Enhancing Software Quality in Agile Software Development through Customer Feedback & Reviews Analysis

P. R. H. Sachithra Vinodani Thilakarathne

198776E

Faculty of Information Technology

University of Moratuwa

July 2022

Enhancing Software Quality in Agile Software Development through Customer Feedback & Reviews Analysis

P. R. H. Sachithra Vinodani Thilakarathne

198776E

Faculty of Information Technology

University of Moratuwa

MSc / PG Diploma in Information Technology

July 2022

Declaration

I declare that this is my work and dissertation's content does not include any material previously submitted for a diploma or degree in any other institute or university without acknowledgment. To the belief and best of my knowledge, it does not include any material previously published by another. I hereby grant the nonexclusive right with in whole or in part in print of distribute my dissertation using electronic or other medium to the University of Moratuwa.

Signature: UOM Verified Signature Date: .....

The supervisor should be certified the research dissertation with the following declaration.

I hereby confirm that the above candidate has carried out research for the master's degree dissertation under my supervision.

Signature of the Supervisor: ..... Date: ……

Acknowledgment

The work included in this research is continued as my Master's degree research project. The completed final project is the outcome of combining all support, encouragement, and guidance given by many others. Furthermore, I take opportunity to express my gratitude who gave me the support and keep up to complete this considerable task.

I deeply express my gratitude to supervisor Dr. Chaman Wijesiriwardana and the lecturers of the University of Moratuwa whose encouragement suggestions, and support for the development, instructive discussions and constant guidance.

I am extremely grateful to my supervisor Dr.Chaman Wijesiriwardana, (Senior Lecturer) who gave and confirmed the permission to continue this research, and for all the guidance and encouragement that he gave. I also wish to thank our friends and colleagues for all their support help,interest, and advice. I would like to thank others whose names are not listed particularly but have given their support in many ways and encouraged us to make and complete this as success.

Abstract

The agile software development process is one of the best software development process models which includes an iterative procedure according to agile practices. Kanban, Scrum, XP, and Hybrid frameworks provide based environments for continuing the agile process. Requirements gathering and analysis phase get the major priority within each framework according to the software feasibility. The research focused on the Scrum framework's requirements gathering and analysis process along with related problems that the software team members faced. The research objective is to provide a better solution to overcome the requirements analysis problems by using a decision support system.

Most of the software teams fail to identify the enhancements from the released software version's feedback and review analysis. The proposed novelty system aids software team members to identify the failures and related enhancements that need to be improved at the next level.

The decision support system is constructed by using three separate components and each analysis helps to identify the failures and enhancements. The decision support system accurately analysis the user requirements in each phase with high accuracy. The results of the analysis provide new insight into the software engineering research area. Research phenomenon makes a coherent interrelation between software quality assurance and quality engineering.

Keywords — Agile software development; Customer feedback and reviews; Product backlog; Scrum framework; Requirements analysis

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1. Introduction

The introduction of this document brings a brief overview of the entire document with the scope of the research, research problem, background context of the research, aim and objectives, research gap, and research questions. The main purpose of this section is to propose a brief commentary about the observation of the entire “Enhancing Software Quality in Agile Software Development through Customer Feedback & Reviews Analysis”.

1.2. Scope of the Research

Software Development Life Cycle

Research is mainly based on the software development process that supports and guides the development of the software continuously and sequentially. In software engineering, a software development process is a process that divides software development into some smaller tasks and step by step improves subtasks sequentially[1].

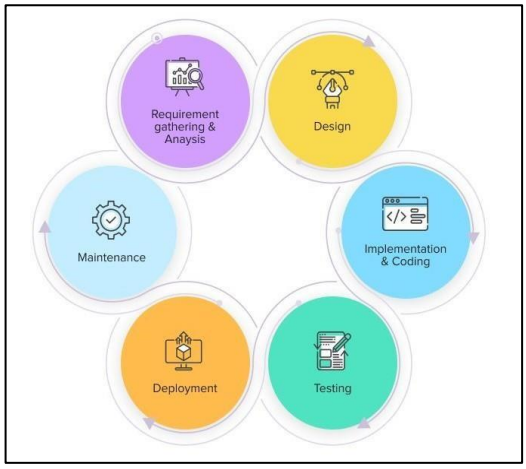


Figure 1. 1: Software Development Life Cycle

SDLC mainly includes planning and requirement analysis, defining requirements, designing the product architecture building or developing the product, testing the product, deployment in the market, and maintenance. The software process models derive from the software life cycle.

There are many kinds of process models included according to the different requirements. Some examples are the waterfall model, the v model, and the spiral model. Currently, most companies use the agile software development process model, because it has high flexibility, team collaboration, quicker & more efficiency, and high customer interaction[1].

Agile Software Development Process Model

In the agile software process model, there are many meetings and analyzing sections included for facilitating get better decisions. Customer interaction between agile software development processes is higher than when compared with other development process models. Scrum and Kanaban are the most famous agile frameworks when compare with the crystal framework, dynamic systems development method, and feature-driven development[2].

Scrum framework is the most famous and utility approach which currently software industries use and it includes events, roles, artifacts, and scrum rules and has specific connections between these tasks. The software development team should be oriented with the scrum framework and need to align with the agile development process[3].

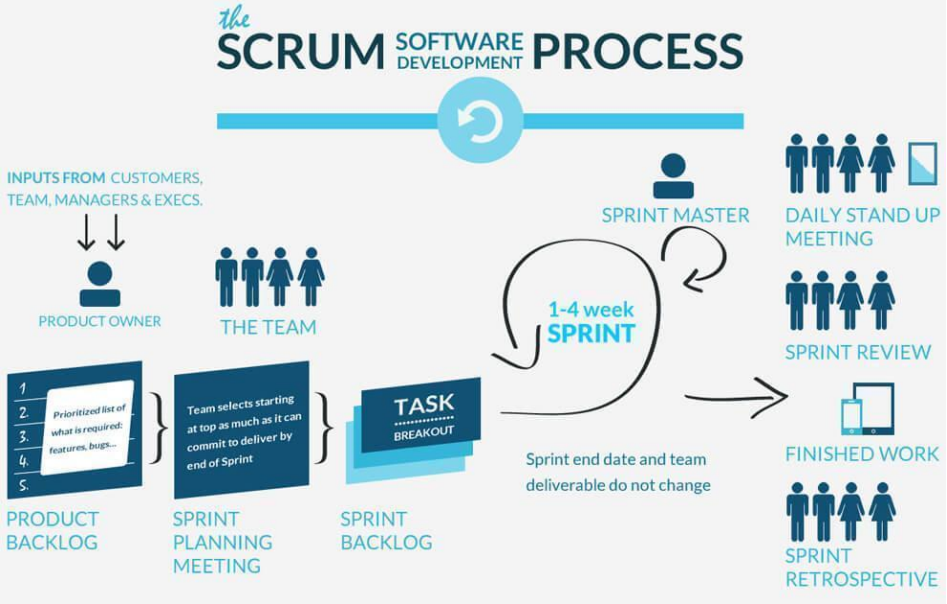


Figure 1.2: Scrum Software Development Process Model

Research only focuses on the process of requirements gathering and analysis stage in the agile scrum software development process model. When analyzing the existing responsibilities of each software team member, the product owner should manage the requirements gathering and analysis stage and create a product backlog by identifying the inputs from customers, teams, managers, and executives.

The product owner involves in gathering and identifying the requirements and creates and manages a product backlog before the sprint plan meeting. Identifying essential requirements and requirements prioritization is the most important process in creating a product backlog process. It highly depends on the product owner's decision. The product owner should know about the business problem, progress, productivity of the team members, and some technology-based knowledge. Business analytics play this role and carry the responsibility[3].

The Product Owner's Responsibilities

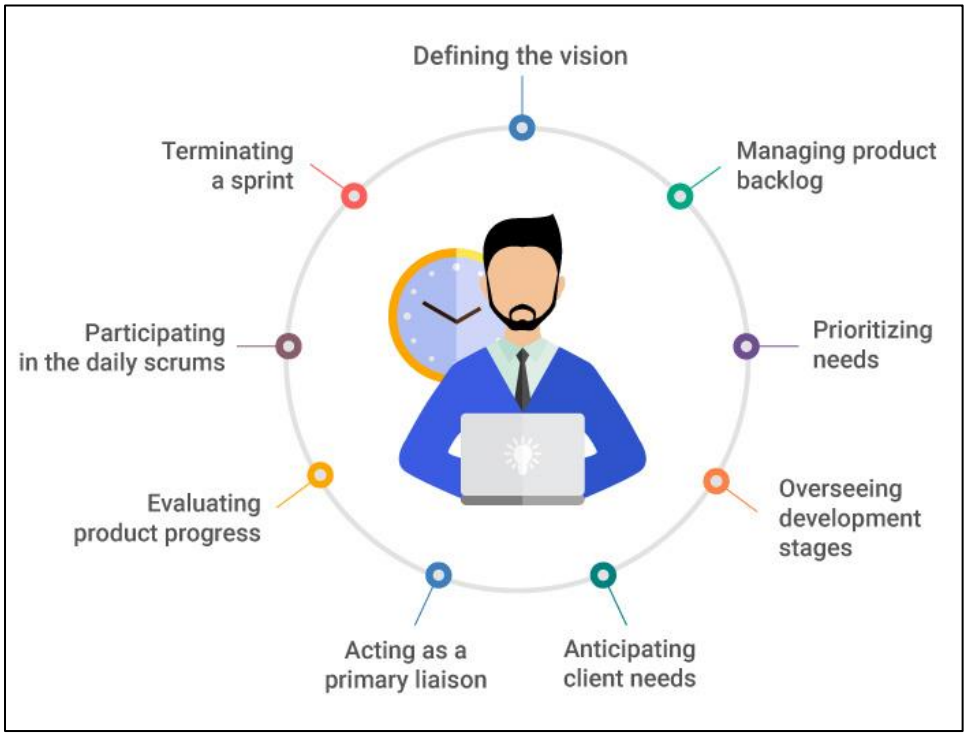


Figure 1.3: The Product Owner's Responsibilities

The research focused on the specific environment in product backlog creation which collects customer feedback/reviews from released software versions and enhances the software quality by extracting failures and enhancements. It is the most difficult and critical section that product owners faced during creating a product backlog[3].

1.3. Research Problem

Problem Related to Agile Scrum Development Process Model

In the agile software process model, requirements and outcome solutions are generated by the connection between self-organizing and the team's collaboration. The process of agile development is which enables teams to deliver the outcome by using utility faster with greater quality and predictability. It has a greater aptitude according to the response changes[4].

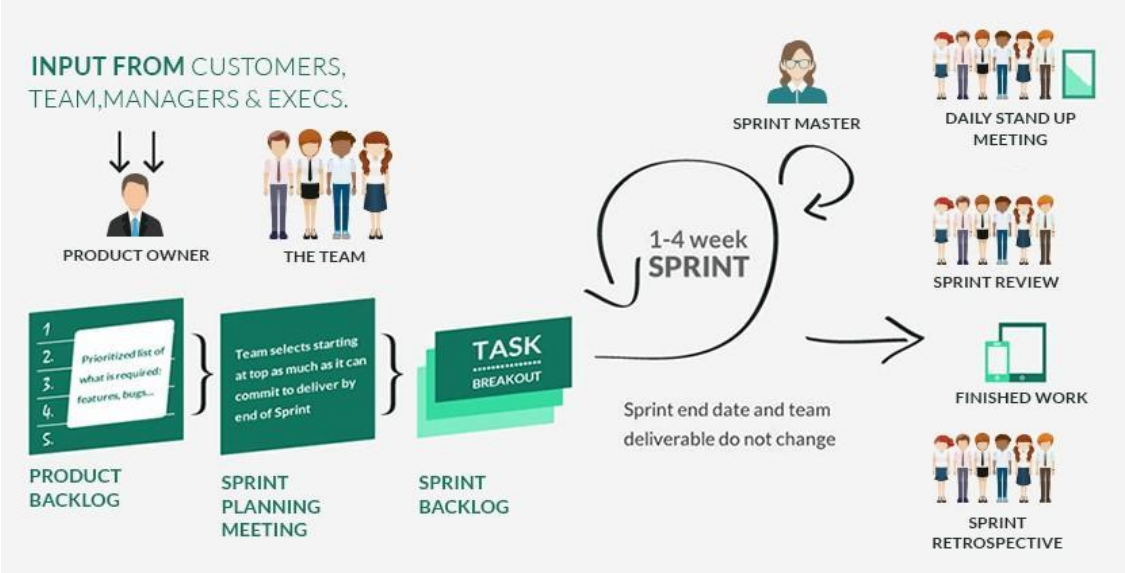


Figure1.4: Agile Scrum Development Process with Sprint Review

Scrum and Kanaban are the most famous agile frameworks when compare with the crystal framework, dynamic systems development method, and feature-driven development. Scrum framework is the most famous and utility approach which currently software industries use and it includes events, roles, artifacts, and scrum rules and has specific connections between these tasks. The software development team should be oriented with the scrum framework and need to align with the agile development process.

In the agile scrum software process model, there are many meetings and analyzing sections included for facilitating get better decisions. Customer interaction between agile software development processes is higher than when compared with other development process models. Higher customer interaction happens during the first stage which means the requirement gathering phase[4].

Process of the Agile Scrum Development Model

- Gather inputs from customers, teams, managers, and executives
- Create product backlog
- Sprint planning meeting
- Select essential requirements from the product backlog and add all selected requirements into a sprint backlog
- Assign requirements into the first sprint
- Finalize the sprint week
- Develop the software during a sprint
- Continue the requirement selection from the product backlog and sprint process
- Testing and deployment

Agile Scrum Roles and Responsibilities

Figure 1.5: Agile Scrum Role and Responsibilities

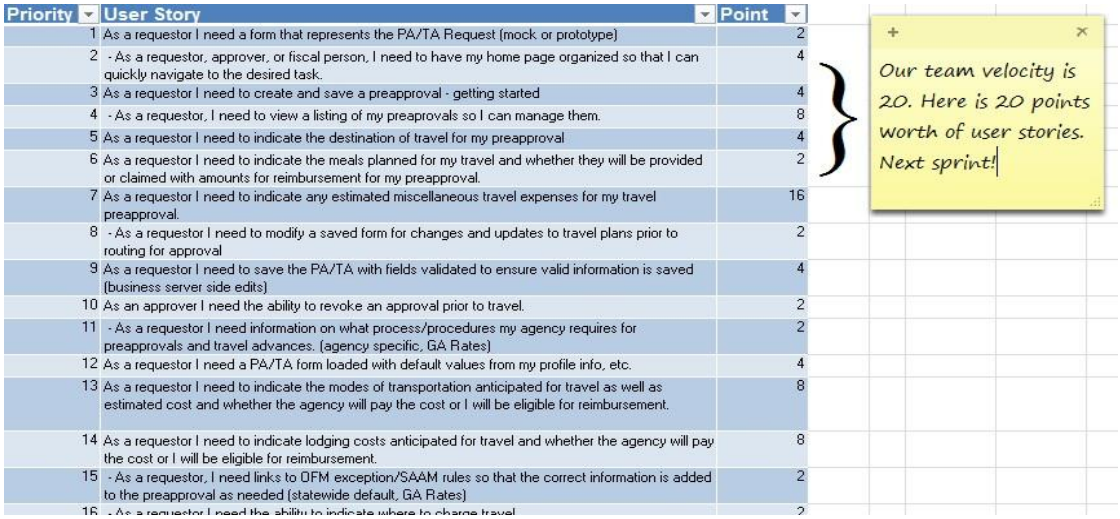
- **Customers, Managers, Teams, and Executives:** Give input and requirements of the requested product[5].
- **Product Owner:** The product owner is one of the stakeholders of the project. The role is primarily responsible for the direction of project progress. The Product Owner understands the requirements of the project from an external perspective and creates a product backlog. He / She Collects data from end-user feedback and determines appropriate next-best-action plans throughout the development process. The key responsibilities are Scrum backlog management, Release management, and Stakeholder management[5].

- **Scrum Master:** Ensures team coordination and helps the development process of the project. The Scrum Master gets the requirements from the Product Owner and ensures and assigns the tasks[5].
- **Software Development Team:** They have their responsibilities and sometimes include cross-functional responsibilities. Because they need to transform an idea or a requirement with end-users. Product designer, Writer, Programmer, Tester, and UX specialist[5].

  All team members need to work together and each member should have a better understanding of the business process and other related skills. When analyzing the existing problems of the agile software development process model, the requirement gathering and analyzing stage is the most challenging section to handle. This research focuses on this problem and tries to understand the background and proposed a new idea to overcome the problem. After reviewing background information on the agile software development process, the product backlog creation section (One main section of the requirements gathering and analysis) is a challenging and important process in the requirement gathering and planning stage[3].

Product Backlog Creation

Mostly, overall project depends on the requirement gathering and analyzing sections. The product backlog creation is important process in agile requirement and gathering section. It helps to analyze, estimate and prioritize requirements for developing future products. Product backlog looks like a to-do list and helps to software team to get a decision. The product owner is one of the stakeholders and he/she creates a product backlog using some specific method [6].



