Ajesh Kumar

**REENGINEERING OF LEGACY SOFTWARE**

Examiners:  Professor Dr. Ajantha Dahanayake  
Associate Professor Dr. Elena Ruskovaara

Supervisors:  Professor Dr. Ajantha Dahanayake
ABSTRACT

LUT University
School of Engineering Science
Software Engineering
Master's Programme in Software Engineering and Digital Transformation

Ajesh Kumar

TITLE OF THE WORK – Reengineering of legacy software

Master’s Thesis

85 pages, 40 figures, 7 tables, 1 appendix

Examiners: Professor Dr. Ajantha Dahanayake
Associate Professor Dr. Elena Ruskovaara

Keywords: Software, Software Reengineering, Legacy Systems, User Interface

In seven decades of software engineering history, legacy systems are becoming increasingly complex to operate in the newly broadened market of digitalized products and easy availability of services. Yet, legacy systems are so important that it can’t be discontinued in absence of any reengineered solutions. MTEE is one such application that is used for a specific entrepreneurship education research and now requires a comprehensive reengineered software with added functionalities. The goal is to reengineer by following an approach best suited for reengineering legacy software. As a result, a new version of MTEE application was developed based on important concepts identified during the research.
ACKNOWLEDGEMENTS

To my parents and family: It’s all been possible because of you all and I owe it all to you. Many Thanks!

To my supervisor, Professor Dr. Ajantha Dahanayake: I am deeply grateful and indebt of your guidance, support and immense patience with which she helped me from the very beginning of my life at LUT. From the day she interviewed me for admission in this university, she has been entrusting me with different set of responsibilities and has always believed in my ability. Her guidance and encouragement has made it possible for me to work hard and not lose hope when things were not going smooth. I owe my success at LUT to her care, concern and affection and I fully realize that I will never be able to repay for all that she has done for me.

To my second supervisor, Associate Professor Dr. Elena Ruskovaara: I am immensely thankful to you for believing in my capability and allowing me to work on a project which is such an important part of your life and trusting me with development of the new version of MTEE. Your encouragement, gentle nature and sense of humor has made my experience of working in this project very enjoyable and I can’t help myself feel exited to know that our engagement will continue further.

To my colleague, Pirjo Kuru: A very special gratitude and thanks for being there with me at every step of this journey and making my life a billion times easier by translating all the Finnish text and testing out each feature of the application so meticulously. You have been a remote partner through many sleepless nights without showing any signs of discomfort while testing the application. This journey wasn’t possible without your help.

To my colleague, Anil Kumal: I am grateful to you for being the part of the team and help create many user interface screens. Your knowledge and ability to work without much supervision is a great skill to have and I will be forever be grateful for the work you have contributed in this project.

I am also grateful to the following university staff: Matti Janis, Otto Myyrää, Tarja Nikkinen for their unfailing support and assistance

And finally, last but by no means least, also to my friends Krishna Teja Vaddepalli, Tania Islam, Tuwin Choudhury, Suraj Jaiswal, Ali Saud, Jai Hardasani, Giota Goswami, Rakitha Dedigama, Soumyajit Chaterjee, Shah Alam Malik and his wife for all help, encouragement, motivation without which this would not have been an enjoyable work.

Thanks for all your encouragement!
# TABLE OF CONTENTS

**Contents**

**TITLE OF THE WORK – REENGINEERING OF LEGACY SOFTWARE ........II**

1 INTRODUCTION ..................................................................................................................4

1.1 BACKGROUND..................................................................................................................4
1.2 RESEARCH PROBLEM .................................................................................................6
1.3 RESEARCH METHODOLOGY ......................................................................................8
1.4 STRUCTURE OF THE THESIS ......................................................................................11

2 LITERATURE REVIEW ......................................................................................................12

2.1 SOFTWARE DEVELOPMENT HISTORY – A BRIEF INTRODUCTION ..................12
2.2 SOFTWARE DEVELOPMENT AND GRAPHICAL USER INTERFACE ........14
2.3 USER INTERFACE AND USER ENGAGEMENT .......................................................15

   2.3.1 Good UI vs Bad UI ...............................................................................................17
   2.3.2 Principle of User Interface Design ......................................................................19
   2.3.3 User Interface Design Patterns ...........................................................................22

2.4 ISSUES OF LEGACY SOFTWARE .............................................................................25

2.5 USABILITY EVALUATION METHODS ........................................................................35

   2.5.1 Cognitive Walkthrough ......................................................................................36
   2.5.2 Heuristic Evaluation ............................................................................................38
   2.5.3 Feature Inspection ...............................................................................................39

3 ANALYSIS OF LEGACY MTEE ......................................................................................40

3.1 OVERVIEW OF LEGACY MTEE ..............................................................................40
3.2 OVERVIEW OF LEGACY MTEE DATABASE .............................................................44

4 IMPLEMENTATION ...........................................................................................................46

4.1 NEW MTEE ARCHITECTURE ....................................................................................47
4.2 TECHNOLOGICAL STACK ..........................................................................................49

5 EVALUATION ....................................................................................................................52

6 DISCUSSION ......................................................................................................................62

7 CONCLUSIONS ..................................................................................................................64

REFERENCES ..........................................................................................................................66
APPENDIX 1........................................................................................................71

8  VISION DOCUMENT..............................................................................71

SOFTWARE REQUIREMENTS SPECIFICATION........................................75

INTRODUCTION .................................................................................................. 76

Purpose ............................................................................................................... 76

Document Conventions .................................................................................. 76

Intended Audience and Reading Suggestions .............................................. 76

Product Scope .................................................................................................. 76

OVERALL DESCRIPTION .............................................................................. 76

Product Perspective ......................................................................................... 76

Product Functions .......................................................................................... 76

User Classes and Characteristics .................................................................. 77

Operating Environment .................................................................................. 77

Design and Implementation Constraints ....................................................... 77

User Documentation ......................................................................................... 77

ASSUMPTIONS AND DEPENDENCIES ....................................................... 77

EXTERNAL INTERFACE REQUIREMENTS ............................................... 77

User Interfaces ................................................................................................ 77

Hardware Interfaces ....................................................................................... 77

Software Interfaces ........................................................................................ 77

Communications Interfaces .......................................................................... 77

SYSTEM FEATURES ....................................................................................... 78

MTEE Application Access ............................................................................... 78

Responding to Questionnaire ......................................................................... 78

Profile Management ......................................................................................... 78

Admin Module ................................................................................................ 79

Admin Reports ................................................................................................ 79

OTHER NONFUNCTIONAL REQUIREMENTS .............................................. 79

Performance Requirements ........................................................................... 79

Safety Requirements ....................................................................................... 79

Security Requirements ................................................................................... 80

SOFTWARE QUALITY ATTRIBUTES ........................................................... 80

BUSINESS RULES .......................................................................................... 80
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACM</td>
<td>Association for Computing Machinery</td>
</tr>
<tr>
<td>EBSCO</td>
<td>Elton B. Stephens Company</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HCI</td>
<td>Human Computer Interaction</td>
</tr>
<tr>
<td>HMI</td>
<td>Human Machine Interaction</td>
</tr>
<tr>
<td>IBM</td>
<td>International Business Machine</td>
</tr>
<tr>
<td>IEEE</td>
<td>Institute of Electrical and Electronics Engineers</td>
</tr>
<tr>
<td>LOC</td>
<td>Lines of Code</td>
</tr>
<tr>
<td>LUT</td>
<td>Lappeenranta University of Technology</td>
</tr>
<tr>
<td>MTEE</td>
<td>Measurement Tool for Entrepreneurship Education</td>
</tr>
<tr>
<td>SOA</td>
<td>Service Oriented Architecture</td>
</tr>
<tr>
<td>SSL</td>
<td>Secure Socket Layer</td>
</tr>
<tr>
<td>URL</td>
<td>Uniform Resource Locator</td>
</tr>
<tr>
<td>UX</td>
<td>User Experience</td>
</tr>
</tbody>
</table>
1 INTRODUCTION

1.1 Background

All software is mortal. Manny Lehman and Les Belady are the first two individuals to propose fundamental laws for software evolution, where they described two important laws governing the lifespan of a software [1].

They are as follows:
• The Law of Continuing Change: “A software program being used in a real-world environment must undergo some change or become increasingly less useful in that environment” [1].

• The Law of Increasing Complexity: “As a software program evolves, it becomes more complex and due to this extra effort or resources are required to preserve and simplify its underlying structure” [1].

These two laws are true for all software, irrespective of their size and domain, though the size of a software, if measured as number of Lines of Codes (LOC) does play a role along with the domain of the functionality [2]. For example, for an average software, the lifetime due to some replacement has been found to be approximately 9 years, with the maximum duration of 20 years and the minimum duration of 2 years as found out in a survey, caused due to multiple factors such as hardware replacement, change of system architecture which may include change from batch system to an online or real time system, change in ways data is entered, changing of business procedures or change in social systems such as introduction of new consumption tax, new business tie-up and other required functional enhancement or significantly high maintenance expenditure which may be due to structural deterioration [2].

Software engineers generally desire to build software systems that are long-lasting but it has been generally seen that the quality of a software product decreases as time pass by. This due to many factors such as system becoming more complex over time, new problems creeping into a software program and which may manifest themselves as software bugs, give rise to unwanted behavior, show undesired side effects or impact the system’s quality
negatively. This depletion of quality could be due to many factors, such as non-standard coding practices, requirements that were unimplemented, flawed architectural design decisions, and in many cases, they lead to technical flaws and may negatively affect the system’s behavior, performance, and maintainability in subtle or unsubtle ways [3].

Software developments has over time become increasingly large, diverse and complex in years due to the penetration of computational technologies in every sphere of human life, due to which extending the life-span of software has grown into a very challenging problem. Object-oriented technology can be one of the solutions. Object-oriented design is the most feasible solution as it helps us by improving the modularity and readability of software. It also helps us to understand the structure of the software and to maintain, modify and extend it. However, there are certain challenges in Object-oriented design, for example, achieving well defined object-oriented designs requires skilled developers who must have spent considerable time in gaining experience of object-oriented design. Design patterns, which can be understood as abstract descriptions of object-oriented design solutions to problems that appear repeatedly in software development and which have succeeded as solutions to past design problems, can help to overcome this problem [4].

Apart from architectural changes, a good software reengineering also incorporates principles of user experience design where any new design that is being introduced should be neutral, simplified, coherent and systematic. Moreover, the elegant typography and the grid-based layout should balance in a minimalistic way with empty space of the page which also aids the look and feel of a software. The complete layout should be more user friendly, should provide users with a higher level of visual attractiveness and should introduce the user to an upgraded and contemporary Interface design [5].

When it comes to reengineering legacy systems, the software development process needs to achieve a better result as compared to the existing legacy software, in terms of usability, maintainability, performance, functionality and other aspects of software quality, in order to justify the effort put in reengineering and the cost associated with it [6].

