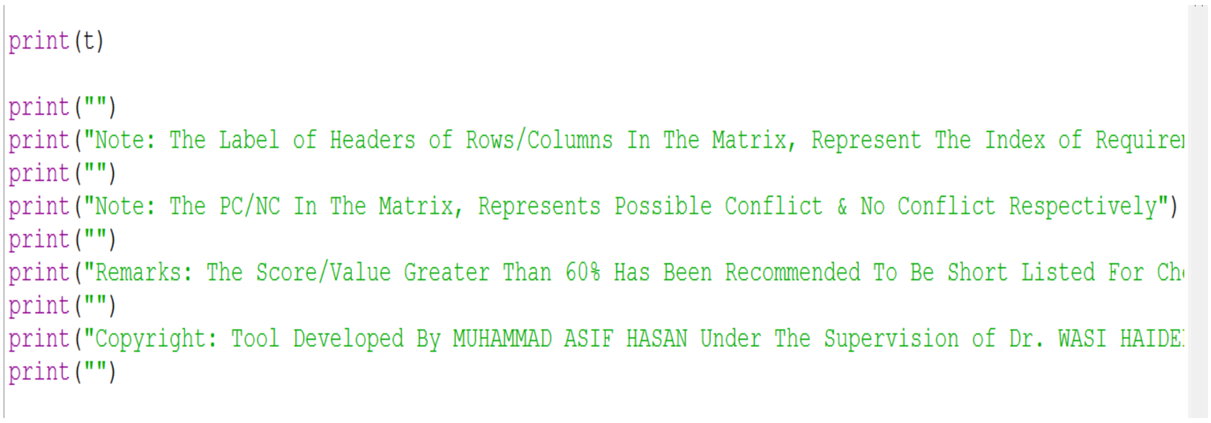
The code to create another type of traceability matrix was written as below.

**Python Algorithm**

```
#Traceability Matrix (Possible Conflict/No Conflict)
print("")
print("The Automated NxN Traceability Matrix of Requirements, After NLP Processing Is As Follows::")
print("")

PC_NC = []

for s in similarity:
    if s >= 60:
        PC_NC.append('PC')
    else:
        PC_NC.append('NC')

from prettytable import PrettyTable
a = ['Traceability Matrix']
a.extend(list(range(1,length+1)))
t = PrettyTable(a)

for m in range(length):

    row=[m+1]
    for x in range(length):

        row.append(PC_NC[x+(length*m)])
    t.add_row(row)

print(t)

print("")
print("Note: The Label of Headers of Rows/Columns In The Matrix, Represent The Index of
Requirement")
print("")
print("Note: The PC/NC In The Matrix, Represents Possible Conflict & No Conflict Respectively")
print("")
print("Remarks: The Score/Value Greater Than 60% Has Been Recommended To Be Short Listed For Checking")
print("")
print("Copyright: Tool Developed By MUHAMMAD ASIF HASAN Under The Supervision of Dr. WASI HAIDER BUTT At College of Electrical & Mechanical Engineering, NUST University, Pakistan")
print("")
```


Automated SRS Checker.py - C:\Users\Asif\Desktop\Thesis\Report\Automated SRS Checker.py (3.7.4)
File Edit Format Run Options Window Help

```
#Traceability Matrix (Possible Conflict/No Conflict)
print("")
print("The Automated NxN Traceability Matrix of Requirements, After NLP Processing Is As Follow
print("")

PC_NC = []

for s in similarity:
    if s >= 60:
        PC_NC.append('PC')
    else:
        PC_NC.append('NC')

from prettytable import PrettyTable
a = ['Traceability Matrix']
a.extend(list(range(1,length+1)))
t = PrettyTable(a)

for m in range(length):

    row=[m+1]
    for x in range(length):

        row.append(PC_NC[x+(length*m)])
    t.add_row(row)
```



```
print(t)

print("")
print("Note: The Label of Headers of Rows/Columns In The Matrix, Represent The Index of Requirer
print("")
print("Note: The PC/NC In The Matrix, Represents Possible Conflict & No Conflict Respectively")
print("")
print("Remarks: The Score/Value Greater Than 60% Has Been Recommended To Be Short Listed For Ch
print("")
print("Copyright: Tool Developed By MUHAMMAD ASIF HASAN Under The Supervision of Dr. WASI HAIDE
print("")
```

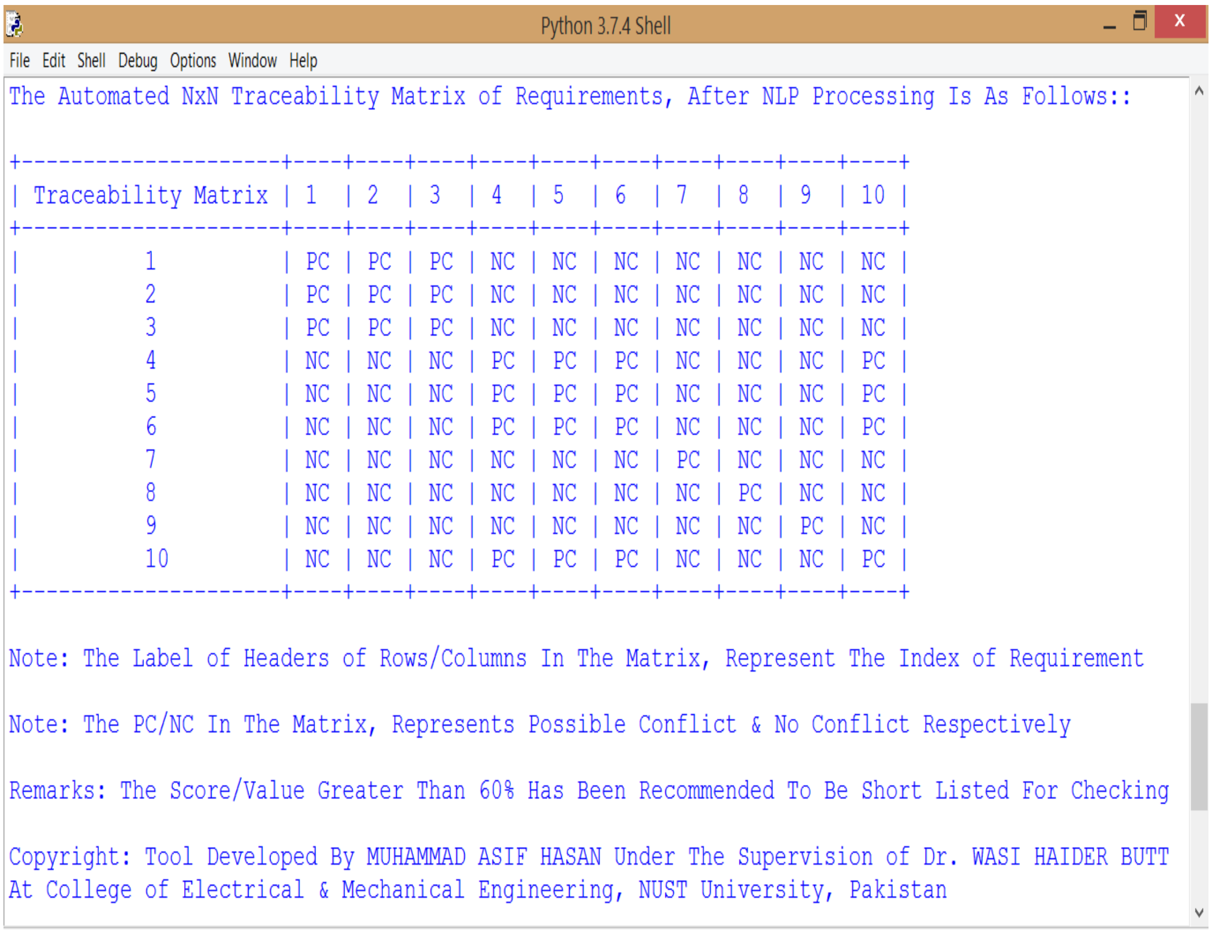
**Figure 4.29:** Code for creating another type of traceability matrix

**Output**

```
The Automated NxN Traceability Matrix of Requirements, After NLP Processing Is As Follows::


+--------------------+----+----+----+----+----+----+----+----+----+----+
| Traceability Matrix | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 |
+--------------------+----+----+----+----+----+----+----+----+----+----+
|         1          | PC | PC | PC | NC | NC | NC | NC | NC | NC | NC |
|         2          | PC | PC | PC | NC | NC | NC | NC | NC | NC | NC |
|         3          | PC | PC | PC | NC | NC | NC | NC | NC | NC | NC |
|         4          | NC | NC | NC | PC | PC | PC | NC | NC | NC | PC |
|         5          | NC | NC | NC | PC | PC | PC | NC | NC | NC | PC |
|         6          | NC | NC | NC | PC | PC | PC | NC | NC | NC | PC |
|         7          | NC | NC | NC | NC | NC | NC | PC | NC | NC | NC |
|         8          | NC | NC | NC | NC | NC | NC | NC | PC | NC | NC |
|         9          | NC | NC | NC | NC | NC | NC | NC | NC | PC | NC |
|         10         | NC | NC | NC | PC | PC | PC | NC | NC | NC | PC |
+--------------------+----+----+----+----+----+----+----+----+----+----+


Note: The Label of Headers of Rows/Columns In The Matrix, Represent The Index of
Requirement


Note: The PC/NC In The Matrix, Represents Possible Conflict & No Conflict Respectively
```

Remarks: The Score/Value Greater Than 60% Has Been Recommended To Be Short Listed For Checking

Copyright: Tool Developed By MUHAMMAD ASIF HASAN Under The Supervision of Dr. WASI HAIDER BUTT At College of Electrical & Mechanical Engineering, NUST University, Pakistan



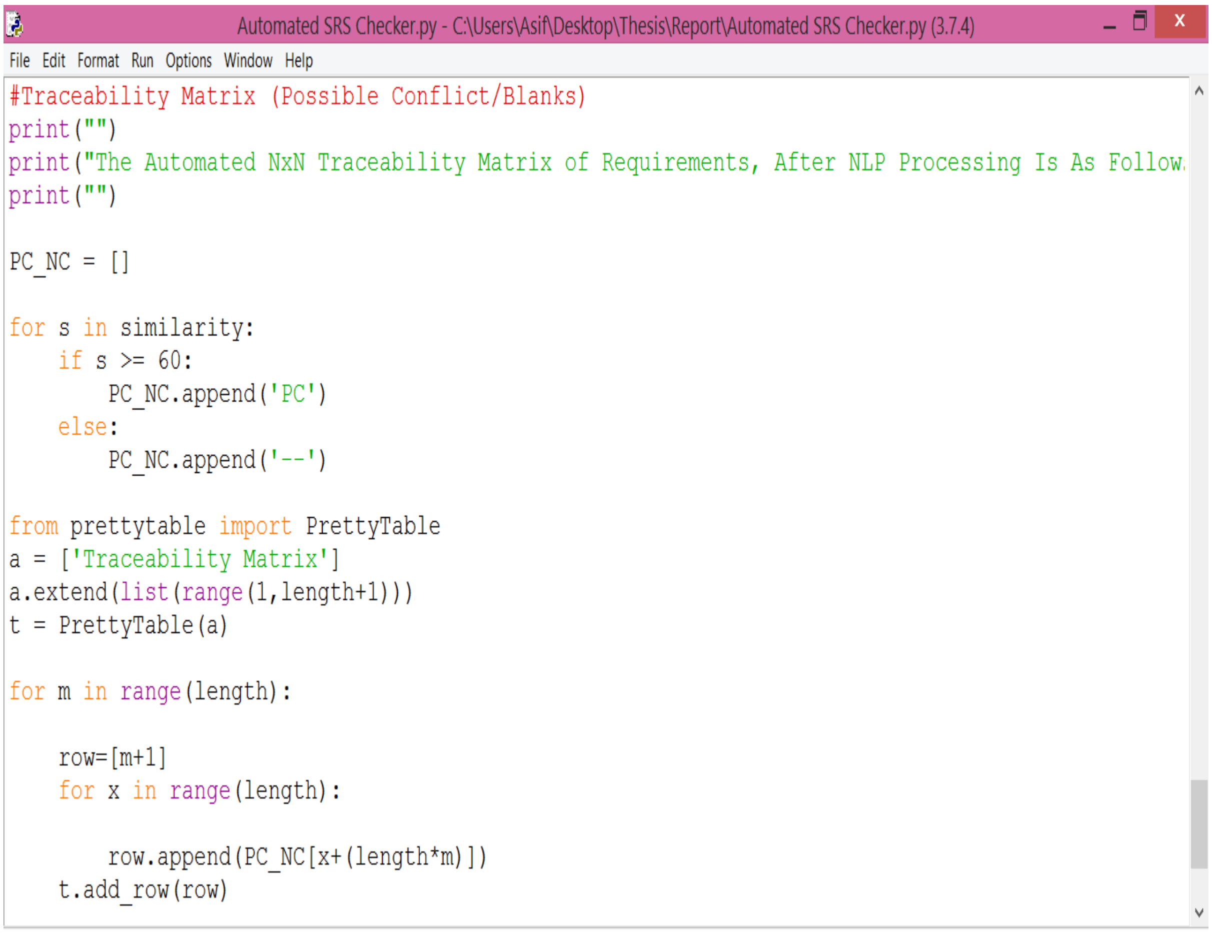
```
The Automated NxN Traceability Matrix of Requirements, After NLP Processing Is As Follows::

+---------------------+----+----+----+----+----+----+----+----+----+----+
| Traceability Matrix | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 |
+---------------------+----+----+----+----+----+----+----+----+----+----+
|          1          | PC | PC | PC | NC | NC | NC | NC | NC | NC | NC |
|          2          | PC | PC | PC | NC | NC | NC | NC | NC | NC | NC |
|          3          | PC | PC | PC | NC | NC | NC | NC | NC | NC | NC |
|          4          | NC | NC | NC | PC | PC | PC | NC | NC | NC | PC |
|          5          | NC | NC | NC | PC | PC | PC | NC | NC | NC | PC |
|          6          | NC | NC | NC | PC | PC | PC | NC | NC | NC | PC |
|          7          | NC | NC | NC | NC | NC | NC | PC | NC | NC | NC |
|          8          | NC | NC | NC | NC | NC | NC | NC | PC | NC | NC |
|          9          | NC | NC | NC | NC | NC | NC | NC | NC | PC | NC |
|          10         | NC | NC | NC | PC | PC | PC | NC | NC | NC | PC |
+---------------------+----+----+----+----+----+----+----+----+----+----+

Note: The Label of Headers of Rows/Columns In The Matrix, Represent The Index of Requirement

Note: The PC/NC In The Matrix, Represents Possible Conflict & No Conflict Respectively

Remarks: The Score/Value Greater Than 60% Has Been Recommended To Be Short Listed For Checking

Copyright: Tool Developed By MUHAMMAD ASIF HASAN Under The Supervision of Dr. WASI HAIDER BUTT
At College of Electrical & Mechanical Engineering, NUST University, Pakistan
```