Priority	User Story	Point
1	As a requestor I need a form that represents the PA/TA Request (mock or prototype)	2
2	- As a requestor, approver, or fiscal person, I need to have my home page organized so that I can quickly navigate to the desired task.	4
3	As a requestor I need to create and save a preapproval - getting started	4
4	- As a requestor, I need to view a listing of my preaprovals so I can manage them.	8
5	As a requestor I need to indicate the destination of travel for my preapproval	4
6	As a requestor I need to indicate the meals planned for my travel and whether they will be provided or claimed with amounts for reimbursement for my preapproval.	2
7	As a requestor I need to indicate any estimated miscellaneous travel expenses for my travel preapproval.	16
8	- As a requestor I need to modify a saved form for changes and updates to travel plans prior to routing for approval	2
9	As a requestor I need to save the PA/TA with fields validated to ensure valid information is saved (business server side edits)	4
10	As an approver I need the ability to revoke an approval prior to travel.	2
11	- As a requestor I need information on what process/procedures my agency requires for preapprovals and travel advances. (agency specific, GA Rates)	2
12	As a requestor I need a PA/TA form loaded with default values from my profile info, etc.	4
13	As a requestor I need to indicate the modes of transportation anticipated for travel as well as estimated cost and whether the agency will pay the cost or I will be eligible for reimbursement.	8
14	As a requestor I need to indicate lodging costs anticipated for travel and whether the agency will pay the cost or I will be eligible for reimbursement.	8
15	- As a requestor, I need links to OFM exception/SAAM rules so that the correct information is added to the preapproval as needed (statewide default, GA Rates)	2
16	- As a requestor I need the ability to indicate where to charge travel	2

Figure 1.6: Product Backlog

Product backlog includes:

- New features
- Enhancements (change existing features and improve)
- Bug fixes
-Technical debt and refactoring
- Other Activities

After analyzing background information of this product backlog creation process, can be finalized this process is the most difficult process than other tasks. Because the product owner has to gather inputs from the customer, manager, executives, and team members and analyzes all the data manually by using experience, sense, and thought. Gathering inputs from the customer stage can divide into two sections gathering customer requirements before developing software, and gather customer and other's requirements (feedback /reviews) after releasing the versions. Gather requirements (feedback /reviews) after releasing the versions directly affected to identify the enhancements and this circumstance support to improve the outcome than previously handover or released versions.

This research focuses on the problem related to extracting enhancements from released software versions' failures through customer feedback and reviews.

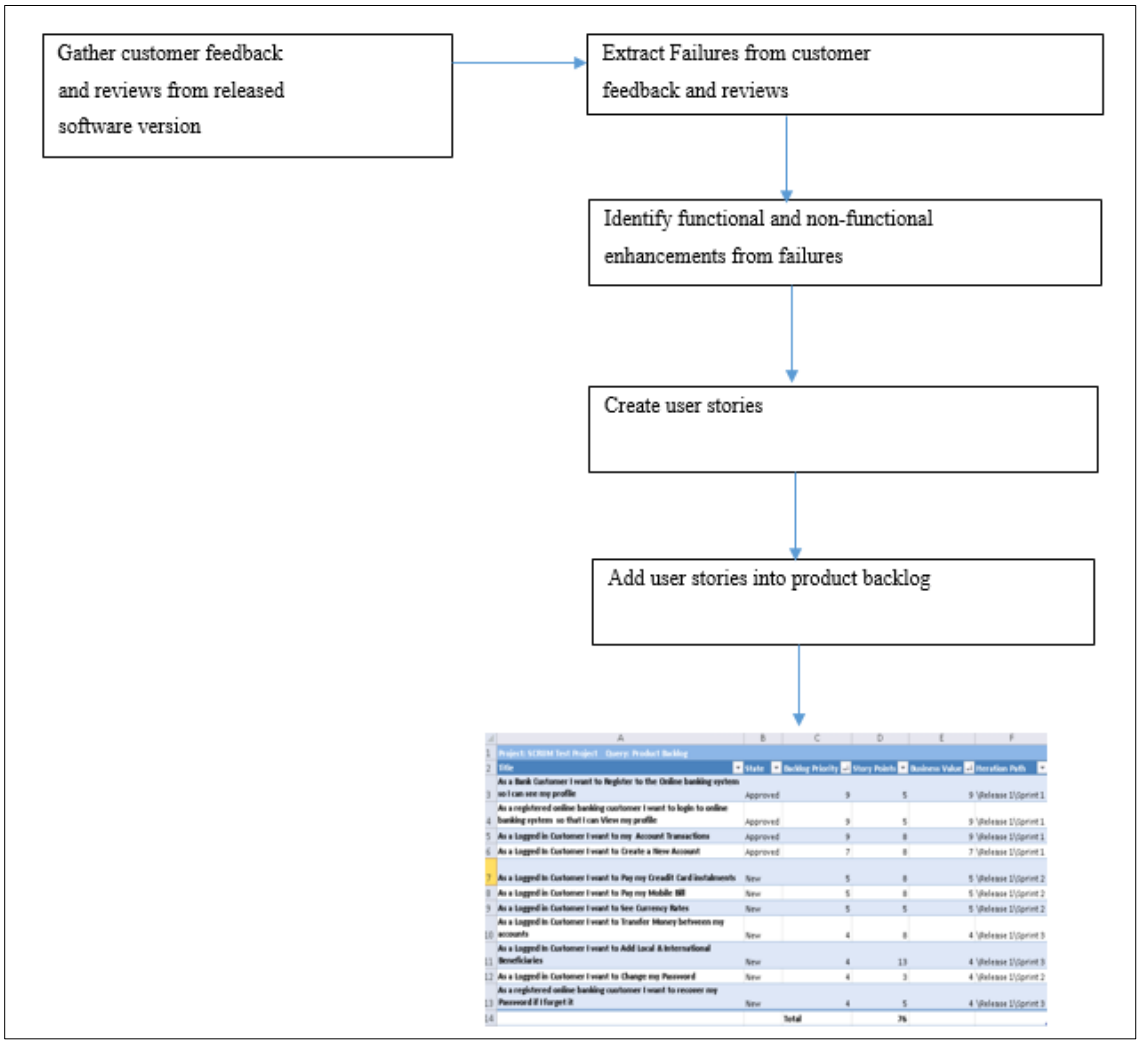


Figure 1.7: The process of Extracting Enhancements from Customer Feedback and Reviews

<u>The problem factors arise when extracting enhancements from previously released software versions.</u>

- What are the existing failures of the released s/w versions?
- How do we identify enhancements from the released s/w version's failures through customer feedback and review analysis?
- What enhancements need to be more concerned?

- How analyze the existing level /status of previously released software versions?
- What are the future impactions of the released software versions?
- How analyze the prioritization level and points that need to be given for each requirement?

When consideration about the problem factors related to the above progress, most of the issues arise when identifying failures and enhancements from released software versions. Moreover, identifying the non-functional and functional enhancements is the second most critical situation that product owners face before creating user stories. Because incorrect enhancements extraction and identification process, directly impact to failure of the project[7].

1.4. Background of the Research

The product owner has the responsibility to create a well-oriented product backlog and needs to make the product backlog align with customer requirements. This research focuses on analyzing customer feedback /reviews and making a decision support system/proposed tool to get better decisions. This proposed tool analyzes the existing quality level of released versions and future variations/impactions of the S/W version using customer feedback /reviews. Finally extracts non-functional and functional enhancements that need to be improved. After finalizing the non-functional and functional enhancements, add them to the product backlog. The research approach and methodology section describe more details of the deep discussion of the solution and research finding.

Suggested decision support tools enhance the quality of software. This decision support tool helps the product owner to make decisions using customer feedback and finalize the most required requirements for improving the next software versions. There are three main components included in this system. Below mentioned all three main components of the decision support system.

The Process of the Decision Support Tool

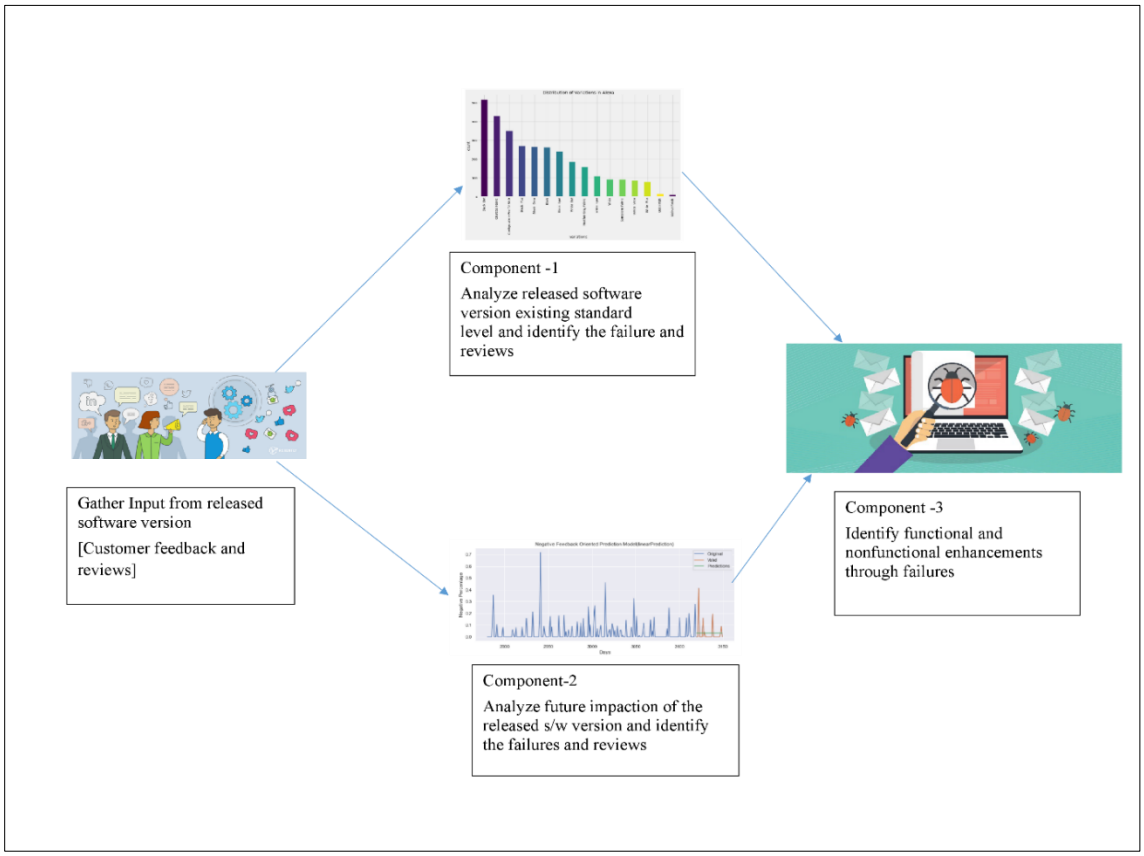


Figure 1.8: The Process of Decision Support Tool

Table 1.1:The Components Descriptions

Components	Description
1	Analyze the existing level /status of the previously released software version and identify the enhancement. This component's main task is to support the product owner to get an idea about the existing status of the released versions and support to identify the failures.

2	Analyze the future variations, and software impaction and identify enhancements. This component predicts sentiment features and the product owner can get an idea about the released software version's future impact and future failures.
3	Extract functional & non – functional enhancements from identified failures from the released S/W version’s feedback and reviews.

Component 1 and component 2 directly interact with component 3 to complete the tool progress. Component 1 and component 2 provide advanced visualizations to extract existing failures and future failures of the released S/W version. Component 3 aid to identify the functional and non-functional enhancements that need to be improved in the next level from extracted failures from components 1 and component 2. The above components are detailed and discussed in the methodology and approach section and clearly define technologies, aim, and each related objective. To create an accurate suggested system model, use advanced visualizations, data procedure methods, data preprocessing techniques, and specific algorithms based on NLP and machine language.

1.5. Research Aim and Objectives

The scope of the question is determined by the research aim and research objectives. The research aim represents the intention of the research study which means this section describes the final target of the research finding. It summarizes all the achievements using a few sentences and determines the specific aim. This section describes the main target of the research study using selected research questions. Before discussing the aim and objective of the research question, identifying the research question is an important factor in research investigation. This section briefly discusses the research question and finally describes the aim and objectives of the research study[8].

Research Aim

The research aim describes the main task that needs to be acquired after studying and implementing the research finding. The main research aim is to create a decision support tool for identifying the essential changes/enhancements through customer feedback /reviews from released software versions and get tool support to create a well-oriented product backlog during the agile scrum software development process.

This suggested system aid to fulfill customer requirements and help to enhance the quality of the software version before deployment. This Table describes the objective and sub-objectives of the decision support tool using brief descriptions.

Table 1.2: The Description of the Component's Objectives

Objectives	**Description and Sub Objectives**
Objective 1/ Component 1: Analyze the existing level /status of the previously released software version.	Analyze existing records and identify the features/attributes variations. This component's main task is to support the product owner to get an idea about the existing status and existing failures of the released versions. ***Objective 1.1:*** Data visualization using different existing features (feedback, rating, customer reviews length, variations). ***Objective 1.2:*** Analyze the overall existing negative, positive, and neutral status of the customer

	reviews using sentiment analysis and customer reviews. ***Objective 1.3:***Analyze existing S/W products trending
Objective2 / Component 2: Predict customer feedback and reviews status and analyze future variations /impaction of the S/W products.	Support for identifying the future changes besides the past customer feedback and reviews changes. This component predicts sentiment features and the product owner can get an idea about the future impactions and future failures of the released S/W. ***Objectives 2.1 / 2.2 / 2.3 / 2.4 / 2.5 /2.6 :*** Predict different parameters (sentiment status ,star rating value , ratings, feedback, reviews ,product trending)

Objective 3 / Component 3 :Extract functional and Non- Functional Enhancements from Selected Negative Reviews	This component uses to extract functional and non-functional enhancements from selected negative reviews. The product owner can easily identify the functional and non-functional enhancements that need to be improved in next level. ***Objective 3.1:*** System support for identifying the functional and non–functional enhancements separately.

1.6. Research Gap

A research gap is a problem that has not been answered by any of the existing papers/commercially available products. Moreover, a research gap exists when there is a new concept that hasn't been studied at all. After comparing with existing tools and research papers, can determine some new performance according to the existing features[9].

After comparing with existing tools, most of the tools only target extracting customer reviews and display existing features using data visualization such as *Mopinion*[10], *Feedbackify*[11], *etc.* Commercially available tools do not consider advanced visualization and usability factors. There are no decision support tools to support creating a product backlog in the agile scrum development process model. After investigating the existing problem of the tools, these are not specific to a particular one-task and not mentioned the usage of the analyzed details. There are no available commercial tools that support analyzing the existing level/status of previously released software versions, predicting customer review status and analyzing future variations, and Extracting essential functional and non-functional enhancements using different properties. Some of the research papers

only mentioned the process of extracting non-functional requirements and each related model creation[12].

Currently, product owners analyze customer reviews using a manual process and get a decision without using any tools. When consideration of the research gap, the suggested system clearly defined the problem and solution that was not previously mentioned or studied. The research tries to make an accurate model and usability research outcome according to the background of the literature review findings.

1.7. Research Questions

When indicating research questions during the research studying, some specific research questions can consider the most essential factors.

- What are the main policies and rules of the agile software development process model when applying the scrum framework approach?
- What are the available existing methods for extracting functional and nonfunctional enhancements?
- What are the possible methods for identifying the functional and non-functional enhancements?
- How to extract enhancements from failures?
- How full fill the technology usage?
- What is the level of user technology awareness?
- What are the main research ethical considerations?

1.8. Summary

Chapter 1 describes a detailed description of the research introduction and covers the scope of the research, research problem, background context of the research, aim and objectives, research gap, and research questions.

Chapter 2

Literature Review

2.1. Introduction

Background of literature means refers to written records, documents, scientific studies, biographies, and virtually anything that help to gain knowledge in a selected area of question. It includes any related examination, critique, synthesis, and integration of sources and theories. Literature reviews provide an overview of research references and explore some research findings according to the particular topic. Research references demonstrate to readers how the research fits into the larger field of study. After analyzing the background of the literature concept, I divide the literature review into three sections such as the difference between commercially available tools and suggested systems, and a literature review of similar research papers[13].

2.2. Commercially Available Tools

Some commercially available tools exist to analyze customer reviews when developing software or application. Below mentioned some tools which partially similar tools to the suggested system. After reviewing some websites of the related tools, identified the performances, technologies, and new features according to the tool background review and customer feedback.

Mopinion is one of the voices of the customer tool used to analyze customer feedback. This is used to get all-in-one user feedback from all digital channels. This tool uses to collect and analyzes feedback from websites, mobile apps, and email. When consideration of the method, it has a user-friendly interface and users can easily build, design, and configure feedback forms. Customer feedback can be visualized in customizable dashboards. Charts use for advanced analyses[11].

UsersThink uses emails and other media to send the report within 24 hours and ***User Zoom* is** a user testing tool that manages the logistics of recruiting test takers and provides a testing platform[11].

Feedbackify is also one important and famous tool used for analyzing customer feedback. This helps to collect real-time and indirect feedback and represent data from the visual dashboard. Analysis of the responses, correlation matrix, user Stories, NPS, and many others and generate report of customer feedback satisfaction. Only one task improves the usability of the system. This system not only focuses gather feedback but also uses customer feedback to process some tasks[11].

2.3 Literature Review of Similar Research Papers

Similar research papers provide an overview of research references and explore some research findings according to the particular topic. Research references represent to readers, how they need to identify the problems related to research and the direction they need to follow.

The first reference research paper is ***"SCRUM model for agile methodology"***. This paper was published at the 2017 International Conference on Computing, Communication, and Automation (ICCCA). The research paper focuses on the problem faced and relevant solution and identify the gains when selecting the SCRUM model for agile methodology according to organization requirements. This paper includes a detailed description of the scrum framework and research questions that they faced during the research study. The research paper defines some main concepts which are the introduction of the agile software development, an overview of the assessment methodology, research result, future work, summary, and conclusion. Mainly analyze the adequacy, capability, and effectiveness during assessing time for each selected organization[3].

- "Adequacy"- Sufficiency of the method concerning meeting its stated objectives.
- "Capability" – Ability of an organization to provide a supportive environment. This ability is represented in the nature of an organization's processes, people, and projects.

NOMATIC tool describes the non-functional requirements extraction, modeling, importance, and usage of agile software development. The research paper title is ***"NORMATIC: A Visual Tool for Modeling Non-functional Requirements in Agile Processes".*** NORMATIVE tool supports analyzing Non-Functional Requirements for Agile Processes. This tool can potentially help agile software development teams early on during the requirements gathering and analysis phases. The tool can also help project managers and Scrum teams estimate user stories and identify risk-driven planning. Normatic includes new features that aid agile teams when they work in NFR's in an agile environment[14].

- The main identification can represent as it has a lightweight process.
- Using Normatic model can access through a risk-driven process.
- It mentioned future improvements details and future approaches.