Reengineering legacy systems requires mitigating few set of challenges ranging from lack of documentation, inflexible architecture, outdated technological stack, hardware
incapability etc. Certain processes and modeling methods helps us to understand the flaws and document the new requirement and vision of the end product. This also helps us to consider the estimated cost, duration of the project and develop a project plan. The development method can be chosen based on the scope, size and complexity of the functional and non-functional requirements [7] [8].

1.2 Research Problem

In this project, an existing software ‘Measurement Tool for Enterprise Education (MTEE)’, which was developed between 2008-2012 as a self-assessment tools for teachers, to enable them to monitor the implementation of enterprise education, needs to be reengineered, as the existing tool has become obsolete due to many factors. This project is carried out as Master Thesis project for LUT University’s research on Entrepreneurial Education through MTEE tool.

MTEE is a web based application that requires a user to register and only registered user can answer a fixed set of questionnaire pertaining to the domain of entrepreneurial education teaching. MTTE development was funded by European Social Fund and Finnish National Board of Education and Co-ordinated by Lappeenranta University of Technology (LUT). In the existing application user responses are stored and can be converted into spreadsheet by running jobs manually. Adding or updating questions were achieved through running queries manually on console by the developer. The end users receives feedback based on their responses and average results across categories called ‘themes’ and can access previous responses. The application is currently running successfully.

The requirement for reengineering MTEE is made possible after a new set of questions were formed for entrepreneurial education research. The existing application has no mechanism for researchers to replace the questionnaire with these new set of questions. Also, features like the ability to update question text, feedback text, etc. is not available to the researchers. Though User Interface (UI) is provided to the researchers to set different filter criteria for reports to be generated in spreadsheet format, they do not have any authentication and is being served as a non-promoted address living on application server.
In this project, the existing MTEE application and new requirements are analyzed to identify a set of processes that can be employed to reengineering this legacy system. For example, analysis to ascertain the extent of code changes required to add the new set of functionality with current application stack was carried out, database structure were analyzed, codes were reviewed and documents were studied. An understanding of these analysis results is helpful to decide the next course of action which may range from code refactoring to code rewrite depending upon different factors such as resource availability, project duration, technical experience of the workforce, team size, etc. [8].

The objective of this research is to find suitable approach that can be applied to reengineering an existing software, so as to enhance user experience, increase user engagement, provide new features, remove architectural short comings and increase the longevity, with as little effort as possible and as quickly as possible.

The quest is to identify those approach that can be applied to any type of software reengineering scenario, where the challenge is to identify the existing short comings of the software, enhance user experience, increase user engagement, increase software robustness and provide new features. To be able to do these things, it require us to first find solution to following set of research questions and then to apply them.

Following questions are important to us for meeting our research objective:

• RQ1: What are the methods to identify short comings of an existing software?
• RQ2: What are the methods to enhance user experience?
• RQ3: What can be done to increase user engagement on the application?
• RQ4: How to make an application robust?
• RQ5: Which design principles suits best when it comes to reengineering a software?

With these questions in mind, the goal is then to meet the defined objective successfully. However, in-spite of these above basic questions, there could be some other additional relevant questions. The intended outcome of this implementation based project is a set of development practices and approach that can be applied in case of application reengineering.
in which it is required that the new application is more robust, has longer lifespan, has better user experience and engagement and thus is more sustainable.

1.3 Research methodology

A systematic approach is used to carry out a survey of literature related to legacy software reengineering. The literature survey process followed an extensive stepwise method to find relevant research material on the topic which is as shown in Figure 1 below:

1. Listing related keywords
2. Scoping through different scientific database
3. Using different time frames for searching
4. Looking for relevant concepts

Figure 1. Steps for literature survey.

Table 1 below provides a list of scientific databases along with the keywords used and the number of hits received in different scientific databases for research work related to keywords searched.
As we can see from Table 1, there is a large number of scientific work present for different keywords used for the search. A more context refined search results by adding more than one keyword is used to obtain more relevant results. The bar chart in Figure 2 below depicts the number of search results in different database. Moreover, articles are selected across the life cycle of software development process and its history to incrementally review the work already done in the research domain.
The research process has broadly covered onion model, where the research philosophy has a positive outlook towards the research methodology with an underlying assumption that a thorough literature review will lead to process that will help reengineer the MTEE application and satisfy the project goal. A deductive approach is employed along with a systematic literature review as a research strategy in this work to obtain a set of specific processes to be followed that could lead us in reengineering legacy systems, which may have similar characteristics to MTEE. The research choices are restricted to the specific keywords and research findings have been used in developing new MTEE system. The project duration is stretched in two phases from 15.06.2018 to 12.06.2018 as first phase and in second phase from 01.02.2019 to 15.08.2019. A self-assessment of project work span and effort is given in Figure 3.
1.4 Structure of the thesis

This work is divided into seven sections. The first section introduces the topic in some detail and also introduces the project scope and limitation. The second section lists existing scientific work carried out in the relevant field. The third section discusses the analysis of legacy MTEE application. The fourth section describes the implementation of reengineering methods. The fifth section presents an evaluation of reengineered application in comparison to the legacy system and presents the improvements in the reengineered MTEE application. The sixth section discusses the experience of reengineering the legacy system, the challenges faced and the future scope. In the end, the seventh section is for conclusion and summarizes the work carried out.
2 LITERATURE REVIEW

2.1 Software development history – a brief introduction

Barry Boehm, who is also considered as father of software engineering, in his paper “A view of 20th and 21st century software engineering”, has documented the philosophy that was prevalent in post 1950, when the era of software engineering began [8]. In the earlier days, software were mainly used in mission critical systems or in other engineering systems that mostly depend on critical hardware and the software were developed with the same philosophy as the hardware, which is best described by Boehm as “measure twice and cut once”. Since, in earlier days of software engineering the cost associated with using computer hardware exceeded greatly with respect to the software, the software development process required the developers to review and test there codes multiple times before they were run. Moreover, the use of single use punch cards that required the developer to punch holes based on the codes they wanted to run were costly and time consuming, making software developers were more cautious of the validity of their code [8].

In the decade that followed, software development became the part of university curricula in the field of engineering and natural sciences, where complex mathematical problems were solved by using mathematical formulas. It also resulted into the creation of formal notations which we call as programming languages. The onset of programming languages such as Fortran in 1957 and COBOL in 1962 bought computing to business and industry. The increase in computing power also bought with it, a new profession for software programmers who soon realized that programming was a difficult task [9].

The initial days for the computer programmers were hugely challenging as they had to write their programs, submit their codes to laboratories, where they would run it in a queue and wait for the result, which could vary from hours to days. This type of processing was called batch processing. This gave rise to the time-sharing concept as the computational resources were still quite expensive. The time sharing system was first introduced at MIT and it bought with it the interactivity between the programmer and the computer. These time sharing systems would mainly run on large mainframes systems developed by organizations such as IBM [10].
Though time sharing system was bought as a solution to overcome the delay in batch processing, the practical implementation was difficult to achieve, as the problem of transition from batch processing to time sharing systems were more complex than initially imagined. It was in the year 1968, in a conference sponsored by NATO that the contemporary challenges of software computing were frankly discussed among the participants and the limitations were acknowledged and the term software engineering was coined [9][10].

During the time that above mentioned changes were being introduced in computing, there were other philosophical changes in the way programming was perceived. For example, the concept of structured programming, sequential execution of instructions and software development life cycle models such as Waterfall model came into existence. Also, data structures concept were introduced that resulted into creation of relational storage mechanism that came to be known as databases for storing information, which was proposed by Edgar Codd in his 1970 paper titled ‘A Relational Model of Data for Large Shared Data Banks’ [11]. These new concepts were a welcome change from the earlier ‘code and fix’ approach and the programming was developed as a process, where coding was carefully carried out and preceded by design which in turn was preceded by gathering the requirements [8][9].

The decade following 1970 was the first time when Graphical User Interface (GUI) was first introduced by Xerox in its Xerox Alto systems, which supported an operating system having a GUI [11]. The introduction of Xerox Alto was a cornerstone event in computing as it provided users a completely new mechanism away from command based control, for using the operating system of the computer [12]. The Xerox Alto personal computer came with attached peripheral device such as a display monitor, a keyboard and pointing device named mouse [13]. It was the technological concepts derived from Xerox Alto systems that impressed the then Apple’s founder Steve Jobs, who received a demonstration of the Xerox’s GUI technology in exchange of shares of Apple and used the technology extensively in Apple’s Lisa and Macintosh personal computers [13]. The 1970’s decade established that software applications would in future would have a greater GUI component.
The increased availability and use of computers in workplaces and home which began in 1970’s has brought required attention to how people interact with computer systems which resulted into emergence of Human-Computer Interaction (HCI) [14].

2.2 Software Development and Graphical User Interface

Usability is a core concept in HCI [15]. Usability has been defined as “the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use” [16]. In 1980’s the term usability started getting used more prominently in the domain of software engineering, than the earlier vogue term “user-friendly”, which was quite subjective in nature [15]. Along with usability, another term ‘user experience’ came into prominent use with respect to GUI [17]. User experience (UX) has been defined as “the combined experience of what a user feels, perceives, thinks, and physically and mentally reacts to before and during the use of a product or service”[18]. In UX the most important concept is “the process by which users form experiences” from the time they first interact with the product or application to through the period of its use [15].

The goal of any UX expert is to accommodate diverse set of user, thus the philosophy is to always strive to achieve an universal design which is possible by creating an application or product that can be easily and comfortably be used by as many people as possible, with as diverse range of abilities as possible, operating in as diverse range of situations as possible [18] [19].

Another important consideration in UI design is culture. Cultural differences play quite significant role in applications usability and may include aspects such as date and time format, interpretation of symbols, colors, gestures, text direction and language. Thus, UI designers at the time of prototyping their design must be sensitive to these cultural differences during the development process and must avoid considering all cultures the same, when it comes to using an application or any software product [20].
Moreover, apart from cultural differences, other factors such as age influence the usability of navigational menus. It has been observed that as people age, there is a decline in sensation and perception, cognition, and movement control. As for instance, decline in vision acuity, color discrimination, hearing, selective attention, working memory, and force controls effect the usability [21]. These changes due to aging influence application use. Thus, a software’s user interface should be designed to accommodate special needs of older users, if the application caters to them. Studies conducted by researchers have shown that aging has significant impact on menu navigation. Menu navigation is an important aspect of software design as an efficient menu design allows users to follow correct navigation. Efficient menu design is based on several factors such as naming, its depth and breadth, the structure of the menu bar and allocation of menu items and their respective position. Menu navigation is also dependent on certain individual factors: verbal memory, spatial ability, psychomotor abilities, self-efficacy and visual abilities. These individual factors are age-related [21].

### 2.3 User Interface and User Engagement

In UI design there are four principle concepts that must be considered for increasing the user engagement of a software application [15] [22]:

- Usability
- Visualization
- Functionality and
- Accessibility

Usability is a quality measure attribute that can be defined as the measure of the ease of use of user interface of an application or a software program [23]. Usability can be assessed by asking simple questions, such as if user is able to navigate through application pages, or if the user can go back to previous page and other process of visual and functional aspects [22][15].

