

# INTERNATIONAL SCHOOL OF MANAGEMENT AND TECHNOLOGY

KATHMANDU, NEPAL

Qualification		Unit Number & Title	
BTEC HND IN COMPUTING		J/615/1631 – Unit 9: Software Development Lifecycle	
Student Name		Assessor Name	
Sparsh Shrestha		Roshan Kandel	
Assignment Launch Date	Due Date		Completion Date
17 October 2022	16 December 2022		
Session/Year	01/2019	Assignment Number	1/1
Assignment Title		Birtamod Technical College Assignment Management System	

## Assignment submission format

Each student has to submit their assignment as guided in the assignment brief. The students are guided what sort of information is to produce to meet the criteria targeted. You are required to make use of headings, paragraphs and subsections as appropriate, and all work must be supported with research and referenced using the Harvard Referencing Style.

### Important:

- **Read the plagiarism notice and requirements at Page 6**
- **Word-limit- 8000 words** (*excludes cover page, table of content, figures, graphs, reference list, appendix and logbook*)
- **Accepted Sources: Research Papers** (*Journal Articles, Conference Proceedings, Thesis, Text Books, Governmental Data, Websites (only a registered organization, an educational institution, government agency)*)
- **Information taken from unreliable sources will not be accepted**
- **Must follow Harvard Reference Style**

### Learning outcomes covered

- LO1 Describe different software development lifecycles.
- LO2 Explain the importance of a feasibility study.
- LO3 Undertake a software development lifecycle.
- LO4 Discuss the suitability of software behavioural design techniques.

### Scenario 1

Birtamod Technical College is a renowned educational institution in eastern part of Nepal. It is located at Birtamod that provides wide range of programs like IT, Engineering & MLT. It has around 200 students, 3 Bachelor programs & over 20 highly qualified faculties that is serving as a technical hub in Jhapa district.

Birtamod Technical College is seeking an enhanced solution that automates assignment submitted by students to their concerned departments. You are hired as a System Analyst (SA) which is developing a new Assignment Management System. Proposed system will be able to handle assignment submitted by each student, submission records of each student, feedback provided by each tutor and tracking of the assignments submitted.

You are responsible for managing the project including analysis and design stage of the new system. Being a SA, you are suggested to use modern software development model in developing the Assignment Management System.

#### **Features to be included in your proposed project are**

1. Login authentication for each Student, Tutor & Assignment Department
2. Subject-wise Assignment with multiple Attachment Available
3. Assignment Approve/Reject by Tutor
4. Grades to be provided to students via notification.
5. Record of assignment, submitted by each student

Also consider non-functional requirements yourself after discussion with your development team.

As a part of your responsibility, you are required to **prepare a report** as guided below.

### Assignment Task – Part 1

As a part of your responsibility, you are required to **prepare a report** that describes different software development cycles. Your report must include the following:

1. Description of predictive and adaptive software development models considering at least two iterative and two sequential models.
2. The risks involved in each of the models and how the risk can be mitigated /managed in each model by taking a reference of the spiral model.

Once you have prepared the **report** you are required to produce documentation that

3. Describe with an example why a particular lifecycle model is selected for a development environment. You may consider a different development environment with different business logics, platform, etc., and the applicability of the particular software development model over the development environment.
4. Assess the merits of applying the waterfall model to a large software development project with appropriate evidenced researched work with reference to the current context.

### Assignment Task – Part 2

1. You are required to **produce a documentation** that **explains** the purpose of the feasibility report and **describe** how technical solutions can be compared.
2. **Prepare a brief report** discussing the components of the feasibility report.
3. Carry out the feasibility study as per the best of your previous research work against the solution to be developed for the given problem and **assess** the impact of different feasibility criteria on the software investigation.

### Assignment Task – Part 3

1. Undertake the **software investigation** to meet the business need using appropriate software analysis tools/techniques to carry out a software investigation and create a supporting documentation. You may submit this task in the form a **report structured** with background information, problem statements, data collection process and summary etc.

*In order to carry out the systems investigation you are required to identify the stakeholders, identify the requirements of the client, specify the scopes like inputs, outputs, processes and the process descriptors, consideration of alternative solutions and security considerations and the quality assurance applied.*

You are also required to identify the constraints like costs, organizational policies, legacy systems, hardware requirements etc.

For software analysis you may use the following tools:

- Data Flow Diagram up to second level
- Entity Relationship Diagram

2. Reference to your task above that required some level of intensive research work **analyse** how software requirements can be traced throughout the software lifecycle.
3. **Discuss** different approaches to improve the software quality and considering the above context discusses the two approaches that can be applied at this context to improve the software quality.
4. **Critically evaluate** how the use of the function design paradigm in the software development lifecycle can improve the software quality. Support your ideas with reference to the tasks you have done.

#### Assignment Task – Part 4

1. **Prepare a documentation** that explains how user and software requirements have been addressed. You may tabulate this task with the columns that has the expected client requirements and the actual output of the product to be developed after the appropriate analysis.
2. **Discuss** about the different software specification methods and suggest two software behavioral specification methods and illustrate their use with an example relevant to the project that needs to be constructed for the given context. Some of the software specification techniques include flowcharts, pseudo code and formal specification methods and so on.
3. **Differentiate** between finite state machines (FSM) and an extended Finite State providing an application for both.
4. **Present** justifications of how data driven software can improve the reliability and effectiveness of the software.

#### Learning and Assessment Criteria

Pass	Merit	Distinction
LO1 Describe different software development lifecycles		D1 Assess the merits of applying the Waterfall lifecycle model to a large software development project.
P1 Describe two iterative and two sequential software lifecycle models.	M1 Discuss using an example, why a particular lifecycle model is selected for a development environment.	
P2 Explain how risk is managed in software lifecycle models.		
LO2 Explain the importance of a feasibility study		
P3 Explain the purpose of a feasibility report.	M2 Discuss the components of a feasibility report	D2 Assess the impact of different feasibility criteria on a software investigation.
P4 Describe how technical solutions can be compared.		

<b>LO3 Undertake a software development lifecycle</b>		
<b>P5</b> Undertake a software investigation to meet a business need.	<b>M3</b> Analyse how software requirements can be traced throughout the software lifecycle.	<b>D3</b> Evaluate the process of undertaking a systems investigation with regards to its effectiveness in improving software quality.
<b>P6</b> Use appropriate software analysis tools/techniques to carry out a software investigation and create supporting documentation	<b>M4</b> Discuss two approaches to improving software quality.	
<b>LO4 Discuss the suitability of software behavioural design techniques</b>		
<b>P7</b> Discuss using examples the suitability of software behavioural design techniques.	<b>M5</b> Analyse a range of software behavioural tools and techniques.  <b>M6</b> Differentiate between a finite state machine (FSM) and an extended-FSM, providing an application for both.	<b>D4</b> Present justifications of how data driven software can improve the reliability and effectiveness of software.

### Grades Achieved

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

**Note:** Refer the unit details provided in your handbook when responding all the tasks above. Make sure that you have understood and developed your response that matches the highlighted key words in each task.

## Plagiarism Notice

You are reminded that there exist **Academic Misconduct Policy and Regulation** concerning **Cheating and Plagiarism**.

### Extracts from the Policy:

**Section 3.4.1:** Allowing others to do assignments / Copying others assignment is an offence

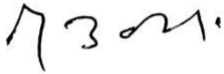
**Section 3.4.2:** Plagiarism, using the views, opinion or insights / paraphrasing of another person's original phraseology without acknowledgement

### Requirements

- It should be the student's own work – **Plagiarism is unacceptable**.
- Clarity of expression and structure are important features.
- Your work should be submitted as a **well presented**, word-processed document with headers and footers, and headings and subheadings.
- You are expected to undertake research on this subject using books from the Library, and resources available on the Internet.
- Any sources of information should be **listed as references** at the end of your document and these sources should be referenced within the text of your document using **Harvard Referencing** style
- Your report should be illustrated with screen-prints, images, tables, charts and/or graphics.
- All assignments must be typed in **Times New Roman, font size 12, 1<sup>1/2</sup> spacing**.

**The center policy is that you must submit your work within due date to achieve “Merit” and “Distinction”. Late submission automatically eliminates your chance of achieving “Merit and Distinction”. Also, 80% attendance is required to validate this assignment.**

I declare that all the work submitted for this assignment is my own work and I understand that if any part of the work submitted for this assignment is found to be plagiarised, none of the work submitted will be allowed to count towards the assessment of the assignment.

<b>Assignment Prepared By</b>	<b>Signature</b>	<b>Date</b>
Roshan Kandel		Oct 10, 2022
<b>Brief Checked By</b>	<b>Signature</b>	<b>Date</b>
Dhruba Babu Joshi		Oct 14, 2022



## Table of Contents

Introduction.....	1
Software .....	1
Types of software.....	1
Application Software .....	1
Types of the application software .....	2
System software .....	3
Important features of system software .....	3
Types of system software.....	3
Attributes of Software .....	4
SDLC .....	4
Why SDLC is need .....	4
Phases of SDLC .....	5
Requirement & Analysis (Stage 1) .....	5
Project planning (Stage 2).....	6
Design (Stage 3).....	6
Coding & implementation (Stage 4) .....	6
Testing (Stage 5) .....	7
Deployment (Stage 6) .....	7
Maintenance (Stage 7) .....	7
Iterative Model.....	7
Phases.....	8
Requirement and Planning Stage .....	8
Design Stage .....	9
Coding Stage.....	9

Testing Stage.....	9
Evaluation Stage .....	9
Two types of Iterative Model.....	10
Agile Model .....	10
Phases.....	11
Concept .....	11
Inception or Requirement Identification.....	11
Development .....	11
Release .....	11
Maintenance .....	11
Retirement.....	12
DSDM.....	12
Phases.....	12
Feasibility Study .....	12
Business Study .....	12
Functional Model Iteration .....	12
Design and Build Iteration .....	13
Implementation .....	13
Sequential Model .....	13
Waterfall Model .....	13
Phases.....	14
Analysis.....	14
Design .....	14
Implementation .....	14
Testing.....	14

Maintenance .....	15
Merit of applying water fall model .....	15
Why waterfall model is best for large company .....	15
RAD Model.....	16
Phases.....	16
Requirement gathering and analysis .....	16
Prototyping.....	17
Development .....	17
Integration and testing.....	17
Deployment.....	17
Spiral Model.....	17
Phases.....	18
Identification .....	18
Design .....	18
Construct or Build.....	18
Evaluation and Risk Analysis .....	18
About Risk .....	19
Roles of Spiral Model to Mitigate a Risk .....	20
Identifying and Evaluating Risks.....	20
Planning and Design .....	20
Building and Testing.....	20
Evaluating and Revising .....	20
Conclusion .....	21
Introduction.....	21
Feasibility Study .....	22

Requirement gathering techniques.....	22
Interview .....	22
Focus group.....	22
Survey .....	23
Prototype .....	23
Document analysis .....	23
Purpose of Feasibility Study .....	23
Some of the reason why Feasibility Study is need .....	24
Avoiding costly mistakes .....	24
Maximizing the potential returns .....	24
Ensuring the availability of resources.....	24
Determining the market demand.....	24
Providing a detail plan .....	24
What is Feasibility Report .....	24
Component of Feasibility Study .....	24
Project Scope .....	24
Analyzing the Present Situation.....	25
The necessities .....	25
Review .....	25
Technical solution.....	25
Technical solution to the software .....	26
Benefits of using Feasibility Study .....	27
Identifying Potential Problems and Risks.....	27
Determining the Feasibility of the Project .....	27
Identifying the Resources needed .....	28

Establishing a Baseline .....	28
Building Consensus and Support .....	28
Feasibility Criteria .....	28
Technical Feasibility .....	28
Financial Feasibility .....	28
Economic Feasibility .....	29
Legal Feasibility .....	29
Operational Feasibility .....	29
Conclusion .....	29
Introduction .....	30
Background Information of new software .....	30
Problem Statement .....	30
Identifying and defining the problem that the software will solve .....	30
Developing a feasible and effective solution .....	31
Managing resources and staying within budget .....	31
Ensuring user adoption and satisfaction .....	31
Stakeholder .....	31
Different types of Stakeholders .....	32
Identify Stakeholders .....	32
Requirements .....	32
Requirement Identification .....	33
Owner .....	33
Staff .....	33
Teacher .....	34
Requirement Specification .....	34

Scope.....	35
Scope of the Project .....	35
Consideration of alternate solution .....	37
Identification of Different constraints.....	38
Cost .....	38
Legacy.....	39
Organizational Policy.....	39
Hardware Requirement .....	39
Background Information.....	40
Problem Statement .....	40
Data collection Process .....	40
Methods of Data collection process .....	40
Surveys.....	41
Interviews.....	41
Focus Groups .....	41
Observation Studies .....	41
Experiments .....	41
Case Studies .....	41
Secondary Data .....	41
Software Analysis .....	42
DFD.....	42
Context Diagram.....	43
Level 0 diagram .....	44
Level 1 Diagram .....	45
Level 2 Diagram .....	46

ER Diagram .....	46
Case Tool .....	47
Software Requirements Specification (SRS) Document .....	48
The uses of SRS .....	49
Defining the scope of the project .....	49
Establishing a common understanding .....	49
Providing a reference point .....	49
Facilitating communication .....	49
Improving the quality .....	49
Steps to Improve Software Quality .....	49
Some of the advantage of software testing .....	50
Improved Quality .....	50
Increased reliability .....	50
Reduced costs .....	50
Improved user satisfaction .....	50
Increased efficiency .....	50
Enhanced security .....	50
The suitability of software behavioral design techniques .....	51
Software Performance Specification .....	52
Improving of Software Behavior Specification in SDLC .....	52
Defining requirements .....	52
Facilitating communication .....	53
Improving quality .....	53
Reducing rework .....	53
Conclusion .....	53

Introduction.....	54
Documentation of user and system software .....	54
Algorithm.....	55
Example .....	56
Benefits .....	56
Efficiency .....	56
Accuracy .....	56
Repeatability .....	56
Adaptability.....	57
Automation .....	57
Flow chart .....	57
Benefits .....	58
Visualize and Document Processes .....	58
Identify Inefficiencies and Problems .....	58
Facilitate Communication and Collaboration .....	58
Improve Process Efficiency .....	58
Enhance Decision Making .....	58
FSM (Finite State Machine).....	58
EFSM (Extended Finite State Machine) .....	60
Data Driven Software .....	61
Application driven software.....	62
Improving the reliability and effectiveness of the software using data driven software .....	62
Data-driven decision-making.....	62
Improved accuracy .....	62
Enhanced adaptability .....	62



Increased efficiency .....	63
Conclusion .....	63
References .....	64

## List of Figure

Figure 1 .....	2
Figure 2 .....	3
Figure 3 .....	5
Figure 4 .....	8
Figure 5 .....	10
Figure 6 .....	19
Figure 7 .....	21
Figure 8 .....	22
Figure 9 .....	31
Figure 10 .....	43
Figure 11 .....	44
Figure 12 .....	45
Figure 13 .....	46
Figure 14 .....	47
Figure 15 .....	57
Figure 16 .....	59
Figure 17 .....	61

## **Introduction**

In this part, I discussed software and its types, like application software and system software. After that, I discussed the attributes of software and its application. The software part is finished. After finishing the software section, I discussed SDLC, including why it is necessary and what the SDLC phases are. And then I jumped to the models. models like adaptive models, sequential models, agile models, spiral models, and waterfall models, and also write about the phases of all these models. After that, I write about the merits of applying a water model to a large project.