The NERV Methodology is a lightweight process and uses to analyze the non-functional requirements. This research paper's name is ***"The NERV Methodology: A Lightweight Process and describes non-functional requirements in agile software development".*** This research paper mentions the importance of non-functional requirements and the methodology of non-functional requirements. NERV methodology includes non-functional requirements elicitation, reasoning, and validation in agile processes." Current results show the artifacts developed in this research can potentially help software development organizations address NFRs in early agile processes. Darshan Domah / Frank J. Mitropoulos find this methodology and publish it at the 2015 IEEE Annual Conference[15].

"The Product Backlog" research paper describes a detailed description of the artifacts. This research has found how the backlog is generated and how the team's role plays in a project[6].

- Objective: determine what is a product backlog, roles and responsibilities, and how does it manage?
- Method: the study around eight / nine software development projects at an international software company.

They interviewed 56 s/w engineers, product managers, and product designers. They surveyed 27 product designers. In research papers include, they use sampling theory to process the research. When considering the results, 13 practices related to product backlog generation[6].

"Information Extraction on Requirement Prioritization Approaches in Agile Software Development Processes" research paper includes requirements extraction and prioritization process. Requirement engineering is new research finding section and subject on which most of the research focuses. The requirement engineering process means which gathers the software requirements from end-users[16].

Prioritization of collected requirements of a project is the main task in requirement engineering. The first phases of the development need to complete the prioritization analysis. Team members can identify the specific requirements and the time that needs to be allocated through the relevant requirement prioritization process. The main objective of requirement prioritization is can decrease the development of the risk which means can reduce the risk-driven process. Agile software development processes can minimize the risk driven because it includes feasibility and iteration process. This paper mentioned a detailed description of the requirement prioritization approaches with their weaknesses and strength. According to the behavior and other constraints, the project owner can finalize the prioritization levels for requirements[16].

"*A Survey on Product Backlog Change Management and Requirement Traceability in Agile (Scrum)*" research paper which includes an introduction, development, and usage of the product backlog. The product backlog includes the basic outline of the own functionality in a prioritized order in Scrum. Scrum is one of the approaches in agile software development which includes an iterative, flexible, and collaborative process to develop a software product. Scrum projects accept changes at any stage of a project. But these changes directly affect project management activities and quality, budget, time and overrun. These changes also need to include architectural integrity. The objectives of this research mentioned how product backlog changes affect quality, budget, time, and overrun

and how managed by expertise in Scrum projects. Finally can assure the scrum approach validity correctness and user requirements traceability using the outcome of the research[17].

"Prioritizing User Requirements for Agile Software Development" research paper mentioned a detailed description of how to identify prioritizing user requirements in agile software development. The agile methodology helps in performing these user requirements through user stories iteratively. This can only be done through continuous and incrementally prioritization. This prioritization is essential for focusing on the budgetary plan. This research paper mentioned for prioritizing processes can proceed using the UML diagram. This dependence can be identified by visualizing the process of the UML Activity Diagram. This converting of user requirements into a UML Diagram can be done using NLP. Once the UML is ready, it supports prioritizing the user stories according to the flow. This paper's objective is to provide an approach to identifying the issue of prioritization when consideration of both business value and process flow aspects. Finally, prioritize the requirements using a UML diagram and add all specifications into the product backlog[18].

2.4 The Difference of the Suggested System

The most commercially available tools only consider the extract customer review features and data visualization through customer reviews and feedback. The suggested system not only considers the advanced visualization of the customer reviews. The proposed system analyzes the existing level /status of the previously released software version and identifies the existing failures through negative customer reviews and feedback, analyzes the future variations, and software impaction and identifies the future failures through prediction, and extracts functional & non – functional enhancements through identified failures. After investigating the existing research papers, the suggested system /decision support system is not only specific for extracting nonfunctional requirements like *normap, nomatic, and nerv*. But these methodologies help to understand the tool requirements clearly. *"The Product Backlog"* and *"A Methodology for Assessing Agile Software Development Methods"* are mentioned in all descriptions of the agile process model and product backlog content.

2.5 Summary

Chapter 2 describes a detailed description of the literature review and covers the description of the literature review, commercially available tools, literature review of similar research papers, and the difference of the suggested system.

Chapter 3

Research Methodology & Approach

3.1 Introduction

This section includes detailed descriptions of the techniques of creating decision support tool which helps to enhance the quality of software in the agile development process through customer feedback analysis. The descriptions include how software implementation of the project is continued, what are materials and data needed, and how will be collected [19].

When considering the above factors, divided research methodology into some specific sections such as gathering requirements and planning according to the selected problem, analyzing and designing, and identifying the performance of the components based on functional and non-functional requirements.

3.2 Requirement Gathering and Planning

Requirement gathering is an essential section before understanding the analysis and design overview of the suggested system. According to this research study need to interact with software team members who work in the agile development process[20]. When compare with other stage in software development life cycle(SDLC),this stage performance directly impact to the software /outcome failure.This circumstance not only frame fro agile software development process model and it can be applied into other software development process model when this stage performances became failure.

Figure 3.1: The Requirements Gathering Steps

Below mentioned all functional and non-functional requirements gathering steps that help to design the overview of the suggested system.

Step 1: As the first step, focused on the research problems and identified the research problem. The suggested system focused on the agile scrum development process model failures and successes. As the next step, identified the most critical section that carries on the project/software into failure status. According to research findings, the most critical section of the agile scrum software development process model is to identify the enhancements from customer feedback and reviews at product backlog creation[20].

Step 2: Then conduct a background study about the current situation of the construct product backlog in agile scrum software development by referring to literature and case studies[20].

Step 3: Gathered key insight on how the top-level strategic decision-maker rely on their data to come up with new goals and objectives and interviewed a few people who hold the key decision-making positions in the companies (ex: Product Owners) on how existing data can be analyzed[20].

Step 4: Furthermore, got the views and ideas from the business professional about their needs when it comes to decision support tools or software through questionnaires, face-to-face interviews, and focused-group discussions[20].

Step 5: Identified the main research objective and other specific objectives according to the requirements gathering section and convert objectives into functional requirements. These selected functional requirements assign to the suggested system as main components[20].

Step 6: After finalizing functional requirements, next consider the non-functional requirements such as performance, availability, reliability, maintainability, security, and usability[20].

- **Performance** - Analyze throughput, response time, utilization, and performance tests. This is use to exams to determine how a system works according to the responsiveness and stability and check whether under a workload [21].

- **Availability** - This suggested system is an available to users. The system not fail and usually have a fault tolerance availability. System downtime less than 15 seconds. Database backups taken continuously and data availability is opened [21].

- **Maintainability** –Proposed system maintenance done properly. SRS document also can be referred within each validation during the maintenance. All maintenance records documented for purposes. System continuously added requirements to the system as features, and bugs identified during the system running time. These new

requirements and bugs will be identified and the development team and take the responsibility [21].

- **Reliability** - This system is processed according to the software development process and keep all requirements secure way. Customer feedback recorded accurately and complete validating user input. System's non-functional requirements performances validate and testing. As a end note, confirmed system accuracy and suggested system performances [21].

3.3 Analysis and Designing

Analyze and design section describes the specific concept of the high-level architecture diagram and other related component diagrams using the aim and objectives of the research. After completing the requirement gathering and planning stage, need to construct the diagrams before moving to the implementation. This section describes the Overview of the decision support tool, high-level architecture diagram, and components diagrams separately. As the next step, explains the aim and objectives, and related components according to the suggested system functional requirements[22].

The Overview of the Decision Support Tool

The suggested tool analyzes the existing standard levels and future impactions of the released S/W versions using customer feedback /reviews analysis. The System help to identify the failures of the released S/W versions using related customer reviews/feedback analysis. As a final step, the system aid to identify the non-functional and functional enhancements through identified failures. Extracted enhancements add to the product backlog after approval from the team and these enhancements extraction process directly impact the next version released gain.

The High-Level Architecture Diagram of the Decision Support Tool

The high-level architecture diagram is the main diagram that is used to develop a software product[23].

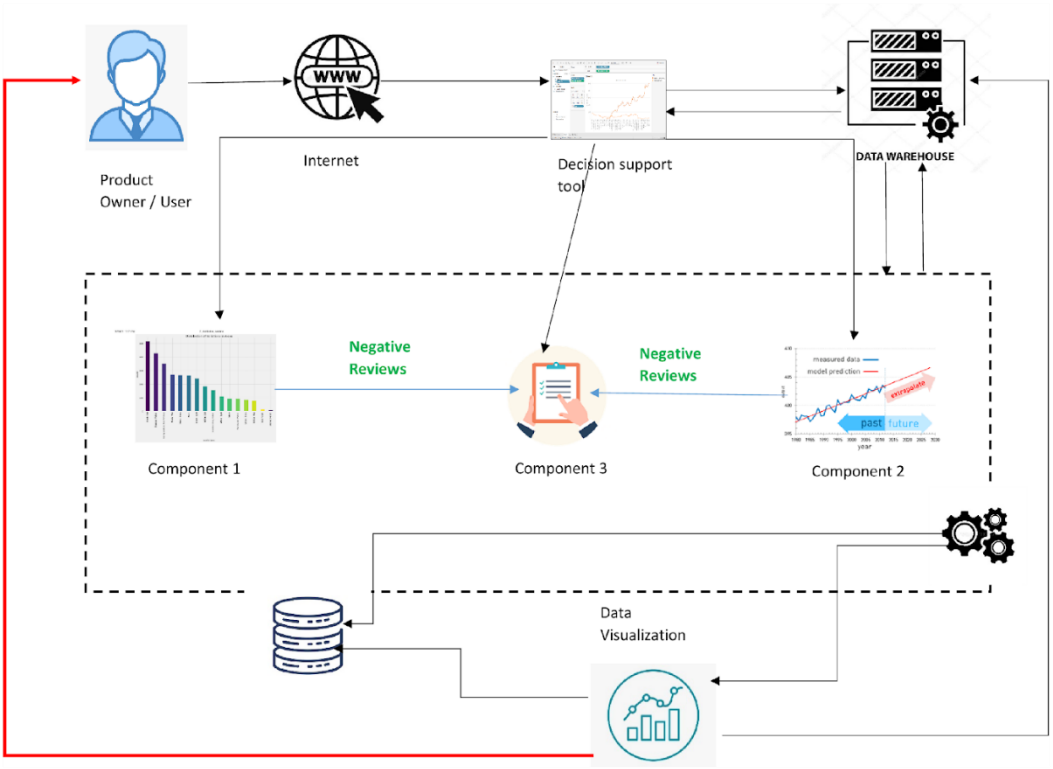


Figure 3.2:High-Level Architecture Diagram

Component 1: Analyze Existing Level /Status of the Previously Released Software Versions and Identify the Existing Failures through Negative Reviews and Feedback

Component 2: Analyze the Future Variations, Software Impaction and Identify the Future Failures through Prediction

Component 3: Extract Functional & Non – Functional Enhancements through Failures and Negative Reviews

Component 1 [*"Analyze existing level /status of the previously released software versions and identify the existing failures through negative reviews and feedback"*] and **Component 2** [*"Analyze the future variations, software impaction and identify the future*

failures through prediction"] directly interact with **Component 3** ["*Extract functional & non – functional enhancements through failures and negative reviews*"] to complete the proposed tool progress. Component 1 and Component 2 give advanced visualizations to the user to extract existing failures and future failures of the released S/W version. Component 3 aid to identify the functional and non-functional enhancements that need to be improved in the next level from extracted failures from Component 1 and Component 2. The proposed system uses advanced visualizations, data procedure methods, data preprocessing techniques, and specific algorithms based on NLP and machine language to create an accurate suggested system model.

Component 1: Analyze Existing Level /Status of the Previously Released Software Versions and Identify the Existing Failures through Negative Reviews and Feedback

- Component 1- Objective 1.1: Analyze Distributions of Alexa S/W Products and Identify Related Failures
- Component 1- Objective 1.2: Analyze Feedback variation in Alexa and Identify Related Failures
- Component 1- Objective 1.3: Analyze Variation vs. Ratings in Alexa S/W Products and Identify Related Failures
- Component 1- Objective 1.4: Analyzed Rating Variation in Alexa and Identify Related Failures
- Component 1- Objective 1.5: Analyzed Sentiment Status (Negative, Positive, Neutral) in Alexa and Identify Related Failures
- Component 1- Objective 1.6: Analyzed Product Trending in Alexa S/W Products and Identify Failures

Component 2: Analyze the Future Variations, Software Impaction and Identify the Future Failures through Prediction

- Component 2-Objective 2.1: Predict Sentiment Status for Newly Added Customer Reviews and Identify Related Failures
- Component 2-Objective 2.2: Dynamic Changing of the Future Feedback Variation and Identify Related Failures
- Component 2-Objective 2.3: Predict Star Rating value (1-5) for Newly Added Customer Reviews and Identify Related Failures
- Component 2-Objective 2.4: Dynamic Changing Of The Future Star Rating Variation and Identify Related Failures
- Component 2-Objective 2.5: Analyze And Identify the Future Products Trending and Identify Related Failures

Component 3: Extract Functional & Non – Functional Enhancements through Failures and Negative Reviews

- Component 3-Objective 1: Extract Functional & Non – Functional Enhancements from Selected Negative Customer Reviews through Identified Failures

User Interaction Process of the Decision Support Tool

This illustration explained the user interaction process of the decision support tool using particular steps.

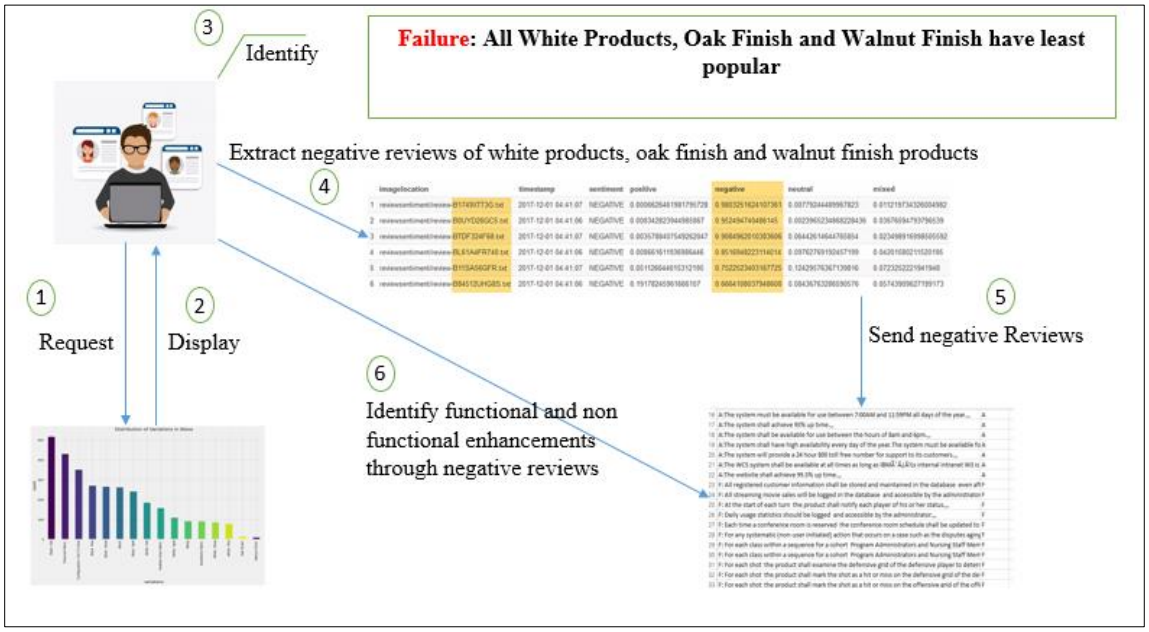


Figure 2.3: The User Interaction Process of the Decision Support Tool

Step 1: The user request distribution of the details of the S/W product.

Step 2: System response and display details using advanced visualization.

Step 3: User identify the least popular S/W products among others

Step 4: The system extracts negative reviews of the least popular S/W products according to the user's request.

Step 5: System sends negative reviews to component 3

Step 6: The user can identify the functional and non-functional enhancements through a negative review

The User Interaction Process of the Component Diagrams

Component 1: Analyze Existing Level /Status of the Previously Released Software Versions and Identify the Existing Failures through Negative Reviews and Feedback

Analyze existing attributes of previously released software versions (feedback, rating, customer reviews length, variations) and identify the features/attributes variations. This component's main task is to support the product owner to get an idea about the existing status of the released S/W versions and identify the failures of the released versions. Below mentioned component 1 process diagram using Objective 1.1: Analyze Distributions of Alexa S/W Products and Identify Related Failures.

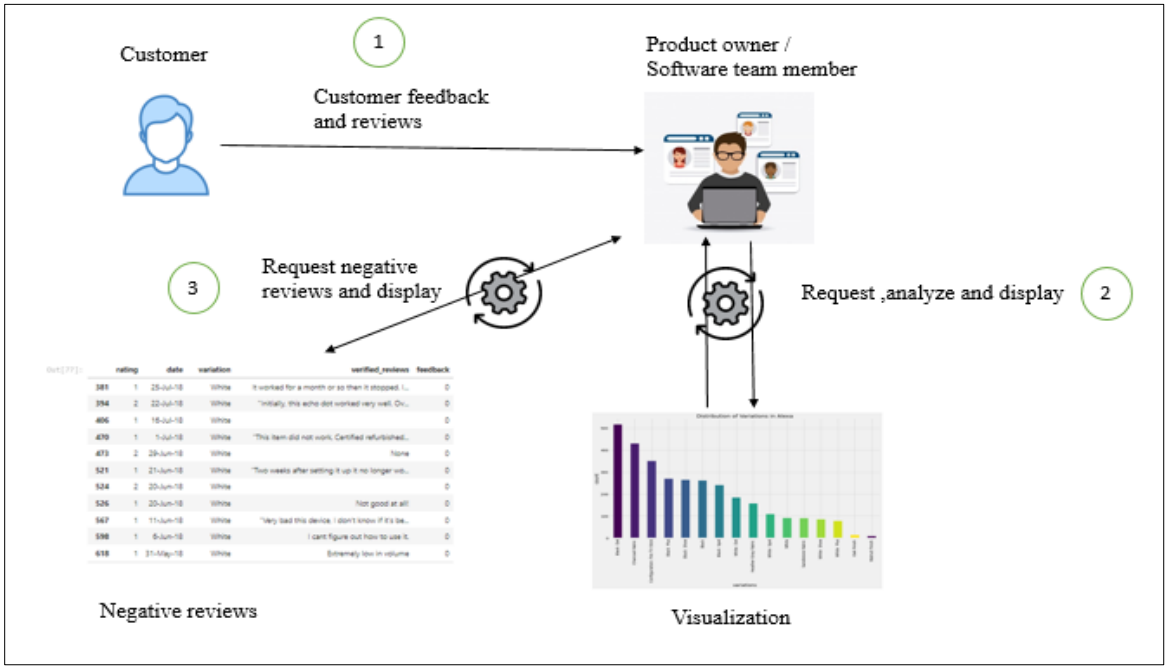


Figure 3.4: The User Interaction Process of Component 1

Component 2: Analyze the Future Variations, Software Impaction and Identify the Future Failures through Prediction

Analyze future impactions of previously released software versions through prediction and identify the future impact of the released s/w. This component's main task is to support the product owner to get an idea about the future impaction and failures of the released s/w versions. The system displays negative reviews of the most affected s/w products in the future according to analytics. Below is mentioned component 1 process diagram using Objective 2.5: Analyze Distributions of Alexa S/W Products and Identify Related Failures.