**Figure 4.30:** Output on shell

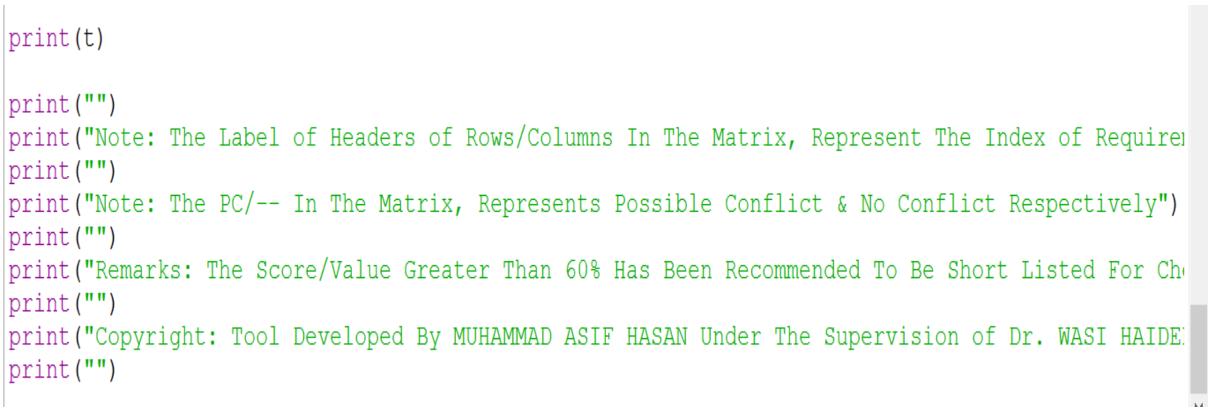
The code to create another type of traceability matrix was written as below.

**Python Algorithm**

```
#Traceability Matrix (Possible Conflict/Blanks)
print("")
print("The Automated NxN Traceability Matrix of Requirements, After NLP Processing Is As Follows::")
print("")

PC_NC = []

for s in similarity:
    if s >= 60:
        PC_NC.append('PC')
    else:
        PC_NC.append('--')

from prettytable import PrettyTable
a = ['Traceability Matrix']
a.extend(list(range(1,length+1)))
t = PrettyTable(a)

for m in range(length):

    row=[m+1]
    for x in range(length):

        row.append(PC_NC[x+(length*m)])
    t.add_row(row)

print(t)

print("")
print("Note: The Label of Headers of Rows/Columns In The Matrix, Represent The Index of
Requirement")
print("")
print("Note: The PC/-- In The Matrix, Represents Possible Conflict & No Conflict Respectively")
print("")
print("Remarks: The Score/Value Greater Than 60% Has Been Recommended To Be Short Listed For Checking")
print("")
print("Copyright: Tool Developed By MUHAMMAD ASIF HASAN Under The Supervision of Dr. WASI HAIDER BUTT At College of Electrical & Mechanical Engineering, NUST University, Pakistan")
print("")
```


Automated SRS Checker.py - C:\Users\Asif\Desktop\Thesis\Report\Automated SRS Checker.py (3.7.4)
File Edit Format Run Options Window Help

```
#Traceability Matrix (Possible Conflict/Blanks)
print("")
print("The Automated NxN Traceability Matrix of Requirements, After NLP Processing Is As Follow
print("")

PC_NC = []

for s in similarity:
    if s >= 60:
        PC_NC.append('PC')
    else:
        PC_NC.append('--')

from prettytable import PrettyTable
a = ['Traceability Matrix']
a.extend(list(range(1,length+1)))
t = PrettyTable(a)

for m in range(length):

    row=[m+1]
    for x in range(length):

        row.append(PC_NC[x+(length*m)])
    t.add_row(row)
```



```
print(t)

print("")
print("Note: The Label of Headers of Rows/Columns In The Matrix, Represent The Index of Requiren
print("")
print("Note: The PC/-- In The Matrix, Represents Possible Conflict & No Conflict Respectively")
print("")
print("Remarks: The Score/Value Greater Than 60% Has Been Recommended To Be Short Listed For Ch
print("")
print("Copyright: Tool Developed By MUHAMMAD ASIF HASAN Under The Supervision of Dr. WASI HAIDE
print("")
```

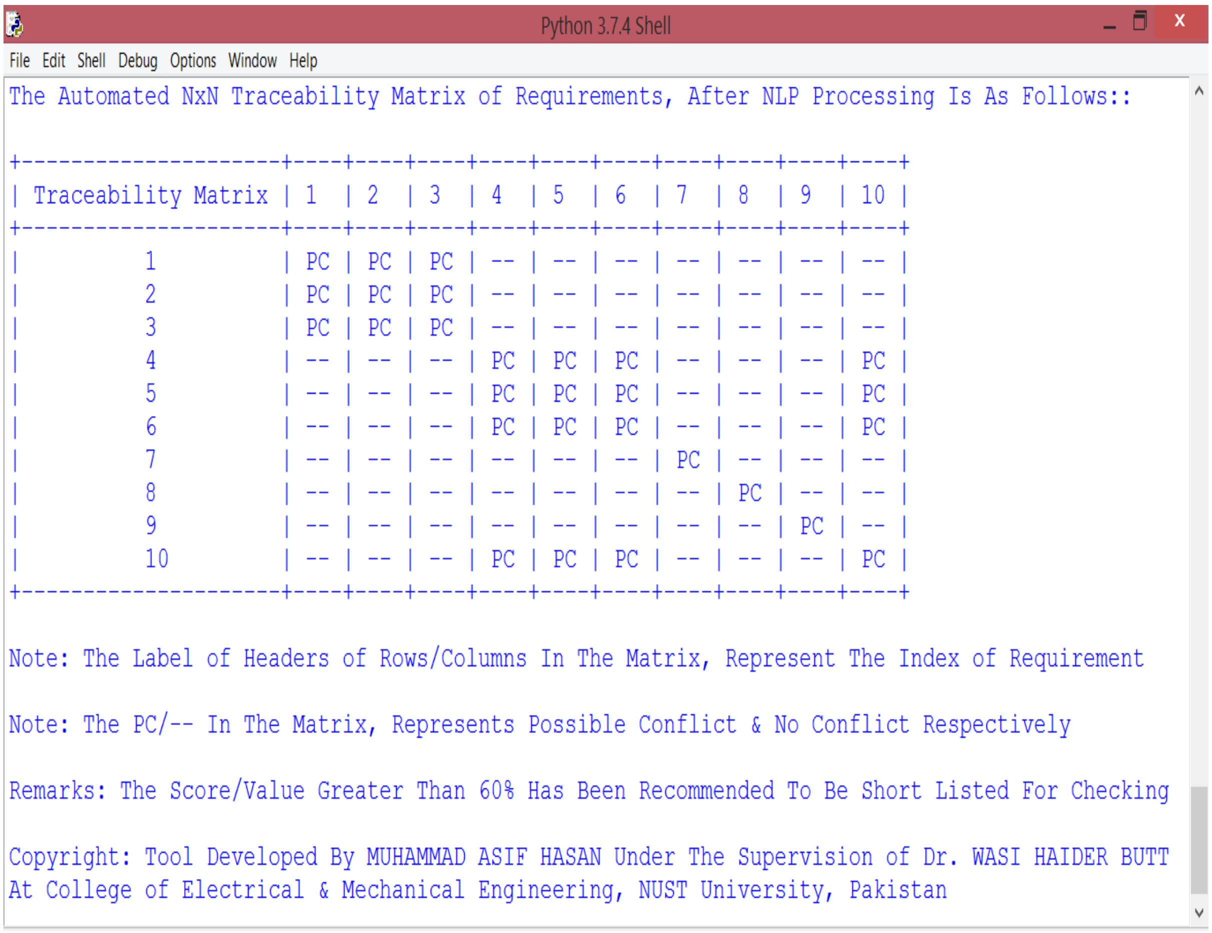
**Figure 4.31:** Code for creating another type of traceability matrix

**Output**

```
The Automated NxN Traceability Matrix of Requirements, After NLP Processing Is As Follows::


+------------------+----+----+----+----+----+----+----+----+----+----+
| Traceability Matrix | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 |
+------------------+----+----+----+----+----+----+----+----+----+----+
|       1        | PC | PC | PC | – | -- | -- | -- | -- | -- | -- |
|       2        | PC | PC | PC | – | -- | -- | -- | -- | -- | -- |
|       3        | PC | PC | PC | – | -- | -- | -- | -- | -- | -- |
|       4        | – | -- | -- | PC | PC | PC | -- | -- | -- | PC |
|       5        | – | -- | -- | PC | PC | PC | -- | -- | -- | PC |
|       6        | – | -- | -- | PC | PC | PC | -- | -- | -- | PC |
|       7        | – | -- | -- | – | -- | -- | PC | -- | -- | -- |
|       8        | – | -- | -- | – | -- | -- | -- | PC | -- | -- |
|       9        | – | -- | -- | – | -- | -- | -- | -- | PC | -- |
|       10        | -- | -- | -- | PC | PC | PC | -- | – | -- | PC |
+------------------+----+----+----+----+----+----+----+----+----+----+


Note: The Label of Headers of Rows/Columns In The Matrix, Represent The Index of
Requirement


Note: The PC/– In The Matrix, Represents Possible Conflict & No Conflict Respectively


Remarks: The Score/Value Greater Than 60% Has Been Recommended To Be Short Listed
```