## **Software**

Software is the collection of programs. It is also programs and other operating information that is used by the computer. Software is also the opposite of hardware, which describes the physical aspects of a computers. A device's running programs, scripts, and applications are collectively referred to as "software" in this context. It can be compared to the variable component of a computer, whereas the invariable component is the hardware.

## **Types of software**

There are various types of software which are:

### **Application Software**

Another name of application software also called software application, or application, or app. An application software is a computer program that designed for a specific task relating to the operation of the computer itself and also it is also used by end users. Application software is also the most common software.

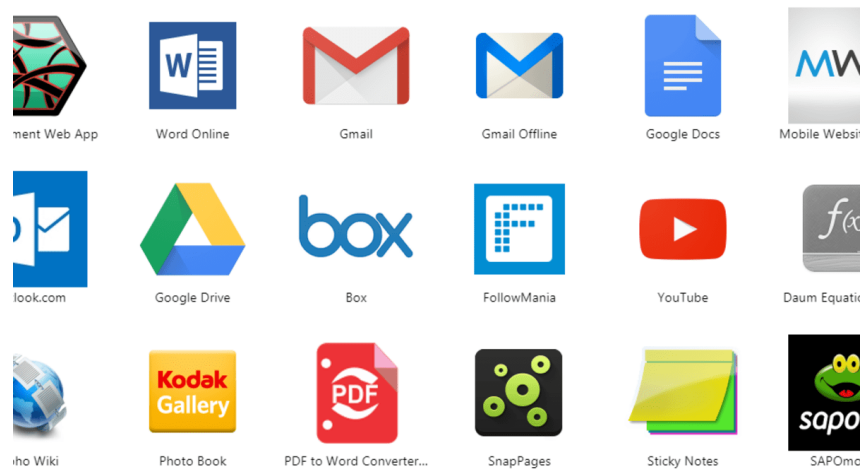


Figure 1

## Types of the application software

### 1) Microsoft software like:

- a) MS Office
- b) PowerPoint
- c) Word
- d) Excel and Outlook.

### 2) Common Internet Browser like:

- a) Chrome
- b) Firefox
- c) Opera
- d) Safari

### 3) Graphic design software like:

- a) Adobe Photoshop
- b) Adobe Premier Pro
- c) Adobe Illustrator
- d) CorelDraw
- e) Canva

## System software

System software is a types of computer program which is designed to run computer hardware and application. System software is also used to manage the computer itself, which runs in the background to maintain the computer basic functions so user can run higher end application software.



Figure 2

## Important features of system software

System software is typically created by computer makers as an important component of the computer. This software's main duty is to establish a conduit between the end user and the manufactured computer hardware.

## Features of system software

- High speed
- Hard to manipulate
- Written in a low-level language
- Versatile

## Types of system software

- Operating system like: Windows, Linux, MAC
- Device driver

- Firmware
- Utilities

### Attributes of Software

Numerous linked attributes of software products reflect the quality of the software. These characteristics have little to do with what the software really accomplishes. They depict how the program behaves when it is running as well as the organization and structure of the source code and any related documentation. The following qualities are crucial for good software:

- **Acceptability:** Software needs to be suitable for the users it is intended for. This calls for it to be comprehensible, practical, and interoperable with the other systems they employ.
- **Dependability and security:** Several characteristics, including as dependability, security, and safety, are included in software dependability. If a system fails, dependable software shouldn't result in harm to people or property. Software must be protected against malicious users so that the system cannot be accessed or damaged.
- **Efficiency:** Software shouldn't use system resources like memory and CPU cycles inefficiently. Therefore, efficiency involves responsiveness, processing speed, resource use, etc.
- **Maintainability:** Software should be created in a way that allows it to change over time to accommodate customers' evolving needs. This is a crucial quality since a changing business environment necessitates software modification, which is unavoidable.

### SDLC

The Software Development Life Cycle (SDLC) is a method for creating software that is as high-quality and inexpensive as possible in the least amount of time. Additionally, the SDLC offers a well-organized phase structure that enables an organization to quickly create high-quality software that has undergone thorough testing and is prepared for usage in production.

### Why SDLC is need

- System Development Life Cycle offers a basic for project planning, scheduling and estimating
- It also provides a framework for a standard set of activities and deliverables

- SDLC is increase visibility of project planning to all involved stake holders of the development process.
- It helps to decrease project risk and project management plan overhead

### Phases of SDLC

Here are the some of the phases of SDLC:

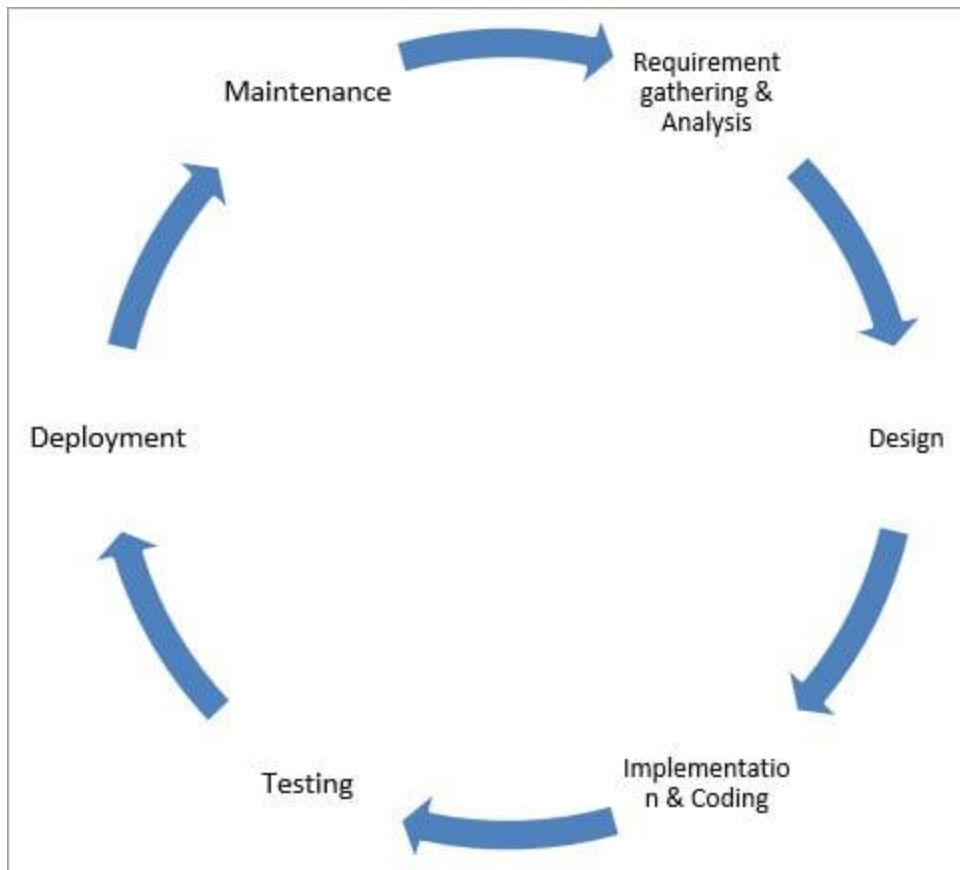


Figure 3

### Requirement & Analysis (Stage 1)

This is the first and fundamental stage of SDLC. Business analysts gather requirements from their customers, target market, and industry experts to create a Business Specification (BS) document. Other organizations and teams may refer to this document as Customer Requirement Specification (CRS) (EmergentSoftware, 2021).

## **Project planning (Stage 2)**

Senior members of the development team plan a software development project based on the BS while including input from stakeholders and subject matter experts. The project's goal could be to create a brand-new software product or enhance an existing one.

During this initial development phases, team members work together and plan out:

- a. Intentions behind the project
- b. Requirements of the project
- c. Anticipated issues
- d. Opportunities
- e. Risks

## **Design (Stage 3)**

This phase focuses on product design. Product architects and developers are involved, and they will come up with and provide a design for the product. They might offer multiple design approaches, and a Design Document Specification documents these concepts (DDS).

The DDS will be a pivotal part of the production process (Stage 4), as developers will lean on it as their primary resource to build their code. Developers must also refer back to the SRS document (from Stage 2) to ensure the product's design safeguards the team from the anticipated issues and risks noted earlier (EmergentSoftware, 2021).

## **Coding & implementation (Stage 4)**

The product is constructed in Stage 4 as production gets under way. To build the product as quickly and efficiently as possible, the programming code is created in accordance with the DDS (Stage 3). The code is created by developers using a variety of tools and programming languages. These are chosen in accordance with the requirements of the software being created.

Some of the programming tools are:

- a. Compiler
- b. Interpreters
- c. Debuggers

The programming languages are:

- a. C



- b. C++
- c. Pascal
- d. Java
- e. PHP

### **Testing (Stage 5)**

Stage 5 is where the development team tests the software for errors and deficiencies. Does the software produce the right results? Does it fulfill the requirements and objectives initially outlined in the SDLC? These are examples of key questions that could be asked during the testing phase (EmergentSoftware, 2021).

Some teams may employ automated testing methods or test the software manually. Regardless of the path they choose, the software's functionality should be checked throughout testing. The program should go through a QA procedure after testing to confirm the product's quality.

### **Deployment (Stage 6)**

After testing and quality assurance, the customer receives the software. Engineers who deploy software typically work on this stage and make it accessible to clients. They might aid individual clients in using the program on their laptops or set up the software at a business.

### **Maintenance (Stage 7)**

Because a software product's usage varies from customer to customer (each person has different needs), there may be unique issues that come up and need to be addressed. These customer issues are solved in this maintenance stage (EmergentSoftware, 2021).

To help mitigate the amount of maintenance that needs to be done, teams may choose to first release the product to a smaller population of customers. This can offer insight into how the product is performing and the development teams can make any last adjustments prior to its final release (EmergentSoftware, 2021).

### **Iterative Model**

In the Iterative model, iterative process starts with a simple implementation of a small set of the software requirements and iteratively enhances the evolving versions until the complete system is implemented and ready to be deployed (Tutorialspoint, 2021). An iterative model is a sort of model

that includes repeatedly repeating a process in order to increase the model's efficacy or accuracy. This could entail performing optimization tasks repeatedly or looking for solutions to issues. It could also entail repeating the process of training a machine learning model on a dataset.

Because they enable the model to continuously learn and advance as it processes additional data, iterative models are frequently utilized in machine learning and artificial intelligence. For instance, a machine learning model might be trained on one dataset, then its correctness might be assessed using a different dataset. The model can be adjusted and the procedure repeated until the required degree of accuracy is reached if the accuracy is not as high as intended.

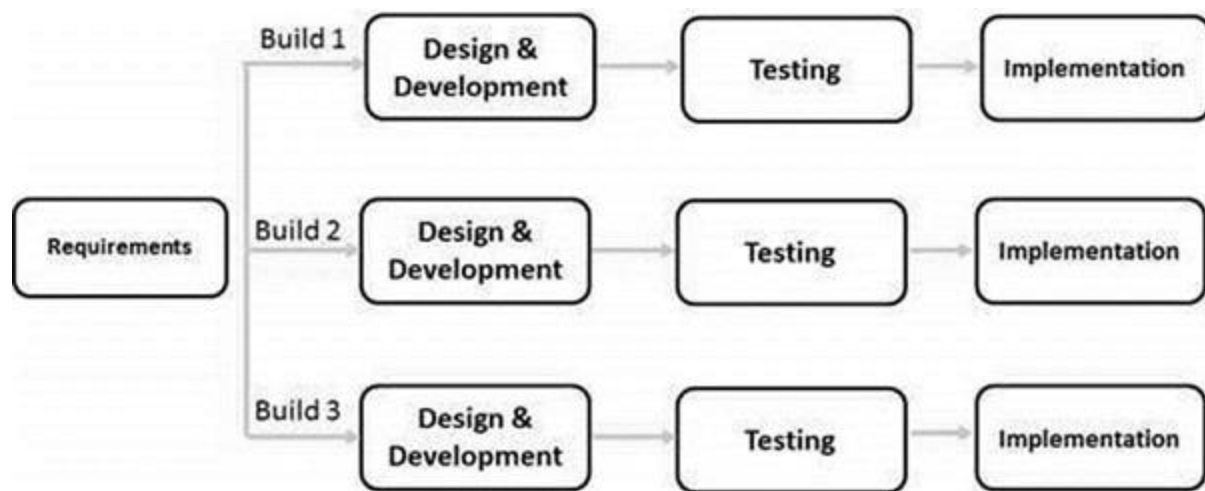


Figure 4

It is not the goal of an iterative life cycle model to begin with a complete set of criteria. Instead, only a portion of the software is specified and implemented at the beginning, and then it is inspected to find any additional requirements. After each iteration of the model, this procedure is repeated to create a new version of the software.

## Phases

### Requirement and Planning Stage

During this phase, the business requirements are collected, and an analyst determines whether or not they will be met within the allocated budget. It is used to layout the business needs in detail and the System information (hardware or software) is gathered and evaluated for feasibility (ScalerAcademy, 2022).

## **Design Stage**

In this phase, the project team gets the complete set of requirements to begin their work in a particular direction (ScalerAcademy, 2022). They employ a variety of diagrams, including state transition diagrams, class diagrams, activity diagrams, data flow diagrams, and others to gain a thorough understanding of the program architecture and assist in the development process. Based on their investigation, developers come up with a variety of potential solutions. A significant component in deciding the degree and complexity of design for the project is also its size and criticality.

## **Coding Stage**

At this stage of the project, the system's real construction starts. The analysis and design that were produced during the design stage will serve as the stage's guidelines. Every necessity, strategy, and design plan has been carried out and coded. The developer will use established coding and measurement standards to execute the selected design. They must put in place a unit test at each level of code development. For that iteration, this stage should work toward producing a fully tested, functional system. The intricacy of the work and the amount of time required for this iteration will vary depending on the project.

## **Testing Stage**

In this step, the current build iteration is tested against a set of standards and guidelines to determine whether it complies with them. This sort of testing includes, but is not limited to, performance testing, stress testing, security testing, requirements testing, usability testing, multi-site testing, disaster recovery testing, and others. A developer or tester must make sure that correcting one bug doesn't cause the system to create new bugs. Testing is a major priority since any faults would impair the software's specification, which would have an influence on the business. The tester can create new test cases or reuse ones from past releases, but testing is a top priority. To conduct some tests and get any input, we can also check in with the project's key players.

## **Evaluation Stage**

This is the last stage of the iterative model. After all the processes are complete, the system constructed up to this point is thoroughly evaluated (ScalerAcademy, 2022). The development team, stakeholders, and other teams in charge of the project's development review the system to

determine whether the results meet their expectations. Based on this, a new requirement plan is created and put into practice during the subsequent iteration cycle.