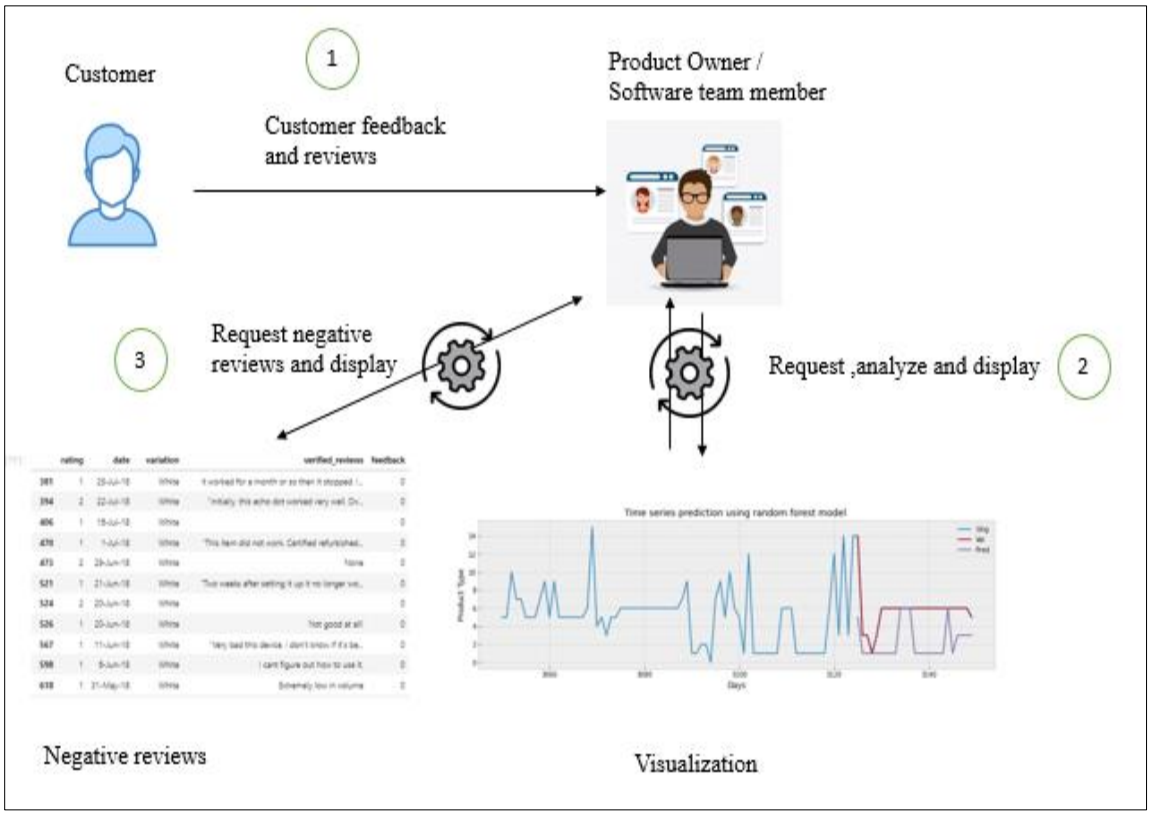


Figure 3.5: The User Interaction Process of Component 2

Component 3: Extract functional & non – functional enhancements using selected negative reviews through identified failures

Analyze and select negative reviews from component 1 and component 2 and display functional and non-functional negative reviews according to the analysis. This component's main task is to support the product owner to get an idea about functional and non-functional enhancements that need to be improved in the next level through selected negative reviews analysis.

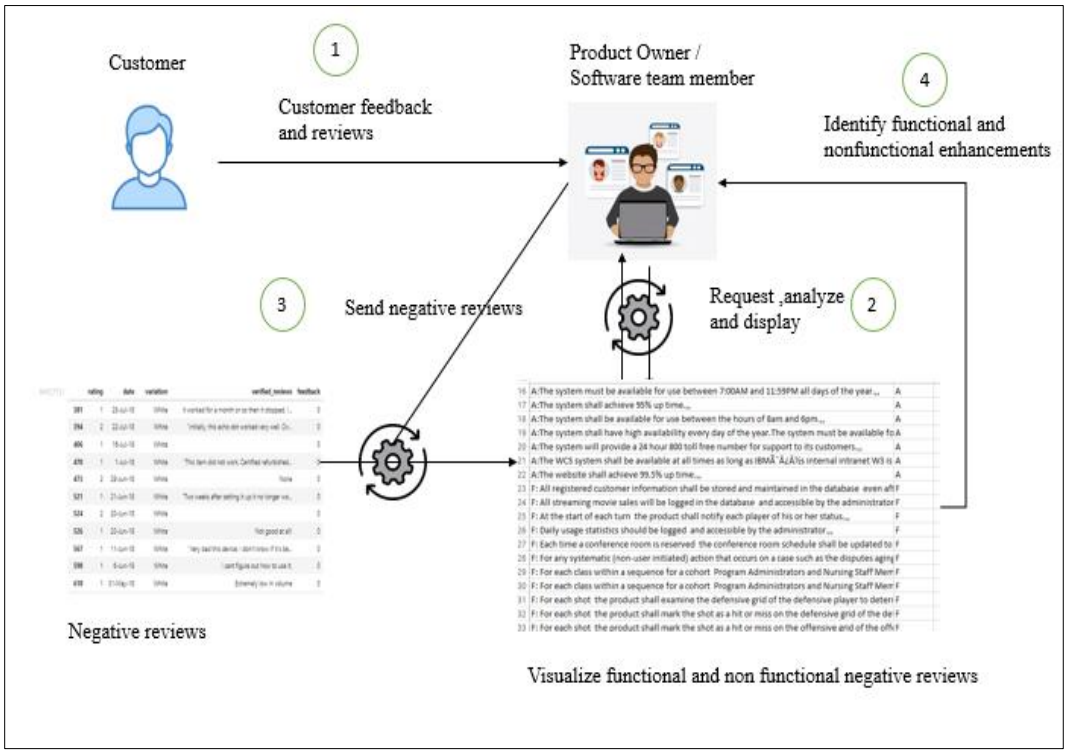


Figure 3.6: The User Interaction Process of Component 3

3.4. Implementation

The research report described the high-level architecture diagram and sub-component diagrams through the analysis and design section. Moreover, the requirement gathering section mentioned the functional and non-functional requirements of the proposed system.

The implementation section provides implementation details of each functional requirement (components) with aiding technologies support[24].

Component-1 Implementation: Analyzed existing level /status of the previously released software versions and identify the existing failures through negative reviews and feedback

The analysis and design section confirms the process of component 1 and this component construct by using 6 objectives to achieve the relevant task. This component's main task is to analyze the existing level /status of the previously released software versions and identify the existing failures through negative reviews and feedback. Below mentioned each implementation of the objectives for acquiring the implementation of the whole component 1 process.

- Component 1-Objective 1.1: Analyze Distributions of Alexa S/W Products and Identify Related Failures through Negative Reviews Analysis

The bar chart uses to represent the objective 1 task of component 1, and the y-axis represents the count of the distribution and the x-axis represents Alexa S/W product types. According to the bar chart representation, this objective represents negative reviews according to the user request and uses tables to represent data. Finally, users can get an idea about the failures of the released s/w product version by referring bar chart and negative reviews.

The requested results (bar chart) were implemented using python programming language, jupyter notebook, and related libraries.

```
#for advanced visualizations
import plotly.offline as py
from plotly.offline import init_notebook_mode, iplot
import plotly.graph_objs as go
from plotly import tools
init_notebook_mode(connected = True)
import plotly.figure_factory as ff

#read data from the CSV
data = pd.read_csv('amazon_alexa.tsv', delimiter = '\t', quoting = 3)

#Display Details
color = plt.cm.viridis(np.linspace(0, 1, 15))
data['variation'].value_counts().plot.bar(color = color, figsize = (15, 9))
plt.title('Distribution of Variations in Alexa', fontsize = 20)
plt.xlabel('variations')
plt.ylabel('count')
plt.show()

#Display Count of the distribution

print(data['variation'].value_counts())

#Display Negative reviews and producct details

products= data.loc[(data['variation'].str[0].isin(['W', 'O'])) & (data['feedback'] == 0)]
products
```

Figure 3.7: Implementation Process, and Libraries of the Objective 1.1

- Component 1-Objective 1.2: Analyze Feedback variation in Alexa and Identify Related Failures through Negative Reviews Analysis

The pie chart uses to construct objective 2 of component 1 and represent feedback variation in Alexa. According to the pie chart representation, this objective represents negative reviews according to the user request and uses tables to represent data. Finally, users can get an idea about the failures of the released s/w product version by referring pie chart and negative reviews.

The requested results (pie chart) were implemented using Python Programming Language, Jupyter Notebook, and related libraries.

```
# for advanced visualizations
import plotly.offline as py
from plotly.offline import init_notebook_mode, iplot
import plotly.graph_objs as go
from plotly import tools
init_notebook_mode(connected = True)
import plotly.figure_factory as ff

#read data from the CSV
data = pd.read_csv('amazon_alexa.tsv', delimiter = '\t', quoting = 3)

feedbacks = data['feedback'].value_counts()
label_feedback = feedbacks.index
size_feedback = feedbacks.values
colors = ['tan', 'gray']

feedback_piechart = go.Pie(labels = label_feedback,
                           values = size_feedback,
                           marker = dict(colors = colors),
                           name = 'Alexa', hole = 0.3)

df2 = [feedback_piechart]
layout = go.Layout(
          title = 'Distribution of Feedbacks in Alexa')
fig = go.Figure(data = df2,
                layout = layout)
py.iplot(fig)

#Display Negative reviews and related producct details
products= data.loc[(data['feedback'] == 0)]
products
```

Figure 3.8: Implementation Process, and Libraries of the Objective 1.2

- Component 1-Objective 1.3: Analyze Variation vs. Ratings in Alexa S/W Products and Identify Related Failures through Negative Reviews Analysis

The bivariate chart uses to construct objective 3 of component 1 and represent Alexa product types vs. ratings. According to the bivariate chart representation, this objective represents negative reviews according to the user request and uses tables to represent data. Finally, users can get an idea about the failures of the released s/w product version by referring bivariate chart and negative reviews.

The requested results (bivariate chart) were implemented using Python Programming Language, Jupyter Notebook, and related libraries.

```
# for advanced visualizations
import plotly.offline as py
from plotly.offline import init_notebook_mode, iplot
import plotly.graph_objs as go
from plotly import tools
init_notebook_mode(connected = True)
import plotly.figure_factory as ff

#read data from the CSV
data = pd.read_csv('amazon_alexa.tsv', delimiter = '\t', quoting = 3)

plt.rcParams['figure.figsize'] = (15, 9)
plt.style.use('fivethirtyeight')

sns.boxenplot(data['variation'], data['rating'], palette = 'winter')
plt.title("Variation vs Ratings")
plt.xticks(rotation = 90)
plt.show()

#Extract and Display Negative reviews and producct details
products= data.loc[(data['variation'].str[0].isin(['Walnut Finish', 'Oak Finish','B']))
                   & (data['feedback'] == 0)]
products
```

Figure 3.9: Implementation Process, and Libraries of the Objective 1.3

- Component 1-Objective 1.4: Analyzed Rating Variation in Alexa and Identify Related Failures through Negative Reviews Analysis

The pie chart uses to construct objective 4 of component 1 and represents Alexa rating variation. According to the pie chart representation, this objective represents negative reviews according to the user request and uses the table to represent data. Finally, users can get an idea about the failures of the released s/w product version by referring pie chart and negative reviews. *The requested results (pie chart) were implemented using Python Programming Language, Jupyter Notebook, and related libraries.*

```
#read data from the CSV
data = pd.read_csv('amazon_alexa.tsv', delimiter = '\t', quoting = 3)
ratings = data['rating'].value_counts()
print(ratings)
label_rating = ratings.index
size_rating = ratings.values
colors = ['green', 'lightblue', 'aqua', 'gold', 'crimson']
rating_piechart = go.Pie(labels = label_rating,
                         values = size_rating,
                         marker = dict(colors = colors),
                         name = 'Alexa', hole = 0.3)
df = [rating_piechart]

layout = go.Layout(
           title = 'Distribution of Ratings for Alexa')

fig = go.Figure(data = df,
                layout = layout)
py.iplot(fig)

#Display Negative reviews and related producct details
products= data.loc[((data['rating'] == 1) | (data['rating'] == 2)
                    | (data['rating'] == 3)) & (data['feedback'] == 0)]
products
```

Figure 3.10: Implementation Process, and Libraries of the Objective 1.4

- Component 1-Objective 1.5: Analyzed Sentiment Status (Negative, Positive, Neutral) in Alexa and Identify Related Failures through Negative Reviews Analysis

The pie chart uses to construct objective 5 of component 1 and represent Alexa existing sentiment status variation. According to the pie chart representation, this objective represents negative reviews according to the user request and uses the table to represent data. Finally, users can get an idea about the failures of the released s/w product version by referring pie chart and negative reviews. *The requested results (pie chart) were implemented using python programming language, Jupiter notebook, and related libraries and the system used sentiment analysis to analyze customer reviews to acquire positive, negative, and neutral percentages of Alexa.*

```
data = pd.read_csv('amazon_alexa.tsv', delimiter = '\t', quoting = 3)
print(data.head())
#describe the data
print(data.describe())
#check the in the dataset include any null value
print(data.isnull().sum())
#check what are the coloumns
print(data.columns)
#display the graph for ratings
ratings = data["rating"].value_counts()
numbers = ratings.index
quantity = ratings.values
custom_colors = ["skyblue", "yellowgreen", 'tomato', "blue", "red"]
plt.figure(figsize=(5, 5))
plt.pie(quantity, labels=numbers, colors=custom_colors)
central_circle = plt.Circle((0, 0), 0.5, color='white')
fig = plt.gcf()
fig.gca().add_artist(central_circle)
plt.rc('font', size=12)
plt.title("Amazon Alexa Reviews", fontsize=20)
plt.show()

# Amazon Alexa Reviews Sentiment Analysis
sentiments = SentimentIntensityAnalyzer()
data["Positive"] = [sentiments.polarity_scores(i)["pos"] for i in data["verified_reviews"]]
data["Negative"] = [sentiments.polarity_scores(i)["neg"] for i in data["verified_reviews"]]
data["Neutral"] = [sentiments.polarity_scores(i)["neu"] for i in data["verified_reviews"]]
print(data.head())
#Overall summation which sum the sentiment scores for each column to understand
#what most of the customers of Amazon Alexa think about it
x = sum(data["Positive"])
y = sum(data["Negative"])
z = sum(data["Neutral"])
```

Figure 3.11: Implementation Process, Code, and Libraries of the Objective 1.5

- Component 1-Objective 1.6: Analyzed Product Trending in Alexa S/W Products and Identify Failures through Negative Reviews Analysis

The star plot in the statistic is used to construct objective 6 of component 1 and represent product trending in Alexa s/w products. According to the star plot representation, this objective represents negative reviews according to the user request and uses the table to represent data. Finally, users can get an idea about the failures of the released s/w product version by referring to the star plot and negative reviews. *The requested results (star plot) were implemented using python programming language, Jupiter notebook, and related libraries.*

```
data = pd.read_csv('amazon_alexa.tsv', delimiter = '\t', quoting = 3)
print(type(data.date[0]))
data['date'] = pd.to_datetime(data['date'])
print(type(data.date[0]))
df = data.sort_values(by='date')
df
# reset the index
df.reset_index(inplace=True)
# delete the index
del df['index']
df.head()
#-----------------------------------------------------------------------

# x axis values
x = np.array(df['date'])
# corresponding y axis values
y = np.array(df['variation'])

plt.scatter(x, y, label= "stars", color= "green",
            marker= "*", s=30)

# x-axis label
plt.xlabel('date')
# frequency label
plt.ylabel('Variation')
# plot title
plt.title('Product Trending distribution')
# showing legend
plt.legend()
# function to show the plot
plt.show()
```

Figure 3.12: Implementation Process, and Libraries of the Objective 1.6

Component 2: Analyze the Future Variations, Software Impaction and Identify the Future Failures through Prediction

The analysis and design section confirmed the process of component 2 and this component included 5 objectives to achieve the component 2 task. This component is used to analyze future variations, and software impaction and identify future failures through prediction. These 5 objectives complete the process of component 2 and below mentioned each implementation of the objectives for acquiring the implementation of the whole component 2 process.

- Component 2-Objective 2.1: Predict Sentiment Status for Newly Added Customer Reviews and Identify Related Failures through Negative Reviews Analysis

According to the predictive analytics using past data, the system analyzes newly added reviews' sentiment status and identifies the negative reviews from newly received customer reviews. Moreover, users can be able to identify the failures of the ongoing released version's status.