For Checking

Copyright: Tool Developed By MUHAMMAD ASIF HASAN Under The Supervision of Dr. WASI HAIDER BUTT At College of Electrical & Mechanical Engineering, NUST University, Pakistan



```
The Automated NxN Traceability Matrix of Requirements, After NLP Processing Is As Follows::

+--------------------+----+----+----+----+----+----+----+----+----+----+
| Traceability Matrix | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 |
+--------------------+----+----+----+----+----+----+----+----+----+----+
|          1          | PC | PC | PC | -- | -- | -- | -- | -- | -- | -- |
|          2          | PC | PC | PC | -- | -- | -- | -- | -- | -- | -- |
|          3          | PC | PC | PC | -- | -- | -- | -- | -- | -- | -- |
|          4          | -- | -- | -- | PC | PC | PC | -- | -- | -- | PC |
|          5          | -- | -- | -- | PC | PC | PC | -- | -- | -- | PC |
|          6          | -- | -- | -- | PC | PC | PC | -- | -- | -- | PC |
|          7          | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- |
|          8          | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- |
|          9          | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- |
|          10         | -- | -- | -- | PC | PC | PC | -- | -- | -- | PC |
+--------------------+----+----+----+----+----+----+----+----+----+----+

Note: The Label of Headers of Rows/Columns In The Matrix, Represent The Index of Requirement

Note: The PC/-- In The Matrix, Represents Possible Conflict & No Conflict Respectively

Remarks: The Score/Value Greater Than 60% Has Been Recommended To Be Short Listed For Checking

Copyright: Tool Developed By MUHAMMAD ASIF HASAN Under The Supervision of Dr. WASI HAIDER BUTT
At College of Electrical & Mechanical Engineering, NUST University, Pakistan
```

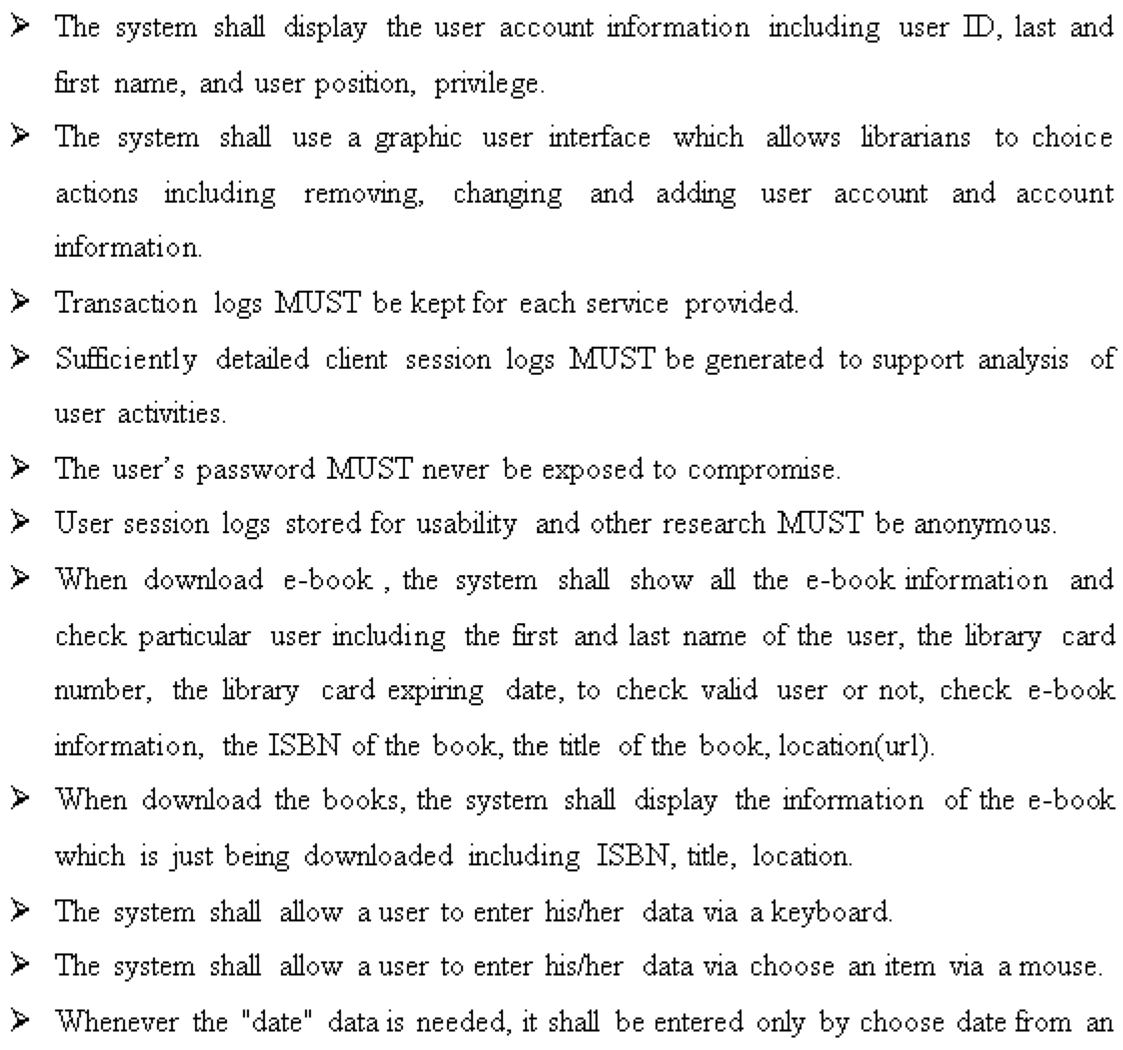
**Figure 4.32:** Output on shell

# CHAPTER 5: VERIFICATION AND VALIDATION

In this chapter the detailed results of the testing that was done over this tool to check its performance will be shown. Three case studies to test this tool were used. **Section 5.1** shows the case study based on Digital Library System – SRS. **Section 5.2** shows the case study based on Hotel Management System - SRS. **Section 5.3** shows the case study based on a custom made SRS out of the first two SRS already used, having some alterations done by the author to test this tool further.

## 5.1 Case Study No.1 (Digital Library System)

The first case study is based on the Software Requirements Specification for the DLS (Digital Library System) [1] which was used to test this tool. There were 60 requirements in this SRS that were input into this tool to be checked for consistency issues and poor practices. Following is the list of these requirements:-



- ➢ The system shall display the user account information including user ID, last and first name, and user position, privilege.
- ➢ The system shall use a graphic user interface which allows librarians to choice actions including removing, changing and adding user account and account information.
- ➢ Transaction logs MUST be kept for each service provided.
- ➢ Sufficiently detailed client session logs MUST be generated to support analysis of user activities.
- ➢ The user's password MUST never be exposed to compromise.
- ➢ User session logs stored for usability and other research MUST be anonymous.
- ➢ When download e-book , the system shall show all the e-book information and check particular user including the first and last name of the user, the library card number, the library card expiring date, to check valid user or not, check e-book information, the ISBN of the book, the title of the book, location(url).
- ➢ When download the books, the system shall display the information of the e-book which is just being downloaded including ISBN, title, location.
- ➢ The system shall allow a user to enter his/her data via a keyboard.
- ➢ The system shall allow a user to enter his/her data via choose an item via a mouse.
- ➢ Whenever the "date" data is needed, it shall be entered only by choose date from an

- The system shall allow the user to enter the library card number and ISBN both by typing or scanning.
- The system shall allow the user to enter book borrowing, recalling data as frequently as required.
- The system shall allow the user to attach notes to each account.
- The system shall allow the user to add or change information in an account including: last name, first name, user ID, user position, user privilege.
- The system shall allow the user to delete an entire account.
- The system shall allow the user typing in search criteria including book title, key word in title, ISBN, subject, category.
- The system shall allow the user choose language option which the searched book is used including English, Arabic.
- If the search result are a list of books, the system shall allow the user to choose any one of them to see the details.
- The system shall allow the user to add or change the record information including the category, the title, the ISBN, the publisher, the brief description of the book, the location in library, the purchase date, the price.
- The system shall allow the user to put "delete" for an existing ebook and specify the deleting reason.
- The system shall have a report feature that will allow the user to generate a report showing the information of a particular patron.
- The system shall have a report feature that will allow the user to generate a report showing the information of book purchase information in a period including the book titles, category, the author, the publisher, the price.
- It also shall give statistic data about the total number of books purchased, the money paid by category.
- The system shall be generate those reports to the display, a file or a printer which is linked to the system.
- The system shall be installed in a windows-NT network.
- The account management system shall only be used by managers or users with defined privileges.
- The Patron information report shall be generated by users who have librarian account.
- The book purchase report shall only be generated by managers or users with defined privileges.

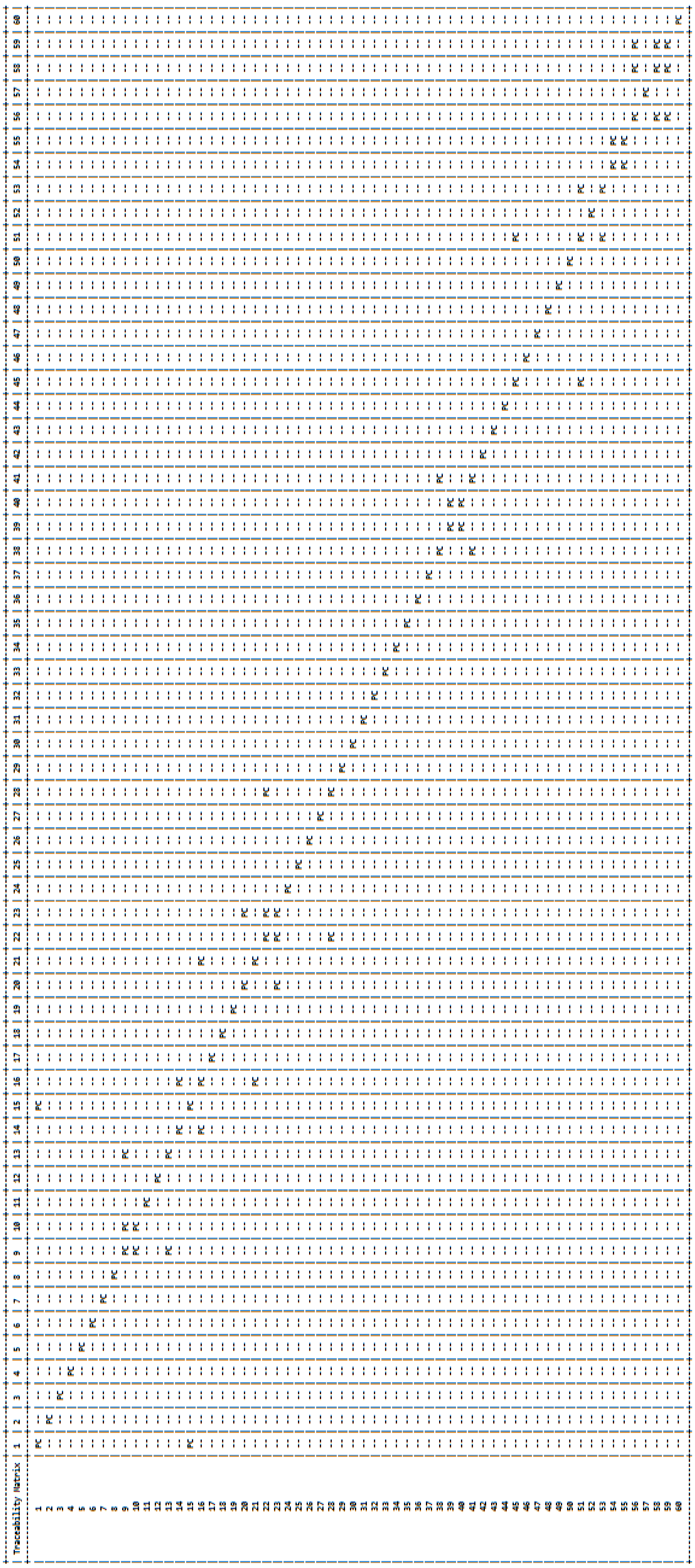
- Database update data shall be committed to the database only after the managers have approved.
- The system shall be recovered within 10 minutes if it is down.
- The system shall be recovered without intervention at user terminal if it is down.
- The system shall show appropriate messages at terminal when system is down.
- The system shall have 99% reliability during library operating hours.
- Scheduled down time after library operating hours shall not be more than 1 hour per day.
- The system shall generate error messages when the user attempts to enter invalid data.
- System must be able to extend to store and deliver new content media types.
- System must be able to extend to support synchronization of content media based on shared work/item structure.
- System MUST be able to extend to include music thesaurus in later versions.
- System MUST be able to extend support to MMTT components built in later versions.
- System MUST be able to extend to support data sharing between records.
- System MUST be able to extend to support more sophisticated bookmaking including additional context (e.g. size and configuration of viewer) and book marking of other record types.
- Users MUST be able to search for content using Work (title, composer name, subject heading, and key), Instantiation (performer names) and Container (title, publisher, editor, type, and format) attributes.
- Library Staff MUST be able to search on record creation and update dates.
- Simple free text search MUST be provided against like records representing containers that will be generated from the metadata stored in the data model.
- Searches over names and titles MUST support matching where diacritical markings are ignored.
- Users MUST always receive feedback on their search in the form of a result set that contains matching entries and/or information to further assist in the query.
- Users MUST be able to retrieve appropriate help in each interface.
- Users MUST be able to download in each e-book.
- Library Staff MUST be able to create/delete/edit records.
- Library Staff MUST be able to load new or update old content.
- Library Staff MUST be able to create/delete/edit structural declarations and bindings and associate them with records.