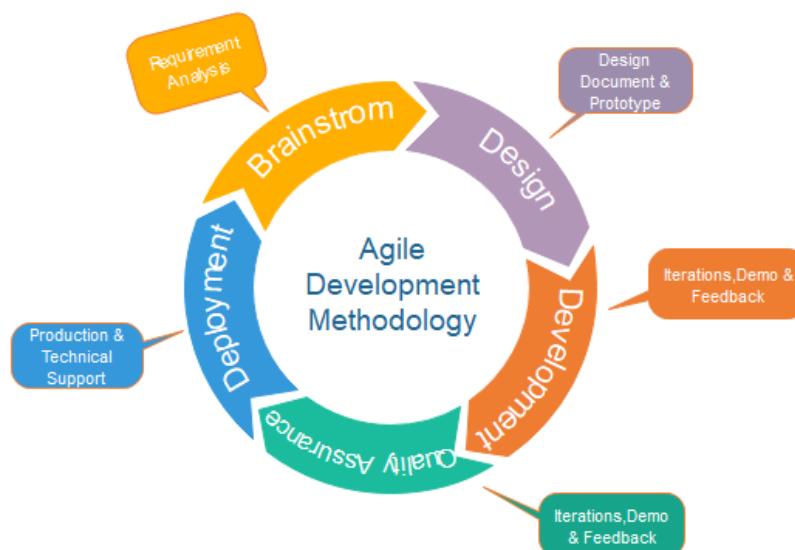
## Two types of Iterative Model

The two types of Iterative Model are given below:

### Agile Model

The meaning of Agile is swift or versatile. "Agile process model" refers to a software development approach based on iterative development. Agile methods break tasks into smaller iterations, or parts do not directly involve long term planning (JavaTpoint, 2021). The project's requirements and scope are established at the start of the development phase. Plans for the quantity, length, and scope of each iteration are spelled out in detail in advance.

In the Agile process model, each iteration is viewed as a brief time "frame" that typically lasts one to four weeks. The project risk is reduced and the overall project delivery time requirements are lowered thanks to the project's breakdown into smaller components. A team goes through the entire software development life cycle at each iteration, including planning, requirements analysis, design, coding, and testing, before showing the client a functional end result.



**Fig. Agile Model**

Figure 5

## **Phases**

### **Concept**

The first phase of the agile development lifecycle is the concept. In this phase, the stakeholder will determine the objective and scope of the software (Nehra, 2022). The product owner will produce a paper outlining the essential specifications for the product. The time needed to execute this task and any potential commercial prospects will also be determined by this paper.

### **Inception or Requirement Identification**

After outlining the concept, the next step is to create the software development team (Nehra, 2022). The product owner will check the co-developer's availability and select the best individuals to ensure the project's success. The chosen front-end and back-end developers will also receive the necessary materials and equipment from the product owner.

### **Development**

Development, often known as iteration, is the third stage of the agile development process. Since it carries the most of the labor, it is the longest of all. The development team will now begin putting all of the product requirements gathered during the concept and conception phase together. Before it is finished, it goes through numerous evaluations and changes for improvement.

### **Release**

Before releasing the product, the quality assurance team runs some tests that assure the functionality of the software. The QA team members will test the software to make sure that the code is clean and if there are any potential errors or bugs, they are addressed by the development team rapidly. (Martin, 2022)

Additionally, this stage permits ongoing updates and enhancements based on user feedback. Training is provided to customers and users on how to utilize the software. And after everything is finished, the product is put into production.

### **Maintenance**

The consumers can now access the software in its entirety. Consequently, the product enters the maintenance phase. In this stage, the development team works closely with the client to ensure that the program functions properly and is bug-free. To improve the usability and convenience of the current product for the users, additional revisions may be made throughout time.

## **Retirement**

A system may be retired because it is replaced by new software or because it has outlived its usefulness or become incompatible with the organization. The program will first be retired after notifying users. Users will be moved to the most recent system when it is replaced. Last but not least, the developers will complete the last tasks and stop providing support for the outdated program. Agile uses several iterations to improve deliverables and produce outstanding results.

## **DSDM**

The Dynamic Systems Development technique (DSDM) is an associate degree agile code development approach that provides a framework for building and maintaining systems (Greeksforgreeks, 2019). A modified version of the sociologist's principle, which states that an application is typically generated in 80% of the time it would take to supply a 100% application, is the foundation of the DSDM philosophy.

The 80% rule, which stipulates that each increment only takes as much effort as is necessary to enable advance to the upcoming increment, is followed in every iteration of the DSDM iterative coding process. The last detail is typically completed later after several business requirements have been established or revisions have been requested and accommodated.

## **Phases**

Some to the phases are given below:

### **Feasibility Study**

It establishes the essential business necessities and constraints related to the applying to be designed then assesses whether or not the application could be a viable candidate for the DSDM method (Greeksforgreeks, 2019).

### **Business Study**

It establishes the use and knowledge necessities that may permit the applying to supply business value; additionally, it is the essential application design and identifies the maintainability necessities for the applying (Greeksforgreeks, 2019).

### **Functional Model Iteration**

It produces a number of iterative prototypes that demonstrate to the client their utility. (Note: Every DSDM prototype is intended to be developed into the delivered application.) This never-ending

loop seeks to identify new demands by soliciting feedback from customers as it employs the paradigm.

### **Design and Build Iteration**

It goes back and looks at the prototypes produced during iterative useable model development to make sure that each was built in a way that might alter it and give end users operational business value. It is conceivable for significant model iterations and style and build iterations to take place at the same time.

### **Implementation**

It places the newest code increment (an “operationalized” prototype) into the operational surroundings. It ought to be noted that:

- a. The increment might not 100% complete or,
- b. Changes are also requested because the increment is placed into place. In either case, DSDM development work continues by returning to the useful model iteration activity (Greeksforgreeks, 2019).

### **Sequential Model**

Something that is arranged in a specific order or sequence is referred to as sequential. This can refer to a number of things, including a list of things, an orderly sequence of things happening, or a set of instructions. A grocery list, for instance, might be a sequential list of products, whereas a recipe might be a sequential list of instructions. The phrase is frequently used in computing to describe data structures like linked lists and arrays that store and arrange data in a predetermined sequence.

### **Waterfall Model**

The waterfall model is a method of developing software in which the various stages, including requirement gathering, design, implementation, testing, and deployment, are finished one after the other in order. This implies that each step must be finished before the subsequent one may start. Agile development methodologies, on the other hand, support more flexibility and iteration.

## **Phases**

### **Analysis**

Every software project starts with an analysis phase that involves a requirements specification and a feasibility investigation. The software project's costs, revenue, and viability are evaluated in the feasibility study. A requirement specification a general explanation of the requirements a project plan, the project calculation, and, if appropriate, an offer to the client are all included in the feasibility study.

### **Design**

The design phase serves to develop a concrete solution concept based on the previously determined requirements, tasks, and strategies (DigitalGuide, 2019). In this phases, software engineers create the software architecture and a thorough plan for building the software, focusing on particular elements like libraries, frameworks, and interfaces. A draft document with a software construction plan and test plans for specific components is the end product of the design process.

### **Implementation**

The software architecture designed in the design phase is implemented in the implementation phase, which includes software programming, troubleshooting, and module testing (DigitalGuide, 2019). The software design is put into practice in the desired programming language during the implementation phase. Separate parts are created, tested within the context of module testing, and gradually incorporated into the final product. A software product that is the end result of the implementation phase is tested for the first time as a finished product in the following phase (alpha test).

### **Testing**

The software's integration into the desired target environment is part of the testing phase. Typically, beta versions of software items are first distributed to a limited number of end users (beta tests). You can check whether the program complies with the requirements by using the acceptance tests created during the analysis phase. A piece of software that has successfully passed beta testing is prepared for public consumption.



## **Maintenance**

After successful completion of the test phase, the software is released for productive use. The final phase of the waterfall model includes delivery, maintenance, and improvement of the software (DigitalGuide, 2019).

## **Merit of applying water fall model**

As I am system analysis of the Birtamod Technical College, the waterfall model's straightforward and structured approach to software development is one of its key benefits, since it can help guarantee that all requirements are well captured and understood before moving on to the next step. When needs are not clearly defined at the beginning of a project, scope creep and other problems may occur.

Since each stage of the project can be planned and finished in a predictable and linear manner, the waterfall model may also help huge businesses allocate resources more effectively. This can ensure that the appropriate resources are available at the appropriate moment, which will increase the project's overall efficiency and effectiveness.

Particularly in large and complex projects where adjustments are frequently required, the inability to make changes after a phase has been finished can be a significant disadvantage. Furthermore, the waterfall approach may not be well adapted to the fast-paced and dynamic nature of major enterprises due to its inherent lack of flexibility and adaptation. These factors suggest that in some circumstances it could be important to think about utilizing a different software development paradigm.

**Why waterfall model is best for large company?** As the System Analysis of Birtamod Technical College, Large projects with clearly specified needs that are unlikely to change considerably over the life of the project are a good fit for the waterfall approach. This is due to the waterfall model's linear, sequential approach, in which the success of each phase of the project is dependent upon the success of the phase that came before it. This can aid in ensuring that the project stays on course and that the finished result satisfies all necessary criteria.

Think about a sizable initiative to create a new enterprise resource planning (ERP) system for a large manufacturing organization. The project team would start by meticulously compiling and examining the specifications for the new system, including the particular demands and

specifications of each of the company's numerous divisions and locations. They would then be able to design and specify the new system in great detail.

The development team would start putting the system's various components, like the database, user interface, and integration with the business's current systems and procedures, into place once the design and specification were finished. Each component would be finished and tested before going on to the next one in a sequential way.

The new ERP system would be implemented in the organization's production environment and made accessible to users once all of the components were finished and extensively tested. This would entail putting up the system on the organization's servers, configuring the required network and security settings, and giving users any necessary instruction and support.

Overall, the waterfall model offers a clear and systematic way for managing the project and guaranteeing that the finished result satisfies the established objectives, making it well-suited for this kind of large, complicated project. Additionally, it enables effective coordination and communication between the various project teams and stakeholders, which can help to lessen misinterpretations and the chance of delays or mistakes.

## **RAD Model**

The Rapid Application Development (RAD) model is a speed- and efficiency-focused approach to software development. RAD Model or Rapid Application Development model is a software development process based on prototyping without any specific planning (Martin, 2022). In the RAD paradigm, development teams create functional prototypes of a new application in a short amount of time through iterative cycles. This enables quick iteration and feedback, which can assist guarantee that the finished product satisfies the user's needs. The RAD paradigm is frequently employed when time to market is a crucial consideration or when the requirements for the new application are ill-defined or subject to change.

## **Phases**

### **Requirement gathering and analysis**

In this phase, the client's needs and specifications for the new application are worked out between the development team and the client. This entails determining the application's main functions and features, as well as any restrictions or limitations.

## **Prototyping**

In this phase, the development team develops one or more low-fidelity (such as paper-based) or high-fidelity (such as digital) prototypes of the new application. With the help of the customer and the intended audience, these prototypes are utilized to swiftly test and confirm the proposed design and functionality.

## **Development**

Based on the suggestions and information obtained during the previous phase, the development team builds the application's final version during this stage. This could entail developing databases, designing user interfaces, writing code, and integrating any required third-party parts or services.

## **Integration and testing**

The many parts of the application are put together in this step and evaluated to make sure they function as intended and adhere to the criteria laid down in phase one. Functional testing, performance testing, security testing, and user acceptance testing all fall under this category.

## **Deployment**

The finished application is made available to the intended users and deployed to the production environment in this last stage. This could entail setting up the program on the client's servers, creating the required infrastructure and security settings, and giving the end users any necessary instruction or support.

## **Spiral Model**

The spiral model is a systems development lifecycle (SDLC) method used for risk management that combines the iterative development process model with elements of the Waterfall model. The spiral model is used by software engineers and is favored for large, expensive and complicated projects (Contributor, 2019).

Software development is done using the spiral model, which combines aspects of iterative and incremental development with those of the waterfall paradigm. It is a risk-driven process model, which means that it is created to manage risk by dividing a project into smaller, more manageable portions and taking care of possible problems as they come up.

## **Phases**

### **Identification**

Gathering the business needs in the baseline spiral is the first step in this phase. Identification of system requirements, subsystem requirements, and unit requirements are all completed in this phase, which is followed by spirals as the product matures.

Understanding the system requirements during this phase also entails constant contact between the customer and the system analyst. The product is introduced into the identified market at the bottom of the spiral.

### **Design**

The Design phase includes architectural design, logical module design, physical product design, and final design in the succeeding spirals. It begins with conceptual design in the baseline spiral.

### **Construct or Build**

The actual software product is produced at each spiral throughout the construct phase. A POC (Proof of Concept) is created in this phase of the baseline spiral, when the product is still only an idea and the design is being worked on, to gather consumer feedback.

Then, in later spirals with more precise requirements and design specifics, a workable version of the software known as build is created. The consumer is contacted for comments on these builds.

### **Evaluation and Risk Analysis**

Identification, estimation, and monitoring of management and technical risks, such as schedule slippage and cost overrun, are all part of the risk analysis process. At the conclusion of the first iteration, the customer analyzes the software and offers comments after testing the build.

The Spiral Model is represented in the following illustration, which lists the tasks involved in each phase.

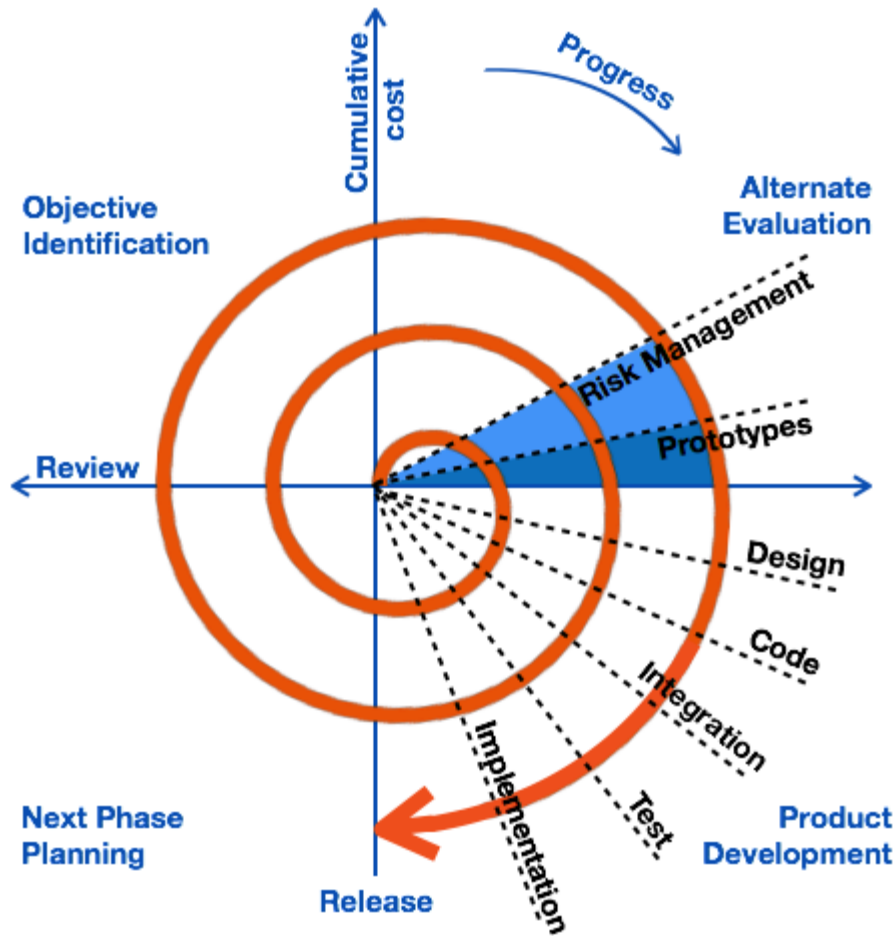


Figure 6

The software development process enters the following iteration based on the customer evaluation and then adopts a linear strategy to execute the consumer feedback recommendations. Throughout the software's lifespan, revisions along the spiral continue to be made.