When considering the implementation of objective 2.1, this section used *supervised learning and well-label data* for implementation. *Random forest classifier* is the best algorithm when compared with other classifications according to the objective 2.1 implementation requests and after accuracy testing. *The requested results were implemented using python programming language, Jupiter notebook, and related libraries*.

```
#-----------------------------------------------------------------------------------------------------
# Creating the Bag of Words model
"""Step 3: Tokenization"""
"""Step 4: Making the bag of words via sparse matrix and need to import countvectorizer"""

from sklearn.feature_extraction.text import TfidfVectorizer

cv = CountVectorizer(max_features = 1500)

count_matrix = cv.fit_transform(corpus)
x = count_matrix.toarray()
#df = pd.DataFrame(data=x,columns = cv.get_feature_names())

y = data.iloc[:, 4].values
#print (y)
#-----------------------------------------------------------------------------------------------------
"""Step 5: Splitting the dataset into the Training set and Test set"""
#splitting the dataset into Training and Test set
from sklearn.model_selection import train_test_split

# experiment with "test_size"
# to get better results -80/20
#if a fixed value is assigned like random_state = 0 or 1 or 42 then no matter how many times
#you execute your code the result would be the same
#.i.e, same values in train and test datasets.
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 0)

#-----------------------------------------------------------------------------------------------------
"""Step 6 : Random forest is an ensemble model (made of many trees) from sklearn.ensemble,
import RandomForestClassifier class"""
#Fitting Random Forest classifier with 100 trees to the Training set
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators = 100, criterion = 'entropy', random_state = 0)
classifier.fit(X_train, y_train)
```

Figure 3.13: Implementation Process, and Libraries of the Objective 2.1

```
#Newly added customer review
text = 'The sound is terrible and not working properly'
print(text)
x = clean(text)
vec = cv.transform([x])
predn=classifier.predict(vec)
print(predn)
```

Figure 3.14: Newly Added Customer Review Result

- Component 2-Objective 2.2: Dynamic Changing of the Future Feedback Variation and Identify Related Failures through Negative Reviews Analysis

This objective mainly focuses on feedback variation along with time prediction. Users can easily detect future feedback variation according to predict time changes. When the negative feedback percentage increases in the future (*implement to forecast 25 days*), users can extract the negative reviews based on the related negative feedback.

Objective 2.2 is implemented using python time series prediction with aiding python programming language, Jupiter notebook, and related libraries.

```
#show the model tree prediction
print('tree prediction for 25 days')
tree_prediction =tree.predict(x_future)
print(tree_prediction)
print()

#show the model lr prediction
print('lr prediction for 25 days')
lr_prediction =lr.predict(x_future)
print(lr_prediction)

print()
#show the model lr prediction
print('svr prediction for 25 days')
svr_prediction =svr.predict(x_future)
print(svr_prediction)

pos =df.tail(600)
#visualize the data
predictions =tree_prediction
valid =df[X.shape[0]:]
valid['Predictions'] = predictions
plt.figure(figsize=(10,2))
plt.title('Time series prediction using Tree model ')
matplotlib.pyplot.xlabel('Days')
matplotlib.pyplot.ylabel('Ratings')
plt.plot(pos['feedback'])
plt.plot(valid[['feedback', 'Predictions']])
plt.legend(['Orig', 'Val', 'Pred'])
plt.show()
```

Figure 3.15: Implementation Process, and Libraries of the Objective 2.2

- Component 2-Objective 2.3: Predict Star Rating value (1-5) for Newly Added Customer Reviews and Identify Related Failures through Negative Reviews Analysis

According to the predictive analytics using past data, the system analyzes newly added review's star ratings and identifies the negative reviews related to the fewer star ratings (1,2,3) from newly received customer reviews. Moreover, users can be able to identify the failures of the ongoing released version's status.

When considering the implementation of objective 2.3, this section used *supervised learning and well-label data* for implementation. *Random forest classifier* is the best algorithm when compared with other classifications according to the objective 2.1 implementation requests and after accuracy testing. *The requested results were implemented using the python programming language, Jupiter notebook, and related libraries*.

```
#-------------------------------------------------------------------------------------------------------------------
# Creating the Bag of Words model
"""Step 3: Tokenization"""
"""Step 4: Making the bag of words via sparse matrix and need to import countvectorizer"""
from sklearn.feature_extraction.text import TfidfVectorizer
cv = CountVectorizer(max_features = 1500)
count_matrix = cv.fit_transform(corpus)
x = count_matrix.toarray()
#df = pd.DataFrame(data=x,columns = cv.get_feature_names())
y = data.iloc[:, 4].values
#print (y)
#-------------------------------------------------------------------------------------------------------------------
"""Step 5: Splitting the dataset into the Training set and Test set"""
#splitting the dataset into Training and Test set
from sklearn.model_selection import train_test_split
# experiment with "test_size"
# to get better results -80/20
#if a fixed value is assigned like random_state = 0 or 1 or 42 then no matter how many times
#you execute your code the result would be the same
#.i.e, same values in train and test datasets.
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 0)

#-------------------------------------------------------------------------------------------------------------------
"""Step 6 : Random forest is an ensemble model (made of many trees) from sklearn.ensemble,
import RandomForestClassifier class"""
#Fitting Random Forest classifier with 100 trees to the Training set
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators = 100, criterion = 'entropy', random_state = 0)
classifier.fit(X_train, y_train)
```

Figure 3.16: Implementation Process, and Libraries of the Objective 2.3

- Component 2-Objective 2.4: Dynamic Changing Of The Future Star Rating Variation and Identify Related Failures through Negative Reviews Analysis

This objective mainly focuses on star rating variation along with time prediction. Users can easily detect the less star rating variation according to the predicted time change. When fewer star ratings (1, 2, 3) percentage increases in the future (*implement to forecast 25 days*), users can extract the negative reviews based on the related fewer star ratings.

Objective 2.4 is implemented using python time series prediction with aiding python programming language, Jupiter notebook, and related libraries.

```
df =df [['rating']]
df.head()
#create a variable to predict the 'x date out into the future'
future_day=25
#craete a new column(target) shifted 'x ' unit /days up
df["Prediction"] = df[["rating"]].shift(-future_day)
df.head()
df.tail()
#craete the feature data set (x) and convert it to a numpy array and remove the last 'x' rows/days
X=np.array(df.drop(['Prediction'],1))[:-future_day]
X
#Create the target data set (y) and convert it to a numpy array
y=np.array(df['Prediction'])[:-future_day]
y
#split the data into 75% traing and 25% testing
from sklearn.model_selection import train_test_split
xtrain, xtest, ytrain, ytest = train_test_split(X, y, test_size=0.25)
#decision tree
tree =DecisionTreeRegressor().fit(xtrain,ytrain)
#Linear
lr =LinearRegression().fit(xtrain,ytrain)
#svr
svr =LinearRegression().fit(xtrain,ytrain)
#Get the last 'x' rows of the feature data set
x_future =df.drop(['Prediction'],1)[:-future_day]
x_future=x_future.tail(future_day)
x_future=np.array(x_future)
x_future
```

Figure 3.17: Implementation Process and Libraries of the Objective 2.4

- Component 2-Objective 2.5: Analyze And Identify the Future Products Trending and Identify Related Failures through Negative Reviews Analysis

This objective mainly focuses on product trending variation along with time prediction. Users can easily detect the less product trending types according to to predict time changes (*implement to forecast 25 days*). Users can extract the negative reviews and product details based on the fewer products trending.

Objective 2.5 is implemented using python time series prediction with aiding python programming language, Jupiter notebook, and related libraries.

```
#show the model tree prediction
print('tree prediction for 25 days')
tree_prediction =tree.predict(x_future)
print(tree_prediction)
print()

#show the model lr prediction
print('lr prediction for 25 days')
lr_prediction =lr.predict(x_future)
print(lr_prediction)

print()
#show the model lr prediction
print('svr prediction for 25 days')
svr_prediction =svr.predict(x_future)
print(svr_prediction)

pos =df.tail(600)
#visualize the data
predictions =tree_prediction
valid =df[X.shape[0]:]
valid['Predictions'] = predictions
plt.figure(figsize=(10,2))
plt.title('Time series prediction using Tree model ')
matplotlib.pyplot.xlabel('Days')
matplotlib.pyplot.ylabel('variation')
plt.plot(pos['variation'])
plt.plot(valid[['variation', 'Predictions']])
plt.legend(['Orig', 'Val', 'Pred'])
plt.show()
```

Figure 3.18: Implementation Process, and Libraries of the Objective 2.5

Component 3: Extract Functional & Non – Functional Enhancements through and Negative Reviews send from Component 1 and Component 2

- Component 3-Objective 1: Extract Functional & Non – Functional Enhancements from Selected Negative Customer Reviews through Identified Failures

This objective help to extract functional and non-functional requirement from selected negative customer reviews through identified failures.

When considering the implementation of objective 3.1, this section used *unsupervised learning with text clustering* for implementation. *K-Mean* is the best algorithm when compared with other classifications according to objective 3.1 implementation requests and after accuracy testing. *The requested results were implemented using python programming language, Jupiter notebook, and related libraries*.

```
vectorizer = TfidfVectorizer(stop_words='english')
features = vectorizer.fit_transform(documents)
k = 2
model = KMeans(n_clusters=k, init='k-means++', max_iter=100, n_init=1)
model.fit(features)

data['cluster'] = model.labels_

data.tail()
# output the result to a text file.

clusters = data.groupby('cluster')

for cluster in clusters.groups:
    f = open('cluster'+str(cluster)+ '.csv', 'w',encoding="utf-8") # create csv file
    data = clusters.get_group(cluster)[['variation','verified_reviews']] # get title and overview columns
    f.write(data.to_csv(index_label='id')) # set index to id
    f.close()

print("Cluster centroids: \n")
order_centroids = model.cluster_centers_.argsort()[:, ::-1]
terms = vectorizer.get_feature_names()

for i in range(k):
    print("Cluster %d:" % i)
    for j in order_centroids[i, :15]: #print out 10 feature terms of each cluster
        print (' %s' % terms[j])
    print('------------')
```

Figure 3.19: Implementation Process, and Libraries of Component 3

3.5. Technology

This section describes a detailed description of the technology that is applied to implement the components. Moreover, described NLP and Machine Learning, Python, Anaconda, and Jupiter notebook.

NLP and Machine Learning

This research-based is mainly based on software engineering but carried on implementation section needs to use data mining. After identifying each feature and particular components of the suggested system, finalized the technology needed to be used to develop the system. NLP and machine language were used to develop this suggested system during the implementation stage.

This automated product backlog uses NLP and machine learning technology to analyze customer feedback from previously released versions, extract nonfunctional requirements, identify the quality levels, etc. Machine learning uses to analyze the software quality-oriented using a prediction model. To complete these tasks, use preprocessing, several algorithms, and finally understand the particular pattern. The important thing is to use the NLT Tool kit to deal with NLP and machine language[25].

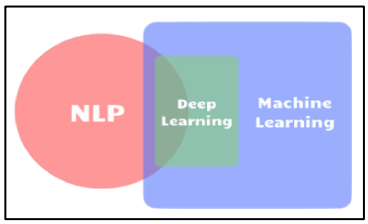


Figure 3.20: NLP and Machine Learning Interaction

Programming Language - Python

I used python language to implement each component. Python is a high-level language and this language constructs using an object-oriented approach and helps programmers to write clear and logical code[26].

Programming Tools or Software Development Tool

A programming tool or software development tool used to create, maintain, debug and otherwise support other programs and applications.

Jupyter Notebook**:** Jupiter notebook is used for scientific python development. This is a very productive tool and very interactive tool[27]. Anaconda Navigator: Anaconda Navigator is a desktop graphical user interface. It allows us to launch applications easily and can manage conda packages, environments and channels without using command-line commands[28].

3.7. Summary

Chapter 3 describes a detailed description of the research methodology and approach and covers the description of the requirement gathering, analysis and design, implementation, and technology usage

Chapter 4

Results and Evaluation

This section described a details description of the results generated from the implementation. According to the proposed approach, relevant results represent using advanced visualization (charts and tables) and each result has different relationships to acquire the outcome. Each results representation is explained using examples and mentioned the significance of the initiated results[29].

Moreover, this section separately mentioned results test accuracies and confirm the trustworthiness of the decision support system.

4.1. Decision Support Tools Results & Evaluation

Component 1-Component 1: Analyzed existing level /status of the previously released software versions and identify the existing failures through negative reviews

This component's main task is to support the product owner to get an idea about the existing status of the released S/W versions and identify the failures of the released versions through negative review analysis. This component uses advanced visualization to explain the variations of the features and select the *Amazon Alexa* dataset to represent the results. The product owner can get an idea about the released software version's status and related failures of the S/W versions using ratings, customer reviews, variations, and feedback visualizations.

According to the requirement gathering and component 1 diagram (see figure 3.2) in the research methodology and approach, the confirmed aim, each sub-objective, and the process of component 1. This component is based on six separate sub-objectives and each

objective initiates results according to the user request. These 6 sub-objective results represent the complete generated result of component 2 and below mentioned each result of the objectives and evaluation process, significance, and accuracy testing for acquiring the whole component 2 process with the best approach.

- Component 1-Objective 1.1: Analyze Distributions of Alexa S/W Products and Identify Related Failures through Negative Reviews Analysis

Out[77]:

	rating	date	variation	verified_reviews	feedback
381	1	25-Jul-18	White	It worked for a month or so then it stopped. I...	0
394	2	22-Jul-18	White	"Initially, this echo dot worked very well. Ov...	0
406	1	16-Jul-18	White		0
470	1	1-Jul-18	White	"This item did not work. Certified refurbished...	0
473	2	29-Jun-18	White	None	0
521	1	21-Jun-18	White	"Two weeks after setting it up it no longer wo...	0
524	2	20-Jun-18	White		0
526	1	20-Jun-18	White	Not good at all!	0
567	1	11-Jun-18	White	"Very bad this device, I don't know if it's be...	0
598	1	6-Jun-18	White	I cant figure out how to use it.	0
618	1	31-May-18	White	Extremely low in volume	0

Figure 4.1: Negative Reviews

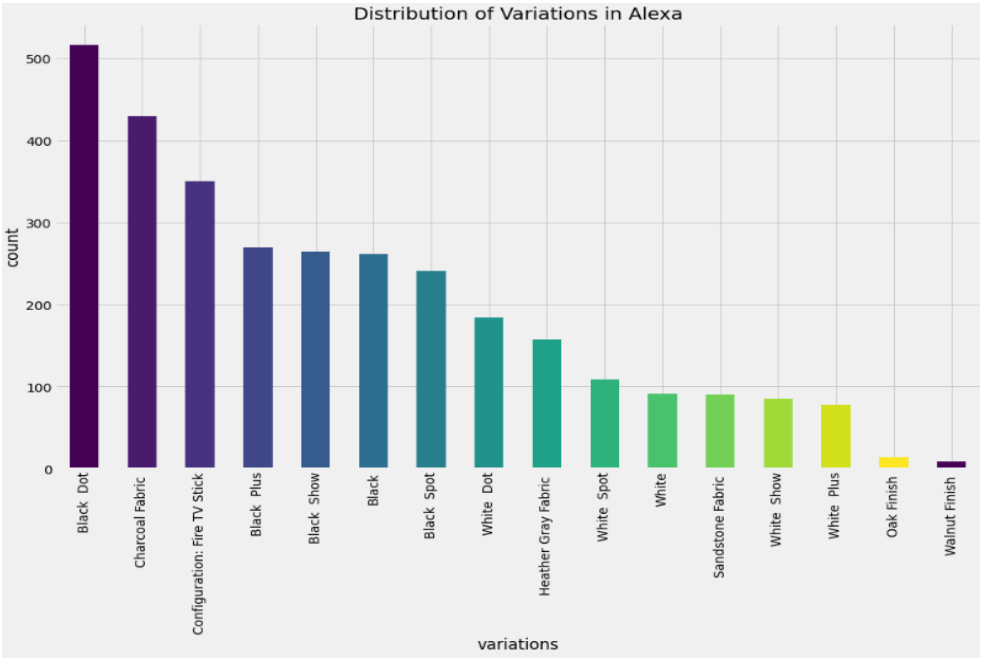


Figure 4.2: Distributions of Alexa S/W Products

Analyzed distribution of the Alexa s/w products and get the count of each product. Users can be able to identify the least popular products (Oak Finish and Walnut Finish have the least popular) among others through advanced visualization. The system extracts negative reviews and other related details of the least popular products according to the user's request.

- Component 1-Objective 1.2: Analyze Feedback variation in Alexa and Identify Related Failures through Negative Reviews Analysis

Analyzed feedback from Alexa and get the whole percentages of the positive and negative feedback.

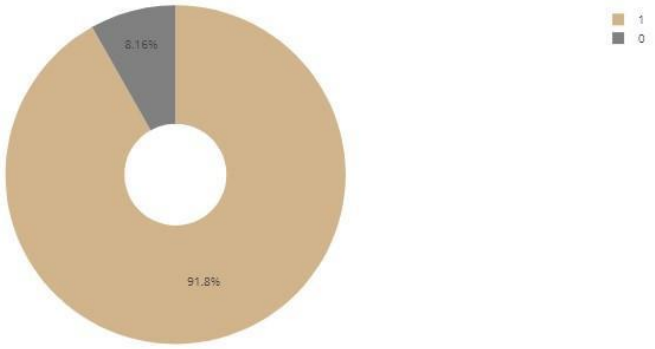


Figure 4.3: Feedback Variation in Alexa

Users can see the whole negative feedback percentage (8.16%) through advanced visualization. If the user needs to extract all negative reviews/comments based on the negative feedback percentage in Alexa, the user has the opportunity to extract all negative reviews/comments and related product details through system support.