Visualization is a process to making computable data and other content of a website or an application clear and presentable. Visualization is not restricted to making application pages designed in a fancy fashion, but means making its content easily readable and data easily
understandable. For example, putting a long list of numbers for users to read could be very confusing or frustrating for the user. However, presenting the same information in form of charts or tables could make information more presentable and readable [22]. Visualization stimulates visual thought which cannot be done through plain text. Visualizing data can help user understand the underlying knowledge behind the object and help them focus on their meaning as visualization helps user in stressing less, understanding more, seeing more and enjoying more. Data visualization is thus about presentation of meaningful abstract data or information to the users [24].

Functionality is life of a system [22]. Designing layout does not do anything significant until we do not integrate it with programming logic. Designing an interface not only takes care of visual aspects of a software system, but helps us in developing logic behind it too. Designing functionality is about considering an element and determining the task it must do or how the element must behave. Functionality is also considered as a feature of an element that is visible to the user [23]. For example, when a user has to access a system which has access control, the user is required to insert his credentials in a form which is a visual representation and press a button to verify the given credentials to allow the user to access the system, which is made possible by the integrating functionality with visual elements such as form and button. There are many ways in which software developers can make use of functional aspect of designing and the aim should be to make things simple for user. Overdoing this is also not advisable, as it can makes system complex and can create other kinds of trouble [23]. For example, if user has every other text of the content box as click and navigate option, it will make user interface quite terrible and won’t be very usable for a user. On the contrary, features that aid users to seamlessly use the system should be implemented such as hotkeys, keyboard shortcuts, etc. For example, pressing enter button on keyboard to submit a form could make it easier for the user to use the system as it will save time and users does not need to move the mouse pointer and click the specified button on the page.

Accessibility is the measure of ability of the system to allow widest possible set of users to use the system [25]. The goal of a software developer should not be to design software for the majority group of users or for users with some specific technical knowhow. Since, not all users are of same category, designers must take into consideration various sets of users that can access their systems, as much as possible. All users including those who do not have
as much familiarity of using computer as well as others should also be considered when designing a software. Moreover, users with slow internet connection, or old peripheral devices should be considered as much as possible. Similarly, people with special needs should be considered too [23].

2.3.1 Good UI vs Bad UI

Creating a good user interface, which can be used easily by a wider set of user is not a simple process. User Interface should not be just a grouping of design elements without any logical or visual relation, which makes using application difficult for the user. It is quite easy to implement a bad design. Even though the goal of a software developer is to make user interface more engaging for the user, it requires that the designers invest considerable thought in the planning of the interface design and associated functionality before implementing their designs. For a good user interface design, it is required that a well-planned design is prepared beforehand, which could help the developer to achieve a simple, easily navigable and well-structured application design. Such applications take into account all good user interface design considerations along with all the design elements thoughtfully [23].

Table 1 below presents some important difference between good and bad application design considerations [23] [24] [25] [26] [27].

<table>
<thead>
<tr>
<th>Sl. No</th>
<th>Good UI</th>
<th>Explanation</th>
<th>Bad UI</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>No unnecessary animation</td>
<td>Animations are used to draw and focus user’s attention to a specific element.</td>
<td>Flashy animation</td>
<td>Difficult to focus on important content.</td>
</tr>
<tr>
<td>02</td>
<td>Well-structured</td>
<td>Well-structured application design makes using the application</td>
<td>Inconsistency</td>
<td>No common page layout or structure makes the</td>
</tr>
<tr>
<td></td>
<td>Consistent design and fonts</td>
<td>Consistent design and fonts makes it easy for the user to get familiar with the application.</td>
<td>Multiple fonts</td>
<td>Using many fonts reduces the readability and user trust on the software.</td>
</tr>
<tr>
<td>---</td>
<td>-----------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>----------------</td>
<td>--------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>03</td>
<td>Clear links and navigation</td>
<td>Clear links and navigation helps user to navigate through different features of the application increasing its usability.</td>
<td>Small fonts</td>
<td>This makes it difficult for the user to read the text of the content.</td>
</tr>
<tr>
<td>04</td>
<td>Correct use of colors</td>
<td>Using a well suited color scheme with set of complementary colors enhances the look and feel of the application.</td>
<td>Focus on unnecessary part</td>
<td>Focusing on side aspects, instead of main feature defeats the purpose of the application</td>
</tr>
<tr>
<td>05</td>
<td>Access control</td>
<td>Allowing users to be able to register or login into application enhances user’s trust in application.</td>
<td>Too much information</td>
<td>Putting every information on same page makes it unreadable and categorize information for user.</td>
</tr>
</tbody>
</table>

Table 1. Difference between Good and Bad User Interface application.
2.3.2 Principle of User Interface Design

In a research conducted by David Chek Ling Ngo, et al. esthetic measures for UI design have been investigated by considering some fourteen most important characteristics. The esthetics of the user interface is critical aspect for drawing user’s attention and to keep them engaged with the application, and a well implemented esthetic concepts in an application can aid acceptability and learnability [28] [29]. The esthetic measures and what they mean are explained in Table 2 below [29]:

<table>
<thead>
<tr>
<th>Important Esthetic Measure for UI</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Balance</td>
<td>Balance has been defined as the distribution of optical weight in a picture or screen. Optical weight is the relative appearance of pictures, where one picture appears heavier or lighter than the other. In application’s view design balance can be achieved by making sure that the design elements are at equal weightage from left, right, top and bottom.</td>
</tr>
<tr>
<td>• Equilibrium</td>
<td>Equilibrium is achieved when design elements are stabilized with respect to the midway center of suspension. Equilibrium in a screen can be achieved through centering the layout by making sure that the center of layout coincides with the center of the screen.</td>
</tr>
<tr>
<td>• Symmetry</td>
<td>Symmetry can be defined as the extent to which the screen is symmetrical in all three directions namely vertical, horizontal, and diagonal. Symmetry can be understood as axial duplication, where the dimensions of one half side of central line is exactly same on the other side of central line. Vertical symmetry is when equivalent elements are arranged in a balanced way.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>• Sequence</td>
<td>Sequence can be defined as a measure of how information which is displayed is ordered in relation to a reading pattern, which could be left to right in most cultures and right to left in middle eastern cultures or top down in some eastern cultures. Sequence in user interface design is about the placement of objects in a layout in such a way that it helps the movement of the eye to flow through the information displayed.</td>
</tr>
<tr>
<td>• Cohesion</td>
<td>Cohesion can be defined as a measure of how cohesive the screen is. In UI design, it has been observed that similar aspect ratios promote cohesion. The term aspect ratio is the relationship between width and height. A change in the aspect ratio of a visual field may affect eye movement. The eye movement patterns affect the cohesion considerably.</td>
</tr>
<tr>
<td>• Unity</td>
<td>Unity can be defined as the extent to which the screen elements seem to belong together. Unity can also be understood as coherence of design elements, where all elements in totality seems visually as one piece. Unity allows elements to seem to belong together in such a way that they are seen as one thing.</td>
</tr>
<tr>
<td>• Proportion</td>
<td>Proportion can be defined as the comparative relationship between the dimensions of the screen components and proportional shapes. In UI design, major design components of the UI screen, including window, grouping of data and text should be of esthetically pleasing proportions and designers should consider this at the time of layout design. In many cultures, traditional design esthetics have preferred proportional components. However, things that may be considered beautiful in one culture may</td>
</tr>
</tbody>
</table>
not necessarily be considered the same in another cultural setting, yet proportional shapes have largely been used in all cultures.

| **Simplicity** | Simplicity can be defined as “directness and singleness of form” in such a way that “combination of elements results in ease in comprehending the meaning of a pattern”. In UI design, simplicity can be achieved by “optimizing all the elements of the screen” and “minimizing the alignment points” [31]. |
| **Density** | Density can be defined as the “extent to which the screen is covered with UI elements”. In UI design the goal is to restrict density levels in a screen to an optimal percentage, so as to make it easier for user to follow the screen elements. Tullis derives measure of density as the percentage of character positions on the entire visible frame containing data [31]. |
| **Regularity** | Regularity can be defined as “a measure of how regular the screen is” and can be understood as the extent of “uniformity of elements based on some principle or plan”. Regularity in UI can be achieved by “establishing standard and consistently spaced horizontal and vertical alignment points for screen elements, and minimizing the alignment points”. |
| **Economy** | Economy in UI design can be defined as “a measure of how economical the screen is”, which can be achieved by “careful and discreet use of display elements” to present the message in as simple manner as possible. A UI design would be considered economic if the display elements do not have many varying size. |
| **Homogeneity** | Homogeneity is defined as “a measure of how evenly the objects are distributed among the four quadrants” |
The esthetics of the user interface is the most critical factor in gaining the user’s attention, trust and loyalty. Thus by carefully using esthetic concepts, UI designers can aid acceptability and learnability [29].

### 2.3.3 User Interface Design Patterns

A pattern can be described as a formalized description of proven approaches to address problems that are common, difficult and regularly encountered [32]. For a design problem, a pattern represents a format for describing the solution. Patterns were first introduced in the field of architecture by Christopher Alexander, who identified that certain solutions can
always be applied to similar problems that reoccur and developed patterns as a design knowledge documentation method [33] [34]. In software engineering, patterns were adopted so as a way to encourage reuse of software components, or code components [35].

For UI design, designers also noticed that certain design problems occurred regularly, which generally have a known solution that could be effective in solving a specific problem, but not many designers know about them or they are not communicated effectively. Moreover, guidelines were seen as difficult to interpret or were considered effort intensive to find relevant materials [36]. Over the years, many UI specific patterns were created that solved some of the most reoccurring problems in UI design and are given below in Table 3 with their explanation [15].