- Administrative users MUST be able to create, edit, and delete groups.
- Administrative users MUST be able to create, edit, and delete users.
- Authorization to play/view/update/add/delete media content MUST be controllable based on location and/or properties associated with the user and/or group.
- All completed descriptive metadata records within the system MUST be readable by all users.
- Authorization to update/add/delete metadata MUST be controllable based on location and/or properties associated with the user and/or group.
- Authorization to update/add/delete users and groups MUST be controllable based on location and/or properties associated with the user and/or group.
- Mechanisms for updating group membership information (course enrolment) MUST be provided to instructors.

### 5.1.1 Results

**Table 5-1:** Summary of Results

| | |
|---|---|
| **Total Number of Requirements** | **60** |
| **Total NxN Cases** | **3600** |
| **Shortlisted Cases By Tool** | **16** |
| **Efforts Now Required [16/((3600-60)/2)*100]** | **0.90%** |
| **Requirement Engineer's Time Saved** | **> 99.1%** |
| **Total Number of Inconsistent Cases In SRS** | **2** |
| **Number of Inconsistent Cases Detected By Tool** | **2** |
| **Number of Poor Practices Detected By Tool** | **12** |
| **Number of Neutral Cases Detected By Tool** | **2** |
| **Accuracy [(16-2)/16*100]** | **87.5%** |



| Traceability Matrix | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 2 | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 3 | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 4 | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 5 | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 6 | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 7 | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 8 | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 9 | -- | -- | -- | -- | -- | -- | -- | -- | PC | PC | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 10 | -- | -- | -- | -- | -- | -- | -- | -- | PC | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 11 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 12 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 13 | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 14 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 15 | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 16 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | PC | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 17 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 18 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 19 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 20 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 21 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 22 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | PC | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 23 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | PC | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 24 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 25 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 26 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 27 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 28 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 29 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 30 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 31 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 32 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 33 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 34 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 35 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 36 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 37 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 38 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 39 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 40 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 41 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 42 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 43 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 44 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 45 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 46 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 47 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 48 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 49 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 50 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 51 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | PC | -- | PC | -- | -- | -- | -- | -- | -- | -- |
| 52 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- | -- | -- | -- | -- | -- |
| 53 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | PC | -- | -- | -- | -- | -- | -- | -- |
| 54 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | PC | -- | -- | -- | -- | -- |
| 55 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | PC | -- | -- | -- | -- | -- |
| 56 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | PC | PC | -- |
| 57 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | -- | -- |
| 58 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | PC | PC | -- |
| 59 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC | -- | PC | PC | -- |
| 60 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | PC |

**Figure 5.1:** Traceability Matrix Generated By Tool

**Table 5-2:** Details of the Shortlisted Cases

| Shortlisted Case No. | Pair of Requirements Shortlisted For Checking | Comments |
|---|---|---|
| 1 | 'the system shall display the user account information including user id, last and first name, and user position, privilege.'<br><br>'the system shall allow the user to add or change information in an account including: last name, first name, user id, user position, user privilege.' | It's a poor practice |
| 2 | 'the system shall allow a user to enter his/her data via a keyboard.'<br><br>'the system shall allow a user to enter his/her data via choose an item via a mouse.' | It's a consistency issue |
| 3 | 'the system shall allow a user to enter his/her data via a keyboard.'<br><br>'the system shall allow the user to enter book borrowing, recalling data as frequently as required.' | It's a neutral case |
| 4 | 'the system shall allow the user to attach notes to each account.'<br><br>'the system shall allow the user to delete an entire account.' | It's a poor practice |
| 5 | 'the system shall allow the user to delete an entire account.'<br><br>'the system shall allow the user to put "delete" for an existing ebook and specify the deleting reason.' | It's a poor practice |
| 6 | 'the system shall allow the user to add or change the record information including the category, the title, the isbn, the publisher, the brief description of the book, the location in library, the purchase date, the price.'<br><br>'the system shall have a report feature that will allow the user to generate a report showing the information of book purchase | It's a neutral case |

| | | |
|---|---|---|
| | information in a period including the book titles, category, the author, the publisher, the price.' | |
| 7 | 'the system shall have a report feature that will allow the user to generate a report showing the information of a particular patron.'<br><br>'the system shall have a report feature that will allow the user to generate a report showing the information of book purchase information in a period including the book titles, category, the author, the publisher, the price.' | It's a poor practice |
| 8 | 'the system shall have a report feature that will allow the user to generate a report showing the information of a particular patron.'<br><br>'the patron information report shall be generated by users who have librarian account.' | It's a consistency issue |
| 9 | 'system must be able to extend to support synchronization of content media based on shared work/item structure.'<br><br>'system must be able to extend to support data sharing between records.' | It's a poor practice |
| 10 | 'system must be able to extend to include music thesaurus in later versions.'<br><br>'system must be able to extend support to mmtt components built in later versions.' | It's a poor practice |
| 11 | 'library staff must be able to search on record creation and update dates.'<br><br>'library staff must be able to create/delete/edit records.' | It's a poor practice |
| 12 | 'library staff must be able to create/delete/edit records.'<br><br>'library staff must be able to create/delete/edit structural declarations and bindings and associate them with records.' | It's a poor practice |

| | | |
|---|---|---|
| 13 | 'administrative users must be able to create, edit, and delete groups.'<br><br>'administrative users must be able to create, edit, and delete users.' | It's a poor practice |
| 14 | 'authorization to play/view/update/add/delete media content must be controllable based on location and/or properties associated with the user and/or group.'<br><br>'authorization to update/add/delete metadata must be controllable based on location and/or properties associated with the user and/or group.' | It's a poor practice |
| 15 | 'authorization to play/view/update/add/delete media content must be controllable based on location and/or properties associated with the user and/or group.'<br><br>'authorization to update/add/delete users and groups must be controllable based on location and/or properties associated with the user and/or group.' | It's a poor practice |
| 16 | 'authorization to update/add/delete metadata must be controllable based on location and/or properties associated with the user and/or group.'<br><br>'authorization to update/add/delete users and groups must be controllable based on location and/or properties associated with the user and/or group.' | It's a poor practice |

## 5.2 Case Study No.2 (Hotel Management System)

The second case study is based on the Software Requirements Specification for the Hotel Management System [2] prepared by Fenil Desani, which was used to test this tool. There were 53 requirements in this SRS that were input into this tool to be checked for consistency issues and poor practices. Following is the list of these requirements:-

- The system shall record reservations.
- The system shall record the customer's first name.
- The system shall record the customer's last name.
- The system shall record the number of occupants.
- The system shall record the room number.
- The system shall display the default room rate.
- The system shall allow the default room rate to be changed.
- The system shall require a comment to be entered, describing the reason for changing the default room rate.
- The system shall record the customer's phone number.
- The system shall display whether or not the room is guaranteed.
- The system shall generate a unique confirmation number for each reservation.
- The system shall automatically cancel non-guaranteed reservations if the customer has not provided their credit card number by 6:00 pm on the check-in date.
- The system shall record the expected check-in date and time.
- The system shall record the expected checkout date and time.
- The system shall check-in customers.
- The system shall allow reservations to be modified without having to reenter all the customer information.
- The system shall checkout customers.
- The system shall display the amount owed by the customer.
- To retrieve customer information the last name or room number shall be used.
- The system shall record that the room is empty.
- The system shall record the payment.
- The system shall record the payment type.
- The system shall charge the customer for an extra night if they checkout after 11:00 a.m.

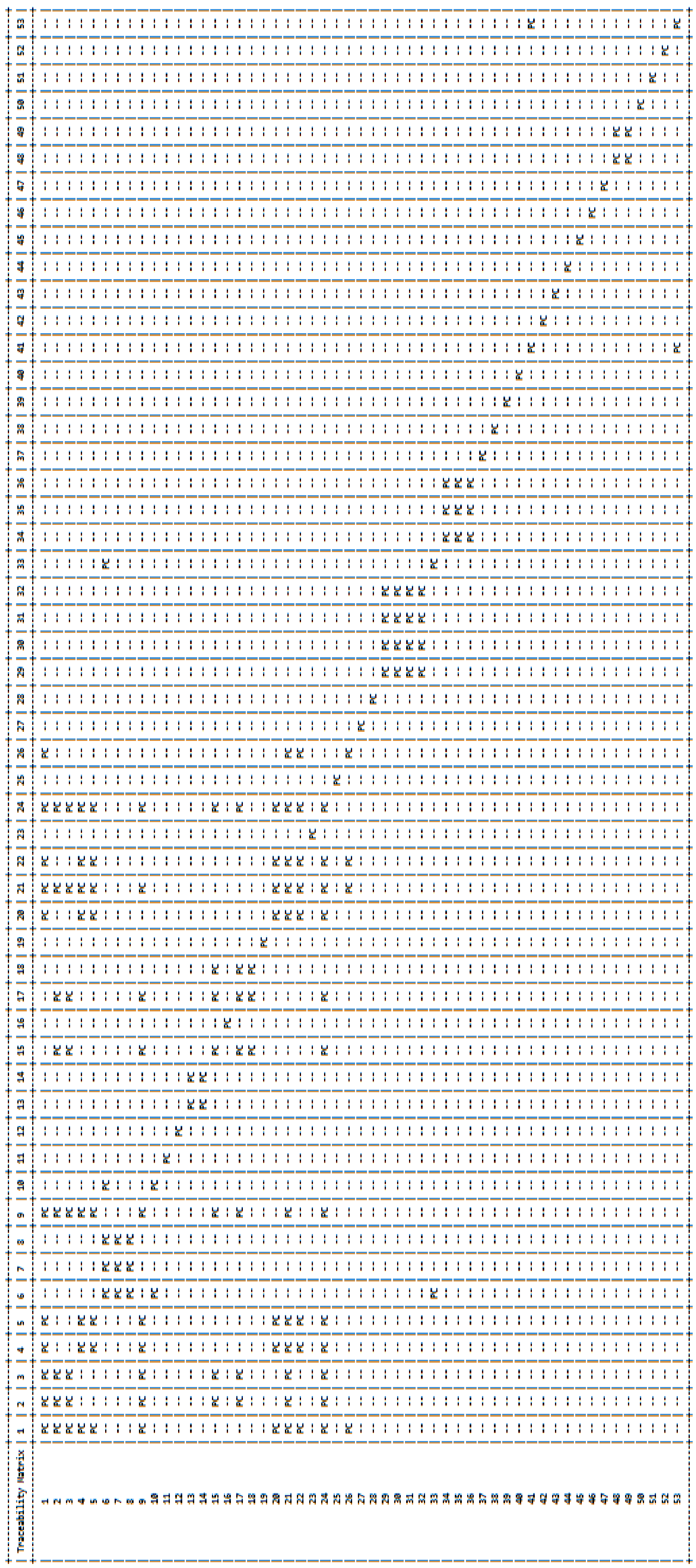
- The system shall mark guaranteed rooms as "must pay" after 6:00 pm on the check-in date.
- The system shall record customer feedback.
- The system shall track all meals purchased in the hotel (restaurant and room service).
- The system shall record payment and payment type for meals.
- The system shall bill the current room if payment is not made at time of service.
- The system shall accept reservations for the restaurant and room service.
- The system shall display the hotel occupancy for a specified period of time (days; including past, present, and future dates).
- The system shall display projected occupancy for a period of time (days).
- The system shall display room revenue for a specified period of time (days).
- The system shall display food revenue for a specified period of time (days).
- The system shall display an exception report, showing where default room and food prices have been overridden.
- The system shall allow for the addition of information, regarding rooms, rates, menu items, prices, and user profiles.
- The system shall allow for the deletion of information, regarding rooms, rates, menu items, prices, and user profiles.
- The system shall allow for the modification of information, regarding rooms, rates, menu items, prices, and user profiles.
- The system shall allow managers to assign user passwords.
- The load time for user interface screens shall take no longer than two seconds.
- The log in information shall be verified within five seconds.
- Queries shall return results within five seconds.
- The Hotel Management System shall be a stand-alone system running in a Windows environment.
- The system shall be developed using Java and an Access or Oracle database.
- There shall be consistency in variable names within the system.
- The graphical user interface shall have a consistent look and feel.
- Specify the factors required to establish the required reliability of the software system at time of delivery.
- The system shall be available during normal hotel operating hours.
- Customer Service Representatives and Managers will be able to log in to the Hotel Management System.