- Component 1-Objective 1.3: Analyze Variation vs. Ratings in Alexa S/W Products and Identify Related Failures through Negative Reviews Analysis

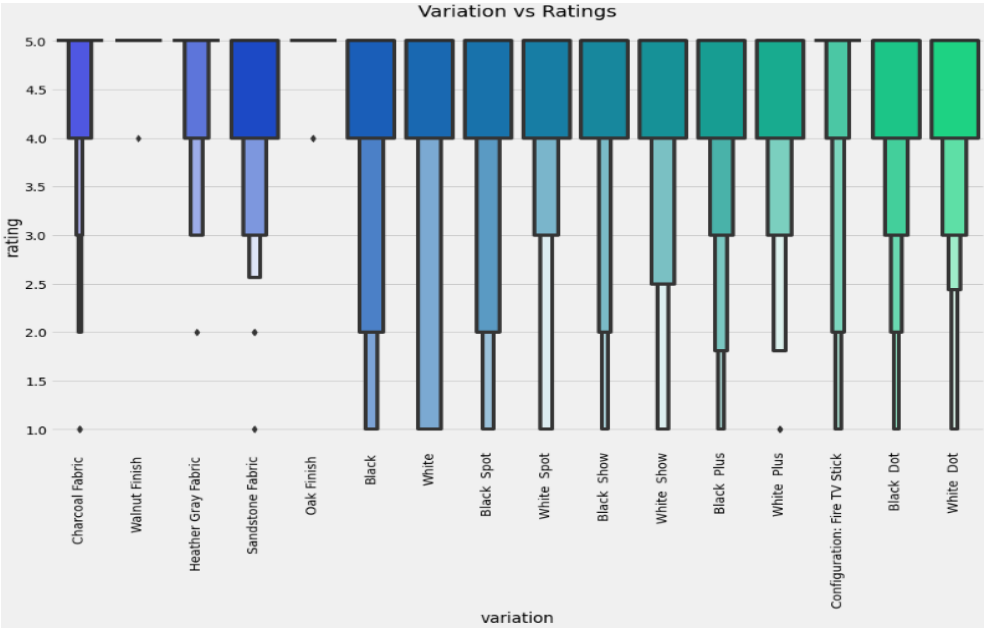


Figure 4.4: Variation Vs. Rating in Alexa S/W Products

This objective is one of the main concepts in component 1 and it makes a new form to reach the target. This is used to analyze s/w products distribution count vs. rating. (Black) S/W products have high populations and less ratings, and (Walnut Finish and Oak Finish) S/W products have less populations and high ratings. This objective help to identify the reasons for the above problems and failures by extracting negative reviews according to the user request.

- Component 1-Objective 1.4: Analyzed Rating Variation in Alexa and Identify Related Failures through Negative Reviews Analysis

This objective is used to analyze rating variation in Alexa and identifies the failures according to the ratings.

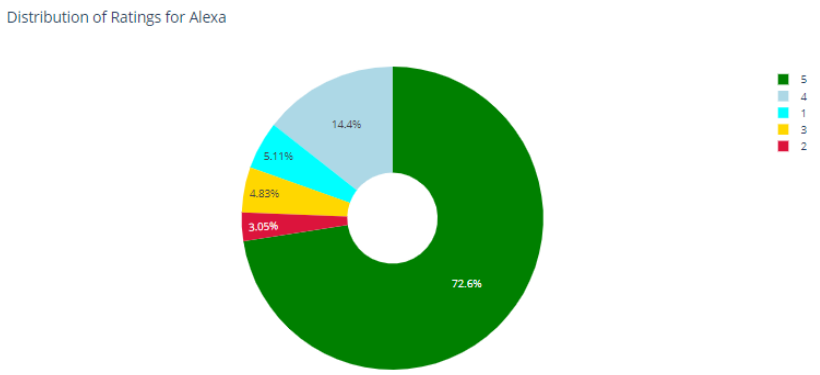


Figure 4.5: Rating Distribution in Alexa

According to the visualization, 3.05% gave a 2-star rating, 4.83% gave a 3-star rating, 5.11% gave a 1-star rating, and a total of 12.99% gave fewer ratings. Users need to be more concerned about 1, 2, and 3 fewer star ratings.[Assume: 1,2 and 3 star ratings mean less rating].This objective mainly extracts negative reviews based on the 1, 2, and 3 star ratings (less ratings) through the decision support tool according to the user request.

- Component 1-Objective 1.5: Analyzed Sentiment Status (Negative, Positive, Neutral) in Alexa and Identify Related Failures through Negative Reviews Analysis

```
Most of the customers of Amazon Alexa think about it :Neutral
Positive:  1020.1069999999986
Negative:  94.30299999999998
Neutral:  1954.592999999997

Positive: 32.38%
Negative: 2.99%
Neutral: 62.05%
```

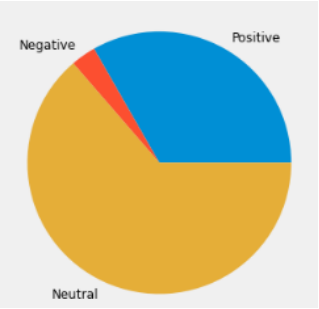


Figure 4.6: Sentiment Analysis

According to the pie chart, 2.99 % are negative and 62.05 % are neutral. Users need to be more concerned about negative and neutral percentage details. This objective mainly extracts reviews based on the neutral and negative through decision support tool according to the user request.

- Component 1-Objective 1.6: Analyzed Product Trending in Alexa S/W Products and Identify Related Failures through Negative Reviews Analysis

The below mentioned chart has fewer products trending and needs to be more concerned about these products than others. The decision support tool displays all less-trending product details and their negative reviews. Users can identify the failures of each product according to the less trending.

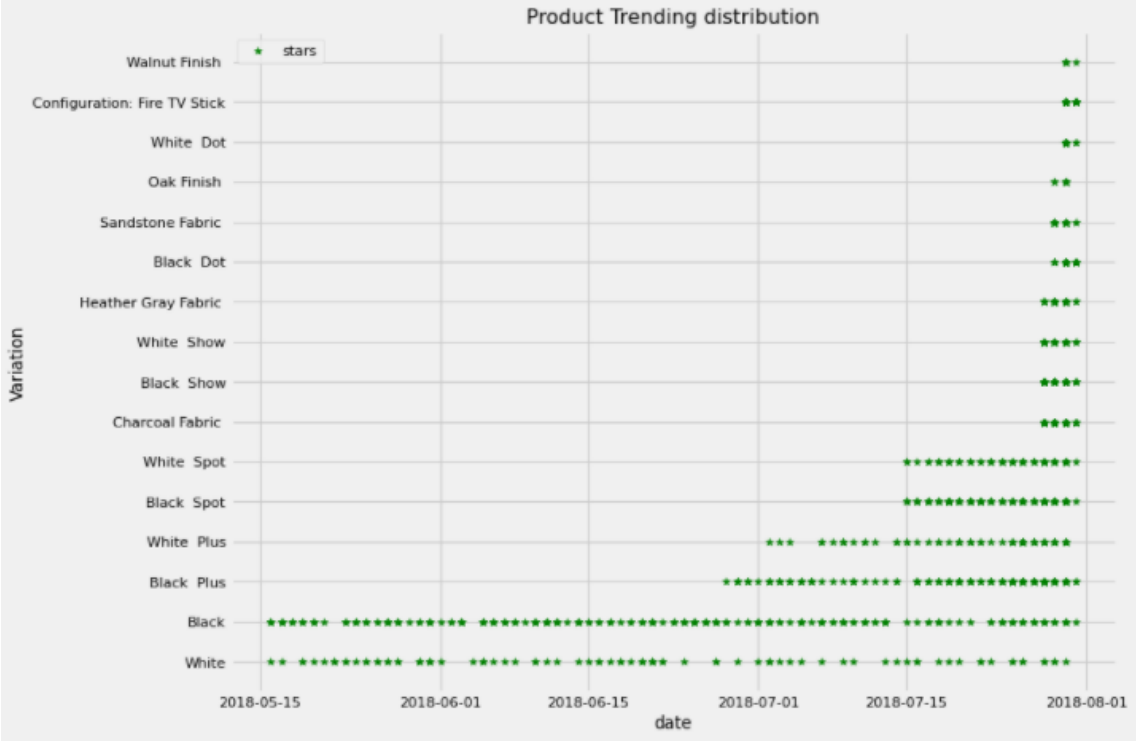


Figure 4.7: Product Trending Distribution

Component 2: Analyze the Future Variations, Software Impaction and Identify the Future Failures through Prediction

Analyze future impactions of previously released software versions through prediction and identify the future impact of the released s/w. This component's main task is to support the product owner to get an idea about the future impaction and failures of the released s/w versions. The system displays negative reviews of the most affected s/w products in the future according to analytics.

- Component 2-Objective 2.1: Predict Sentiment Status for Newly Added Customer Reviews and Identify Related Failures through Negative Review Analysis

According to predictive analytics, the system analyzes newly added reviews' sentiment status and identifies the negative reviews from newly received customer reviews.

```
Sound is terrible if u want good music too get a bose
[0]
------------------------------------------------------------
Love it! I’ve listened to songs I haven’t heard since childhood! I get the news, weather, information! It’s great!
[1]
------------------------------------------------------------
Not much features.
[0]
------------------------------------------------------------
```

Figure 4.8: Predict Sentiment Status

Decision support tool aids to extract negative reviews and failures from newly added reviews. This objective help identify the ongoing process of the s/w version using predictive analytics.

- Component 2-Objective 2.2: Dynamic Changing of the Future Feedback Variation and Identify Related Failures Negative Review Analysis

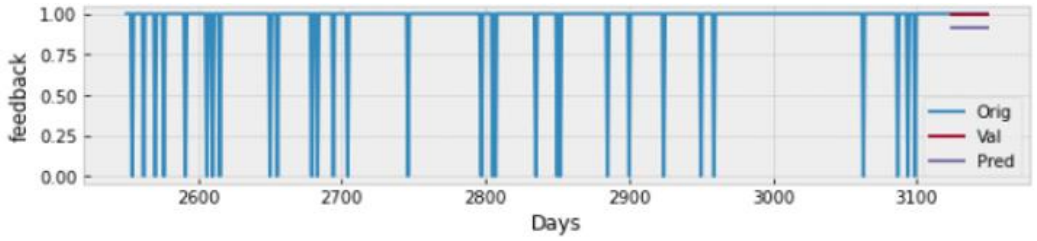


Figure 4.9: Dynamic Changing of the Feedback Prediction

This objective mainly focuses on feedback variation with time prediction (forecasting to 25 days). According to the above graph, the decision support tool displays dynamic changes in the future feedback variation and the user can easily detect the variation

according to time changes. When the negative prediction range increase by 50 % (the user can input a range like 60 %, or 10%) in the future, the system displays negative reviews related to the negative feedback according to the user request.

- Component 2-Objective 2.3: Predict Star Rating value (1-5) for Newly Added Customer Reviews and Identify Related Failures Negative Review Analysis

According to predictive analytics, the system analyzes newly added review star rating values and identifies the less star rating values based on negative reviews from newly received customer reviews.

```
[630 rows x 1 columns]
Sound is terrible if u want good music too get a bose
[2]
--------------------------------------------------------------
Love it! I’ve listened to songs I haven’t heard since childhood! I get the news, weather, information! It’s great!
[5]
--------------------------------------------------------------
Not much features.
[1]
--------------------------------------------------------------
```

Figure 4.10: Star Rating Values Prediction

Decision support tool aids to extract negative reviews and failures from newly added reviews. This objective help identify the ongoing process of the s/w version using predictive analytics.

- Component 2-Objective 2.4: Dynamic Changing Of The Future Star Rating Variation and Identity Related Failures Negative Review Analysis

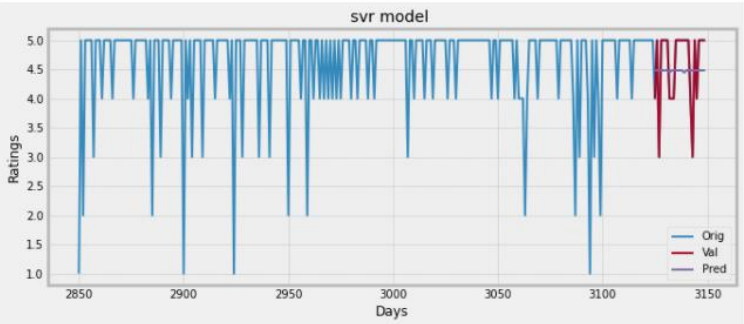


Figure 4.11: Dynamic Changing of the Star rating Values Prediction

This objective mainly focuses on star rating variation with time prediction (forecasting to 25 days). According to the above graph, the decision support tool displays dynamic changes in the future star rating variation and the user can easily detect the star rating variation according to time changes. When less star ratings (1, 2, or 3) increase than 50 % (user can input range like 60 %, 10%) in future, system display negative reviews related to the less star ratings (1,2,or 3) according to the user request.

- Component 2-Objective 2.5: Analyze And Identify the Future Products Trending and Identify Related Failures Negative Review Analysis

0	Black
1	Black Dot
2	Black Plus
3	Black Show
4	Black Spot
5	Charcoal Fabric
6	Configuration: Fire TV Stick
7	Heather Gray Fabric
8	Oak Finish
9	Sandstone Fabric
10	Walnut Finish
11	White
12	White Dot
13	White Plus
14	White Show
15	White Spot

Figure 4.12: Index & Product Types

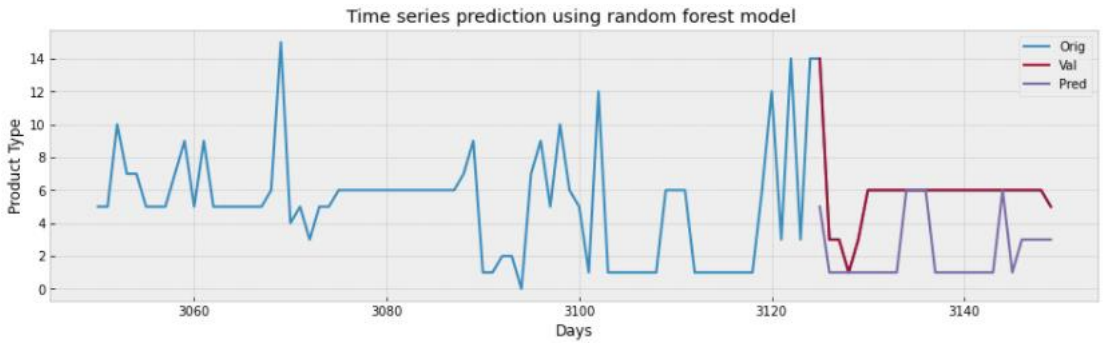


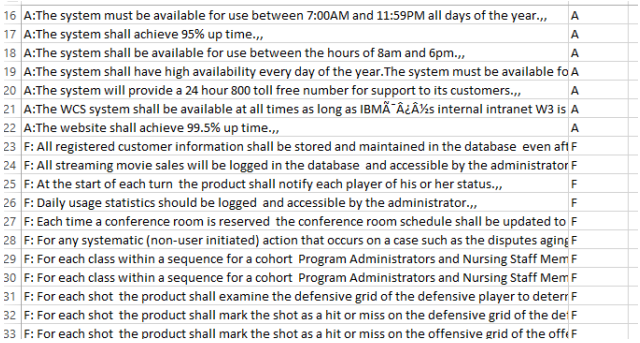
Figure 4.13: Dynamic Changing of the Products Trending Prediction

This objective mainly focuses on product trending variation along with time prediction. Users can easily detect the less product trending types according to to predict time changes (*implement to forecast 25 days*). User can extract the negative reviews and product details based on the less products trending.

Component 3: Extract Functional & Non – Functional Enhancements through Failures and Negative Reviews

Component 3 aid to extract functional and non-functional enhancements from selected negative customer reviews through identified failures from components 1 and component 2. Users can select availability, security, usability (nonfunctional), and functional enhancements using system support.

Types: Non –functional: Availability, Security, Usability, and Functional



16	A:The system must be available for use between 7:00AM and 11:59PM all days of the year.,,	A
17	A:The system shall achieve 95% up time.,,	A
18	A:The system shall be available for use between the hours of 8am and 6pm.,,	A
19	A:The system shall have high availability every day of the year.The system must be available fo	A
20	A:The system will provide a 24 hour 800 toll free number for support to its customers.,,	A
21	A:The WCS system shall be available at all times as long as IBMÃ¯Â¿Â½s internal intranet W3 is	A
22	A:The website shall achieve 99.5% up time.,,	A
23	F: All registered customer information shall be stored and maintained in the database even aft	F
24	F: All streaming movie sales will be logged in the database and accessible by the administrator	F
25	F: At the start of each turn the product shall notify each player of his or her status.,,	F
26	F: Daily usage statistics should be logged and accessible by the administrator.,,	F
27	F: Each time a conference room is reserved the conference room schedule shall be updated to	F
28	F: For any systematic (non-user initiated) action that occurs on a case such as the disputes aging	F
29	F: For each class within a sequence for a cohort Program Administrators and Nursing Staff Mem	F
30	F: For each class within a sequence for a cohort Program Administrators and Nursing Staff Mem	F
31	F: For each shot the product shall examine the defensive grid of the defensive player to deterr	F
32	F: For each shot the product shall mark the shot as a hit or miss on the defensive grid of the de	F
33	F: For each shot the product shall mark the shot as a hit or miss on the offensive grid of the off	F

Figure 4.14:Non-Functional and Functional Enhancements

4.2. Testing Accuracy Level

The previous section describes the results/evaluation, and the significance of each objective along with its components. The tool's trustworthiness and accuracy of the process depend on the accuracy of the component's initiated results. These results generate using different types of models and the model's accuracy depends on the metrics. According to the above factor, the result's accuracy depends on the model's accuracy, selected preprocessing techniques and dataset, etc. According to the research findings, below mentioned specific

model evaluation metrics and preprocessing techniques which used to analyze the accuracy of the components[30].

Model Evaluation Metrics

-*Classification problems*: Classification accuracy, Classification Error, Confusion Metrics and Classification Report[31]

-*Regression problems*: Mean Absolute Error, Mean Squared Error, Root Mean Squared Error

Preprocessing Techniques

-Used more data cleansing techniques to get accurate data

```
for i in range(0, 3150):

    #column : "Review", row ith
    review = re.sub('[^a-zA-Z]', ' ', data['verified_reviews'][i])

    #Convert each word into its lower case
    review = review.lower()

    #split to array(default delimiter is " ")
    review = review.split()


    #creating PorterStemmer object to
    #take main stem of each word
    ps = PorterStemmer()
    review = [ps.stem(word) for word in review if not word in set(stopwords.words('english'))]


    #rejoin all string array elements
    #to create back into a string
    review = ' '.join(review)

    # append each string to create
    # array of clean text
    corpus.append(review)
```

Figure 4.15: Data Preprocessing Techniques

- Accuracy Level of Component 2-Objective 2.1: Predict Sentiment Status for Newly Added Customer Reviews and Identify Related Failures through Negative Review Analysis

According to (see figure 3.3), the system analyzes newly added review's sentiment status and identifies the negative reviews from newly received customer reviews. According to the objective process analysis, this objective is based on the classification problem.