<table>
<thead>
<tr>
<th>UI Design Patterns</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page Composition</td>
<td>Page composition is an all-encompassing term in user interface design. A composition in UI design encompasses different components such as scrolling, annunciator row, notification, title, menu patterns, lock screen, interstitial screen, and advertising [26]. A UI designer should focus on what components are required in the UI design, for effectively carrying out the objective of the user. This could include menu navigation, adaptable interface or adaptive interface [15].</td>
</tr>
<tr>
<td>Display of information</td>
<td>On a computer system, users are generally presented with multitude of information. Using some information display patterns help users in filtering and processing relevant visual information. Some examples of information display patterns include different types of lists, including vertical list, thumbnail list, fisheye list, carousel, grid, and film stripe [26]. An effective information display pattern must reflect user’s mental models and the way users organize and process information that they see [15].</td>
</tr>
<tr>
<td>Control and confirmation</td>
<td>Physical and cognitive limits of human users may lead to unwanted errors that can be minor error to more critical ones like application crash [15]. In software systems, control and confirmation dialogues</td>
</tr>
</tbody>
</table>
can be used to prevent errors, typically user errors, such as restricted access, wrong navigational flow, etc. A confirmation dialogue can be used when in course of application use, users reach a decision point and must confirm an action or choose between options. Control dialogue should be used to prevent against accidental loss of information such as accidental logout or accidental delete [26] [27].

| Revealing more information | In user interface design, information can be revealed or presented in two ways mainly, either by displaying information in a full page or by revealing in a context [15]. When a large amount of content has to be presented or displayed, revealing in a full page should be generally used. In situations where information should be revealed quickly and within a context revealing in context should be generally used. Some of the examples of patterns for revealing information are popup, window shade, hierarchical list and returned results. |
| Lateral access | In user interface design, lateral access components are those, which provide faster access to different categories of information. Two of the most common patterns for lateral access are tabs and pagination which helps designers by limiting number of levels of information users must drill through, reducing constant returning to a main page, and reducing the use of long list. |
| Navigation | In user interface design, links are the most common way to provide navigation. Links helps in navigating the user and provides access to additional content by loading a new page or jumping to another section within the current page. UI designers should make sure that the links are functional, not over used and follow a structured sequence [23]. |
| Button | In user interface design, buttons are the most ubiquitous design component that are used across the platforms. Buttons are used to initiate certain actions, such as login button, which would be a standalone button will initiate authentication logic in the application. |
On the other hand, buttons can also be used to allow users to select among alternatives available in form of radio buttons.

<table>
<thead>
<tr>
<th>Icon</th>
<th>In user interface design, an icon can be defined as a visual representation that aids users to access a specific destination or trigger a function in a cursorily manner. There are three different objectives that can be achieved by using icons, which are (1) provide access to a function or target destination, (2) provide an indicator of different system statuses, and (3) provide a way to change system behavior.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Information control</td>
<td>In user interface design, the amount of information shown is determined based on the size of device screens. Information control mechanisms such as zooming and scaling, searching, and sorting and filtering have been utilized to assist users in finding, accessing, and focusing on intended information while minimizing unrelated information.</td>
</tr>
<tr>
<td>Input mode and selection</td>
<td>Input mode and selection is about how the user communicate with the application. For example, on a desktop, user input is generally captured with the help of peripheral devices such as keyboard or mouse and output could be displayed through different channels. There has been new methods such as touch gestures</td>
</tr>
</tbody>
</table>

Table 3. List of UI design patterns and what they mean.

### 2.4 Issues of Legacy Software

K. Bennett defines legacy software informally as “large software systems that we don't know how to cope with but that are vital to our organization” [37]. Legacy software have certain characteristics like the software may have been written years ago using such technologies which may be considered old in existing context, yet it may still be doing its stated useful work. In software development perspective, migrating and updating this old piece of functioning code may have technical and nontechnical challenges, which could range from the economic ones such as justifying expense in form of payment for employment of external contractors to implementing new requirements such as using
visualization techniques on data, in a technical perspective. The nature and the extent of requirements and associated challenges varies with the software in question [37].

In a seven decade old history of software development, there are many reasons that have forced organizations to consider upgrading legacy systems. These reasons range from economic factors such as reducing cost, increasing availability, increasing capacity, increasing performance, etc. to technical factors such as integrity of software code across multiple systems, increasing maintainability in software product, streamlining business processes, etc. [37].

In some cases, situation may demand scrapping of legacy system and replacing them with more modern software, which could involve various significant risk associated with replacing legacy systems with new model for solution [38]. These risks are taken when the existing legacy system cannot meet the current requirement without significant cost of time and resource or if the existing legacy systems cannot be modified at all [39]. Replacing a legacy systems requires that the new software, which is generally developed from scratch is extensively tested to determine that it first meets the functionality set by the current systems and then secondly if it can provide additional functionality. An assessment of such requirements, needs to be done before making a decision about replacing the legacy software. This process begins by first analyzing the requirements and then analyzing legacy systems and offered functionality to determine if the existing legacy software can be re-designed or needs to be replaced completely. In absence of through testing, new system runs a risk of not being robust or functional as the old one [38].

Compared to replacement, reengineering is considered a more pragmatic approach when it comes to reengineering legacy systems, as in most cases, completely removing the legacy systems is close to impossible, even though it is generally not the cheapest option, but could turn out less risky and can significantly help in extending the useful lifetime of existing system. Moreover, reengineering capabilities and possibilities are dependent on how the legacy system was developed in the first place. Factors such as technology used, architecture, database schema, business value, business models, maintainability, availability of documentation, etc. determine the type of reengineering possible and the complexity involved in providing those possibilities [38] [39] [40].
There are different approaches that can be employed for reengineering legacy system. Currently, there are many available legacy system migration approaches which are discussed in brief further. Although legacy system redevelopment is a major research issue, there are very few comprehensive redevelopment methodologies available as compared to other methodologies available which are mostly not well documented, so general that they are devoid of many specifics or satisfy to a particular aspect of software development and thus can result into situations where other critical aspects are overlooked [40]. Six of the most well-known approaches are documented below in Table 4 with their explanation [40] [41] [42].

<table>
<thead>
<tr>
<th>Redevelopment Approaches for Legacy Systems</th>
<th>Explanation</th>
</tr>
</thead>
</table>
| 1  Big Bang Approach                        | - The Big Bang approach is sometimes also called Cold Turkey Strategy [42], and it means redeveloping a legacy system from scratch by choosing a modern architecture principles, tools and databases, running on a new hardware platform.  
- If a legacy system is reasonably sized, this approach could be termed as huge undertaking.  
- In real scenarios, the risk of this approach getting failed is too big and thus it is not suitable to be considered for scenarios involving big financial systems, or mission critical systems. |

Figure 4. Initial functional space of legacy system
When this approach is applied, it is necessary to guarantee that the redeveloped software will include all the functionality provided by the original legacy software apart from many new additional features as explained in Figure 4 and Figure 5.

Availability of accurate and updated documentation of legacy system plays a critical role in determining duration and complexity of the project as well as the risk of failure.

If the legacy systems are not isolated systems and are also interfaced with other systems then using this approach becomes more complex and the risk of new systems not meeting the integrity requirement of other systems become significantly large.

This approach could be used in situations when the legacy system being considered is not a mission critical system, small in size and have well defined and stable functionality.

<table>
<thead>
<tr>
<th>2</th>
<th>Database First Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The Database First approach is sometimes also called the Forward Migration Method [42], and it</td>
</tr>
</tbody>
</table>
involves the initial movement of legacy data to a modern such as relational or elastic, Database Management System (DBMS) and then incrementally developing the legacy applications and interfaces as shown in Figure 6 and Figure 7.

- Some advantages of applying this approach includes continued operational independence of existing legacy application while interfaces are being redeveloped and it results in interoperability between both the legacy and target systems

- The main advantage of applying this approach is that after successful migration of legacy data it becomes possible to use latest generation language and reporting tools that can create value by accessing the data which could translate into benefits.

- There are couple of disadvantages to this approach as well, as a specific example, this approach is only well suited for a fully decomposable legacy system. Also, in this approach the structure of data in legacy system can adversely limit the database structure in new system.

Figure 6. Initial state of legacy system

Figure 7. Final state of new system after database first approach.
In a nutshell this approach can be considered as a rather simplistic approach to legacy system redevelopment.

- The transfer and translation of the legacy data may take a significant amount of time during which the legacy system may be inaccessible and thus this approach is not suitable for mission critical information systems.

3 Database Last Approach

- The Database Last approach is sometimes also called the Reverse Migration Method [42], which is similar in concept to Database First approach and is also recommended for a well decomposable legacy system.

- Legacy applications in this approach are gradually developed on the new platform while the legacy database remains on the original platform as shown in Figure 8 and Figure 9.

- The Database Last approach has similar characteristic of a client/server paradigm. In this approach, the legacy database acts as
The Database Last approach has its share of problems, the most common is performance issues in gateway application developed to access and map legacy database.

- This approach relies on successful mapping of the new application database schema to the schema of legacy database and achieving a successful mapping may be quite complex and slow which would adversely affect the new application.

- Moreover, if the legacy system has archaic database, many of the standard advantages offered by new database systems in form of complex features such as integrity, consistency constraints, triggers etc. may not be exploitable, defeating the purpose of redevelopment.

- This approach is also more commercially acceptable than the Database First approach because it allows legacy application to operate uninterrupted while new application is being redeveloped.
4 Composite Database Approach

- This approach is also not suitable for mission control system as this approach requires a downtime on legacy system when the new application is accessing legacy database.

- The Composite Database approach [42] is suitable for fully decomposable, semi-decomposable and non-decomposable legacy systems as shown in Figure 10. All legacy systems have components that can be placed under each of the three categories.

In Composite Database approach, the legacy system and new system being developed are in parallel operation throughout the development project.

- The new applications are gradually rebuilt on the new platform by employing modern tools and technology. At first the new application is developed as a small functioning unit and as development progresses, it grows into a fully functioning system encapsulating the functionality of legacy system to a level satisfactory enough for legacy system to be retired as shown in Figure 11.
Some implementation of this approach may involve data being replicated across both the application instances, and thus may have to handle issues related to data integrity which requires a well-oiled sync strategy.

Even though the Composite Database approach eliminates the need for a single large migration of legacy data which is significant in a mission critical system yet it has significant overhead with respect to other two approaches because of added complexity.

The Chicken Little strategy [43] is a modified version of the Composite Database approach. This approach is suitable for fully, semi, and non-decomposable legacy systems as it employs multiple interfaces.

Different interface supports specific needs and achieve different functionality. The shared goal of these interface components is to mediate between operational software components of the two systems as shown in Figure 12.
• One advantage of this approach is that it insulates end users from all background infrastructure and processes during development.

• The interface infrastructure is responsible for handling many important functionalities such as to captures user authentication or makes system interface calls to some applications and redirects them to others.

• This approach can be understood as an 11 step plan that needs to be followed as shown in Figure 13. Steps are designed to handle a specific aspect of redevelopment, e.g. migrating the database or transforming web service response and can be easily adapted to fit individual legacy systems.

• This approach lacks focus on testing steps and methodologies which is a significant aspect of redeveloping legacy system.

Figure 13. 11 steps of Chicken Little Strategy approach.

Butterfly Methodology

• The Butterfly Methodology is suitable for those situations where the data of the legacy system is logically the most important part of the system.
2.5 Usability Evaluation Methods

There are four significant ways by which user interfaces are evaluated. They are either done automatically where a software program measures user interface with respect to some specified specification, or empirically where the application is tested by real users for usability, or formally where certain exact models and formulas are used to calculate usability measure, and informally where the experience of the evaluators are employed and the evaluation is mostly done on the basis of some rule of thumbs in UI design [44].