- Customer Service Representatives will have access to the Reservation/Booking and Food subsystems.
- Managers will have access to the Management subsystem as well as the Reservation/Booking and Food subsystems.
- Access to the various subsystems will be protected by a user log in screen that requires a user name and password.
- The Hotel Management System is being developed in Java. Java is an object oriented programming language and shall be easy to maintain.
- The Hotel Management System shall run in any Microsoft Windows environment that contains Java Runtime and the Microsoft Access database.

### 5.2.1 Results

**Table 5-3:** Summary of Results

| | |
|---|---|
| **Total Number of Requirements** | **53** |
| **Total NxN Cases** | **2809** |
| **Shortlisted Cases By Tool** | **66** |
| **Efforts Now Required [66/((2809-53)/2)*100]** | **4.79%** |
| **Requirement Engineer's Time Saved** | **> 95.21%** |
| **Total Number of Inconsistent Cases In SRS** | **4** |
| **Number of Inconsistent Cases Detected By Tool** | **4** |
| **Number of Poor Practices Detected By Tool** | **48** |
| **Number of Neutral Cases Detected By Tool** | **14** |
| **Accuracy [(66-14)/66*100]** | **78.79%** |



| Traceability Matrix | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | PC | PC | PC | PC | PC | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | PC | PC | PC | - | PC | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 2 | PC | PC | PC | - | - | - | - | - | PC | - | - | - | - | - | PC | - | PC | - | - | - | PC | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 3 | PC | PC | PC | - | - | - | - | - | PC | - | - | - | - | - | PC | - | PC | - | - | - | PC | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 4 | PC | - | - | PC | PC | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | PC | PC | PC | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 5 | PC | - | - | PC | PC | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | PC | PC | PC | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 6 | - | - | - | - | - | PC | PC | PC | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 7 | - | - | - | - | - | PC | PC | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 8 | - | - | - | - | - | PC | PC | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 9 | PC | PC | PC | PC | PC | - | - | - | PC | - | - | - | - | - | PC | - | PC | - | - | - | PC | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 10 | - | - | - | - | - | PC | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 11 | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 12 | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 13 | - | - | - | - | - | - | - | - | - | - | - | - | PC | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 14 | - | - | - | - | - | - | - | - | - | - | - | - | PC | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 15 | - | PC | PC | - | - | - | - | - | PC | - | - | - | - | - | PC | - | PC | PC | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 16 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 17 | - | PC | PC | - | - | - | - | - | PC | - | - | - | - | - | PC | - | PC | PC | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 18 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | PC | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 19 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 20 | PC | - | - | PC | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | PC | PC | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 21 | PC | PC | PC | PC | PC | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | PC | PC | PC | - | PC | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 22 | PC | - | - | PC | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | PC | PC | - | PC | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 23 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 24 | PC | PC | PC | PC | PC | - | - | - | PC | - | - | - | - | - | PC | - | PC | - | - | PC | PC | PC | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 25 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 26 | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | PC | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 27 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 28 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 29 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | PC | PC | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 30 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | PC | PC | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 31 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | PC | PC | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 32 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | PC | PC | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 33 | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 34 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | PC | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 35 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | PC | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 36 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | PC | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 37 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 38 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 39 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 40 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 41 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | PC |
| 42 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - |
| 43 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - |
| 44 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - |
| 45 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - |
| 46 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - |
| 47 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - |
| 48 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | PC | - | - | - | - |
| 49 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | PC | - | - | - | - |
| 50 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - |
| 51 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - |
| 52 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - |
| 53 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | PC |

**Figure 5.2:** Traceability Matrix Generated By Tool

**Table 5-4:** Details of the Shortlisted Cases

| Shortlisted Case No. | Pair of Requirements Shortlisted For Checking | Comments |
|---|---|---|
| 1 | 'the system shall record reservations.'<br><br>'the system shall record the customer's first name.' | It's a poor practice |
| 2 | 'the system shall record reservations.'<br><br>'the system shall record the customer's last name.' | It's a poor practice |
| 3 | 'the system shall record reservations.'<br><br>'the system shall record the number of occupants.' | It's a poor practice |
| 4 | 'the system shall record reservations.'<br><br>'the system shall record the room number.' | It's a poor practice |
| 5 | 'the system shall record reservations.'<br><br>'the system shall record the customer's phone number.' | It's a poor practice |
| 6 | 'the system shall record reservations.'<br><br>'the system shall record that the room is empty.' | It's a poor practice |
| 7 | 'the system shall record reservations.'<br><br>'the system shall record the payment.' | It's a poor practice |
| 8 | 'the system shall record reservations.'<br><br>'the system shall record the payment type.' | It's a poor practice |
| 9 | 'the system shall record reservations.'<br><br>'the system shall record customer feedback.' | It's a poor practice |
| 10 | 'the system shall record reservations.' | It's a poor |

| | | |
|---|---|---|
| | 'the system shall record payment and payment type for meals.' | practice |
| 11 | 'the system shall record the customer's first name.'<br><br>'the system shall record the customer's last name.' | It's a poor practice |
| 12 | 'the system shall record the customer's first name.'<br><br>'the system shall record the customer's phone number.' | It's a poor practice |
| 13 | 'the system shall record the customer's first name.'<br><br>'the system shall check-in customers.' | It's a neutral case |
| 14 | 'the system shall record the customer's first name.'<br><br>'the system shall checkout customers.' | It's a neutral case |
| 15 | 'the system shall record the customer's first name.'<br><br>'the system shall record the payment.' | It's a poor practice |
| 16 | 'the system shall record the customer's first name.'<br><br>'the system shall record customer feedback.' | It's a poor practice |
| 17 | 'the system shall record the customer's last name.'<br><br>'the system shall record the customer's phone number.' | It's a poor practice |
| 18 | 'the system shall record the customer's last name.'<br><br>'the system shall check-in customers.' | It's a neutral case |
| 19 | 'the system shall record the customer's last name.'<br><br>'the system shall checkout customers.' | It's a neutral case |
| 20 | 'the system shall record the customer's last name.'<br><br>'the system shall record the payment.' | It's a poor practice |

| | | |
|---|---|---|
| 21 | 'the system shall record the customer's last name.'<br><br>'the system shall record customer feedback.' | It's a poor practice |
| 22 | 'the system shall record the number of occupants.'<br><br>'the system shall record the room number.' | It's a poor practice |
| 23 | 'the system shall record the number of occupants.'<br><br>'the system shall record the customer's phone number.' | It's a poor practice |
| 24 | 'the system shall record the number of occupants.'<br><br>'the system shall record that the room is empty.' | It's a poor practice |
| 25 | 'the system shall record the number of occupants.'<br><br>'the system shall record the payment.' | It's a poor practice |
| 26 | 'the system shall record the number of occupants.'<br><br>'the system shall record the payment type.' | It's a poor practice |
| 27 | 'the system shall record the number of occupants.'<br><br>'the system shall record customer feedback.' | It's a poor practice |
| 28 | 'the system shall record the room number.'<br><br>'the system shall record the customer's phone number.' | It's a poor practice |
| 29 | 'the system shall record the room number.'<br><br>'the system shall record that the room is empty.' | It's a poor practice |
| 30 | 'the system shall record the room number.'<br><br>'the system shall record the payment.' | It's a poor practice |
| 31 | 'the system shall record the room number.'<br><br>'the system shall record the payment type.' | It's a poor practice |

| | | |
|---|---|---|
| 32 | 'the system shall record the room number.'<br><br>'the system shall record customer feedback.' | It's a poor practice |
| 33 | 'the system shall display the default room rate.'<br><br>'the system shall allow the default room rate to be changed.' | It's a neutral case |
| 34 | 'the system shall display the default room rate.'<br><br>'the system shall require a comment to be entered, describing the reason for changing the default room rate.' | It's a neutral case |
| 35 | 'the system shall display the default room rate.'<br><br>'the system shall display whether or not the room is guaranteed.' | It's a poor practice |
| 36 | 'the system shall display the default room rate.'<br><br>'the system shall display an exception report, showing where default room and food prices have been overridden.' | It's a neutral case |
| 37 | 'the system shall allow the default room rate to be changed.'<br><br>'the system shall require a comment to be entered, describing the reason for changing the default room rate.' | It's a consistency issue |
| 38 | 'the system shall record the customer's phone number.'<br><br>'the system shall check-in customers.' | It's a neutral case |
| 39 | 'the system shall record the customer's phone number.'<br><br>'the system shall checkout customers.' | It's a neutral case |
| 40 | 'the system shall record the customer's phone number.'<br><br>'the system shall record the payment.' | It's a poor practice |

| | | |
|---|---|---|
| 41 | 'the system shall record the customer's phone number.'<br><br>'the system shall record customer feedback.' | It's a poor practice |
| 42 | 'the system shall record the expected check-in date and time.'<br><br>'the system shall record the expected checkout date and time.' | It's a poor practice |
| 43 | 'the system shall check-in customers.'<br><br>'the system shall checkout customers.' | It's a poor practice |
| 44 | 'the system shall check-in customers.'<br><br>'the system shall display the amount owed by the customer.' | It's a neutral case |
| 45 | 'the system shall check-in customers.'<br><br>'the system shall record customer feedback.' | It's a neutral case |
| 46 | 'the system shall checkout customers.'<br><br>'the system shall display the amount owed by the customer.' | It's a neutral case |
| 47 | 'the system shall checkout customers.'<br><br>'the system shall record customer feedback.' | It's a neutral case |
| 48 | 'the system shall record that the room is empty.'<br><br>'the system shall record the payment.' | It's a poor practice |
| 49 | 'the system shall record that the room is empty.'<br><br>'the system shall record the payment type.' | It's a poor practice |
| 50 | 'the system shall record that the room is empty.'<br><br>'the system shall record customer feedback.' | It's a poor practice |

| | | |
|---|---|---|
| 51 | 'the system shall record the payment.'<br><br>'the system shall record the payment type.' | It's a poor practice |
| 52 | 'the system shall record the payment.'<br><br>'the system shall record customer feedback.' | It's a poor practice |
| 53 | 'the system shall record the payment.'<br><br>'the system shall record payment and payment type for meals.' | It's a consistency issue |
| 54 | 'the system shall record the payment type.'<br><br>'the system shall record customer feedback.' | It's a poor practice |
| 55 | 'the system shall record the payment type.'<br><br>'the system shall record payment and payment type for meals.' | It's a consistency issue |
| 56 | 'the system shall display the hotel occupancy for a specified period of time (days; including past, present, and future dates).'<br><br>'the system shall display projected occupancy for a period of time (days).' | It's a poor practice |
| 57 | 'the system shall display the hotel occupancy for a specified period of time (days; including past, present, and future dates).'<br><br>'the system shall display room revenue for a specified period of time (days).' | It's a poor practice |
| 58 | 'the system shall display the hotel occupancy for a specified period of time (days; including past, present, and future dates).'<br><br>'the system shall display food revenue for a specified period of time (days).' | It's a poor practice |
| 59 | 'the system shall display projected occupancy for a period of | It's a poor |

| | | |
|---|---|---|
| | (days).' 'the system shall display room revenue for a specified period of time (days).' | practice |
| 60 | 'the system shall display projected occupancy for a period of time (days).' 'the system shall display food revenue for a specified period of time (days).' | It's a poor practice |
| 61 | 'the system shall display room revenue for a specified period of time (days).' 'the system shall display food revenue for a specified period of time (days).' | It's a poor practice |
| 62 | 'the system shall allow for the addition of information, regarding rooms, rates, menu items, prices, and user profiles.' 'the system shall allow for the deletion of information, regarding rooms, rates, menu items, prices, and user profiles.' | It's a poor practice |
| 63 | 'the system shall allow for the addition of information, regarding rooms, rates, menu items, prices, and user profiles.' 'the system shall allow for the modification of information, regarding rooms, rates, menu items, prices, and user profiles.' | It's a poor practice |
| 64 | 'the system shall allow for the deletion of information, regarding rooms, rates, menu items, prices, and user profiles.' 'the system shall allow for the modification of information, regarding rooms, rates, menu items, prices, and user profiles.' | It's a poor practice |
| 65 | 'the hotel management system shall be a stand-alone system running in a windows environment.' 'the hotel management system shall run in any microsoft | It's a consistency issue |

| | | |
|---|---|---|
| | windows environment that contains java runtime and the microsoft access database.' | |
| 66 | 'customer service representatives will have access to the reservation/booking and food subsystems.'<br><br>'managers will have access to the management subsystem as well as the reservation/booking and food subsystems.' | It's a neutral case |

## 5.3 Case Study No.3 (Custom SRS with Alterations)

The third case study is based on the Software Requirements Specification taken from the first two SRS already used, to test this tool. Some alterations were done by the author, so that a thorough stress testing can be done of this tool. Author himself entered a couple of consistency issues in to the requirements, to see if they can be successfully detected by this tool or not. There were 47 requirements in this SRS that were input into this tool to be checked for consistency issues and poor practices. Following is the list of these requirements:-

- The system shall use a graphic user interface which allows librarians to choice actions including removing, changing and adding user account and account information.
- Transaction logs must be kept for each service provided.
- Sufficiently detailed client session logs must be generated to support analysis of user activities.
- The user's password must never be exposed to compromise.
- User session logs stored for usability and other research must be anonymous.
- When download e-book , the system shall show all the e-book information and check particular user including the first and last name of the user, the library card number, the library card expiring date, to check valid user or not, check e-book information, the isbn of the book, the title of the book, location(url).
- When download the books, the system shall display the information of the e-book which is just being downloaded including isbn, title, location.
- Whenever the "date" data is needed, it shall be entered only by choose date from an online calendar.
- The system shall allow the user to enter the library card number and isbn both by typing or scanning.
- The system shall allow the user typing in search criteria including book title, key word in title, isbn, subject, category.
- The system shall allow the user choose language option which the searched book is used including english, arabic.
- If the search result are a list of books, the system shall allow the user to choose any one of them to see the details.