This section use supervised learning because the dataset has well-label data to perform the analysis. The Random forest classifier is the best algorithm when compared with other classifications. Classification accuracy, classification error, and confusion metrics are used to analyze the accuracy status of objective 2.1 and enhance the random forest classifier performance and use training and testing data to test the result.

Classification Accuracy - The probability of correct prediction in model

```
Decision Tree Accuracy= 0.9317460317460318
Random Forest= 0.9428571428571428
SVM accuracy= 0.9253968253968254
LC accuracy= 0.9412698412698413
gb accuracy= 0.5349206349206349
```

Figure 4.16: Classification Accuracy of the Objective 2.1

Classification Accuracy = (True Positives + True Negatives) / (True Positives + False Positives + False Negatives + True Negatives)

Classification accuracy of the model when using Random forest classifier: 94.28%

Classification Error of the Model (Random forest classifier)

Classification Error: 0.05

Analyzed Predicted Test Results Accuracy using Confusion Metrics (Random forest classifier)

```
[[ 18  36]
 [  0 576]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.33      0.50        54
           1       0.94      1.00      0.97       576

    accuracy                           0.94       630
   macro avg       0.97      0.67      0.73       630
weighted avg       0.95      0.94      0.93       630

0.9428571428571428
```

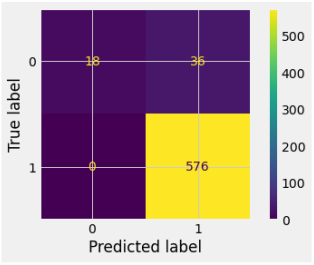


Figure 4.17: Confusion Metrics and Classification Report of the Objective 2.1

36+0= 36 incorrect predictions and 18+576 = 594 correct prediction

In conclusion, the Random Forest classifier algorithm with 100 trees worked efficiently to train the model in predicting positive and negative reviews made by customers, with an overall accuracy of 94.28%.

The Process of Improving Random Forest Classifier Accuracy

```
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 1)

#-----------------------------------------------------------------------------------------------------------------------
"""Step 6 : Random forest is an ensemble model (made of many trees) from sklearn.ensemble, import RandomForestClassifier class""
#Fitting Random Forest classifier with 100 trees to the Training set
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators = 100, criterion = 'entropy', random_state = 1)
classifier.fit(X_train, y_train)
```

Figure 4.18: Random Forest Classification Usage

- Used Random Forest Classification Algorithm to predict the negative and positive status using the feedback column. I select test_size = 0.2 and train_size =0.8. It takes less training time when compared with other algorithms[32].

- Getting the n_estimator value as 100 and the greater number of trees in the forest leads to higher accuracy and avoids the problem of fitting[32].

- Use some actual values in the feature variable of the dataset such as the feedback column so that the classifier can predict accurate results rather than a guessed result[32].

Accuracy Level of Component 2-Objective 2.3: Predict Star Rating value (1-5) for Newly Added Customer Reviews and Identify Related Failures Negative Review Analysis

According to (see figure 3.2), the system analyzes newly added review's star rating value and identifies the less star rating values based on negative reviews from newly received customer reviews. According to the objective process analysis, this objective is based on the classification problem.

This section use supervised learning because the dataset has well-label data to perform the analysis. Random forest classifier is the best algorithm when compared with other classifications and this objective use *Multi-Class Classification.*
Classification accuracy, classification error, and confusion metrics are used to analyze the accuracy status of the objective 2.1 and enhance the random forest classifier performance and use training and testing data to test the results.

Classification Accuracy - The probability of correct prediction in model

```
Decision Tree Accuracy= 0.7412698412698413
Random Forest= 0.807936507936508
SVM accuracy= 0.7666666666666667
LC accuracy= 0.7761904761904762
gb accuracy= 0.32063492063492066
```

Figure 4.19: Classification Accuracy of the 2.3.

Classification Accuracy = (TruePositives_1 + TruePositives_2) / ((TruePositives_1 + TruePositives_2) + (FalsePositives_1 + FalsePositives_2))

Classification accuracy of the model when use Random forest classifier – 80.79%

Classification Error of the model (Random forest classifier)

Classification Error =0.19

Analyzed predicted test results accuracy using confusion metrics (Random forest classifier)

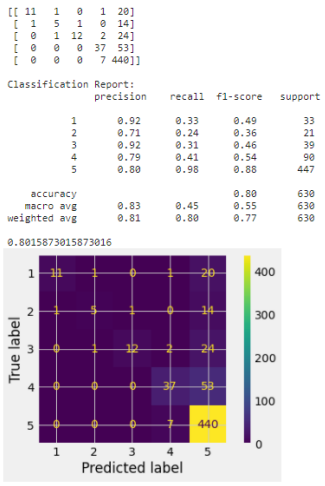


Figure 4.20: Confusion Metrics and Classification Report

Accuracy Level of Component 2- Time series prediction objectives (2.2, 2.4, 2.5, and 2.6)

According to research findings, component 2 time series prediction objectives (2.2, 2.4, 2.5, and 2.6) construct using regression. When considering the regression, mean absolute error, mean squared error, and root means squared error are the main metrics to calculate the accuracy[33].

- Accuracy Level of Component 2-Objective 2.2: Dynamic Changing of the Future Feedback Variation and Identify Related Failures through Negative Reviews Analysis

This objective mainly focuses on feedback variation with time prediction (forecasting to 25 days). According to the dynamic variation plot (see figure 4.9), the decision support tool display dynamic changing in the future feedback variation and the user can easily detect the variation according to time change.

This objective 2.2 construct by using time series prediction with regression algorithms (linear regression, decision tree regression, and support vector machines) and selected mean absolute error, mean squared error, and root mean squared error metrics to calculate the accuracy of objective 2.2.

```
tr Mean Absolute Error:  0.1444086834585822
tr Mean Squared Error :  0.06574884911607498
tr Root Mean Squared Error : 0.2564153839302061

lr Mean Absolute Error:  0.14440868345858213
lr Mean Squared Error :  0.06574884911607498
lr Root Mean Squared Error : 0.2564153839302061

svr Mean Absolute Error:  0.14440868345858213
svr Mean Squared Error :  0.06574884911607498
svr Root Mean Squared Error : 0.2564153839302061
```

Figure 4.21: Accuracy Values of the Objective 2.2

- Accuracy Level of Component 2-Objective 2.4: Dynamic Changing Of The Future Star Rating Variation and Identify Related Failures through Negative Reviews Analysis

This objective mainly focuses on star rating variation with time prediction (forecasting to 25 days). According to the dynamic variation plot(see figure 4.11), the decision support tool display dynamic changing in the future star rating variation and the user can easily detect the star rating variation according to time changes.

Objective 2.4 was constructed by using time series prediction with regression algorithms (linear regression, decision tree regression, and support vector machines) and selected mean absolute error, mean squared error, and root means squared error metrics to calculate the accuracy of objective 2.4.

```
tr Mean Absolute Error:  0.7796367414170228
tr Mean Squared Error :  1.1738564305630514
tr Root Mean Squared Error : 1.083446551779575

lr Mean Absolute Error:  0.7792897997521782
lr Mean Squared Error :  1.1697472074118276
lr Root Mean Squared Error : 1.0815485229113984

svr Mean Absolute Error:  0.7792897997521782
svr Mean Squared Error :  1.1697472074118276
svr Root Mean Squared Error : 1.0815485229113984
```

Figure 4.22: Regression Metrics Errors Objective 2.4

- Accuracy Level of Component 2-Objective 2.5: Analyze And Identify the Future Products Trending and Identify Related Failures through Negative Reviews Analysis

This objective mainly focuses on product trending variation along with time prediction. According to the dynamic variation plot (see figure 4.13), the user can easily detect the less

product trending types according to the predicted time change (*implement to forecast 25 days*).

This objective 2.5 is constructed by using time series prediction with regression algorithms (linear regression, decision tree regression, and support vector machines) and selected mean absolute error, mean squared error, and root means squared error metrics to calculate the accuracy of objective 2.5.

```
tr Mean Absolute Error:  0.1419610865175302
tr Mean Squared Error :  0.06581845637686776
tr Root Mean Squared Error : 0.2565510794693091

lr Mean Absolute Error:  0.1419610865175302
lr Mean Squared Error :  0.06581845637686776
lr Root Mean Squared Error : 0.2565510794693091

svr Mean Absolute Error:  0.1419610865175302
svr Mean Squared Error :  0.06581845637686776
svr Root Mean Squared Error : 0.2565510794693091
```

Figure 4.23: Regression Metrics Errors Objective 2.5

- Accuracy Level of Component 3-Objective 3.1: Extract Functional & Non – Functional Enhancements using Selected Negative Reviews through Identified Failure

Component 3 –Objective 3.1 cluster variation calculate using silhouette score,calanski harabazs ,distortion score and david bouldin.

- Accuracy Level of Component 3-Objective 3.2: Confirm Text Clustering using External Dataset

Component 3 –Objective 3.2 cluster variation calculate using silhouette score,calanski harabazs ,distortion score and david bouldin.

1.3.Summary

Chapter 4 describes a detailed description of the results and evaluation by using advanced visualization. Moreover described the significance of the initiative results. Finally, the report mentioned the accuracy levels of each component through objective accuracy testing and analysis.

Chapter 5

Conclusion

5.1. Introduction

This section describes the research conclusion, commercialization aspects of the research, research challenges, and future work. According to the summarization, this section aid to understand the project status, components and features clearly.

The research described a particular section of the agile software development process model and enclosed problems related to gathering and analyzing customer feedback/reviews from released software versions. The research only focuses on extracting failures and identifying the enhancements of previously released software versions using customer feedback/reviews analysis. After evaluating the results of the research, confirmed the plans and strategies of the software quality improvements according to the identified failures and future enhancements.

The suggested system include three components and each component aid to analyze customer feedback and reviews using various processes. These components are “Analyze Existing Level /Status of the Previously Released Software Version and Identifying Enhancement through Negative Review Analysis, Analyze the Future Variations, Software Impaction and Identify Enhancements through Negative Review Analysis, Extract Functional & Non – Functional Enhancements from Selected Negative Reviews”.

The proposed system helps to analyze the customer feedback and reviews status and this tool helps to extract failures and enhancements. Moreover, the Product owner/software teams can get an idea about the released version's ongoing situation, future impaction and support to improve the next version level features before releasing. To create an accurate software suggested model, use a matrix based on NLP and machine language. The most

important factor is that this decision support system avoids the problems related to the unclear requirements gathering and analysis process.

5.2. Commercialization aspects of the project

This suggested system is a decision support system for the agile development process and mainly supports for the product owner to analyze the customer feedback. The software team can select the correct way to be driven within the development process. In addition, it can be mentioned as an extension sub-system for automated product backlog as a web app. The research plan helps them to uplift their business and their software development without any hesitation. Furthermore, this system can commercialize through the internet as a website. We can aware of this system via newspapers, magazines, the internet, and using social media[34].

5.3. Research Challenges and Future Improvements

Creating a decision support tool is the most useful suggested system for the software development process. This system not only focuses on the agile scrum approach, can be applied to other software development process models. When considering the research challenges, below specific points can consider as the main challenges that faced during research studies and implementation[35].

- Improvement process of the trustworthiness / accuracy level for system
- Research ethical approaches
- User-based technology awareness
- Technology suitability
- Pre Processing
- Algorithms Selection Process
- Results Testing

As future improvements, below mentioned some specific suggestions that can use for next research level improvement.

- Try to improve the trustworthiness of the data science meditated system and more consider about accuracy level of the selected algorithms.
- Focuses on increasing the accuracy level of each components.
- Using customer feedback, try to increase the performance of the app.
- Use more effective data cleansing and transformation techniques and more focuses on research ethical improvements.
- This system not only focuses on agile scrum approach, can be apply for other software development process model

5.5. Summary

This document describes research abstraction, introduction, the problem statement, aim and objectives, literature review, research methodology and approach, results and evaluation, technologies, commercialization and aspects of the project, research challenges and future work.

Chapter 6

References

[1] “SDLC - Overview.” https://www.tutorialspoint.com/sdlc/sdlc_overview.htm (accessed Jun. 20, 2022).

[2] P. Jain, A. Sharma, and L. Ahuja, “The Impact of Agile Software Development Process on the Quality of Software Product,” in *2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, Aug. 2018, pp. 812–815. doi: 10.1109/ICRITO.2018.8748529.

[3] A. Srivastava, S. Bhardwaj, and S. Saraswat, “SCRUM model for agile methodology,” in *2017 International Conference on Computing, Communication and Automation (ICCCA)*, May 2017, pp. 864–869. doi: 10.1109/CCAA.2017.8229928.

[4] “The Problems With Agile and Scrum | PagerDuty.” https://www.pagerduty.com/blog/problems-with-agile-scrum/ (accessed Jun. 20, 2022).

[5] “Scrum Methodology and Project Management.” https://www.mountaingoatsoftware.com/agile/scrum (accessed Jun. 20, 2022).

[6] T. Sedano, P. Ralph, and C. Péraire, “The Product Backlog,” in *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*, May 2019, pp. 200–211. doi: 10.1109/ICSE.2019.00036.

[7] G. Koç and M. Aydos, “Trustworthy scrum: Development of secure software with scrum,” in *2017 International Conference on Computer Science and Engineering (UBMK)*, Oct. 2017, pp. 244–249. doi: 10.1109/UBMK.2017.8093383.

[8] “Aims and Objectives - Guide for Thesis and Dissertations.” https://www.discoverphds.com/advice/doing/research-aims-and-objectives (accessed Jun. 20, 2022).

[9] "Identifying Research Gaps to Pursue Innovative Research - Enago Academy." https://www.enago.com/academy/identifying-research-gaps-to-pursue-innovative-research/ (accessed Jun. 20, 2022).

[10] "Mopinion Feedback for Websites, Apps and Email." https://mopinion.com/ (accessed Jun. 20, 2022).

[11] "13 Ways to Collect Customer Feedback for Your Website | Blog." https://qualaroo.com/blog/collect-customer-feedback-for-your-website/ (accessed Jun. 20, 2022).

[12] "Top 15 Website Feedback Tools: Full Comparison Guide [Updated for 2022]." https://marker.io/blog/website-feedback-tools (accessed Jun. 20, 2022).

[13] "Literature Reviews," *The Writing Center • University of North Carolina at Chapel Hill*. https://writingcenter.unc.edu/tips-and-tools/literature-reviews/ (accessed Jun. 20, 2022).

[14] W. M. Farid and F. J. Mitropoulos, "NORMATIC: A visual tool for modeling Non-Functional Requirements in agile processes," in *2012 Proceedings of IEEE Southeastcon*, Mar. 2012, pp. 1–8. doi: 10.1109/SECon.2012.6196989.

[15] D. Domah and F. J. Mitropoulos, "The NERV methodology: A lightweight process for addressing non-functional requirements in agile software development," in *SoutheastCon 2015*, Apr. 2015, pp. 1–7. doi: 10.1109/SECON.2015.7133028.

[16] N. Govil and A. Sharma, "Information Extraction on Requirement Prioritization Approaches in Agile Software Development Processes," in *2021 5th International Conference on Computing Methodologies and Communication (ICCMC)*, Apr. 2021, pp. 1097–1100. doi: 10.1109/ICCMC51019.2021.9418285.

[17] A. M. Alsalemi and E.-T. Yeoh, "A survey on product backlog change management and requirement traceability in agile (Scrum)," in *2015 9th Malaysian Software Engineering Conference (MySEC)*, Dec. 2015, pp. 189–194. doi: 10.1109/MySEC.2015.7475219.

[18] S. Sachdeva, A. Arya, P. Paygude, S. Chaudhary, and S. Idate, "Prioritizing User Requirements for Agile Software Development," in *2018 International Conference On Advances in Communication and Computing Technology (ICACCT)*, Feb. 2018, pp. 495–498. doi: 10.1109/ICACCT.2018.8529454.

[19] "Research Methodology | Examples, Approaches & Techniques - Video & Lesson Transcript," *Study.com*. https://study.com/learn/lesson/research-methodology-examples-approaches-techniques.html (accessed Jun. 20, 2022).

[20] "Requirements Gathering: A Complete Step-by-Step Guide (2022)." https://nmgtechnologies.com/blog/requirement-gathering-solve-biggest-problems-consulting.html (accessed Jun. 20, 2022).

[21] "Non-functional Requirements: Examples, Types, Approaches | AltexSoft." https://www.altexsoft.com/blog/non-functional-requirements/ (accessed Jun. 20, 2022).

[22] "Study Design and Analysis | Research Connections." https://www.researchconnections.org/research-tools/study-design-and-analysis (accessed Jun. 20, 2022).

[23] "Difference between High Level Design and Low Level Design - GeeksforGeeks." https://www.geeksforgeeks.org/difference-between-high-level-design-and-low-level-design/ (accessed Jun. 20, 2022).

[24] "Implementation research: what it is and how to do it | The BMJ." https://www.bmj.com/content/347/bmj.f6753 (accessed Jun. 20, 2022).

[25] "Machine Learning (ML) for Natural Language Processing (NLP) - Lexalytics." https://www.lexalytics.com/blog/machine-learning-natural-language-processing/ (accessed Jun. 20, 2022).

[26] "Introduction to Python." https://www.w3schools.com/python/python_intro.asp (accessed Jun. 20, 2022).

[27] "Project Jupyter." https://jupyter.org (accessed Jun. 20, 2022).

[28] "Anaconda Navigator — Anaconda documentation." https://docs.anaconda.com/anaconda/navigator/ (accessed Jun. 20, 2022).