### About Risk

Risk is a potential problem that might come up in the future but is not certain to do so. When one of the software's components is gone, there is a risk. This process often takes place when there is no guarantee of time, money, or control over any project. As a result, the loss could be anything, including a waste of time or money, added stress on a worker, a bad software product, or a variety of other things. Small things add up to basic risk to terrible risk in the project, which can enhance the employees' perception of their own physical, psychological, and financial well-being. In order

to minimize them for both present and future needs, I will describe potential hazards in this part along with their mitigating traits. The models may alter the risks.

### **Roles of Spiral Model to Mitigate a Risk**

The spiral model divides a project into smaller, more manageable portions and addresses possible concerns as they develop in order to manage risk. Thus, the spiral model contributes to risk mitigation in a number of ways:

#### **Identifying and Evaluating Risks**

Identifying and assessing possible project risks is the initial stage in any spiral. The team will be better able to comprehend the risks they are facing and take action to solve them by recognizing potential concerns and examining their potential effects.

#### **Planning and Design**

The spiral model's planning and design phase is dedicated to creating a remedy for the identified hazards. The team may lessen the possibility of issues emerging and lessen the impact of any that do by carefully planning and designing the project.

#### **Building and Testing**

The spiral model's construction and testing phases give the team the chance to confirm that the solution they came up with during the planning and design phase is practical and satisfies the project's needs. By putting the solution to the test, the team can spot any problems early on and fix them.

#### **Evaluating and Revising**

Each spiral's last phase entails reviewing the outcomes of the one before and making any required changes to the plan. The team can stay on track and reduce any new risks by constantly monitoring the project's progress and making changes as necessary.

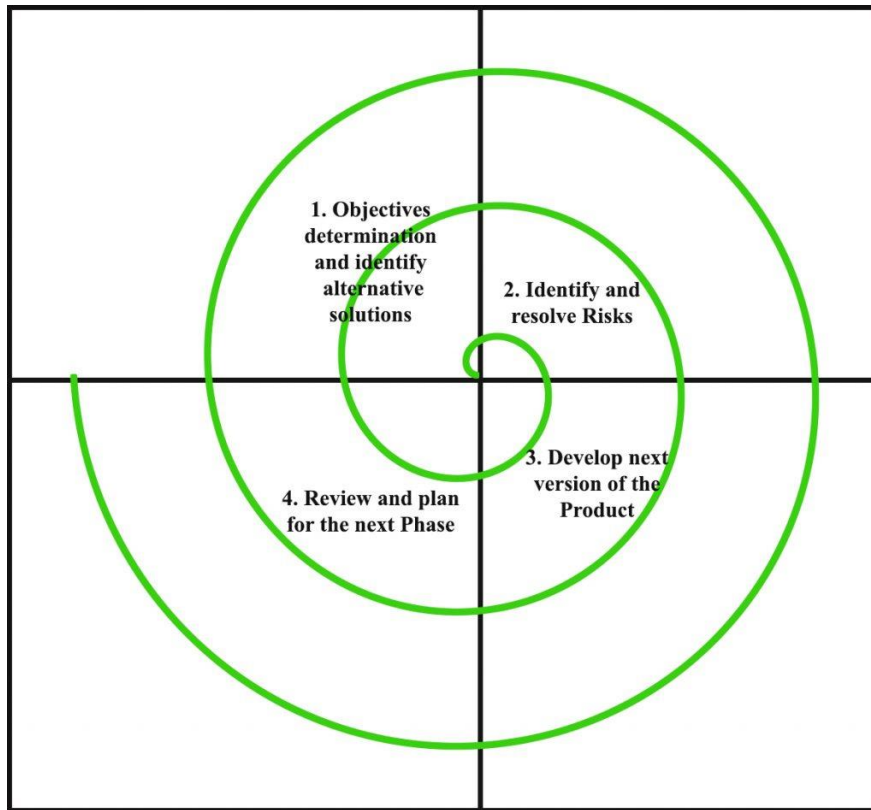


Figure 7

## Conclusion

From this section, we can learn about the many models used in the software lifecycle development. As a result, we can verify that the software development lifecycle complies with all the specifications needed to produce the particular application being discussed. It should also make it simpler to progressively incorporate new features into the software. It is possible to assert that software models meet client expectations in the interim. In the case of the use of particular approaches, it is applicable to software development. It also aids in lowering the risk involved as well as the likelihood of a potential issue materializing through the use of various techniques and proper processes. Every software development process must finally pass through software lifecycle development procedures in order to produce and ensure a superior software product.

## Introduction

In this part, I have discussed the feasibility study and its requirements. I will also be explaining the purpose of the feasibility study in detail. As soon as I finished the feasibility study, I discussed

the technical solutions to the software and how those technical solutions can be compared. After finishing the technical solution, I discussed the components of the feasibility report.

### **Feasibility Study**

An examination of a proposed project or plan of action to determine its likelihood of success is known as a feasibility study. Feasibility studies are frequently used to assess a new project or venture's potential and decide whether it is worthwhile to pursue. A feasibility study is a detailed analysis that considers all of the critical aspects of a proposed project in order to determine the likelihood of it succeeding (IncestopediaTeam, 2022).



Figure 8

### **Requirement gathering techniques**

#### **Interview**

Interviews are conversations with specific stakeholders, such as users, developers, and project managers, to get their feedback and project requirements. These might be unstructured interviews where the conversation develops naturally or organized interviews with predetermined questions.

#### **Focus group**

Focus groups include gathering a small group of project stakeholders to talk about their needs and suggestions. A variety of viewpoints and ideas can be obtained in this method.



## **Survey**

Surveys entail distributing a questionnaire to participants in order to get their feedback and needs. Surveys can be carried out offline or online and are a quick and effective approach to collect a lot of data.

## **Prototype**

Early iterations of the product, known as prototypes, can be tested and utilized to solicit input from key stakeholders. By observing how consumers engage with and respond to the product, this can be a helpful technique to gather needs.

## **Document analysis**

Document analysis involves reviewing existing documents, such as requirements specifications, user manuals, and also design the documents, to gather information about the project. This can be a useful way to gather information that may have been overlooked in other requirement gathering techniques.

## **Purpose of Feasibility Study**

Determine the viability of a project or idea by conducting a feasibility study. This entails evaluating the project's operational, financial, and technical components to see if it is feasible and attainable. The outcomes of a feasibility study can give decision-makers important information for project planning and execution as well as assist in deciding whether to move forward with the project.

Here are the some of the purpose of Feasibility Study

- Recognizing potential issues and dangers related to the project and creating solutions for them
- Calculating the project's costs and benefits, and assessing if the anticipated returns are sufficient to justify the expenditure
- Determining whether the project is technically feasible, considering the accessibility of the required materials, equipment, and manpower
- Assessing the project's market potential and its likelihood of success in the target market
- Supplying a thorough project execution plan with schedules, budgets, and important milestones

## **Some of the reason why Feasibility Study is need**

Here is the reason why Feasibility Study is need

### **Avoiding costly mistakes**

A feasibility study can assist prevent expensive errors and guarantee that the project is carried out successfully by identifying potential issues and hazards related to the project.

### **Maximizing the potential returns**

A feasibility study can assist decision-makers in determining if the anticipated returns are sufficient to justify the investment by assessing the project's expenses and benefits.

### **Ensuring the availability of resources**

A feasibility study can assist in ensuring that the required resources, such as materials, equipment, and staff, are available by assessing the technical feasibility of the project.

### **Determining the market demand**

A feasibility study can assist in determining if a project is likely to succeed in the target market by analyzing the market demand for it.

### **Providing a detail plan**

A feasibility study helps ensure that the project is carried out successfully and efficiently by offering a thorough plan for its execution.

## **What is Feasibility Report**

A feasibility report is a report that evaluates a set of proposed project paths or solutions to determine if they are viable (Team, 2021). The person who prepares a feasibility report evaluates the feasibility of different solutions and then chooses their recommendation for the best solution (Team, 2021).

## **Component of Feasibility Study**

Here are some components:

### **Project Scope**

The scope of the project to determine the business challenge and/or opportunities. It's an appropriate ancient saying: “The well-declared issue is half solved” (Entrepreneurindia, 2021). The scope should be clear and current; rambling tales serve no purpose and may very likely lead

the project members astray. It is necessary to describe either directly or indirectly the parts of the businesses involved, including project participants and end-users, who are impacted by the project. If the project sponsor must cover the expense, this should be made clear.

### **Analyzing the Present Situation**

The present analysis is used to identify and comprehend the current technique, such as a system, a product, and so on (Entrepreneurindia, 2021). In this study, it is common to discover that the existing system or product only truly has a few misunderstandings about it, or that only a few minor adjustments are needed rather than a significant rewrite.

Additionally, the current methodology revealed advantages and disadvantages (pros and cons). Aspects of the current system or product can also be used, which will save time and money when it comes time to replace it. Without such a study, this could never be discovered.

### **The necessities**

Depending on the project's goal, requirements and the methods used to define requirements. For instance, how needs for a product are articulated is very different from how requirements for a building, bridge, or information system are described. Each exhibit wholly unique characteristics and is characterized as such. Additionally, there are substantial differences in how requirements are expressed for systems and for software.

### **Review**

Examine the assembly of all the above aspects into a feasibility study and a formal review with all concerned parties (Entrepreneurindia, 2021). The review's purpose is to decide whether to move forward with the project and to attest to the feasibility studies accuracy and depth. Before a final choice is made, it may be accepted, rejected, or necessary. This review has two objectives.

### **Technical solution**

An approach to leveraging technology to address a need or solve a problem is known as a technological solution. It might just be a piece of software automating a time-consuming process, or it might be a complex network of interconnected technology supporting a massive business. In order to create and construct dependable, efficient, and effective systems, technical solutions frequently make use of scientific and engineering principles.

## Technical solution to the software

Depending on the particular requirements and objectives of the project, a variety of technical solutions might be used for software. Some common technical solution for software includes:

- **Performance and Efficiency:** Using algorithms and data structures created to optimize the performance and efficiency of the software would be a technological solution to a software problem based on performance and efficiency. This might entail employing data structures that are optimal for the particular issue at hand as well as methods with low time and space complexity. The software's functionality and efficiency can also be increased by putting effective memory management and caching mechanisms into place. The best technical solution will ultimately depend on the particulars of the issue and the limitations of the software system.
- **Legacy system upgrade:** Technically speaking, a legacy system upgrade problem would most likely be solved by moving the legacy system to a more advanced and scalable platform. This could entail adding new features and functionality to enhance the system's overall functionality and user experience, as well as reworking the existing code to make it more extendable and manageable. In order to utilize their capabilities and maintain the system's long-term survival, it can also be required to interface the legacy system with other contemporary systems and technologies, such as cloud services or artificial intelligence. The technological solution will ultimately be determined by the specifics of the old system and the upgrade's objectives.
- **Automation:** If a software issue involved automation, a technological solution would entail applying algorithms and technologies that allow the software to carry out tasks automatically, without requiring human participation. To enable software to learn from data and make predictions or judgments without explicit programming, this may involve implementing machine learning algorithms. It might also entail linking the software with other devices and systems so that it can automatically start operations or activities in response to specific occurrences or conditions. The effectiveness and dependability of the software development process can also be increased by introducing automation solutions like continuous integration and deployment.

- **Elimination of human errors:** Implementing techniques and technologies that lessen the possibility of errors happening in the first place as well as systems for identifying and fixing errors when they do occur are all parts of a technical solution to a software problem involving the eradication of human errors. This may entail putting in place reliable checks for the accuracy and completeness of the input data as well as error-handling procedures to gracefully handle and recover from unexpected input or exceptions. Prior to the software being released to production, putting automated testing and quality assurance methods in place can aid in finding and resolving faults. The technological solution will ultimately depend on the particulars of the issue and the limitations of the software system.
- **Budget/Economic:** Finding cost-effective alternatives that meet the required functionality within the allocated budget is a technological solution to a software challenge based on financial and commercial limitations. In order to lower the initial costs of hardware and infrastructure, this may entail using open-source or inexpensive software tools and technologies as well as utilizing cloud computing services. The use of agile software development approaches, which emphasize delivering tiny, incremental software updates on a frequent basis rather than attempting to build the full system at once, may also be involved. A cost-effective approach to software development and maintenance can also be achieved by putting into practice cost reduction measures like minimizing waste and maximizing resource usage.

### **Benefits of using Feasibility Study**

A feasibility study examines a proposed project or proposal to ascertain its viability from a technical, financial, and economic standpoint. A feasibility study can have a number of advantages, such as:

#### **Identifying Potential Problems and Risks**

A feasibility study can assist in locating potential issues or dangers that can appear during the planning or execution of a project. The team can act to alleviate or eliminate these difficulties by detecting them early on, before they develop into significant issues.

#### **Determining the Feasibility of the Project**

A feasibility study can assist in determining the technical, financial, and economic viability of a project. The team can determine if a project is worthwhile to undertake or if it would be preferable

to pursue an alternate course of action by analyzing the technical requirements, expenses, and potential benefits of the project.

### **Identifying the Resources needed**

A feasibility study can assist in determining the necessary financing, manpower, tools, and supplies to finish the project. This can make it easier for the team to efficiently plan and budget for the project.

### **Establishing a Baseline**

An initial baseline for the project, including a schedule, budget, and scope, can be established by a feasibility study. This can serve as a point of reference for monitoring development and spotting any deviations from the original plan.

### **Building Consensus and Support**

By offering a thorough and impartial examination of the advantages and disadvantages of the project, a feasibility study can assist in fostering consensus and support. This is particularly helpful when there are numerous stakeholders or when the project needs finance from outside sources.

### **Feasibility Criteria**

The standards or guidelines used to determine whether a project or proposal is feasible are called feasibility criteria. Feasibility criteria might vary based on the project's objectives and particular circumstances, but they often involve questions about the project's economic, financial, and technological viability.

### **Technical Feasibility**

The ability of the project to be finished with the resources and technology now in use is referred to as technical feasibility. This could take into account things like if the necessary tools and software are available, whether the team has the requisite experience to finish the project, and whether the project can technically be completed within the allocated timeframe and budget.