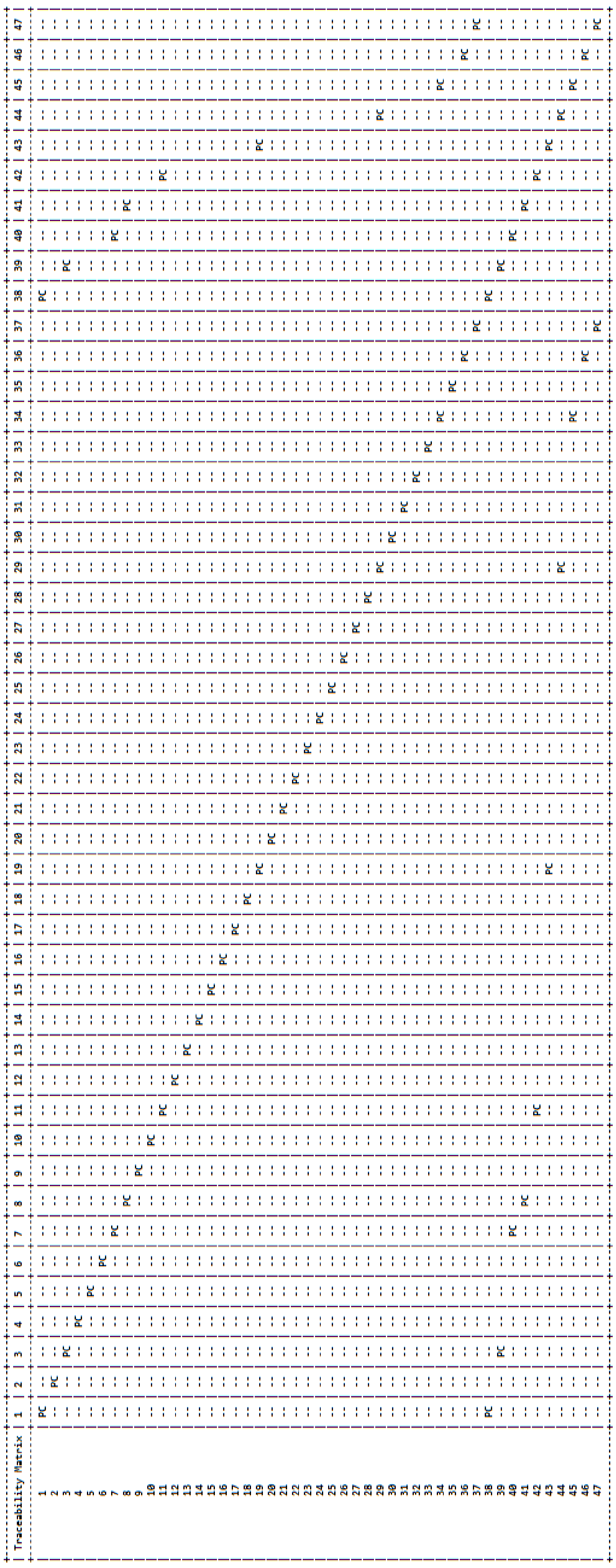
- It also shall give statistic data about the total number of books purchased, the money paid by category.
- The system shall be generate those reports to the display, a file or a printer which is linked to the system.
- The system shall be installed in a windows-nt network.
- The account management system shall only be used by managers or users with defined privileges.
- The book purchase report shall only be generated by managers or users with defined privileges.
- Database update data shall be committed to the database only after the managers have approved.
- The system shall be recovered within 10 minutes if it is down.
- The system shall be recovered without intervention at user terminal if it is down.
- The system shall show appropriate messages at terminal when system is down.
- The system shall have 99% reliability during library operating hours.
- Scheduled down time after library operating hours shall not be more than 1 hour per day.
- The system shall generate error messages when the user attempts to enter invalid data.
- System must be able to extend to store and deliver new content media types.
- System must be able to extend to support more sophisticated bookmaking including additional context (e.g. size and configuration of viewer) and book marking of other record types.
- Users must be able to search for content using work (title, composer name, subject heading, and key), instantiation (performer names) and container (title, publisher, editor, type, and format) attributes.
- Library staff must be able to search on record creation and update dates.
- Simple free text search must be provided against like records representing containers that will be generated from the metadata stored in the data model.
- Searches over names and titles must support matching where diacritical markings are ignored.
- Users must always receive feedback on their search in the form of a result set that contains matching entries and/or information to further assist in the query.
- Users must be able to retrieve appropriate help in each interface.
- Users must be able to download in each e-book.

- Library staff must be able to load new or update old content.
- All completed descriptive metadata records within the system must be readable by all users.
- Mechanisms for updating group membership information (course enrolment) must be provided to instructors.
- The system shall use a graphic user interface which should deny librarians to remove and add user account and account information.
- Client session logs to support analysis of user activities must never be generated during weekends.
- When download the books, the system shall hide the information of the e-book which is just being downloaded including its title.
- Whenever the "date" data is needed, it can be manually entered by keyboard.
- The system shall allow the user choose language option which the searched book is used including english and french, except arabic.
- The system shall be recovered not more than 15 minutes, if it is down.
- Library staff must be able to search on record creation but should not be able to update dates.
- Users must be able to download in each e-book except university published thesis.
- Only descriptive metadata records of students within the system must be readable by all users.
- Mechanisms for updating group membership information (course enrolment) must be provided to students as well.

### 5.3.1 Results

**Table 5-5:** Summary of Results

| | |
|---|---|
| **Total Number of Requirements** | **47** |
| **Total NxN Cases** | **2209** |
| **Shortlisted Cases By Tool** | **10** |
| **Efforts Now Required [10/((2209-47)/2)*100]** | **0.93%** |
| **Requirement Engineer's Time Saved** | **> 99.07%** |
| **Total Number of Inconsistent Cases In SRS** | **9** |
| **Number of Inconsistent Cases Detected By Tool** | **9** |
| **Number of Poor Practices Detected By Tool** | **1** |
| **Number of Neutral Cases Detected By Tool** | **0** |
| **Accuracy [(10-0)/10*100]** | **100%** |



| Traceability Matrix | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - |
| 2 | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 3 | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - |
| 4 | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 5 | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 6 | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 7 | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - |
| 8 | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - |
| 9 | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 10 | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 11 | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - |
| 12 | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 13 | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 14 | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 15 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 16 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 17 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 18 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 19 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - |
| 20 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 21 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 22 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 23 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 24 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 25 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 26 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 27 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 28 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 29 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - |
| 30 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 31 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 32 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 33 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 34 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | PC | - | - |
| 35 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - |
| 36 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | PC | - |
| 37 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | PC |
| 38 | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - |
| 39 | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - |
| 40 | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - |
| 41 | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - |
| 42 | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - |
| 43 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - |
| 44 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - |
| 45 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | - | PC | - | - |
| 46 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | PC | - |
| 47 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | PC | - | - | - | - | - | - | - | - | - | PC |

**Figure 5.3:** Traceability Matrix Generated By Tool

**Table 5-6:** Details of the Shortlisted Cases

| Shortlisted Case No. | Pair of Requirements Shortlisted For Checking | Comments |
|---|---|---|
| 1 | 'the system shall use a graphic user interface which allows librarians to choice actions including removing, changing and adding user account and account information.'<br><br>'the system shall use a graphic user interface which should deny librarians to remove and add user account and account information.' | It's a consistency issue |
| 2 | 'sufficiently detailed client session logs must be generated to support analysis of user activities.'<br><br>'client session logs to support analysis of user activities must never be generated during weekends.' | It's a consistency issue |
| 3 | 'when download the books, the system shall display the information of the e-book which is just being downloaded including isbn, title, location.'<br><br>'when download the books, the system shall hide the information of the e-book which is just being downloaded including its title.' | It's a consistency issue |
| 4 | 'whenever the "date" data is needed, it shall be entered only by choose date from an online calendar.'<br><br>'whenever the "date" data is needed, it can be manually entered by keyboard.' | It's a consistency issue |
| 5 | 'the system shall allow the user choose language option which the searched book is used including english, arabic.'<br><br>'the system shall allow the user choose language option which the searched book is used including english and french, except arabic.' | It's a consistency issue |

|  |  |  |
|---|---|---|
| 6 | 'the system shall be recovered within 10 minutes if it is down.'<br><br>'the system shall be recovered not more than 15 minutes, if it is down.' | It's a consistency issue |
| 7 | 'library staff must be able to search on record creation and update dates.'<br><br>'library staff must be able to search on record creation but should not be able to update dates.' | It's a consistency issue |
| 8 | 'users must be able to download in each e-book.'<br><br>'users must be able to download in each e-book except university published thesis.' | It's a consistency issue |
| 9 | 'all completed descriptive metadata records within the system must be readable by all users.'<br><br>'only descriptive metadata records of students within the system must be readable by all users.' | It's a consistency issue |
| 10 | 'mechanisms for updating group membership information (course enrolment) must be provided to instructors.'<br><br>'mechanisms for updating group membership information (course enrolment) must be provided to students as well.' | It's a poor practice |

# CHAPTER 6: DISCUSSION AND LIMITATION

In this chapter the Section 6.1 does a discussion on the use of NLP techniques along with the concepts of cosine similarity to provide a solution to facilitate software requirement engineer in detecting conflicts in SRS requirements. Section 6.2 discusses about the limitations of this methodology and tool, and possible ideas to provide solution to cater them.

## 6.1 Discussion

A category of artificial intelligence that make computers understand, interpret and manipulate natural language is called Natural language processing (NLP). This branch has to be drawn out from many disciplines which includes computer science and computational linguistics. It is filling the gap between human language and computer understanding.

This technology is gaining advancement because of an increase in the interest of human-to-machine communications. Also because now a days big data, powerful computing and enhanced algorithms are also getting available and cheaper.

As a person, the speaking and written is done in some kind of natural language like English, Spanish or Chinese. But computers understand machine code or machine language. This language not understandable by people at large. Communication happen not with words but through a lot of 0’s and 1’s that produce logical actions at computer device level.

Computers can now communicate with people in their own language and scale other language-related tasks by the help of this technology. As an example to take, this all makes devices to read text, hear speech, interpret it and also even measure sentiment.

Computer of today can process more language-based information than humans, without fatigue and in a consistent, unbiased way as well. From medical records to social media, a huge amount of unstructured data is being created every day. So an automation will be critical to fully process text and speech data fast and efficiently.

Specifications of multiple stakeholders are often detailed into use case models in object-oriented software development. This will repalce the static domain model by dynamic and functional specifications. These specifications are analyzed and integrated to create a consistent overall SRS. Conflicts between requirements of different parties can cause iterations of the model to happen with time. Conflicts are quite difficult to find out because there is a diversity, incompleteness, and informal nature of functional and dynamic specifications. Based on logic, formal methodologies to requirements engineering, can cater these issues. But a requirement of highly specialized experts of this field are required, to write and reason about such requirements.