[29] "Evaluation Research: Definition, Methods and Examples | QuestionPro." https://www.questionpro.com/blog/evaluation-research-definition-methods-and-examples/ (accessed Jun. 20, 2022).

[30] J. Brownlee, "Difference Between Classification and Regression in Machine Learning," *Machine Learning Mastery*, Dec. 10, 2017. https://machinelearningmastery.com/classification-versus-regression-in-machine-learning/ (accessed Jun. 20, 2022).

[31] “Classification: Accuracy | Machine Learning Crash Course | Google Developers.” https://developers.google.com/machine-learning/crash-course/classification/accuracy (accessed Jun. 20, 2022).

[32] “The Ultimate Guide to Random Forest Regression.” https://www.keboola.com/blog/random-forest-regression (accessed Jun. 20, 2022).

[33] “Time Series Forecast Error Metrics you should know | Towards Data Science.” https://towardsdatascience.com/time-series-forecast-error-metrics-you-should-know-cc88b8c67f27 (accessed Jun. 20, 2022).

[34] “Five steps to successful commercialization of an innovation.” https://www.valmet.com/media/articles/experts-voice/five-steps-to-successful-commercialization-of-an-innovation/ (accessed Jun. 20, 2022).

[35] “7 Research Challenges (And how to overcome them) | Articles | Walden University.” https://www.waldenu.edu/news-and-events/publications/articles/2010/01-research-challenges (accessed Jun. 20, 2022).

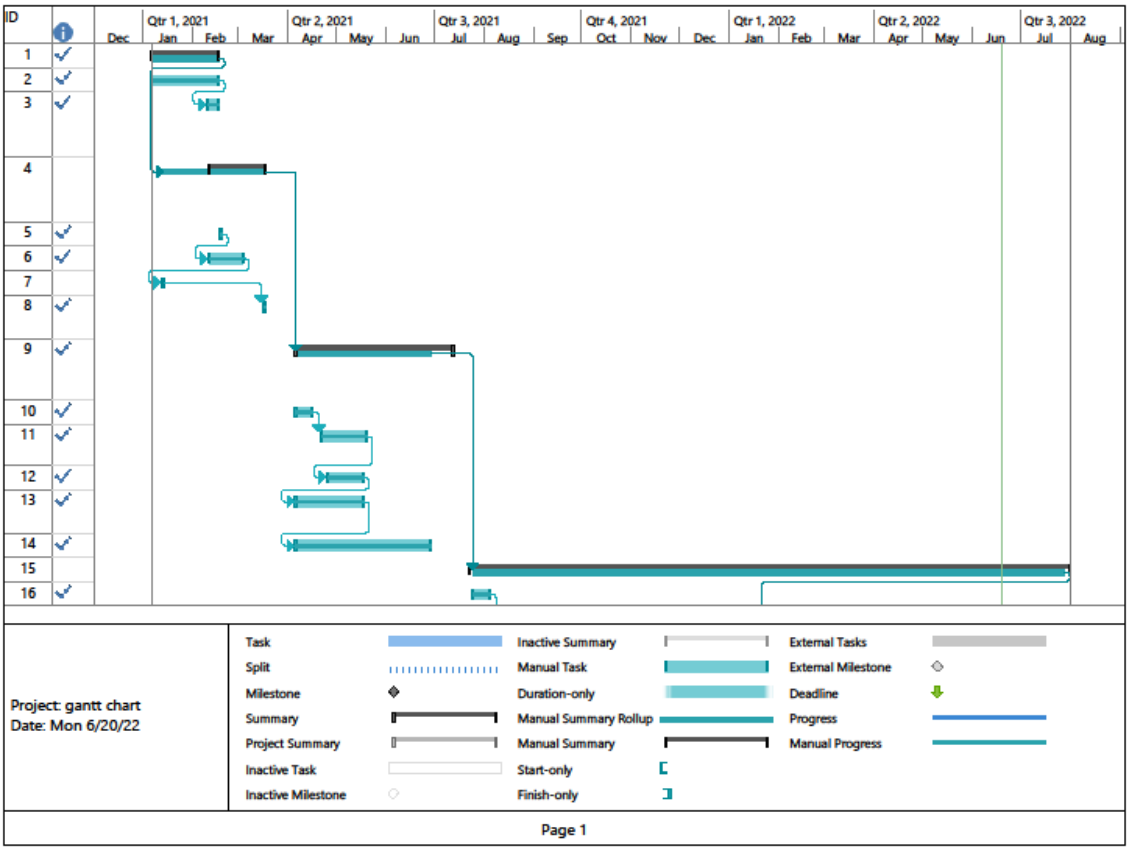
Appendixes

Appendix A –The Research Project Progress

ID	Task Mode	Task Name	Duration	Start	Finish	Predecessors	Resource Names
1		**Finding Research Topic**	**30 days**	**Wed 1/6/21**	**Tue 2/16/21**		
2		Refer Research Papers	30 days	Wed 1/6/21	Tue 2/16/21		
3		Write document about selected Topic	5 days	Wed 2/10/21	Tue 2/16/21		
4		**Research Topic Approved Process and Finding Supervisor**	**25 days**	**Thu 2/11/21**	**Wed 3/17/21**		
5		Supervisor Discussion	1 day	Thu 2/18/21	Thu 2/18/21		
6		Refer Research Papers	15 days	Thu 2/11/21	Wed 3/3/21		
7		Systematic Review	10 days	Sun 2/21/21	Thu 3/4/21		
8		Approve Topic and Select Supervisor	1 day	Wed 3/17/21	Wed 3/17/21		
9		**Research Interim Presentation and Report Writing**	**70 days**	**Tue 4/6/21**	**Mon 7/12/21**		
10		Supervisor Discussion	8 days	Tue 4/6/21	Thu 4/15/21		
11		Refer Research Papers and Other Related Document	20 days	Thu 4/22/21	Wed 5/19/21		
12		Find Technology	16 days	Mon 4/26/21	Mon 5/17/21		
13		Create Research Interim Presentation	30 days	Tue 4/6/21	Mon 5/17/21		
14		Create Interim Report	60 days	Tue 4/6/21	Mon 6/28/21		
15		**Research Paper Ceation**	**267 days**	**Fri 7/23/21**	**Sun 7/31/22**		
16		Supervisor Discussion	9 days	Sun 7/25/21	Wed 8/4/21		
17		Systematic Reviews	200 days	Wed 7/28/21	Tue 5/3/22		
18		Create Research Paper	264 days	Wed 7/28/21	Sun 7/31/22		
19		**Final Presentation and Report Creation**	**145 days**	**Tue 1/11/22**	**Sun 7/31/22**		
20		Supervisor Discussion	8 days	Wed 6/1/22	Fri 6/10/22		
21		Systematic Review	45 days	Fri 1/28/22	Thu 3/31/22		
22		Final Presentation Creation	59 days	Sun 5/1/22	Wed 7/20/22		
23		Final Report Creation	50 days	Tue 4/12/22	Mon 6/20/22		

Page 1

Appendix B – Gantt Chart



Appendix – Component 1 – Selected Implementation Codes

In [82]:

```
#2-distribution_variation
#-----------------------------------------------------------------------------------------
#Distribution of Variation in Alexa

#for basic operations
import numpy as np

#importing pandas package
import pandas as pd
#for basic visualizations
import matplotlib.pyplot as plt
import seaborn as sns
plt.style.use('fivethirtyeight')

#for advanced visualizations
import plotly.offline as py
from plotly.offline import init_notebook_mode, iplot
import plotly.graph_objs as go
from plotly import tools
init_notebook_mode(connected = True)
import plotly.figure_factory as ff

#read data from the CSV
data = pd.read_csv('amazon_alexa.tsv', delimiter = '\t', quoting = 3)

#Display Details
color = plt.cm.viridis(np.linspace(0, 1, 15))
data['variation'].value_counts().plot.bar(color = color, figsize = (15, 9))
plt.title('Distribution of Variations in Alexa', fontsize = 20)
plt.xlabel('variations')
plt.ylabel('count')
plt.show()

#Display Count of the distribution

print(data['variation'].value_counts())

#Display Negative reviews and producct details

products= data.loc[(data['variation'].str[0].isin(['W', 'O'])) & (data['feedback'] == 0
products
```

In [1]:

```
# 3.Distribution of feedback in Alexa
#0-negative feedback
#1-positive feedback

# for basic operations
import numpy as np
import pandas as pd

# for basic visualizations
import matplotlib.pyplot as plt
import seaborn as sns
plt.style.use('fivethirtyeight')

# for advanced visualizations
import plotly.offline as py
from plotly.offline import init_notebook_mode, iplot
import plotly.graph_objs as go
from plotly import tools
init_notebook_mode(connected = True)
import plotly.figure_factory as ff

#read data from the CSV
data = pd.read_csv('amazon_alexa.tsv', delimiter = '\t', quoting = 3)

feedbacks = data['feedback'].value_counts()
label_feedback = feedbacks.index
size_feedback = feedbacks.values
colors = ['tan', 'gray']

feedback_piechart = go.Pie(labels = label_feedback,
                           values = size_feedback,
                           marker = dict(colors = colors),
                           name = 'Alexa', hole = 0.3)

df2 = [feedback_piechart]
layout = go.Layout(
           title = 'Distribution of Feedbacks in Alexa')
fig = go.Figure(data = df2,
                layout = layout)
py.iplot(fig)

#Display Negative reviews and related producct details
products= data.loc[(data['feedback'] == 0)]
products
```

Appendix – Component 2 – Selected Implementation Codes

In [18]:

```
#14.NLP prediction  analysis
#predict negative or positive feedback state
#library to clean data
import re
# Natural Language Tool Kit
import nltk
#to remove stopword
nltk.download('stopwords')
from nltk.corpus import stopwords

# for Stemming propose
from nltk.stem.porter import PorterStemmer
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

#for basic operations
import numpy as np
import pandas as pd
import warnings
warnings.filterwarnings("ignore")

# for basic visualizations
import matplotlib.pyplot as plt
import seaborn as sns
plt.style.use('fivethirtyeight')

# for advanced visualizations
import plotly.offline as py
from plotly.offline import init_notebook_mode, iplot
import plotly.graph_objs as go
from plotly import tools
init_notebook_mode(connected = True)
import plotly.figure_factory as ff
from sklearn.metrics import plot_confusion_matrix

#extract features
from sklearn.feature_extraction.text import CountVectorizer

#-----------------------------------------------------------------------------------------
"""Step 1:Import dataset with setting delimiter as '\t' as columns are separated as tab
# Import dataset with setting delimiter as '\t' as columns are separated as tab space
data = pd.read_csv('amazon_alexa.tsv', delimiter = '\t', quoting = 3)

#-----------------------------------------------------------------------------------------
"""Step 2:Text Cleaning / Preprocessing"""
"""A corpus is a large and structured set of machine-readable texts that have been prod
# Initialize empty array
# to append clean text and create corpus
corpus = []

# 3150 (reviews) rows to clean
for i in range(0, 3150):

    #column : "Review", row ith
    review = re.sub('[^a-zA-Z]', ' ', data['verified_reviews'][i])

    #Convert each word into its lower case
    review = review.lower()
```

```
    #split to array(default delimiter is " ")
    review = review.split()


    #creating PorterStemmer object to
    #take main stem of each word
    ps = PorterStemmer()
    review = [ps.stem(word) for word in review if not word in set(stopwords.words('engl


    #rejoin all string array elements
    #to create back into a string
    review = ' '.join(review)

    # append each string to create
    # array of clean text
    corpus.append(review)

#------------------------------------------------------------------------------------
# Creating the Bag of Words model
"""Step 3: Tokenization"""
"""Step 4: Making the bag of words via sparse matrix and need to import countvectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
cv = CountVectorizer(max_features = 1500)
count_matrix = cv.fit_transform(corpus)
x = count_matrix.toarray()
#df = pd.DataFrame(data=x,columns = cv.get_feature_names())
y = data.iloc[:, 4].values
#print (y)
#------------------------------------------------------------------------------------
"""Step 5: Splitting the dataset into the Training set and Test set"""
#splitting the dataset into Training and Test set
from sklearn.model_selection import train_test_split
# experiment with "test_size"
# to get better results -80/20
#if a fixed value is assigned like random_state = 0 or 1 or 42 then no matter how many
#you execute your code the result would be the same
#.i.e, same values in train and test datasets.
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state

#------------------------------------------------------------------------------------
"""Step 6 : Random forest is an ensemble model (made of many trees) from sklearn.ensemb
import RandomForestClassifier class"""
#Fitting Random Forest classifier with 100 trees to the Training set
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators = 100, criterion = 'entropy', random_s
classifier.fit(X_train, y_train)

# n_estimators can be said as number of
# trees, experiment with n_estimators
# to get better results
#------------------------------------------------------------------------------------
"""Step7 : Predicting the Test set result"""
# Predicting the Test set results
#y_pred = classifier.predict(X_test)
#print(y_pred)

y_pred=classifier.predict(X_test)
#------------------------------------------------------------------------------------
```

```
"""Accuracy with the random forest may be different when performed an experiment with d
Step 8: To know the accuracy, a confusion matrix is needed.Confusion Matrix is a 2X2 Ma

#Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)

"""TRUE POSITIVE : measures the proportion of actual positives that are correctly ident
TRUE NEGATIVE : measures the proportion of actual positives that are not correctly iden
FALSE POSITIVE : measures the proportion of actual negatives that are correctly identif
FALSE NEGATIVE : measures the proportion of actual negatives that are not correctly ide
#-------------------------------------------------------------------------------------

print(cm)
print('\nClassification Report:\n', classification_report(y_test, y_pred))
print(accuracy_score(y_test, y_pred))

#-------------------------------------------------------------------------------------
plot_confusion_matrix(classifier, X_test, y_test)
plt.show()
df = pd.DataFrame(y_pred, columns = ['Sentiment'])
print(df)
def clean(review):
  #column : "Review", row ith
    review = re.sub('[^a-zA-Z]', ' ', review)

    #Convert each word into its lower case
    review = review.lower()

    #split to array(default delimiter is " ")
    review = review.split()

    #creating PorterStemmer object to
    #take main stem of each word
    ps = PorterStemmer()
    review = [ps.stem(word) for word in review if not word in set(stopwords.words('engl


    #rejoin all string array elements
    #to create back into a string
    review = ' '.join(review)
    return review

#Newly added customer review
text = 'The sound is terrible and not working properly'
print(text)
x = clean(text)
vec = cv.transform([x])
predn=classifier.predict(vec)
print(predn)

print(1 - metrics.accuracy_score(y_test, y_pred))

import sklearn.metrics as metrics
# IMPORTANT: first argument is true values, second argument is predicted probabilities

# we pass y_test and y_pred_prob
# we do not use y_pred_class, because it will give incorrect results without generating
# roc_curve returns 3 objects fpr, tpr, thresholds
# fpr: false positive rate
```

Appendix – Component 3 – Selected Implementation Codes

In [52]:
```
#ML with Python | Text Clustering | K-Means

#library to clean data
import re
# Natural Language Tool Kit
import nltk
#to remove stopword
nltk.download('stopwords')
from nltk.corpus import stopwords

# for Stemming propose
from nltk.stem.porter import PorterStemmer
from sklearn.metrics import classification_report, confusion_matrix, accuracy_sc

#for basic operations
import numpy as np
import pandas as pd
import warnings
warnings.filterwarnings("ignore")

# for basic visualizations
import matplotlib.pyplot as plt
import seaborn as sns
plt.style.use('fivethirtyeight')

# for advanced visualizations
import plotly.offline as py
from plotly.offline import init_notebook_mode, iplot
import plotly.graph_objs as go
from plotly import tools
init_notebook_mode(connected = True)
import plotly.figure_factory as ff
from sklearn.metrics import plot_confusion_matrix
import warnings
warnings.filterwarnings("ignore")

#extract features
from sklearn.feature_extraction.text import CountVectorizer

#-------------------------------------------------------------------------------
# Step 1: Load the data
"""Step 1:Import dataset with setting delimiter as '\t' as columns are separated
# Import dataset with setting delimiter as '\t' as columns are separated as tab s
data = pd.read_csv('amazon_alexa.tsv', delimiter = '\t', quoting = 3)


data=data.loc[data['feedback'] == 0]

print(data)

#-------------------------------------------------------------------------------
# Step 2: Explore the data
data.info()
#-------------------------------------------------------------------------------
# Step 3: Data preprocessing
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
from sklearn.cluster import KMeans

documents = data['verified_reviews'].values.astype("U")

vectorizer = TfidfVectorizer(stop_words='english')
features = vectorizer.fit_transform(documents)
k = 2
model = KMeans(n_clusters=k, init='k-means++', max_iter=100, n_init=1)
model.fit(features)

data['cluster'] = model.labels_

data.tail()
# output the result to a text file.

clusters = data.groupby('cluster')

for cluster in clusters.groups:
    f = open('cluster'+str(cluster)+ '.csv', 'w',encoding="utf-8") # create csv j
    data = clusters.get_group(cluster)[['variation','verified_reviews']] # get t
    f.write(data.to_csv(index_label='id')) # set index to id
    f.close()

print("Cluster centroids: \n")
order_centroids = model.cluster_centers_.argsort()[:, ::-1]
terms = vectorizer.get_feature_names()

for i in range(k):
    print("Cluster %d:" % i)
    for j in order_centroids[i, :15]: #print out 10 feature terms of each cluster
        print (' %s' % terms[j])
    print('------------')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\Sachithra\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
      rating       date            variation  \
46         2  30-Jul-18      Charcoal Fabric
111        2  30-Jul-18      Charcoal Fabric
141        1  30-Jul-18      Charcoal Fabric
162        1  30-Jul-18     Sandstone Fabric
176        2  30-Jul-18  Heather Gray Fabric
...      ...        ...                 ...
3047       1  30-Jul-18           Black  Dot
3048       1  30-Jul-18           White  Dot
3067       2  30-Jul-18           Black  Dot
3091       1  30-Jul-18           Black  Dot
3096       1  30-Jul-18           White  Dot

                                     verified_reviews  feedback
46    "It's like Siri, in fact, Siri answers more ac...         0
111   Sound is terrible if u want good music too get...         0
141                                 Not much features.         0
```