### **Financial Feasibility**

The capacity of the project to be completed within the allocated budget and to yield sufficient financial returns to cover the investment is referred to as financial feasibility. This may consider

factors including the project's estimated expenses, anticipated earnings or benefits, and return on investment.

### **Economic Feasibility**

The ability of the project to be completed in a way that is economically viable is referred to as economic feasibility. This could consider things like the product or service's market demand, prospective economic effects, and potential competition.

### **Legal Feasibility**

Legal viability is the project's capacity to adhere to all pertinent rules and regulations. Considerations including zoning laws, environmental restrictions, and intellectual property laws may be included in this.

### **Operational Feasibility**

The ability of the project to be incorporated into the organization's current operations and processes is referred to as operational feasibility. Considerations including the effect on the organization's resources, the possibility of disrupting ongoing activities, and compatibility with the organization's mission and goals might be included in this.

Overall, a project's or idea's potential viability is assessed using a set of criteria, including technical, financial, economic, legal, and operational aspects.

### **Conclusion**

Every key aspect of the feasibility study and how the various technology options may be compared overall have been covered. Confirming that the proposal is technically, legally, economically, and socially feasible is the study's main objective. We may therefore draw the conclusion that conducting a feasibility study is the best way to approach planning. Every significant step has been successfully performed compared to a technical solution in a really admirable manner. This section further clarifies the goal of a feasibility study, which is to assess and show a company proposal's commercial and technological viability. This section has led us to the reasonable conclusion that the feasibility study is helpful in a number of situations.

## **Introduction**

I work as a system analyst for the New Birtamod Technical College in Nepal, which offers instruction to both local and international students. The organization wants a new assignment management system that can track both college and other assignments. I am in charge of managing a project as well as the analysis and design of the program. I must first identify the stakeholders because operating it is a really challenging task. Along with discussing the software's quality, I'll mention all the many stakeholders and their individual requirements. However, there are a few challenging impediments in the road. I'm hoping it will eliminate as it travels. Various tool and technique kinds, including measures and supporting documentation, will be applied through the usage of a program. The superior software version will be evaluated during the development process, and problems will be shortlisted and managed in accordance with the suggested design and planning.

## **Background Information of new software**

What will be the background if I create new Software's? In my view the background will be like this if I am developing new software, the background data may contain the software's objectives and aims, its intended users, its capabilities and features, prospective advantages and benefits, and any pertinent data regarding the market or industry in which the software will be applied. You could also wish to provide details on the software development process, such as the team members engaged, the technologies and tools employed, and the development schedule. You could also wish to provide details on any earlier iterations of the program, as well as any lessons discovered or user input that can guide the creation of the new software.

## **Problem Statement**

There will be many potential problems that are:

## **Identifying and defining the problem that the software will solve**

Before creating new software, it's crucial to recognize and comprehend the issue or requirement that it will try to solve. To make sure that the program will be useful and pertinent, this may entail performing research and obtaining input from potential users.



## Developing a feasible and effective solution

The next stage after identifying the issue is to create a solution that is both technically doable and successful in solving the issue. To make sure the software will function as intended, this may take careful planning and design, testing, and iterating on various strategies.

## Managing resources and staying within budget

It can take a lot of time and effort to create new software, therefore it's critical to efficiently manage resources to keep the project on schedule and within budget. This may entail establishing precise objectives and deadlines, allocating resources wisely, and keeping an eye on development to spot and resolve any potential obstacles.

## Ensuring user adoption and satisfaction

Once the software has been created, it is crucial to make sure that the target audience uses it. This may entail marketing and outreach initiatives, as well as offering assistance and resources to help users get started with the program. To make sure that the program is meeting the needs and expectations of its users, it is also crucial to keep an eye on user input and address any problems or worries.

## Stakeholder

A stakeholder is a party that has an interest in a company and can either affect or be affected by the business. The primary stakeholders in a typical corporation are its investors, employees, customers, and suppliers (Fernando, 2022). **Why it is important**-Stakeholders are crucial for several reasons. They are significant to internal stakeholders since the operations of the company depend on their capacity to cooperate in achieving the company's objectives. On the other hand, external stakeholders may have an indirect impact on the company. Customers can alter their



Figure 9

purchasing patterns, suppliers can alter how they manufacture and distribute their products, and governments can alter the laws and regulations. The secret to a company's long-term success is ultimately managing relationships with internal and external stakeholders.

### **Different types of Stakeholders**

Shareholders, clients, vendors, and staff are a few examples of crucial stakeholders for a company. Internal to the company are some of these stakeholders, including the shareholders and the staff. Others, such as the company's clients and suppliers, are not a part of the company but are nonetheless impacted by its decisions. These days, a wider range of external stakeholders, such as the governments of the nations in which the corporation works or even the general public, are increasingly frequently mentioned.

### **Identify Stakeholders**

Stakeholders are individuals or organizations with an interest or concern in something, particularly a business. The person or group who assigned the work, the person or group who will be impacted by the assignment's findings, or the person or group who will be in charge of putting those results into practice are all examples of stakeholders in the context of an assignment. Some examples of stakeholders in an assignment are the instructor who gave the task, the group of classmates who will work on it, the assignment's subject or topic, and any pertinent organizations or people who might be impacted by the task's results.

### **Requirements**

Here are some requirements:

- Login authentication for each Student, Tutor & Assignment Department
- Subject-wise Assignment with multiple Attachment Available
- Assignment Approve/Reject by Tutor
- Grades to be provided to students via notification
- Record of assignment, submitted by each student

## **Requirement Identification**

The process of locating and identifying a project's or system's requirements is known as requirement identification. In this process, information regarding the project's requirements and objectives is gathered, and the features, functionality, and capabilities required to satisfy those requirements are identified.

In the software development process, requirement identification is a crucial step since it ensures that the project is precisely specified and that all stakeholders are aware of the expectations. Additionally, it assists in ensuring that the team has the necessary tools and resources to finish the job.

The team may employ a range of methods, including interviews, workshops, surveys, and examination of current systems, to determine the needs. A requirements specification, which serves as a guide for the remainder of the project, is often used to record the requirements that are discovered during this process.

According to the scenario above, the following stakeholders are regarded as having an interest in the project's successful completion:

### **Owner**

Owner, who is in control of every department, is the key and most significant member of the college. Anything they desire to do, he or she is capable of doing. He has a responsibility to maintain and upgrade the college's equipment. He also has the authority to implement a specific law in accordance with the assurance of the government. With the assistance of the college's staff, employees, and other subordinates, he must essentially oversee every aspect of the institution. He must spend all of his time managing a college.

### **Staff**

Staff in a college or university refers to the workers who do a variety of tasks on campus to support the operations and duties of the college. In addition to administrative staff like office managers and receptionists, there may also be technical and support staff like IT specialists, custodians, and maintenance personnel.

A college or university may also include academic staff, which consists of professors and other educators who are in charge of both teaching and research, in addition to these other categories of employees. These employees may work full- or part-time shifts, and they may have tenure or not.

In a college or university, staff members support the daily activities of the institution and contribute to the development of a stimulating and productive learning environment for students. A staff member's duties may include handling administrative and support chores, instructing classes, carrying out research, and offering support and help to students and other college community members.

### **Teacher**

A teacher is a member of the faculty who is in charge of instructing students and teaching courses in a college or university. In a college or university, instructors may also take part in academic counseling, research, and other initiatives relevant to their areas of specialization.

A college or university may employ full-time, part-time, tenured, or non-tenured teachers. Non-tenured faculty members may have contract or temporary roles, while tenured faculty members normally have permanent posts in the college or university.

Depending on their degree of education and experience, the subject they teach, and the college or university they teach at, teachers in those institutions may hold a number of titles. For instance, a master's-educated teacher might be referred to as an instructor, whereas a doctoral-educated teacher might be referred to as a professor.

Teachers also serve as the major decision-makers for the college. One of the many professors at a college who neither has tenure nor is a permanent member of the faculty is lectures. As the system's primary content administrator, they need a distinct username and password to access the device. The system must allow professors to incorporate files, connections, and multimedia while also assessing assignments, establishing a new platform for students to interact with lecturers, and doing other things.

### **Requirement Specification**

The requirement specification that are required according to scenario

## **Scope**

Scope refers to the combined objectives and requirements needed to complete a project. The term is often used in project management as well as in consulting (Grant, 2021). The boundaries and specifications of a project are referred to as its scope in project management terminology. The work that will be done and the deliverables that will be generated are both clearly and precisely defined.

The objectives, deliverables, and important stakeholders of a project are often outlined in a project charter or scope statement, which defines and documents the project's scope. A list of the tasks that will be included in the project as well as any tasks that will be eliminated is also supplied.

The resources and finances that will be needed to execute a project, as well as its timeline, will be determined in large part by its scope. Additionally, it serves to outline the project's parameters and keep the team's attention on the project's goals and deliverables.

## **Scope of the Project**

The parameters and specifications of a project are referred to as its scope. The work that will be done and the deliverables that will be generated are both clearly and precisely defined.

The objectives, deliverables, and important stakeholders of a project are often outlined in a project charter or scope statement, which defines and documents the project's scope. A list of the tasks that will be included in the project as well as any tasks that will be eliminated is also supplied.

The resources and finances that will be needed to execute a project, as well as its timeline, will be determined in large part by its scope. Additionally, it serves to outline the project's parameters and keep the team's attention on the project's goals and deliverables.

The progress of this project's development appears to be improving. Some of the problems that can arise during the development process will be lessened. This project aims to enhance the management assignment system at the college. The ability of the staff members to swiftly and efficiently summarize intricate material in an electronic format, ensuring that it cannot be lost outside of the server, is another advantage of this project. The staff employees gain from the time and money saved by the college.

Here is the table, it includes input, output, process which involve in this software.

S. N	Description	Input	Output	Process	Process Descriptor
1	Login	In input, there are Username/Email and Password	Login Authentication	Check Authentication	If the user or administrator provides the proper email address and password, they can login to the system.
2	Assignment Submit	PDF upload file	All of the assignment is related information	Checking the data, before deadline or after deadline	Employee, administrator, or student
3	Assignment Tracking System	It doesn't require any requirements if staff, user or admin once they have logged	Assignment is related to Information	Assignments data's and other information	Employee, administrator, or student
4	Delete or Update	It doesn't require once staff, user admin had logged or login	According to the current scenario, data can be deleted or update easily	Review or update of the essential data	Employee, administrator, or student
5	Exit	It doesn't require anything. Without anything it can exit easily	It quit the software easily	-	Employee, administrator, or student

## **Consideration of alternate solution**

This section should outline the prospective outcomes so that everyone involved in the project can better understand them. The software must have multiple levels of security one for customer protection and another for corporation protection in order to be finalized. Because we are talking about a different approach, the program needs to be connected to the main server. It will ensure the safety of the local and central databases linked to the software's front end. If the database malfunctions, the data can be accessed from the main server. Since the databases are perfectly connected, it is possible to swiftly and effectively move data from one port to another in order to update it for each user using different database methods.

### **Security Consideration**

The Security Considerations Assessment (SCA) process ensures security-related vulnerabilities are considered across a range of activities and processes within an organization. This includes physical, personnel, cyber and cross-cutting security measures (CPNI, 2022).

Security worries are the things that throw off the suggested software system's balance. This system software's data may have been hacked. We are aware that the data belonging to the company contains sensitive and private information. The amount of data it can handle at once might be limited. There could be a security concern here. But with good planning and cooperation, this might be avoided. To ensure that all data and files are protected from dangers, the antivirus provider should be linked to the software. Additionally, the company should regularly backup the important files to save space and improve the software's performance.

Security considerations are the steps and safeguards put in place to guard against potential risks to the security of a system or network. These safeguards may be organizational safeguards like access control policies and procedures as well as technology safeguards like firewalls and encryption.

Quality assurance (QA) is any systematic process of determining whether a product or service meets specified requirements (Gillis, 2020). Quality Assurance is referred to as QA. It alludes to the procedure of confirming that a good, service, or system complies with rules and specifications. QA in software development refers to testing the software to make sure it is of a high standard, trustworthy, and functions as planned.

Before the software is made available to the public, QA seeks to locate and fix any problems or flaws. To make sure that the software complies with the necessary standards and specifications, QA methods may include manual testing, automated testing, and other techniques.

QA plays a crucial role in ensuring that the software is of the highest caliber and functions as intended during the software development process. The team can prevent expensive and time-consuming issues later on by detecting and resolving issues as soon as they arise during the development process. Quality assurance is crucial during SDLC phases. In essence, it aids in the prevention of incoming flaws as well as issues that occur during software development. It must ensure that the software meets all quality standards. There will be concerns raised about the company if the software fails to achieve the need. Testing is necessary before the system is finished so that any potential dangers and flaws can be re-coded. This software is considered to be a well-deserving software system for the Birtamod Technical college because it has been tested in a variety of ways.

### **Identification of Different constraints**

A constraint is a restriction on the degree of freedom you have in providing a solution. Constraints are effectively global requirements, such as limited development resources or a decision by senior management that restricts the way you develop a system (Ambler, 2018). Every business face limitation when a project first begins. Each restriction has a unique limit that establishes a barrier for the project process, and the software team is obliged to emphasize each restriction's restriction. The following list of restrictions is necessary for the system software to advance:

#### **Cost**

The cost of something is the sum of the money, time, and resources needed to manufacture, obtain, or complete it. Cost plays a significant role in resource allocation and decision-making in the context of business and economics since it influences the viability and profitability of various solutions. Similar to that, Birtamod Technical College needs resources for this project in order to sustain its management system and accounting software. The project should receive a particular amount so that it can compete with it and stay within its budget, which is why there will be cost restrictions. like the price of the personnel required for software development. The software developer's office is separate and has a computer. These are the expenses involved in developing the program. The structure needed to construct the software is shown in the tables below.



S. N	Equipment and things	Cost
1	Hardware components like Processor, RAM.	Rs. \$2,000
2	Software	Rs. \$500
3	System Investigation	Rs. \$500
4	Debuggers	Rs. \$550
5	Developers	Rs. \$500
6	Total of everything's	Rs. \$4,050

### Legacy

A legacy system is any outdated computing system, hardware or software that is still in use. Legacy systems include computer hardware, software applications, file formats and programming languages. However, not all legacy systems are obsolete technologies. Most legacy systems work even if they are outdated, and enterprises will often continue using legacy systems that are critical to their daily functions and business needs (Barney, 2022).

### Organizational Policy

An organization's organizational policy is a set of rules or guidelines that it creates to direct the conduct of its stakeholders and employees. These guidelines may address a variety of subjects, such as moral conduct, workplace behavior, health and safety, data protection, and business operations. Organizational policies serve to build a consistent and professional corporate culture, provide clear and consistent guidance for employees and stakeholders, and assist in ensuring compliance with laws and regulations. The leadership of a business often develops and implements organizational policies, which may be disseminated through employee handbooks, training manuals, and other corporate publications.

### Hardware Requirement

The specific types and numbers of physical components needed to execute a given software application or system are referred to as the hardware requirements. The types and specifications of the computer processors, memory, storage, and other hardware elements that are required for the software to run effectively are often included in these requirements.