It has been seen that automatic and formal methods are difficult to use and apply as they are quite complex and don’t work so well for applications that have large user interfaces. Generally, projects rely on empirical method to evaluate user interfaces and thus employ experienced testers who acts as the end users and evaluates the application interface. Since, it is always difficult to find real users in large numbers to evaluate application in development stage and as there is generally a limited budget to hire real users for very long duration of development lifecycle, or due to project timeline constraints, informal methods of inspecting user interface are most widely used methods to evaluate user interfaces [44].

Usability inspection methods are set of different methods that employ different techniques to evaluate user interface, but all these methods rely on evaluators to inspect the interface. Mostly, usability inspection is carried out to find usability problems in application design,
however, certain methods also dwell upon the severity of problems in user interface and in the whole application design. Some of these inspections can be carried out even when the user interface specification is not formally documented, thus these methods can be applied quite early in the application development lifecycle [44] [45].

Usability inspection methods are not devoid of its fair share of problems. The most widely studied problem in user inspection method is called ‘The Evaluator Effect’ which is a problem that creeps in due to the different evaluators taking part in the evaluation of user interface. It has been found that when multiple evaluators evaluate the same user interface with same usability evaluation method, they come up with different set of problems and the amount of common problems in interface are generally quite less. Some studies have shown that the agreement between two evaluators on the usability problem of an interface they both have inspected using same methods can deviate widely between the range of 5% to 65%. Thus having multiple evaluators, inspecting the same interface using same methods can lead to a very extensive list of problems which may not be feasible to address and may have conflicting nature, however standardized evaluation goal, evaluation procedures and defect criteria can greatly improve the outcome of usability inspection method [46].

In the section below, few usability inspection methods are discussed in some details.

2.5.1 Cognitive Walkthrough

Cognitive Walkthrough (CW), which is based on a psychological theory by Polson and Lewis called CE+ that describes what makes a user interface easy to learn, is a usability inspection method [47]. The principle and the execution of this method is similar to requirement gathering or code walkthrough. This method has evolved considerably since it was first introduced in 1990 and the most widely used version was developed in 1994 by Warthon et al. [48] [49] [50].

The Cognitive Walkthrough methodology has two critical phases of preparation and execution. In preparation phase, the evaluator has to specify the tasks that will be evaluated and the knowledge, experience, and skills a user is expected to have. For each task in the list prepared by evaluator, the evaluator specifies the desired system action and expected system
response. In execution phase, the evaluator closely examines each action in sequence and asks the following four questions for each action:

- “Will the system allow user to achieve the right effect”?
- “Will the user notice that the correct action is available in the system”?
- “Will the user be able to associate the correct action with the effect trying to be achieved”?
- “If the correct action is performed, will the user see that progress is being made toward solution of the task”? [49][50]

If, for any question pertaining to any given action in the evaluator’s list, the evaluator can answer “yes”, then the evaluator should provide evidence to support by means of a credible application example for that question. If all four questions, that are stated above, for any given action has been answered by credible application example, then it may be considered that the user can be expected to have no difficulty completing the action. If, for any question pertaining to any given action in the evaluator’s list, the evaluator can answer “no”, then evaluator should be able to provide evidence to support by means of by a credible application example for that question which will indicate that a usability problem has been discovered.

We can summarize that for each question about any given task of the evaluator, the evaluator should identify a success or a failure scenario of the application which should be credible and only answering “yes” or “no”, without any credible example from the application won’t suffice [49][50].

There are different versions and variations of cognitive walkthrough that has been developed over the years. Even though these variations are still dependent on its theoretical foundations that aren’t related to software development, yet the CW method has evolved over time, mostly due to its easy applicability and practicality. These evolutions has compensated for some limitations and also takes into account the new progress made in the field of human-machine interaction (HMI) or the different HMI types such as web or multimedia [51][52].

Thus, cognitive walkthroughs can be an efficient method that can be applied for identifying certain issues of a Web site. This methods helps especially to determine how easy or difficult it is to learn or to navigate any application or website without reading the documentation. One of the main disadvantage of this method is that for more complex application design,
the process could be too time-consuming and since redesigning the UI is more difficult at later stages of software development life cycle, the cognitive walkthrough should usually be used early in development [50] [51] [52].

2.5.2 Heuristic Evaluation

Heuristic Evaluation is a method where a small group of usability experts evaluate an application using simple and general heuristics to create a structured critique. In Heuristic evaluation, the evaluators focus on examining and then judging the application interface based on well recognized usability principles, which are general rules describing common properties of a usable interface [53].

Heuristic evaluation is generally considered as one of the most efficient and cost effective method for identifying usability problems in application user interface design, especially when the project is in an initial phase. Generally, many usability problems can be identified by employing few usability experts as evaluators and they can easily identify usability problems in a short duration using this method of evaluation. The process requires each evaluator to go through the application interface several times in isolation and analyze and compare various elements present with the set of usability rules. The evaluators are also free to consider other usability principles apart from the stated ones and suggest possible improvements in interface design that can enhance application’s usability and aid user experience with using the application [53].

It has been found that 5 or 6 number of evaluators assigned to analyze an application is generally sufficient as they tend to find more than 75 percent of the usability issues in an application. Increasing the number of evaluators does not necessarily means that the defects identified will be significantly higher in number, as more evaluators tend to identify similar issues. Moreover, having a large number of evaluators add ups the cost of inspection and also the time required to complete the inspection. Even though heuristic evaluation is preferred to be performed by usability experts, it’s not a necessary condition as people with little usability experience can also perform this test [53].
In heuristic evaluation, once all evaluation has been conducted, the findings are aggregated and analysis of the finding can be started. The results should be documented in form of a report for each evaluator. Heuristic evaluation is generally best suited for applications that have large number of usability problems or in cases where the application is still being developed and hasn’t matured enough to have real users test the application. Heuristic evaluation for finding out results of usability problems is cheaper to conduct than formal usability testing and it can be completed in a very short period as well. Moreover, heuristic evaluation can be conducted at any stage of the application design phase, which adds to its advantage with respect to other evaluation methods [53].

2.5.3 Feature Inspection

Feature inspection is another method of usability inspection where evaluators lists sequence of features used to accomplish a typical task to check for long sequences required to complete a task, or identify cumbersome steps, or steps that do not follow natural way for users to try, or such steps that require extensive knowledge or experience in order to assess a proposed feature set [44] [54]. In case of feature inspection, the process is carried out by analyzing each set of features required to produce a desired output and the result is determined in terms of its availability, understandability and usefulness by an expert [55].

The usability attributes of an application helps in increasing the effectiveness, efficiency and satisfaction of a user with the application and helps in determining if specified users achieve specified goals in a particular environment. However, the effectiveness, efficiency and satisfaction also depends on other software quality attributes such as functionality, reliability and system efficiency, in addition to the relevant aspects of the context of use. The usability attributes of an application are thus only one aspect of the quality of an overall system [54] [55].
3 ANALYSIS OF LEGACY MTEE

In this section, the work carried out under this project has been described in great details. At first, the old MTEE tool has been analyzed to identify the features that are available to an end user. Also, the old system was reviewed to identify shot comings in terms of usability, accessibility and features present or absent in the existing application. New requirements were also analyzed to identify the set of new features that was needed to be developed. The analysis was first carried out to determine how the new features can be incorporated into the existing MTEE application. After detailed analysis, a development strategy was constituted to reengineer the existing MTEE application.

3.1 Overview of legacy MTEE

MTEE is Measurement Tool for Enterprise Education, which is a research tool having a set of questionnaire that has to be responded by teachers involved in entrepreneurship education across Finland. The development of MTEE was funded by European Social Fund and Finnish National Board of Education and LUT University was the coordinator and needed cooperating with National Board of Education, local authorities, teachers and third sector organizations to effectively use the existing tool for entrepreneurship education research. The objective of this research tool is to evaluate the quality of entrepreneurship education in Finland and provide methods and mechanism to improve the same.

For end users, this tool provides a self-assessment based survey to determine their entrepreneurship education competency and generates a feedback report that advices teachers on how to improve their entrepreneurship education delivery. The questionnaire are standardized research questions that have been carefully constructed to achieve the desired response from the user, and the result is thus comparatively analyzed across Finland.

The old MTEE version is currently running on two different server address with different URL’s. One version supports the MTEE tool in Finnish language and the other version supports the English language. There was another version which supported Swedish version but has been discontinued as the number of user’s using the version was quite few, which did not make it prudent to employ resources for its maintenance and for operational infrastructure. Since different language version had different application instant and
database, compiling the data of user responses for research is also performed separately. The application had a special URL from which the researchers could request user responses for different versions with few filters such as date of response, user location, gender and based on some values associated with user’s background information.

Figure 15 below shows the landing page of old MTEE tool. The landing page consists of a banner with tool name on the top, a navigational menu below it to navigate to different pages, a panel that displays contact information on the left, a central panel that welcomes the visitor and describes the tool and a login panel on the right hand side to allow registered users to login into the application. The login panel also provides the user with an option to change password and to register if the user hasn’t done so.

User registration allows user to create a user profile which consists of many information related to user’s name, sex, experience as entrepreneurship teacher, entrepreneurship experience, qualification, etc. These information helps researchers to compare user responses across different levels of teachers with different set of entrepreneurship experience and generate feedback suitable for them, based on the average response for the level considered. Figure 16 and Figure 17 below shows the user registration page and the information required to be filled by the user in order to register themselves on the application. Once registered, the application sends a verification link via a simple SMTP mail to the
registered email address. Once the user clicks on the link the verification process is complete and the user can proceed with taking the survey.

Apart from user login and registration the application landing page also has a navigational link to guidelines page, which provides links to resources that helps teachers to improve their entrepreneurship education skills. The English version in the old MTEE tool does not have a translated version in English for the guidelines and the page is not constructed well as the readability is low as shown in Figure 18 below. The page font’s and color selection make reading the text difficult and is also not well aligned. The page has a text which states...
“English guidelines coming” which makes end user feel that the page hasn’t been completely developed.

Figure 18. User guidelines page of old MTEE tool (English Version).

Once the user logs into the old MTEE webpage, the user is redirected to the questionnaire page where a small text line welcomes the user and provides a short description about the tool and its intended use. The user are also provided with the set of instructions about how they are supposed to answer the questions. Questions are part of a particular theme and can have same set of options or different options, depending on the question and the theme the question represents. Figure 19 below shows a sample of questions as they appear on the first page of the application. In the old version there are around 140 different questions that a user has to respond before a feedback can be generated. Once the user submits all the answers, the feedback is calculated based on average values for the responses.

Figure 19. User questionnaire page of old MTEE tool (English Version).
3.2 Overview of legacy MTEE Database

The old MTEE tool has a relational database structure and uses MYSQL database. There are sixteen relational database that represents the whole database schema. The table name and field names are in the Finnish language as shown in the Figure 20 below. Some of the most important relational tables in the database schema of old MTEE application and their significance is explained below in Table 5. There are other relational tables that are not explained in the Table 5 below, which were used mainly for certain validation.