A quantitative measurement of semantics between two non-zero vectors of an inner product space is called Cosine similarity. This measure the cosine of the angle between the vectors. The cosine of Zero degrees is 1, and it is less than 1 for any angle in the interval [0, 90] degrees. Two vectors with the same orientation have a cosine similarity of 1 and two vectors oriented at 90° relative to each other have a similarity of 0. This gives an idea of orientation and not magnitude. So, two vectors diametrically opposed to each other will have a similarity of -1, independent of their magnitude. In positive space cosine similarity is usually used where the value is in bunds of [0,1]. Its name has a derivation from the term "direction cosine". In direction oscine the unit vectors are maximally "similar" if they're parallel to each other and maximally "dissimilar" if they're orthogonal (perpendicular) to each other. Hence this is different from the cosine similarity in which the unity (maximum value) is when the segments subtend a zero angle and zero (uncorrelated) is when the segments are perpendicular.

The cosine similarity is most often used in high-dimensional positive spaces. As an example to take, in data retrieval and text mining, each word is hypothetical assigned a different dimension and a content is characterized by a vector where the value in each dimension relates to the number of times the word appears in the content. An important measure of how similar two requirements are likely to be in terms of their semantics are revealed by Cosine similarity. To measure cohesion within clusters in the area of data mining can also be done this methodology.

For sparse vectors, cosine similarity is very advantageous because of its simplicity. Only the non-zero dimensions are need to be focused in this scenario. Some word names for it are Orchini similarity and the Tucker coefficient of congruence. In Orchini similarity the cosine similarity is then applied to binary data being considered.

By utilizing the Euclidean dot product formula, cosine of the two non-zero vectors can be calculated:

$$\mathbf{A} \cdot \mathbf{B} = \|\mathbf{A}\| \, \|\mathbf{B}\| \cos\theta$$

Given two vectors of terms, A and B, the cosine similarity, cos(θ), is shown using a dot product and magnitude as follows:

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum\limits_{i=1}^{n} A_i B_i}{\sqrt{\sum\limits_{i=1}^{n} A_i^2} \sqrt{\sum\limits_{i=1}^{n} B_i^2}},$$

where $A_i$ and $B_i$ are components of vector A and B respectively.

The results of cosine similarity rage from -1 to 0 to 1. If it's −1 it means exactly opposite. If it's 0 it means orthogonal or decorrelation. If it's 1 it means exactly the same. While in-between values show intermediate similarity or dissimilarity.

To address the cumbersome job and demanding efforts of requirement engineer to detect problems in the requirements of SRS, concepts of NLP techniques and cosine semantic similarity were used to develop a fully automated tool in Python. This helps to find inconsistency issues in SRS, efficiently and in very little time. The tool further using the concepts of Traceability Matrix generates three types of such tables to show results in a professional and tabulated way. This greatly saves the crucial time of engineer and also reduces the cost of this step of software development life cycle for the software house, ultimately reducing the cost of software project overall.

## 6.2 Limitation

Those features of design or methodology that motivated the understanding of the findings from the research, are covered in the limitations of a work. The limitations on generalizability, applications to practice, and/or utility of findings are covered in this area. The result of the options in which it was originally chosen to create the work or the method used to create internal and external validity are also covered. The outcomes of unknown challenges that happened during the study are also part of this area.

One should always appreciate a study's limitations. It is good if the researcher recognize and acknowledge his/her study's limitations than to have them highlighted out by the supervisor.

It is an phase to make suggestions for future research work if the study's limitations are appreciated. By connecting the study's limitations to future work of research, the ways to elaborate the unanswered questions may become more pronounced because of the work.

This also shows and gives one with an opportunity to show that researcher have thought deeply about the research domain and issues. He/she has also understood the relevant literature published about it in depth. It also shows that one has correctly analyzed the methods chosen for studying the problem domain. The goal of research processes is not just to discover new information but also to face assumptions and find what is not known.

Detailing limitations is greatly a subjective process. One has to calculate the impact of those limitations on the study. It's a poor practice to just write about the weaknesses and the magnitude of a study's limitations. If it is done, it will reduce or eliminate the authenticity of your research because it leaves the reader wondering thoughts. He/she might be thinking whether, or in what ways, limitation(s) in your work may have affected the outcomes and conclusions. Critical, overall appraisal and interpretation of limitations is essential.

So, although this tool performs quite well, but there are always some limitations in every work. It can noticed that this tool can process effectively if the requirement is of one sentence. If a requirement is composed of multiple sentences, it will process them all individually rather than as a single requirement. Although this doesn't affect the performance of this tool and will still provide the requirement engineer with correct results, but still that is a technical limitation of this tool.

To address this limitation, a solution for it has been proposed. The solution is that if a requirement is composed of multiple sentences, replace the full stops with commas. This will make this tool to recognize the entire requirement paragraph as one sentence and perform correct technically.

# CHAPTER 7: FUTURE WORK

In this chapter the Section 7.1 gives a short conclusion to sum up the discussion of this work. This section further discusses about possible future works and upgradations that can be done over this tool, to provide an opportunity for improvement for future research.

## 7.1 Conclusion and Future Work

Natural Language Processing (NLP) has numerous advantages. To enhance the efficiency of documentation processes and improve the accuracy of documentation, NLP can be involved by companies. It can also help companies to identify the most pertinent data from large databases. As an example, with the use of natural language processing in hospitals, a specific diagnosis from a doctor's unstructured notes can be pulled out and allocated a billing code.

To check one's own emotional state, NLP can also be a helpful solution. An electronic therapist that communicates with clients, is Woebot. It connects by Facebook Messenger chatbot or through a stand-alone application available. Obviously there is no high-level sentiment analysis being done in this case. However, Woebot is tracking for depression and anxiety states of one. It searches for words that may reveal users face as an emergency situation.

Sentiment analysis also aids us to know society as well. Periscopic has utilized NLP with visual recognition to develop the Trump-Emoticoaster. This is a data engine that processes language and facial expressions to detect President Donald Trump's emotional state.

To prevent school shootings, same technology can also be utilized. Researchers have analyzed 2 million Tweets posted by 9,000 at-risk youth at Columbia University for deal such problems. It tries to find the answer to one query: How does language change as a teen comes closer and closer to getting violent?

Dr. Desmond Patton is a program director who mentions that "Problematic content can evolve over time". When a help is needed, at-risk youth getting closer to the brink utilizes language expressions. Flagging problematic emotional states using natural language processing can help the social workers to intervene and act on time.

Likewise Periscopic, sentiment analysis has been utilized with image recognition to increase accuracy at Columbia. Computer vision can break down images attached to the Tweets to apply machine learning steps as told by Patton. This can be merged together with the language to tell the actual emotionality of an image. Is this image about threats? Is this image about grief? What else is happening in an image that helps us understand more complexly? Columbia program goals to prevent gang violence as well in addition to school shootings.

Job recruiters can utilize natural language processing to aid them sort through resumes, attract diverse candidates, and hire more qualified workforce. To keep irrelevant emails out of the inbox, spam detection tools utilizes NLP techniques. Outlook and Gmail also utilize it to sort messages from certain people into folders that are created.

Sentiment analysis tools greatly aid companies to quickly check Tweets. It can reveal that whether they are good or bad so they can triage customer concerns. But it just doesn't only process words on social media but also breaks down the semantics in which they appear. Skye Morét who is a data visualizer at analysis firm Periscopic tells that just 30 percent of English words are positive. Rest of the words are neutral or negative. Businesses are greatly aided by NLP to completely understand a post. What's the consumer emotion behind those neutral words?

In past, natural language processing was utilized by corporations to tag feedback as positive or negative. A senior vice president of social and innovation at Fleishman Hillard, Ryan Smith tells that application of nowadays identify more precise emotions, like sadness, anger, and fear than before.

So, it can be seen that there is so much that needs and can be done in this field. Just some areas of this field to facilitate a software requirement engineer to process the SRSs in an efficiently way were utilized, and this saves time and errors. There's a lot of future work that can be done in this tool. A couple of works that are possible to extend this work has been listed as below:-

❖ At the moment it can be noticed that this tool require a software requirement engineer to enter all the requirements into tool by copying them from source. An extension module that is able to read the SRS and input all the requirements from it in to the tool can be a one possible extension possible.

❖ At the moment it can be noticed that this tool is not very GUI intensive. It also shows the traceability matrix within the default python shell window. An extension to this tool to make this tool graphically intensive can make it even more user friendly.

# REFERENCES

[1] Software Requirement Specification Document
URL, http://repository.sustech.edu/bitstream/handle/123456789/7384/chapter%203.pdf?sequence=6&isAllowed=y

[2] Software Requirement Specification Document
URL, https://vdocuments.mx/srs-for-hotel-management-system.html

[3] Agirre, E., Edmonds, P.: Word Sense Disambiguation: Algorithms and Applications, vol. 33. Springer, Berlin (2007).

[4] Ambriola, V., Gervasi, V.: On the systematic analysis of natural language requirements with CIRCE. ASE 13(1), 107–167 (2006).

[5] Arora, C., Sabetzadeh, M., Briand, L., Zimmer, F.: Automated checking of conformance to requirements templates using natural language processing. TSE 41(10), 944–968 (2015).

[6] Dragos, V., Detection of contradictions by relation matching and uncertainty assessment. Procedia Computer Science, 2017. 112: p. 71-80.

[7] Avramidis, E.: Rankeval: open tool for evaluation of machine-learned ranking. Prague Bull. Math. Linguist. 100, 63–72 (2013).

[8] Blei, D.M.: Probabilistic topic models. Commun. ACM 55(4), 77–84 (2012).

[9] Matsumoto, Y., S. Shirai, and A. Ohnishi, A Method for Verifying Non-Functional Requirements. Procedia Computer Science, 2017. 112: p. 157-166.

[10] Malhotra, R., et al. Analyzing and evaluating security features in software requirements. in 2016 International Conference on Innovation and Challenges in Cyber Security (ICICCS-INBUSH). 2016.

[11] Nikora, A.P. and G. Balcom. Automated Identification of LTL Patterns in Natural Language Requirements. in 2009 20th International Symposium on Software Reliability Engineering. 2009.

[12] Ott, D. Defects in natural language requirement specifications at Mercedes-Benz: An investigation using a combination of legacy data and expert opinion. in 2012 20th IEEE International Requirements Engineering Conference (RE). 2012.

[13] Khtira, A., A. Benlarabi, and B.E. Asri. Detecting feature duplication in natural language specifications when evolving software product lines. in 2015 International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE). 2015.

[14] Nguyen, T.H., J. Grundy, and M. Almorsy. GUITAR: An ontology-based automated requirements analysis tool. in 2014 IEEE 22nd International Requirements Engineering Conference (RE). 2014.

[15] Mezghani, M., J. Kang, and F. Sèdes. Industrial Requirements Classification for Redundancy and Inconsistency Detection in SEMIOS. in 2018 IEEE 26th International Requirements Engineering Conference (RE). 2018.

[16] Herrera, J., et al. The Revealing Crosscutting Concerns in Textual Requirements Documents: An Exploratory Study with Industry Systems. in 2012 26th Brazilian Symposium on Software Engineering. 2012.

[17] Sandhu, G. and S. Sikka. State-of-art practices to detect inconsistencies and ambiguities from software requirements. in International Conference on Computing, Communication & Automation. 2015.

[18] Arunthavanathan, A., et al. Support for traceability management of software artefacts using Natural Language Processing. in 2016 Moratuwa Engineering Research Conference (MERCon). 2016.

[19] Camacho-Collados, J., Pilehvar, T.: From word to sense embeddings: a survey on vector representations of meaning. arXiv preprint arXiv:1805.04032 (2018)

[20] Ferrari, A. and Esuli, A., 2019. An NLP approach for cross-domain ambiguity detection in requirements engineering. Automated Software Engineering, 26(3), pp.559-598.

[21] Phalnikar, R. and D. Jinwala, Analysis of Conflicting User Requirements in Web Applications Using Graph Transformation. SIGSOFT Softw. Eng. Notes, 2015. 40(2): p. 1-7.

[22] Sousa, T.C.d., et al., Automatic analysis of requirements consistency with the B method. SIGSOFT Softw. Eng. Notes, 2010. 35(2): p. 1-4.

[23] Perera, S., et al., Challenges in understanding clinical notes: why NLP engines fall short and where background knowledge can help, in Proceedings of the 2013 international workshop on Data management &#38; analytics for healthcare. 2013, ACM: San Francisco, California, USA. p. 21-26.

[24] Ahsan, I., et al., A comprehensive investigation of natural language processing techniques and tools to generate automated test cases. 2017. p. 132:1-132:10.

[25] Sharma, V.S., R.R. Ramnani, and S. Sengupta, A framework for identifying and analyzing non-functional requirements from text, in Proceedings of the 4th International

Workshop on Twin Peaks of Requirements and Architecture. 2014, ACM: Hyderabad, India. p. 1-8.