Depending on the complexity and capabilities of the software, the hardware specifications are frequently provided by the software vendor or developer. In order to ensure optimum performance and prevent any problems or errors, it is crucial to make sure that a computer or device satisfies the hardware requirements for a specific application or system.

### **Background Information**

The organization's main line of business is software development. For the New Republica College, which mostly works in Nepal and educates a number of local and international students, the program needs to be modified. As a result, it was difficult for this college to collect data from all departments and students. A system known as the college management system was developed to administer the college's accounting system, library management system, and assignment tracking system. A multitude of hardware and software elements are required for this strategy. The difference between the various educational specialties leads to several problems. The project development process is known among programmers, but college students might not be familiar with it. As time went on, it was resolved.

### **Problem Statement**

An outline of a problem or issue that has to be addressed or resolved is known as a problem statement. It is a brief statement that expresses the issue at hand succinctly, and it is frequently used to define the parameters of a project or research effort.

The gap or opportunity that the project or research is meant to address should be clearly identified in a proper issue statement, which should be precise and targeted. It should also be pertinent, significant, and should express plainly how the issue will affect or have implications if it is not resolved.

### **Data collection Process**

To answer specified research questions, test hypotheses, and assess results, data collection is the act of acquiring and measuring information on variables of interest in a systematic and defined manner.

### **Methods of Data collection process**

The methods are:

## **Surveys**

Individuals are prompted to fill out a questionnaire or form as part of surveys, a popular technique for gathering data. You can conduct surveys online, over the phone, via mail, or in person.

## **Interviews**

In person or online, questions are posed to participants in interviews. Interviews can be unstructured, with more open-ended and exploratory inquiries, or organized, with a planned list of questions.

## **Focus Groups**

Focus groups include gathering a small group of people to talk about a certain subject or issue. Focus groups can be useful for developing thoughts and learning more about a certain subject.

## **Observation Studies**

Observational studies watch and document how people or groups act without interfering with or changing the situation. Both a controlled laboratory environment and a natural environment can be used to conduct observational research.

## **Experiments**

In experiments, one or more factors are manipulated to see how they affect a dependent variable. In scientific research, experiments are frequently employed to test hypotheses and assess cause-and-effect connections.

## **Case Studies**

Case studies examine a particular person, organization, or circumstance in great detail. Case studies are frequently employed in social science research to offer in-depth understanding of a specific event.

## **Secondary Data**

Data that has already been gathered and is available for reuse is referred to as secondary data. Government statistics, industry reports, and published research studies are examples of secondary data sources.

## **Software Analysis**

Software analysis is basically a requirement analysis that aims to determine the tasks that are needed to get fully functional software. This analysis undergoes various requirement of stakeholders, documenting, and validating software and system requirements. Without requirement analysis, a project will not be completed and would lead to failure as design can only be implemented after this analysis (Waghmare, 2022).

There are three types of requirement analysis. They are given below:

- Eliciting Requirement: Simple collection of data through stakeholders in the form of processing documentation (Waghmare, 2022)
- Recording Requirement: The documented requirements are recorded in various forms that involve the use of summary list (Waghmare, 2022)
- Analyzing Requirement: Last stage where the gathered resources are clear, concise, unduplicated, valid, and consistent without any conflicts arising (Waghmare, 2022)

## **DFD**

A data flow diagram (DFD) is a graphical or visual representation using a standardized set of symbols and notations to describe a business's operations through data movement (Nolle, 2021). Data flow across a system is graphically represented by a data flow diagram (DFD). It shows how data enters a system, how it is processed there, and how it is exported or stored.

A DFD is made up of a number of symbols and notations that stand in for a system's many parts, such as the input, output, and processing of data. A DFD's standardized symbols and nomenclature make it simple to comprehend and interpret the diagram.

When modeling the data flow through a system and looking for potential bottlenecks or inefficiencies, DFDs are frequently employed in systems analysis and design. They can also be used to communicate design concepts to stakeholders and document already-existing systems.

## Context Diagram

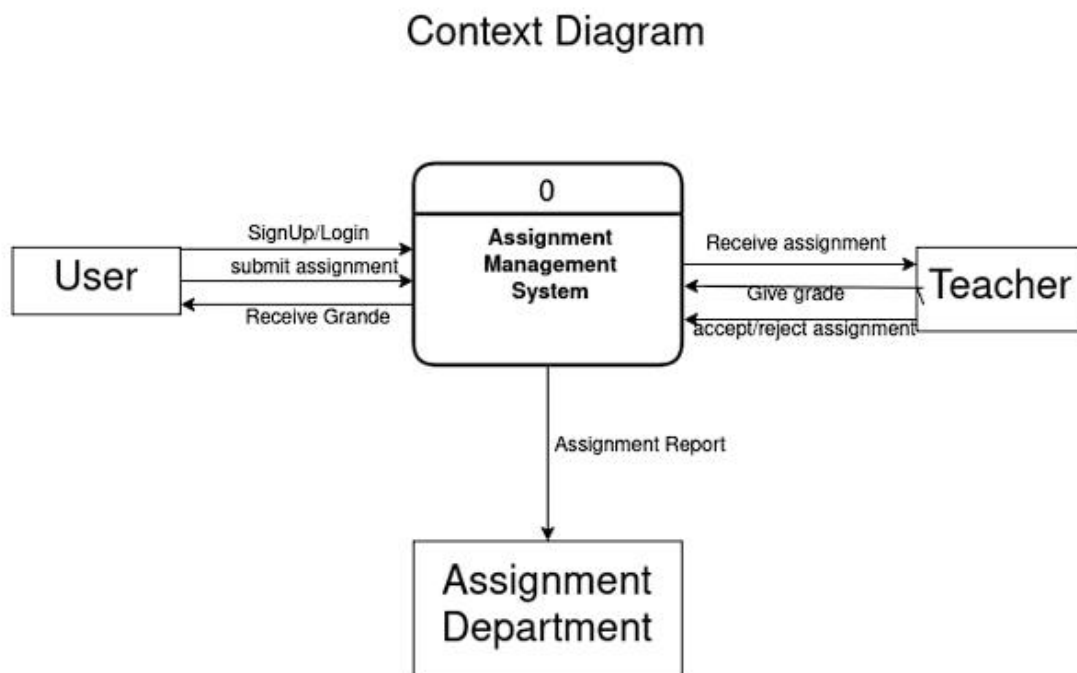


Figure 10

## Level 0 diagram

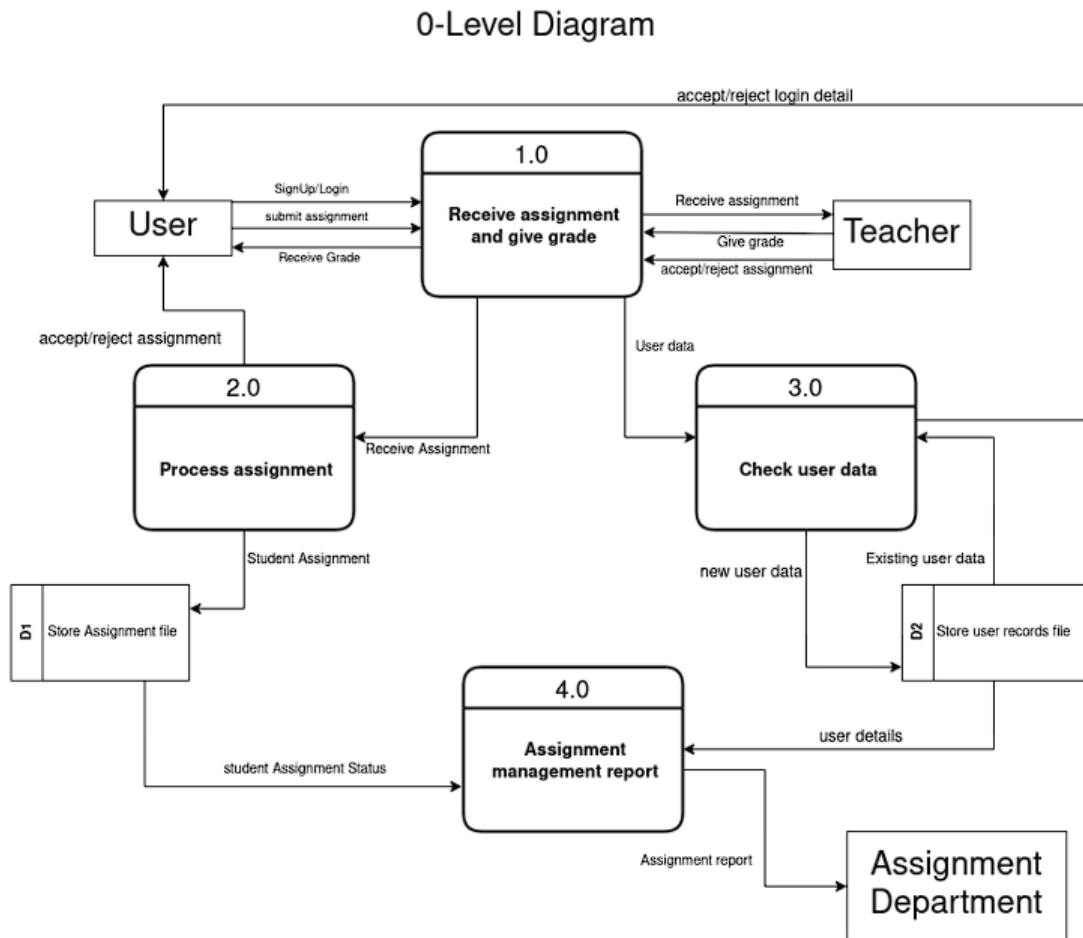


Figure 11

## Level 1 Diagram

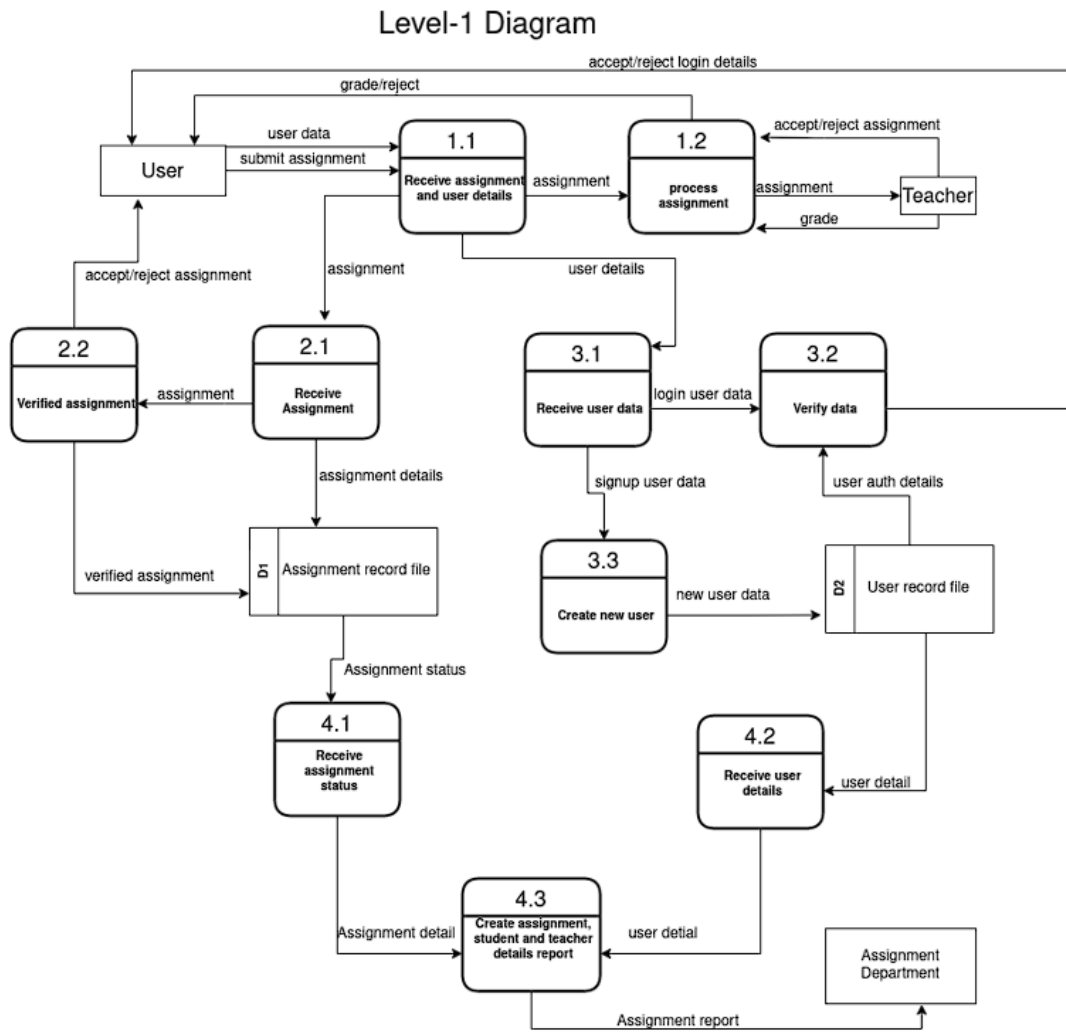


Figure 12

## Level 2 Diagram

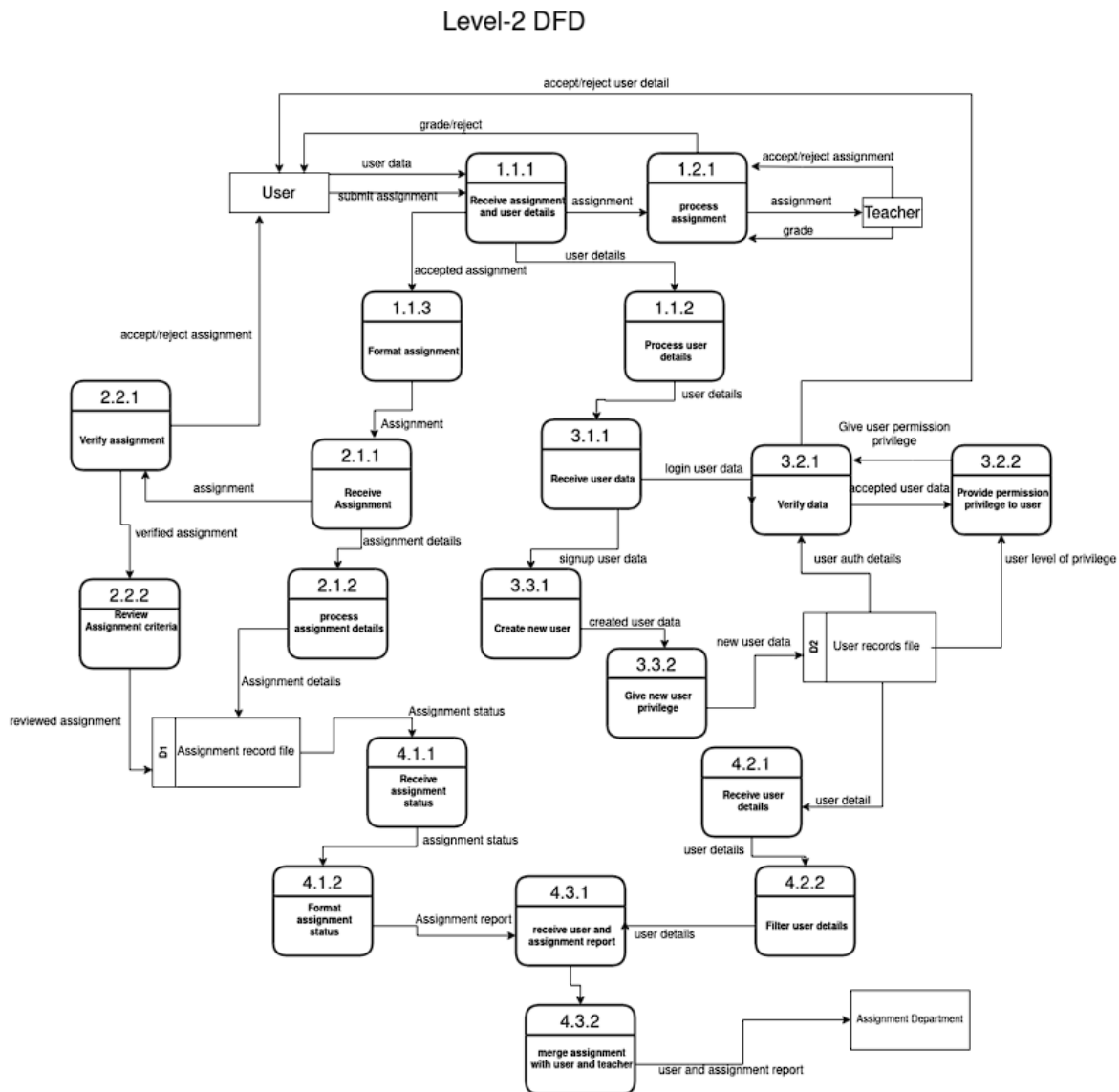


Figure 13

## ER Diagram

ER Diagram stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database (Peterson, 2022). A database's entities, relationships, and characteristics are graphically represented in an entity relationship (ER) diagram. It is a visual tool for modeling and designing databases that aids in explaining to stakeholders how a database is organized.