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Significance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tblteemakysymysyhte</td>
<td>This table stores the relationship between a question and the theme it belongs to. The fields in this table store unique combination of theme id and question id. Each question must be associated with only one theme and this table maintains that relationship.</td>
</tr>
<tr>
<td>Tbleemat</td>
<td>This table stores the details of all the themes under which the questions belong. The fields in this table store theme name, theme information, the sequence in which they appear in the questionnaire page, theme heading and average values of user response pertaining to questions of this theme.</td>
</tr>
<tr>
<td>Tblkysymykset</td>
<td>This table stores the details of all the questions that the user has to respond to in the questionnaire. The fields in this table store question id, question text, question type id, question sequence in the theme, question heading and other information that controls how the question appear in the questionnaire.</td>
</tr>
<tr>
<td>Tblkysymystyypi</td>
<td>This table stores different question type available for the application. The fields in this table store only valid question type id. Different id signifies how the question options are represented in the questionnaire, such as if the option for the question will be a radio button, check box or dropdown.</td>
</tr>
<tr>
<td>Tblpisterajakoment</td>
<td>This table stores the details of feedback that is generated after the user has responded to the questionnaire. The fields in this table store feedback id, feedback text, minimum and maximum range,</td>
</tr>
</tbody>
</table>
theme for which the feedback is applicable and the average value of the user responses for that particular theme.

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TblKayttajat</td>
<td>This table stores the details of user credentials and is used to authenticate the user. The fields in this table store unique user id, password, user permission, user email and date of access.</td>
</tr>
<tr>
<td>TblTaustatiedot</td>
<td>This table stores the details of user information that are used to categorize the users. The fields in this table store user id, year of birth, gender, user’s entrepreneurship experience parameter and user’s location and school where the user is employed. These information constitute user’s background information and creates user profile in the system.</td>
</tr>
<tr>
<td>TblVastaukset</td>
<td>This table stores user responses to the questionnaire. The fields in this table store user id, question id and corresponding answer value and date of response.</td>
</tr>
</tbody>
</table>

Table 5. List of the most important relational tables of old MTEE application.

Figure 20. Database schema representation of old MTEE tool.
4 IMPLEMENTATION

The new MTEE web application was created after thorough analysis of the old tool and the functionality it offered. The new requirement was necessitated due to the change in the questions that were relevant for entrepreneurship education implementation research. The old tool did not have the feature for researchers to modify the questions and change the way they appear in the questionnaire. Since, it was not only the question text that had to be changed but also the corresponding options, which made it difficult to just update the database with executing SQL query as the applications logic also needed to be rewritten. There are specific validations coded into applications logic to make sure the data captured as user responses are suitable for analysis.

Another problem with the old MTEE application is that it did not have any documentation from the previous development activity that could help in understanding the application’s code. The code base are devoid of any comment that would explain the written code and its logic. Moreover, the table names, field names and the variables names were in Finnish language which made it difficult to read the code and understand the application code base for a developer who does not knows the Finnish language. These factors greatly impacted the feasibility of modifying the existing application code base to incorporate the new changes, or to even reuse the code base in new application version. Thus, it was decided that it will be more feasible to create a new version of MTEE application from scratch which would have all the functionalities offered by the old MTEE tool and would not have the drawbacks of the old tool. The best approach thus was to use Big Bang approach of reengineering legacy system, which has been described in great detail in section 2.4.

Once, it was decided that the new version of MTEE application would be developed from scratch, the development process followed a new software development process. The process started with creation of a project vision documentation that has been attached in Appendix 1. The project vision document outlined the project goal and the development outcome that needed to be achieved. Analysis of old MTEE tool helped in creating a Requirement Specification document that documented the different aspect of new MTEE tool such as the list of functionality, the different roles that user would have and the reports that would be required for the researchers. Agile Methodology was chosen as the software development
life cycle and the project requirement specification document was decomposed into small user stories to be developed. The project development plan was created to develop the new version of MTEE tool in two separate phases. In the first phase the emphasis was given on developing end user specific requirements such as welcome page, user registration, user login, questionnaire page and feedback page. However, some of end user specific requirement were dependent on data and parameters that would come from system admins, such as question text, options for the questions, the theme each question belongs to and the feedback parameters. Thus the phase 1 also had some admin user specific stories that were developed to make a functional base for the application. The second phase involved developing admin user specific functionalities such as generating report of user responses based on different filters. Once the report is generated the admin user can download the generated report. Apart from report functionality, the second phase development consists of many features that allows admin user to change the application parameters such as the sequence in which question under a theme appear or the sequence in which the themes are arranged.

4.1 New MTEE architecture

The new MTEE was developed with the goal of removing the shortcoming of the old application, which meant that the new application instance should have features for admin users to change the questions, number of options related to a particular question as well as the type of question, which would determine how the options will be presented to the user. In the legacy MTEE application changing the question type in the database required code changes to be done and this dependency on the developer was not an ideal solution for the project goal. Moreover, the application needed to be available in multiple language, the application architecture was needed to incorporate the same.

Apart from the design challenge of making the application’s logic dynamic to the question parameters, the design challenge also include creating a more usable application design with greater usability factor. The goal is to make new MTEE tool easy to use, simple and more attractive to end user. This can be achieved using a component based architecture, where the application can be constructed as a combination of multiple components that interact with each other using a predefined set of protocol in such a way that changing the data flow
between components do not lead to any or significant change in component design. In figure 21 below the design challenges are shown in part A.

Figure 21. A) Design challenges for new MTEE application. B) Design Solutions for new MTEE application.

We overcome these design challenges by employing a combination of some well-known design solutions in the domain of software engineering as shown in part B of figure 21 above. For example, MVC based UI framework has been known to aid rapid web application development. It’s a design paradigm that encourages modularity in application design, by making sure that the functional units are isolated from each other as much as possible and it helps developers modify the codes for one functional unit without bothering about other functional units in the application [56][57]. Similarly, component based software development process which supports development of web application as an assembly of components, which are reusable, easy to maintain, upgrade, customize or replace was chosen as a design solution [58][59]. Moreover, the application backend was designed keeping in mind Service Oriented Architecture (SOA) principles in mind, as it is well compatible with MVC framework that was employed for the front end development of user interface [60]. The advantage of using a restful architecture is that it can aids separation of concern in application development and separates the user interface design with application’s back end logic. This helps in supporting multiple application interface. The other advantage is that it makes application development faster and can be tested more easily. Apart from being advantageous in development process, using SOA based principles helps reduce maintainability of the application, aids reusability and increases configurability factor of a web application [61].
### 4.2 Technological Stack

The new MTEE application was designed using a combination of modern technologies that were compatible with the design solutions used for the reengineering of the legacy MTEE application. For example, AngularJS which is a new MVC based framework and helps in developing application and uses component service architecture principle in application design was chosen as front end technology framework. Figure 22 below shows the overview of Component Service Architecture which is the underlying principle of AngularJS framework.

![Figure 22. Overview of Component Service Architecture used in Angular.](image)

Table 6 below lists out the different software engineering technologies that is used in both the legacy version of the MTEE and the new reengineered version of MTEE along with the advantages of new technological stack over the old technological choices. The new reengineered MTEE uses a more comprehensive list of technological stack that are carefully chosen based on the specific requirement of application design and also that complemented the design solutions used in the development of reengineered MTEE application. Figure 23 below gives a high level overview of technological stack that has been used in the new reengineered version of MTEE application. The new application architecture uses a three tiered application design. The frontend application tier is encapsulated in AngularJS framework with a host of other technologies that makes user interface more dynamic, attractive and easy to use and consistent. The second tier is encapsulated in a light weight python based framework called Flask which helps in rapidly developing restful application backend that runs on server side and serves the AngularJS based user interface. The Flask
framework is easily to integrate with MongoDB using JSON. Thus the information flow between the frontend and database through flask based backend and vice versa does not require any transformation data as JSON is compatible across the tier.

<table>
<thead>
<tr>
<th>Legacy MTEE Technological Stack</th>
<th>New MTEE Technological Stack</th>
<th>Advantage over old technological stack</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTML 4</td>
<td>HTML 5</td>
<td>More consistent static page elements with better error handling and new UI elements are supported in HTML 5.</td>
</tr>
<tr>
<td>CSS</td>
<td>CSS, CSS3, Bootstrap</td>
<td>Using plain CSS does not support responsive design. Using Bootstrap along with plain CSS helps in creating more responsive web pages that adapt to different screen sizes. The old MTEE application was not responsive at all and display elements did not adapt themselves according to screen size.</td>
</tr>
<tr>
<td>JS</td>
<td>JS, Jquery, JSON, Angular JS</td>
<td>The old application used plain JavaScript to interact with the server side application base. The new technological stacks uses not only the plain old JavaScript but also uses host of other technologies that makes application more coherent, manageable, secure and supports better routing and secured interaction with server side application engine.</td>
</tr>
<tr>
<td>ASP.NET</td>
<td>Python, Flask 3.0</td>
<td>New MTEE application uses python based application backend developed using the light weight flask framework. This allows to develop application according to SOA principles and gives added advantage of modularity and reusability. The old tool used the classic ASP.NET scripting framework that is more cumbersome to learn and code.</td>
</tr>
</tbody>
</table>
A decision to use a NoSQL based database in the new version of MTEE was taken in order to make the application more adaptive, configurable and easily to integrate with the application backend. The fact that Mongodb uses JSON based schema representation makes inserting and updating the records more easy. Moreover, changing the database structure is pretty simple and a change in database structure does not break the application.

Table 6. Advantage of new technological stack over the old technological stack of MTEE application.

Figure 23. High level overview of new MTEE technological stack.
5 EVALUATION

In this section, evaluation of new reengineered MTEE application is presented in comparison to the legacy MTEE tool based on the literature review topics discussed in section 2. The evaluation of the reengineered tool is based on two most critical aspects. The first evaluation criteria is the evaluation of functionality offered by both legacy MTEE and the reengineered MTEE application. The second evaluation criteria is based on the usability evaluation based on the usability evaluation methods discussed in section 2.5.

Since, we have followed Big Bang approach of application reengineering, it is imperative that the new application should offer more functionality than the legacy MTEE tool. Table 7 below lists the difference between functionality offered by both the systems along with a small explanation of the improvements in comparison with legacy MTEE in terms of usability aspects based on esthetic user interface principles and user interface pattern which are explained in section 2.3.1, section 2.3.2 and section 2.3.3.

<table>
<thead>
<tr>
<th>Functionalit y</th>
<th>Legacy MTEE</th>
<th>New MTEE</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Landing Page</td>
<td>✔️</td>
<td>✔️</td>
<td>The landing page is present in both the version of MTEE application, but the landing page in the old version was not designed keeping in mind the aspects of good UI design explained in Section 2.3.1, Section 2.3.2 and Section 2.3.3.</td>
</tr>
</tbody>
</table>

Figure 24. Landing page of new MTEE application.