[26] Iren, D. and H.A. Reijers, Leveraging business process improvement with natural language processing and organizational semantic knowledge, in Proceedings of the 2017 International Conference on Software and System Process. 2017, ACM: Paris, France. p. 100-108.

[27] Sateli, B., E. Angius, and R. Witte. ReqWiki Approach for Collaborative Software Requirements Engineering with Integrated Text Analysis Support. in 2013 IEEE 37th Annual Computer Software and Applications Conference. 2013.

[28] Sengupta, S., et al., Verb-based Semantic Modelling and Analysis of Textual Requirements, in Proceedings of the 8th India Software Engineering Conference. 2015, ACM: Bangalore, India. p. 30-39.

[29] Casamayor, A., Godoy, D., Campo, M.: Functional grouping of natural language requirements for assistance in architectural software design. KBS 30, 78–86 (2012).

[30] Chen, X., Liu, Z., Sun, M.: A unified model for word sense representation and disambiguation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1025–1035 (2014).

[31] Zambrano, A., J. Fabry, and S. Gordillo, Expressing aspectual interactions in requirements engineering: Experiences, problems and solutions. Science of Computer Programming, 2012. 78(1): p. 65-92.

[32] Binkley, D., et al., Increasing diversity: Natural language measures for software fault prediction. Journal of Systems and Software, 2009. 82(11): p. 1793-1803.

[33] Bateman, J.A., et al., A linguistic ontology of space for natural language processing. Artificial Intelligence, 2010. 174(14): p. 1027-1071.

[34] Chentouf, Z., Managing OAM&P requirement conflicts. Journal of King Saud University - Computer and Information Sciences, 2014. 26(3): p. 296-307.

[35] Cleland-Huang, J.: Mining domain knowledge [requirements]. IEEE Softw. 32(3), 16–19 (2015).

[36] Dalpiaz, F., van der Schalk, I., Lucassen, G.: Pinpointing ambiguity and incompleteness in requirements engineering via information visualization and NLP. In: International Working Conference on Requirements Engineering: Foundation for Software Quality. Springer, pp. 119–135 (2018).

[37] Aldekhail, M., Chikh, A. and Ziani, D., 2016. Software Requirements Conflict Identification: Review and Recommendations. International Journal of Advanced Computer Science and Applications, 7(10).

[38] Maysoon Aldekhail and Djamal Ziani, 2017. Intelligent Method for Software Requirement Conflicts Identification and Removal: Proposed Framework and Analysis. International Journal of Computer Science and Network Security, VOL.17 No.12

[39] Maysoon Aldekhail, Azzedine Chikh and Djamal Ziani, 2016. Software Requirements Conflict Identification: Review and Recommendations. International Journal of Advanced Computer Science and Applications, Vol. 7, No. 10

[40] Dieste, O., Juristo, N.: Systematic review and aggregation of empirical studies on elicitation techniques. IEEE Trans. Softw. Eng. 37(2), 283–304 (2011).

[41] Jan Hendrik Hausmann, Reiko Heckel and Gabi Taentzer, 2012. Detection of Conflicting Functional Requirements in a Use Case-Driven Approach. University of Paderborn 33095 Paderborn, Germany

[42] Matias Urbieta, Maria Jose Escalona, Esteban Robles Luna and Gustavo Rossi, 2011. Detecting Conflicts and Inconsistencies in Web Application Requirements. University of Seville, Spain

[43] Jan Hendrik Hausmann, Reiko Heckel and Gabi Taentzer, 2012. Detection of conflicting functional requirements in a use case-driven approach: a static analysis technique based on graph transformation. ICSE '02: Proceedings of the 24th International Conference on Software Engineering May 2002 Pages 105–115

[44] Evans, M.C.,Bhatia, J.,Wadkar, S., Breaux, T.D.: An evaluation of constituency-based hyponymy extraction from privacy policies. In: Requirements Engineering Conference (RE), 2017 IEEE 25th International. IEEE, pp. 312–321 (2017).

[45] Falessi, D., Cantone, G., Canfora, G.: Empirical principles and an industrial case study in retrieving equivalent requirements via natural language processing techniques. IEEE Trans. Softw. Eng. 39(1), 18–44 (2013).

[46] Femmer, H., Kuˇcera, J., Vetrò, A.: On the impact of passive voice requirements on domain modelling. In: ESEM. ACM, p. 21 (2014).

[47] Femmer, H., Fernández, D.M.,Wagner, S., Eder, S.: Rapid quality assurance with requirements smells. JSS 123, 190–213 (2017).

[48] Ferrari, A., Gnesi, S.: Using collective intelligence to detect pragmatic ambiguities. In: RE'12. IEEE, pp. 191–200 (2012).

[49] Ferrari, A., Spoletini, P., Gnesi, S.: Ambiguity and tacit knowledge in requirements elicitation interviews. REJ 21(3), 333–355 (2016).

[50] Ferrari, A., Dell'Orletta, F., Esuli, A., Gervasi, V., Gnesi, S.: Natural language requirements processing: a 4D vision. IEEE Softw. 6, 28–35 (2017a).

[51] Ferrari, A., Donati, B., Gnesi, S.: Detecting domain-specific ambiguities: an NLP approach based on Wikipedia crawling and word embeddings. In: 2017 IEEE 25th International Requirements Engineering Conference Workshops (REW). IEEE, pp. 393–399 (2017b).

[52] Ferrari, A., Esuli, A., Gnesi, S.: Identification of cross-domain ambiguity with language models. In: Groen, E.C., Harrison, R., Murukannaiah, P.K., Vogelsang, A. (eds) 5th InternationalWorkshop on Artificial Intelligence for Requirements Engineering, AIRE@RE 2018, Banff, AB, Canada, 21 Aug 2018. IEEE, pp. 31–38 (2018a).

[53] Ferrari, A., Gori, G., Rosadini, B., Trotta, I., Bacherini, S., Fantechi, A., Gnesi, S.: Detecting requirements defects with NLP patterns: an industrial experience in the railway domain. Empir. Softw. Eng. 23(6), 3684–3733 (2018b).

[54] Gacitua, R., Sawyer, P., Gervasi, V.: On the effectiveness of abstraction identification in requirements engineering. In: Proceedings of the 18th IEEE International Requirements Engineering Conference (RE'10). IEEE, pp. 5–14 (2010).

[55] Gleich, B., Creighton, O., Kof, L.: Ambiguity detection: Towards a tool explaining ambiguity sources. In: REFSQ'10, vol. 6182. Springer, LNCS, pp. 218–232 (2010).

[56] Guo, J., Cheng, J., Cleland-Huang, J.: Semantically enhanced software traceability using deep learning techniques. In: 2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE). IEEE, pp. 3–14 (2017).

[57] Jha, N., Mahmoud, A.: Mining user requirements from application store reviews using frame semantics. In: International Working Conference on Requirements Engineering: Foundation for Software Quality. Springer, pp. 273–287 (2017).

[58] Kof, L.: From requirements documents to system models: a tool for interactive semi-automatic translation. In: 2010 18th IEEE International Requirements Engineering Conference, pp. 391–392 (2010).

[59] Krisch, J., Houdek, F.: The myth of bad passive voice and weak words an empirical investigation in the automotive industry. In: 2015 IEEE 23rd International Requirements Engineering Conference (RE). IEEE, pp. 344–351 (2015).

[60] Levy, O., Goldberg, Y.: Neural word embedding as implicit matrix factorization. In: Advances in Neural Information Processing Systems, pp. 2177–2185 (2019).

[61] Lian, X., Rahimi,M., Cleland-Huang, J., Zhang, L., Ferrai, R., Smith,M.: Mining requirements knowledge from collections of domain documents. In: 2016 IEEE 24th International Requirements Engineering Conference (RE). IEEE, pp. 156–165 (2016).

[62] Manning, C.D.: Part-of-speech tagging from 97% to 100%: is it time for some linguistics? In: International Conference on Intelligent Text Processing and Computational Linguistics. Springer, pp. 171–189 (2011)

[63] Massey, A.K., Rutledge, R.L., Anton, A.I., Swire, P.P.: Identifying and classifying ambiguity for regulatory requirements. In: RE'14. IEEE, pp. 83–92 (2014).

[64] Mikolov,T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations ofwords and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013).

[65] Navigli, R.: Word sense disambiguation: A survey. ACM Comput. Surv. (CSUR) 41(2), 10 (2009).

[66] Pohl, K., Rupp, C.: Requirements Engineering Fundamentals. Rocky Nook Inc, San Rafael (2011)

[67] Quirchmayr, T., Paech, B., Kohl, R., Karey, H.: Semi-automatic software feature-relevant information extraction from natural language user manuals. In: Proceedings of the 23rd International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ'17), pp. 255–272, Springer (2017).

[68] Raganato, A., Bovi, C.D., Navigli, R.: Neural sequence learning models for word sense disambiguation. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 1156–1167 (2017).

[69] Robeer,M., Lucassen, G., van derWerf, J.M.E., Dalpiaz, F., Brinkkemper, S.: Automated extraction of conceptual models from user stories via nlp. In: Proceedings of the 24th IEEE International Requirements Engineering Conference (RE'16), pp. 196–205. IEEE (2016).

[70] Rodriguez, D.V., Carver, D.L., Mahmoud, A.: An efficient wikipedia-based approach for better understanding of natural language text related to user requirements. In: 2018 IEEE Aerospace Conference, pp. 1–16. IEEE (2018)

[71] Rosadini, B., Ferrari, A., Gori, G., Fantechi, A., Gnesi, S., Trotta, I., Bacherini, S.: Using NLP to detect requirements defects: an industrial experience in the railway domain. In: REFSQ, pp. 344–360. Springer (2017)

[72] Sleimi, A., Sannier, N., Sabetzadeh,M., Briand, L., Dann, J.: Automated extraction of semantic legal metadata using natural language processing. In: 2018 IEEE 26th International Requirements Engineering Conference (RE), pp. 124–135. IEEE (2018).

[73] Sultanov, H., Hayes, J.H.: Application of reinforcement learning to requirements engineering: requirements tracing. In: Proceedings of the 21st IEEE International Requirements Engineering Conference (RE'13), pp. 52–61. IEEE (2013).

[74] Taghipour, K., Ng, H.T.: Semi-supervised word sense disambiguation using word embeddings in general and specific domains. In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 314–323 (2015)

[75] Tjong, S., Berry, D.: The design of SREE a prototype potential ambiguity finder for requirements specifications and lessons learned. In: REFSQ'13, LNCS, vol. 7830, pp. 80–95 (2013).

[76] Wang,W., Niu, N., Liu, H., Niu, Z.: Enhancing automated requirements traceability by resolving polysemy. In: 2018 IEEE 26th International Requirements Engineering Conference (RE), pp. 40–51. IEEE (2018).

[77] Yang, H., De Roeck, A.N., Gervasi, V.,Willis, A., Nuseibeh, B.: Analysing anaphoric ambiguity in natural language requirements. Requir. Eng. 16(3), 163–189 (2011).

[78] Yuan, D., Richardson, J., Doherty, R., Evans, C., Altendorf, E.: Semi-supervised word sense disambiguation with neural models (2016).

[79] D. Mairiza, D. Zowghi, and V. Gervasi, Utilizing TOPSIS: A Multi Criteria Decision Analysis Technique for Non-Functional Requirements Conflicts in Requirements Engineering, D. Zowghi and Z. Jin, Eds. Springer Berlin Heidelberg, 2014, pp. 31-44.

[80] T. H. Nguyen, B. Q. Vo, M. Lumpe, and J. Grundy, KBRE: a framework for knowledge-based requirements engineering, Softw. Qual. J., vol. 22, no. 1, pp. 87-119, Apr. 2013.

[81] A. Alebrahim, S. Fabender, M. Heisel, and R. Meis, Problem-Based Requirements Interaction Analysis in Requirements Engineering: Foundation for Software Quality, C. Salinesi and I. van de Weerd, Eds. Springer International Publishing, 2014, pp. 200-215.

[82] M. Kamalrudin, J. Grundy, and J. Hosking, Managing Consistency between Textual Requirements, Abstract Interactions and Essential Use Cases in Computer Software and Applications Conference (COMPSAC), 2010 IEEE 34th Annual, 2010, pp. 327-336.

[83] D. Mairiza, D. Zowghi, and V. Gervasi, Conflict characterization and Analysis of Non Functional Requirements: An experimental approach, in 2013 IEEE 12th International

Conference on Intelligent Software Methodologies, Tools and Techniques (SoMeT), 2013, pp. 83-91.

[84] D. Mairiza, D. Zowghi, and N. Nurmuliani, Managing conflicts among non-functional requirements, University of Technology, Sydney, 2009.