The entities, relationships, and characteristics in a database are represented by a number of symbols and notations in an ER diagram. Rectangles are used to represent the entities in an ER diagram, which often reflect real-world items or concepts like clients, orders, or products. Lines linking the rectangles serve as a visual representation of the relationships between entities, showing how they are related to one another. Finally, oval shapes are used to symbolize an entity's attributes, which are utilized to explain the traits or properties of the entity.

Identifying the entities, relationships, and characteristics that should be included in a database as well as visualizing the structure of the database before it is implemented are two common uses for ER diagrams, which are a crucial tool in database design. They can also be used to describe a database's structure to stakeholders and to record already-existing databases.

### ER diagram according to scenario

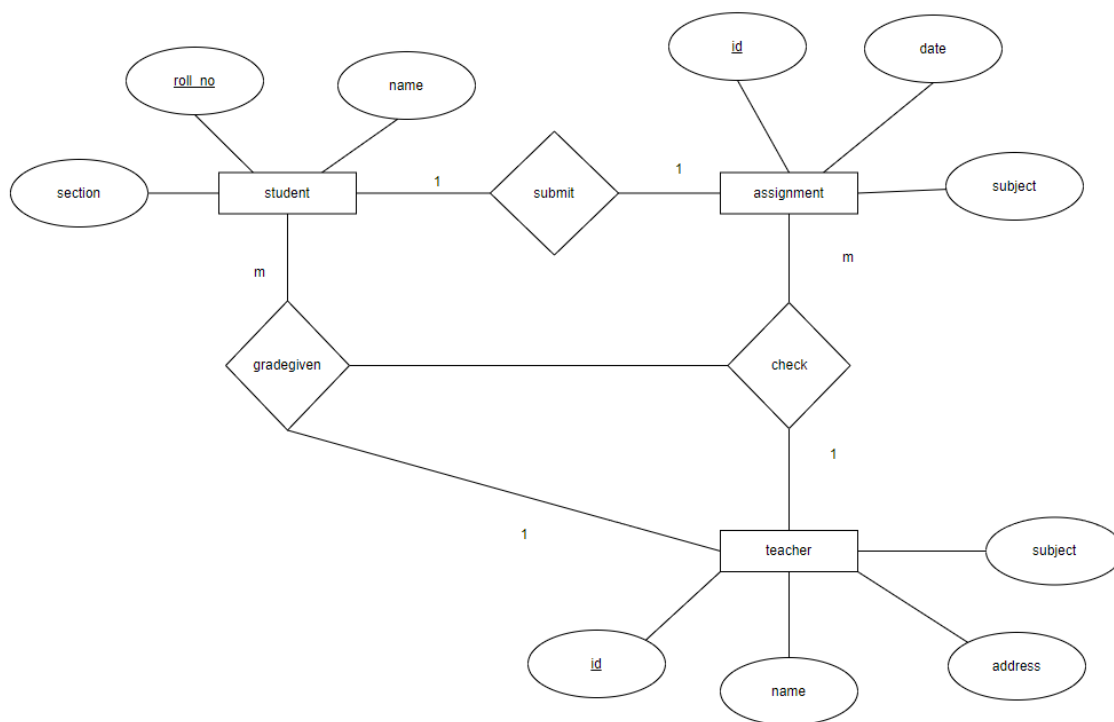


Figure 14

### Case Tool

Computer aided software engineering (CASE) is the implementation of computer facilitated tools and methods in software development. CASE is used to ensure a high-quality and defect-free

software. CASE ensures a check-pointed and disciplined approach and helps designers, developers, testers, managers and others to see the project milestones during development (GreekforGreek, 2020). Software developers, systems analysts, and other IT workers can use case tools to organize the many tasks and activities associated in the software development lifecycle, such as requirements gathering, design, testing, and deployment.

Many of the manual chores associated in software development, such as code generation, testing, and documentation, can be automated with the aid of case tools. By offering features and tools like project management, version control, and collaboration tools, they can also aid in enhancing the effectiveness and quality of the software development process.

Case tools come in a wide variety of forms, each one intended to support particular phases or facets of the software development process. Common case tools include the following:

- Requirement management tools: These tools make it easier to record, arrange, and keep track of a software project's requirements.
- Design tools: These instruments aid in the creation and documentation of the architecture, data model, and user interface of a software system.
- Testing tools: The development and execution of test cases are both automated thanks to these tools.
- Configuration management tools: The codebase and other project assets can be tracked and managed with the use of these tools.
- Project management tools: The resources and activities involved in a software development project can be planned, monitored, and managed with the use of these tools.

### **Software Requirements Specification (SRS) Document**

A software requirements specification (SRS) is a document that describes what the software will do and how it will be expected to perform. It also describes the functionality the product needs to fulfill all stakeholders (business, users) needs (Lane, 2021). The primary criteria that a software system must satisfy are listed in a document called a software requirements specification (SRS). It provides the framework for the system's design, development, and testing. The functional and non-functional needs of the system, as well as any restrictions or limitations, should be clearly and thoroughly described in the SRS.

The goal of an SRS is to create a shared knowledge of the system to be produced between the development team and the stakeholders (such as customers and users). It aids in defining the project's scope and guarantees that the finished result contains all necessary components. It also acts as a point of reference for the duration of the development process, assisting in ensuring that the system is constructed in accordance with the predetermined specifications.

### **The uses of SRS**

The primary criteria that a software system must satisfy are listed in a document called a software requirements specification (SRS). There are several uses of SRS they are:

#### **Defining the scope of the project**

An SRS identifies which features and functions will be included in the project and which will be excluded, helping to define its bounds.

#### **Establishing a common understanding**

An SRS aids in establishing a shared understanding of the system to be produced between the development team and the stakeholders (such as customers and users).

#### **Providing a reference point**

Throughout the development phase, an SRS acts as a reference point to make sure the system is constructed in accordance with the predetermined criteria.

#### **Facilitating communication**

By stating the requirements and limitations of the system, an SRS aids in facilitating communication between the development team and the stakeholders.

#### **Improving the quality**

By giving a thorough and precise description of the requirements and constraints, an SRS can help to increase the quality of the program by allowing for the early detection and correction of flaws.

### **Steps to Improve Software Quality**

Software testing is a process of quality control and assurance, not only testing a product to see if the criteria are met or not.

- Quality control: It is a method of defect detection and correction
- Quality assurance: It is a method of defect prevention when the product is under control

### **Some of the advantage of software testing**

Software testing is a procedure that aids in finding flaws, mistakes, and bugs in a system or piece of software. As it helps to guarantee that the program complies with the requirements and performs as intended, it is a crucial step in the software development process. The following are some advantages of software testing:

#### **Improved Quality**

Before the program is released, problems and errors can be found and remedied thanks to testing. This allows the software's overall quality to improve.

#### **Increased reliability**

Testing makes ensuring that the program is dependable and capable of carrying out its intended tasks without error.

#### **Reduced costs**

Early in the development process, flaws can be found and fixed to save time and money. Additionally, testing can assist in preventing costly issues that might occur if flaws are not found until after the program has been put to use.

#### **Improved user satisfaction**

Testing can help to improve user happiness by ensuring that the program works as intended and satisfies users' needs.

#### **Increased efficiency**

A software's potential for improvement and efficiency can be found through testing, which can ultimately increase productivity.

#### **Enhanced security**

Testing can assist in locating software flaws and ensuring that the program is safe and secure from outside dangers.

### **Function design Paradigm**

A type of design called functional design concentrates on the capabilities and functions of a system, product, or service. Instead of being concerned with how something looks or functions, it is more interested in what it can do.

Software development, product design, and system engineering are a few areas where functional design can be applied. Creating a specification that details the features, capabilities, intended usage, and input and output data that the software will process is a common task in the functional design phase of software development.

When it comes to product design, functional design may entail making prototypes and testing them to make sure the final product can carry out the tasks for which it was created. Creating a system architecture that details the features and capabilities of the system, as well as how it will be connected with other systems or components, is a common task in system engineering called functional design.

Comparable design paradigms to functional design include user-centered design, which prioritizes the requirements and preferences of the user, and aesthetic design, which concentrates on the aesthetics and visual appeal of the system, product, or service.

### **The suitability of software behavioral design techniques**

User requirements, commonly referred to as customer demands, outline what the technique fixes as well as the tasks that users must be able to do. User requirements are frequently identified using description text in a User Requirements Document (URD). The user's needs, which are typically a sign of the user's skills, are the major input for developing system requirements. The creators of the system definition use blocks to construct the mechanism. This is the situation. Conventional "statements," like "must," specify how objects function. Additional requirements and well-designed requirements are two categories for system requirements. We were unable to expand the essential software application for the insurance business organization without the client and the software. It is a challenging task to design a new machine with all of its features and user needs for such a large company. When the work was initially given to our firm, we decided to do a feasibility research on this freshly manufactured equipment. Once the study's results were reliable, we then began the application development process. We examined a range of model types for creating devices, such as Waterfall, Agile, Prototype, Spiral, and others. You will decide to utilize the agile version in this project once we have received feedback from all project participants and you have considered the advantages and disadvantages of each model. ERD and DFD were created to help the project's creator and other participants get feedback on the entire device, how it will operate once finished, and what specific requirements various users have for this system. After

assembling the machine format using a variety of technologies, we take on the leading clothier. Once this phase was complete, we advanced to the improvement portion. The activities are then prioritized for substantial software program requirements, with high order obligations finished first and low arrange responsibilities finished last. This aids in determining the need for software.

### **Software Performance Specification**

The performance requirements and expectations for a software system or application are described in a document known as a software performance specification. It often includes information on the product's anticipated performance traits, such as speed, dependability, and scalability, as well as any particular performance limits or standards that the software must adhere to.

Typically produced as part of the software development process, a software performance specification can be used to direct the development of the software. Once the software is finished, it can be used to assess its performance to make sure it satisfies the required performance standards.

A software performance specification may include information on how the performance of the software will be measured and evaluated in addition to the software's performance criteria. This may entail defining criteria for assessing the performance of the software in relation to specified performance measures.

### **Improving of Software Behavior Specification in SDLC**

The process of specifying a software system or application's behavior and capabilities is known as software behavior specification. Because it helps to guarantee that the software is planned and implemented in a way that satisfies the needs and expectations of the users and stakeholders, it is an essential component of the software development life cycle (SDLC).

There are several ways in which software behavior specification can be important in the SDLC:

#### **Defining requirements**

Software behavior specifications aid in defining the precise needs and expectations for the program, as well as its intended uses, potential, and performance traits. By doing this, you can make sure that the software is created in a way that satisfies the requirements of users and other stakeholders.

### **Facilitating communication**

The specification of software behavior can aid in facilitating communication between various participants in the development process, such as developers, testers, and users. It may be simpler for these stakeholders to comprehend and debate the project if the behavior and capabilities of the software are clearly defined.

### **Improving quality**

It is possible to discover potential problems and faults during the development process by outlining the behavior and capabilities of the software in advance. This can assist to increase the overall quality of the software.

### **Reducing rework**

It is feasible to lower the possibility of misunderstandings or miscommunications that could result in rework later in the development process by precisely specifying the behavior and capabilities of the software.

### **Conclusion**

This section has examined the numerous kinds of components that are required for the creation of software. Both system development and system research and analysis fall under this category. Numerous project stakeholders, client needs, scope definitions, input-output processes, and process descriptors have all been identified. The cost, legacy, hardware component, organizational policies, context, issue statement, data collecting process, DFD, and ER diagram have all been covered as well as other constraints. As a result, this section gently explains all that is required. For instance, the most important formats to guarantee that the stakeholders' knowledge is consistent are DFD and ER diagrams.

## **Introduction**

We will discuss and clarify in particular how the needs of the user and the software have been considered in this part of the assignment. Additionally, we are required to deliver the tables, which must accurately reflect both the finished product and the anticipated customer needs. With the aid of software requirements, we will be able to talk about and address a wide range of related questions. Software requirements offer a description, a codification, and a specification of a software-based solution that may be applied to address and satisfy a perceived need. They provide explanations of how, and to what extent, the user, software, and hardware should interact with the overall and underlying machine. They might be described using guidelines and limitations, and they could be used to denote device attributes or traits. As a result, all prospective outline-related queries will be sorted through in this section.

## **Documentation of user and system software**

The written or electronic materials that offer instructions and direction on how to use and maintain the software are referred to as user and system software documentation. A variety of formats, including as user manuals, system documentation, release notes, help files, and training materials, may be used to create this content.

User guides and instructions, as well as details on the software's features and capabilities, are provided in user manuals. For developers and other technical stakeholders, system documentation provides technical information on the design and implementation of the software. Release notes give details on the modifications and additions made to the software in a certain release. Help files, which can be obtained from the software itself, offer context-specific support and instructions on how to use the program. Guides, tutorials, and other resources might be included in training materials that are developed to assist users in learning how to use the product efficiently.