The new version of landing page as shown in figure 24 above, gives more preference to branding and provides...
only small amount of text that is relevant for user to understand what the tool is all about. The navigational menu on the top provides navigational buttons to user to explore other pages of the application. In the old version of MTEE, the landing page had too much information which was repeated across different pages and also did not had a context relevant branding element.

<table>
<thead>
<tr>
<th>User Registration</th>
<th>✔</th>
<th>✔</th>
</tr>
</thead>
</table>

The user registration page in the new MTEE is completely different from the one in the legacy version. In the legacy version, user’s background information such as their experience in entrepreneurship education domain and school as well as location information was required before the user was registered. The registration process is much simpler in the new version of the MTEE application.

![Registration panel of new MTEE application.](image)

Only 5 set of information, as shown in figure 25 above, is required for registering on the application which
makes it far easier for a user to register on to the new application. Moreover, the registration page has been made more inclusive by supporting other gender types and the registration panel has a more appealing design in comparison to the old version. The fields present in the registration form provides relevant guidance to the user about the type of information that is expected to be filled and also supports a reset option to clear the incorrect entry.

<table>
<thead>
<tr>
<th>Registration Email</th>
<th>✔️</th>
<th>✔️</th>
</tr>
</thead>
</table>

Both the legacy version and the reengineered version of MTEE application has the functionality to send an email message to the registered email address, but in the new version the content of email has been parameterized and stored in the database. Thus it can be easily updated without requiring any code change, which was not the case in the legacy MTEE version.

<table>
<thead>
<tr>
<th>User Login</th>
<th>✔️</th>
<th>✔️</th>
</tr>
</thead>
</table>

User login panel in both the versions are similar. The only change in the new version is that a user can use either email address or the user id and they can remained signed in, if they choose to do so using a toggle button provided in the panel as shown in figure 26. The old version only accepts registered email address and requires user to login again if they close the browser window.

![Login panel of new MTEE application.](image)

Figure 26. Login panel of new MTEE application.
<table>
<thead>
<tr>
<th>Feature</th>
<th>Old Version</th>
<th>New Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forgot Password</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Change Password</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Multilingual Support</td>
<td>✖️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

Forgot password functionality is provided in both the version of the MTEE tool. However, a key difference is that in the new version of the tool, first the email or user id is verified as shown in figure 27 and then a unique link is sent to the registered email address, after clicking which the user can update the password. In the old version, once the user gives the email address, an auto generated password is sent to the user. The user then uses the system generated password and has to again update the password, if they want to use a more memorable password.

![Figure 27. Recover password panel of new MTEE application.](image)

In the new version of MTEE the user can change the password by simply clicking the button and is redirected to update password screen as shown in figure 28 below. In the old version the user can also update the password by going to the profile page and update the password manually.

![Figure 28. Change password panel of new MTEE application.](image)

The legacy MTEE tool did not support multilingual application design. The legacy application had more than one instant running on different address to provide different language version of the tool. This method was...
not an ideal solution as the user registered in English version would not be able to access the Finnish version. Moreover, the changes had to be done in both the version which adds to development cost. In the new reengineered version of MTEE both the English and Finnish versions are running on the same application instance. The user just needs to click on the language change button as shown in figure 29 and figure 30 below and the content is translated without any redirection.

<table>
<thead>
<tr>
<th>Question Page</th>
<th>✔️</th>
<th>✔️</th>
</tr>
</thead>
</table>

The reengineered version of MTEE has a more appealing questionnaire page, where the questions are more aesthetically arranged. Each question panel consists of question text, question instruction and the relevant options as shown in figure 31 below. The way these questions appear are completely dynamic and changing question parameters do not require any change in the application code, which is not the case with legacy
MTEE version. Responding to these questions are far easier and the questionnaire is more readable.

![Question panel of new MTEE application.](image)

<table>
<thead>
<tr>
<th>Feedback Page</th>
<th>✔️</th>
<th>✔️</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>The reengineered version of MTEE has a more appealing feedback page, where the feedback for each response are more aesthetically arranged. User can select the date when the user has responded and can get the feedback for the date selected as shown in figure 32 below. The feedback page is more readable in the new reengineered version of MTEE.</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Feedback version of new MTEE application.](image)

<table>
<thead>
<tr>
<th>Admin Panel</th>
<th>✗</th>
<th>✔️</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>The new reengineered version of MTEE provides admin panel for admin users. The admin panel allows user to update different parameters settings related to the MTEE application. The panel provides option to create new questions, edit questions, change question parameters, change feedback parameters, create campaign code, generate and download reports and many other functionalities as shown in figure 33 below. In the legacy version, there was no admin panel that allowed admin users to change application parameter.</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Old</td>
<td>New</td>
</tr>
<tr>
<td>------------------</td>
<td>-----</td>
<td>-----</td>
</tr>
<tr>
<td>Create Question</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Edit Question</td>
<td>✗</td>
<td>✓</td>
</tr>
</tbody>
</table>

The create question panel is available in the reengineered version of MTEE for admin users to create question. The admin users can select the question parameters and provide the question text in different language as shown in figure 34 below. The user has the option to create multiple questions having similar parameter at one go. The legacy system did not had this feature and creating questions required executing SQL query.

The edit question panel is available in the reengineered version of MTEE for admin users to edit question. The admin users can edit the question parameters and question text in different language as shown in figure 35 below. The user has the option to search and edit multiple questions having similar parameter at one go. The legacy system did not had this feature and editing questions required executing SQL query.
Create Feedback Parameters

The feedback parameter panel is available in the reengineered version of MTEE for admin users to create and edit feedback parameters. The admin users can edit the feedback parameters and feedback text in different language as shown in figure 36 below. The user has the option to edit multiple parameters at one go. The legacy system did not had this feature and editing feedback parameter required executing SQL query.

Create Theme

The create theme panel is available in the reengineered version of MTEE for admin users to create new themes. The admin users can fill the theme details and provide the theme name and instruction text in different language as shown in figure 37 below. The legacy system did not had this feature and creating theme required executing SQL query.
<table>
<thead>
<tr>
<th>Create Question Type</th>
<th>✗</th>
<th>✓</th>
</tr>
</thead>
</table>

The create question type panel is available in the reengineered version of MTEE for admin users to create new question type. The admin users can fill the question type details and provide the different options text in different language as shown in figure 38 below. The legacy system did not have this feature and creating question type required executing SQL query.

<table>
<thead>
<tr>
<th>Progress Bar</th>
<th>✗</th>
<th>✓</th>
</tr>
</thead>
</table>

The reengineered version of MTEE application is more intuitively designed. For example, the user gets to know how many questions they have answered, which theme questions they are answering and unanswered questions are marked in different color as shown in figure 39 below. Once a question is answered the progress bar moves forward and the question number tile changes color to indicate the question has been responded. In the old version, no such visual feedback was provided to user.
The new reengineered version of MTEE application also provides the option to generate report of user responses as in the old tool. However, in the new version the reporting parameters can be dynamically decided based on multiple filters that are provided. The admin users can filter user responses based on multiple filter criteria as well as has the option to filter based on individual questions. In the old version the filter parameters to generate the report was quite limited.

Table 7. Evaluation of usability and functional features between legacy and reengineered MTEE.

Apart from the differences explained in table 7 above, the new version also provides more configurability and maintainability to admin users. For example, the error or success messages are completely configurable and shown according to the user's language preference. Similarly, the applications navigational menu is dynamic and changes according to the page user is on and also according to the user role. Moreover, the application user interface is responsive and supports multiple browser and device screen size without any difficulty. Thus the reengineered version is more usable and provides a better user experience to end user.
6 DISCUSSION

Software development is a complex process and requires mitigating multiple challenges [62]. In case of component based software development the challenges are more difficult as it requires a more robust understanding of software technologies, significant development experience and the ability to integrate multiple components in a single structurally coherent software unit [63]. The development process of reengineering MTEE tool followed the principles of component based software development which helped in creating a more robust MTEE application which is responsive, more usable, configurable and easy to maintain. The system supports dynamic binding between the UI and application data. Moreover, the application also supports easy translation of application content for the end user, which enhances user experience. Visual tools such as progress bar, color changing panels, etc. help simulate user activity and makes the application more interactive.

The reengineering process of MTEE had its own set of challenges. The initial challenge was to analyze the legacy MTEE tool in absence of any development documentation as well code comments which could explain the code logic. Moreover, the relational table in the database and corresponding field names were in Finnish language which made it even more difficult to analyze the legacy systems. Another challenge faced during the development of new MTEE was the significant change of requirements in terms of questions and their corresponding options between phase 1 and phase 2. This change resulted into significant redevelopment with development of additional functionalities, which resulted into significant increase in UI development and development effort and testing. Apart from requirement based challenges, there were other procedural and infrastructural challenges such as availability of secured hosting server in the LUT environment and unforeseeable delay in getting SSL certificate for hosting the site.

Even though the new version of MTEE is significantly better than the legacy MTEE version in terms of usability, functionality offered, aesthetics of user interface and is more interactive, there is still many scope of improvement that can be carried out in the future. For example, the new MTEE version could be made to support more language versions. In fact the next step is to adapt the new MTEE tool to support more languages as the tool will also be used in other European Union countries such as Italy, Spain, Sweden, etc. This
requires that the existing static content should then be made parameterized in the database. Moreover, there are many other reports that could be required in future and should be available to the admin users. Currently, only teachers and admin roles are available, but there could be more stakeholders in future who would be interested in the research results. Additionally, the reporting tool as well as the reporting result could be displayed more graphically in the application itself. Currently the entrepreneurship education researchers use another tool to statistically analyze user responses, which can be integrated in the existing tool in future.
7 CONCLUSIONS

In conclusion, it can be said that the redevelopment of legacy MTEE process met its objective and the process followed helped in meeting the research objective and finding the answer to the research question. Given below are the set of research questions and their respective answer that has been implemented in this project successfully.

• RQ1: What are the methods to identify short comings of an existing software?
Cognitive Walkthrough, Heuristic Evaluation and Feature inspection were some of the methods that helped us in analyzing the legacy system and identify the shortcomings in the legacy version of MTEE. The shortcomings identified ranged from functional shortcomings to usability specific problems that needed to be overcome in the new system.

• RQ2: What are the methods to enhance user experience?
There are many guidelines that describe the characteristics of good user interface as described in section 2.3.1. Following these guidelines we can create a well-structured, useable, interactive, intuitive and coherent user interface.

• RQ3: What can be done to increase user engagement on the application?
Incorporating the principles of aesthetic design described in section 2.3.2 helps us design a more coherent, logical, intuitive and simple user interface, which enhances user engagement and user experience. Additionally, using user interface design principles helps us to create a more appealing interface and present information in a more coherent manner to the end user. These methods enhance user engagement and should be followed.

• RQ4: How to make an application robust?
Developing application using component based software development technique and adopting service oriented architecture makes the application development more modular, less interdependent and easy to change. Thus the application developed is more robust, easily maintainable and configurable.

• RQ5: Which design principles suits best when it comes to reengineering a software?
Section 2.4 lists different reengineering techniques that can be used when it comes to reengineering a legacy application. In this case Big bang approach was used as the reengineered system had a completely new set of requirement which was independent from the old legacy system. Also the product developed was comparatively smaller than enterprise level application, in which case the functional space is quite large as well as historical data holds significant value even for the reengineered system. In cases, where the development activity is significantly greater, other techniques can be used.
REFERENCES


8 VISION DOCUMENT

1. Introduction

a. Overview:

This project goal is to redesign and enhance the existing Measurement Tool for Entrepreneurship Education (MTEE), which was developed in a timeline of 2008-2012, as a self-assessment tools for teachers to enable them to monitor the implementation of entrepreneurship education. MTEE development was funded by European Social Fund and Finnish National Board of Education and Co-ordinated by Lappeenranta University of Technology (LUT).

b. Purpose:

MTEE uses a set of questionnaires related to planning and development of entrepreneurship education that teachers need to fill out every six months and based on the answers, an immediate feedback is generated with respect to the average scores, and additionally tips and ideas are provided to improve the teaching methods. This tool is the data collection tool for research into different teaching methods, their applicability, and effectiveness, as well as to, suggest steps for improvement.

c. Block diagram of system:

Illustrates basic components of the system and depicts external interfaces to the environment.

![Figure 1. MTEE Application Components](image)
2. **Design Constraints and Considerations**

   a. Assumptions / dependencies:
      
      i. This will be used for collecting answers to the questionnaire.

      ii. Primarily, around eight thousand teachers in Finland will use it for few times in a year.

      iii. The application should be configurable.

   b. Design Drivers

      i. Easy to use, reliable and secured system.

      ii. Should be configurable in terms of questions and feedback given.

      iii. Easy to use as an application to conduct a quantitative analysis of user responses.

3. **System Usage**

   a. Describes the domain served by the system.

      This system will be used as a delivery channel of questionnaires related to planning and development of entrepreneurship education to teachers across Finland. Teachers can answer a set of questions that are then used to assess their result compiled as a feedback generated with respect to the average scores and additionally tips and ideas are provided to improve the teaching methods. All the reports, feedback and charts will be generated or stored and can be viewed in the webpage for admin users.

   b. Describes the end users of the system.

      There are three types of users for this system.

      i. Subject group users: These users will be of the primary group in terms on number of users of the system and will answer the questionnaires.

      ii. Application Admin: These persons will have admin privileges and will be able to create questions, access reports, etc.
iii. External Users: The external stakeholders who may have limited privileges to the application.

c. Describe the target environment for the system.

The system would be accessible as a webpage, where a user will be authenticated.

Software platform: Web browser.

Hardware platform: Laptop, Desktops, Tablets.

d. Important use cases:

i. Users are able to register into the system.

ii. Subject group users are able to access the questionnaire and submit their responses. They will also be able to access their previous responses and update their profile information.

iii. Admin user can generate reports based on different filter criterions of users and their user responses.

iv. Admin user can create questions, manage user privileges

4. System Requirements and Supporting Features

a. Desired system features table:

<table>
<thead>
<tr>
<th>Feature</th>
<th>Rationale</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accessible</td>
<td>The application should be available on any web browser.</td>
</tr>
<tr>
<td>Role based access to system</td>
<td>User should have access to only those functionalities that match their roles.</td>
</tr>
<tr>
<td>Security</td>
<td>The application should have proper security system where user information will not be revealed to public.</td>
</tr>
<tr>
<td>Webpage Application</td>
<td>User wants a web page based application where they can interact with the system.</td>
</tr>
</tbody>
</table>
b. User Requirements / system features table:

<table>
<thead>
<tr>
<th>User Requirement</th>
<th>Supporting Feature(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ease of use</td>
<td>Application should be easy to use for an average user.</td>
</tr>
<tr>
<td>System shall store each</td>
<td>Using an open database access API allows for interaction with existing database servers.</td>
</tr>
<tr>
<td>interactions</td>
<td></td>
</tr>
<tr>
<td>Secured system</td>
<td>User cannot use this system without registration so no unauthorized access.</td>
</tr>
<tr>
<td>Configurable</td>
<td>Application should be configurable for similar use.</td>
</tr>
</tbody>
</table>
SOFTWARE REQUIREMENTS SPECIFICATION

For

MTEE platform

Version 1.0

Prepared by Ajesh Kumar

LUT
**Introduction**

**Purpose**
This document describes the software requirements of MTEE web application.

**Document Conventions**
N.A.

**Intended Audience and Reading Suggestions**
This document is aimed mainly for developers, testers, project supervisors and project owners.

**Product Scope**
MTEE is a platform for teachers to answer questionnaire about their teaching methods, in order to evaluate their teaching in the domain of entrepreneurship education. The responses made by teachers are stored in database and is evaluated based on the average scores, and a feedback is generated for the respondents.

The admin panel will be used to create and update the questionnaire, generate required reports, manage users and control application settings.

**Overall Description**

**Product Perspective**
In the earlier version of the MTEE application, many application elements were static. This made it difficult to change the questionnaire. However, the application used to record user data quite meticulously with respect to analysis required in later part of the research, which remains a persistent requirement, even though the questions needs to be changed. Keeping in mind the application specific requirements of configurability, language independence and role based access, a technological stack was chosen, which met these specifications.

The application is designed as composition of three primary components of ‘MVC (Model-View-Controller) based front-end application’ which will be serving application content over HTTPS, a ‘Python Rest Flask framework based back-end web-services application’ which will be processing all the user requests and responses and a ‘NoSQL based MongoDB database server’ to store the data.

**Product Functions**
- The user should be able to register, login and logout while using MTEE application.
- The user should be able to answer the questionnaire about their teaching activity.
- The user should be able to view feedback or suggestions based on the responses.
- The user should be able to review their previous responses.
- The user should be able to update user profile.
- The admin should be able to configure the questionnaire.
- The admin should be able to manage users.
- The admin should be able to generate reports based on different criterion.
User Classes and Characteristics
● Subject Group Users – Answers the questionnaire.
● External Stakeholders – Restricted application access.
● System Administrators – Application Control.

Operating Environment
The application should be hosted on two servers, one to run the application core that will serve the application UI and a database server to host the user data. This type of configuration can be modified easily to run on three server setup or single server setup with different nodes.

Design and Implementation Constraints
The MTEE front-end will be based on HTML5, Bootstrap, Jquery, Javascript, JSON and Angular. The styling of the user interface will be based on CSS and CSS3. The back-end of the MTEE application will be based on Rest Flask framework with python as core. The backend components of MTEE platform will be based on python. The system uses MongoDB as the data storage.

User Documentation
There will be a user guide/help pages in the MTEE platform user interface. The platform itself will have a site guide/offer recommendations/offer tooltips.

Assumptions and Dependencies
The MTEE application will primarily be used by approximately eight thousand teachers across Finland in either English or Finnish few times in a year.

The application will be accessed using a web browser primarily over desktop, laptops, notebooks and tablets. Though the application could be accessed on mobile based browsers, the UI components may not function effectively.

The whole project will be developed in separate phases, hence some of the requirements will be handled in next phase, based on the criticality of the requirement.

External Interface Requirements
User Interfaces
The user interface will comprise of the following software components: HTML5, CSS, CSS3, Angular, JQuery, JavaScript, JSON and Bootstrap.

Hardware Interfaces
The MTEE application will be accessible over a web browser primarily over desktop, laptops, notebooks and tablets.

Software Interfaces
MTEE platform currently does not require any external software interfaces. However, in future such use may be required. For intra-application exchange of payload, JSON based messages are exchanged between the components.

Communications Interfaces
### System Features

#### MTEE Application Access

Description and Priority
The user needs to be able to access MTEE Application. This is critical to the system and therefore High priority.

Stimulus/Response Sequences
The user enters the registered user name and password and is able to access the system. A new user is able to register into the system.

Functional Requirements
AA-REQ-1: The user should be able to register.
AA-REQ-2: The user should be able to log in using the registered user name and password.
AA-REQ-3: The user should be able to logout from the application.
AA-REQ-4: The system should validate the user credentials before access is granted.
AA-REQ-5: The system must give a response to the user so that he/she knows whether his actions were successful or requires guidance.

#### Responding to Questionnaire

Description and Priority
The user should be able to respond the questionnaire, and while doing so should be able to edit their responses before final submission. This is critical to the system and therefore High priority.

Stimulus/Response Sequences
When a user arrives at the questionnaire page, a set of questions with possible answers are shown for user to make a selection and these selections are then stored to the database. The user receives a feedback based on the response and can access response history.

Functional Requirements
RQ-REQ-1: The user should be able to see the questionnaire made available by the system
RQ-REQ-2: The user should be able respond to the questions
RQ-REQ-3: The user should be able to navigate between the questions.

#### Profile Management

Description and Priority
The user should be able to record details about profile and should be able to edit the information provided. This is critical to the system and therefore High priority.

Stimulus/Response Sequences
When a user arrives at the profile page, a set of input boxes requiring a varied set of information about the user is shown for user to fill as best applicable. These information is then stored into the database.

**Functional Requirements**

PM-REQ-1: The system should be able to show profile management page to user.
PM-REQ-2: The system must allow the user to fill or update profile information.
PM-REQ-3: The system must be able to store the user’s profile.
PM-REQ-4: The system must allow admins to access user profile.

**Admin Module**

Description and Priority
The system should allow the admin access, which allows the application admins to access the application’s settings panel, consisting of different application control options that an admin can change. This is critical to the system and therefore High priority.

Stimulus/Response Sequences
When the admin user logs in using his registered credentials, he is redirected to admin panel where different options are available to him.

**Functional Requirements**

AM-REQ-1: The system should allow admin user access.
AM-REQ-2: The system should direct admins to application settings page.
AM-REQ-3: Admins should be able to change different application settings such as related to questionnaire, application user, feedback and reports.
AM-REQ-4: The system must allow admins to logout.

**Admin Reports**

Description and Priority
The system should allow admins to generate different reports in a specified file format, based on different search criteria. This is critical to the system and therefore High priority.

Stimulus/Response Sequences
Admin user requests a report by selecting different search filters and a report is generated in specified file format.

**Functional Requirements**

AR-REQ-1: The system should allow user to request a report.
AR-REQ-2: The system should generate the report in specified file format.
AR-REQ-3: The system must allow the change of filter criteria.

**Other Nonfunctional Requirements**

**Performance Requirements**
The system performance is in some cases could be critical, for example, when undertaking a high user load campaign and analysis.

**Safety Requirements**
N.A.
Security Requirements
The application should only be accessed after providing a registered user credentials. Admin access credentials should not be misused.

Software Quality Attributes
Some of the required quality attribute of MTEE application are adaptability, availability, correctness, flexibility, maintainability, portability, reliability, reusability, robustness, testability, and usability.

Business Rules
User (data provider)
Admin user (administering the system)
City official user (retrieving data from the system and acting if needed)