Users and software requirements have been addressed by the following

The precise demands and expectations that users and stakeholders have for a software system are referred to as user and software requirements. There are several steps involved in meeting these standards, such as:

- Identifying the requirements: This entails compiling data on the requirements and goals of users and stakeholders, as well as any restrictions or limitations that the software must



consider. Interviews, questionnaires, focus groups, and other techniques for getting feedback may be used to do this.

- Analyzing and prioritizing the requirements: The needs must then be discovered, evaluated, and prioritized according to their significance and impact. This can ensure that the most crucial needs are met first and that resources are utilized efficiently.
- Specifying the requirements: The requirements should be precisely defined and documented, including information on how they should be put into practice and assessed to see if they are met.
- Implementing the requirements: Following a specified development process that incorporates testing and quality assurance to guarantee that they are implemented properly, the requirements should be implemented in the program.
- Validating the requirements: Once the program has been put into place, it needs to be tested to make sure it adheres to the specifications. User testing or other techniques for assessing the performance of the software may be used in this.

### **Expected client requirement and Actual Output**

- Easily to accessible the clients this is expected and the actual output will be impressive services and outcome of the software.
- Smooth and correct accounting calculating this will be expected client requirement and the actual output will be necessary fundamental of the software, all the calculation of the accounting department is correct
- High security will be client requirement and the actual output will be system password will be strong

### **Algorithm**

An algorithm is a procedure used for solving a problem or performing a computation. Algorithms act as an exact list of instructions that conduct specified actions step by step in either hardware- or software-based routines (Gillis, 2019). An algorithm is a series of instructions that must be carried out in a particular order in order to solve a problem or complete a task. Numerous fields, such as computer programming, mathematics, and data analysis, use algorithms.

## Example

Here is an example of a simple algorithm for finding the largest number in a list of numbers:

- Set the first value in the list as the value of the "biggest" variable.
- Repeat for the remaining numbers on the list.
- If any number is larger than the value of "biggest" at the moment, for each number, set "largest" to that value.
- The largest number in the list will be found in "largest" after iterating over the full list.

Finding the greatest number in a list is a problem that can be solved using an algorithm that consists of four steps. The algorithm iterates through the list of integers using a loop, comparing each one to the current value of "biggest" as it goes. The value of "biggest" is updated to the current number if the current number is greater. The biggest number in the list will be in "largest" once the loop is finished.

Algorithms can be used to handle a broad range of issues, from straightforward jobs like sorting a list of numbers to more difficult ones like analyzing data sets or optimizing resource allocation. They can be implemented in a variety of programming languages.

## Benefits

There are several benefits they are:

### Efficiency

Algorithms can be created to handle issues effectively and efficiently, frequently requiring less time or effort than alternative methods.

### Accuracy

Algorithms are composed of a series of exact steps that must be carried out in a particular order in order to produce results that are accurate.

### Repeatability

Algorithms can be applied repeatedly to solve the same issue, which is advantageous when the issue needs to be resolved more than once.

## Adaptability

Algorithms are flexible and adaptable because they may be changed or tailored to handle various issues or types of data.

## Automation

Automating procedures and tasks using algorithms can improve efficiency and lower the possibility of human error.

## Flow chart

A flow chart is a graphical representation of a system or process that highlights the many processes or tasks that must be completed as well as the connections between them. To visually depict processes and systems, flow charts are frequently used in a variety of contexts, including business, engineering, and computer science.

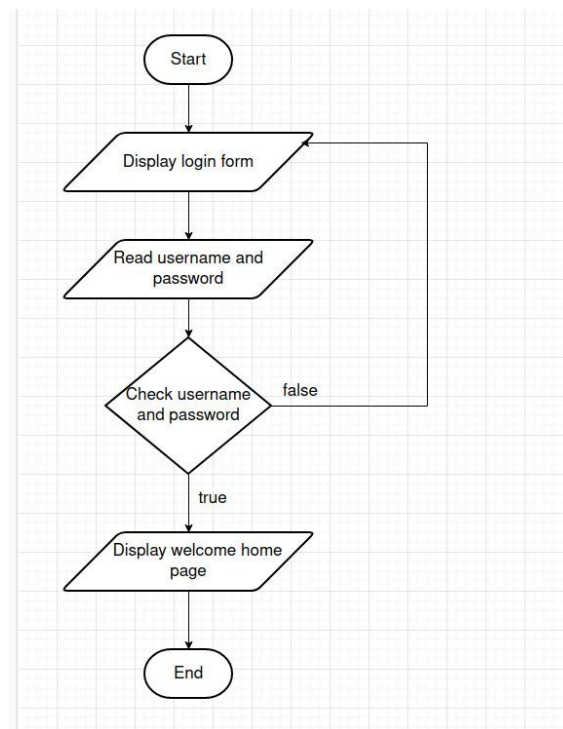


Figure 15

The various steps or tasks in the process or system are represented by a variety of symbols or shapes in flow charts. Arrows that connect these symbols show how the system or process flows

from one stage to the next. Annotations or comments may also be added to flow charts in order to add extra information about the system or process.

From straightforward actions like ordering a product online to more complicated ones like creating a product or rendering a service, flow charts may be used to represent a broad variety of processes and systems. They can be helpful for comprehending and assessing systems and processes as well as for finding room for advancements or enhancements.

## **Benefits**

The benefits are:

### **Visualize and Document Processes**

Flow charts make it simpler to understand and follow a process by clearly communicating the steps involved. For complicated processes that would be challenging to comprehend from a textual description alone, this is very helpful.

### **Identify Inefficiencies and Problems**

Flow charts can help locate bottlenecks or places where the process might be inefficient by outlining the steps in the process.

### **Facilitate Communication and Collaboration**

Flowcharts can be distributed to team members, interested parties, and other pertinent parties to make sure that everyone is on the same page and pursuing the same objectives.

### **Improve Process Efficiency**

Flowcharts can aid in streamlining and improving processes by identifying faults and inefficiencies.

### **Enhance Decision Making**

Flowcharts can be used to aid decision-making by displaying the prospective results and effects of various actions.

## **FSM (Finite State Machine)**

Finite state machine (FSM) is a term used by programmers, mathematicians, engineers and other professionals to describe a mathematical model for any system that has a limited number of conditional states of being (Contributor, 2019). It is a mathematical representation of how a system

behaves that has a finite number of possible states. A system that can only be in one of a small number of states at any given time is said to be a finite state machine.

In computer science and engineering, FSMs are frequently used to describe the behavior of a system that can change depending on input or events. For instance, a traffic light with states like "red," "yellow," and "green" can be described as a finite state machine. Based on a set of guidelines, the traffic light changes states (e.g., when the light is red, it stays red for a certain amount of time before switching to yellow, and then to green).

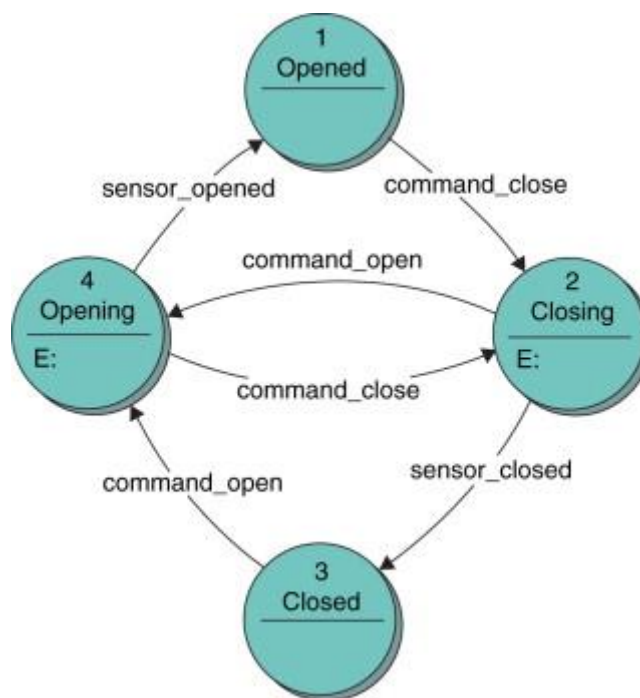


Figure 16

A flow chart or state diagram is frequently used to graphically illustrate FSMs, with the states denoted by circles and the transitions between the states shown by arrows. FSMs can be used to design and manage the behavior of complicated systems, and they can be implemented in hardware or software.

## EFSM (Extended Finite State Machine)

Extended Finite State Machine, often known as EFSM, is a type of Finite State Machine (FSM) model that is used to describe system behavior. In that they are mathematical models used to describe the behavior of a system that can take one of a finite number of states, EFSMs and FSMs are comparable in this regard. But EFSMs are different from FSMs in that they permit the addition of extra information or variables to the state transition procedure.

An EFSM model uses a collection of states, transitions, and variables to represent the behavior of the system. The variables can be used to control the transitions between states, represent the internal status of the system, or both. For instance, in a system that keeps track of the amount of products in a shopping cart, the quantity of items could be a factor that affects how quickly one state changes into another (e.g., moving from a "shopping" state to a "checkout" state when the cart is not empty).

In software engineering and design, EFSMs are frequently used to describe how systems behave when they need to store and manipulate data in order to function. They can be implemented in software or hardware and graphically depicted using state diagrams or flow charts.

Difference between FSM and EFSM are

Features	FSM	EFSM
Number of states	Finite	Finite
State transition process	It determined by current state and input	It determined by the current state, input, and variables
variables	No	Yes (it is used to store data or represent the system internal state)
Graphical representation	State diagram or flow chart	Same as FSM (State diagram or flow chart)
implementation	Hardware or software	Same as FSM (Software or Hardware)
Use cases	Systems that only have a small number of states and transitions	Systems that demand data manipulation or have intricate behavior

## Data Driven Software

Data-driven development is a software engineering approach that relies on data to guide the development process. It involves selecting and monitoring metrics or key performance indicators (KPIs) that help you better understand your product so you can make continuous improvements (CloudZero, 2021). Software that is data-driven uses data as the main input to guide its functioning or behavior. This indicates that rather than being guided by rigid rules or predetermined instructions, the software is created to process, analyze, and make decisions based on data inputs.

Applications like data mining, business intelligence, and machine learning that call for the analysis of massive amounts of data frequently use data-driven software. It can also be applied to situations that call for quick judgment calls or flexibility, such as autonomous vehicles or intelligent personal assistants.

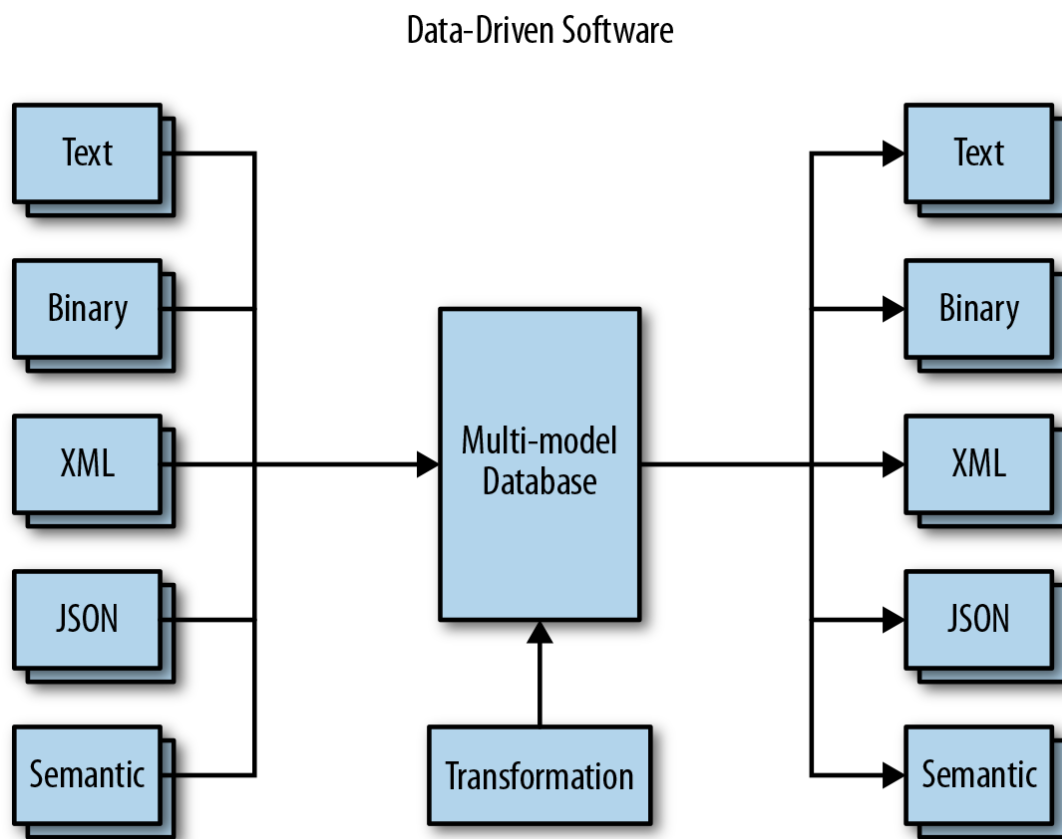


Figure 17

### **Application driven software**

Software that is application-driven is created to support a particular task or application. This kind of software is normally intended to be used in a particular context or environment and is usually focused on enabling users to carry out a specific set of tasks or duties.

Word processors, spreadsheet applications, and graphics editing software are a few examples of application-driven software. The users of these tools can make and edit documents, spreadsheets, and photos, respectively.

Data-driven software, which uses data as the main input to drive its behavior or functioning, is sometimes contrasted with application-driven software. Applications like data mining, business intelligence, and machine learning that call for the analysis of massive amounts of data frequently use data-driven software.

### **Improving the reliability and effectiveness of the software using data driven software**

There are several ways that data-driven software can be improve the following methods are:

#### **Data-driven decision-making**

Data-driven software can assist in enhancing the accuracy and dependability of decision-making processes by using data as the main input for driving its behavior. This is especially helpful for applications like self-driving cars or intelligent personal assistants that need to make decisions or react in real-time.

#### **Improved accuracy**

Large amounts of data may be analyzed and processed using data-driven software, which makes it possible to spot patterns and trends that might not be immediately obvious. Predictions, recommendations, and other software outputs may become more accurate as a result of this.

#### **Enhanced adaptability**

Data-driven software can adapt and enhance its performance over time based on data inputs by utilizing machine learning techniques. This may aid in enhancing the software's performance in dynamic or changing contexts.



### **Increased efficiency**

Data-driven software can assist in streamlining procedures and lowering the need for manual involvement, increasing the program's effectiveness and productivity.

### **Conclusion**

I have completed the last part of my assignment by producing the documentation that demonstrates how user and software criteria have been met. When we were developing software for Bitramod Technical College, it was essential that we consider user and software needs. Without the user and software specifications, we were unable to produce the required software for Apex Educational Institute. The best way to discover user requirements was through documentation. Then I discussed the different software definition methodologies and made two recommendations. A software behavior specification document specifies how something should be done and outlines the software's step-by-step functioning. A specifications document may include all of the possible error scenarios and suitable error messages for a certain form. Project Management Information System, then I differentiated between finite state machines and extended finite state machines, and last I provided explanations and examples of how data-driven software can improve the reliability and efficiency of the software.

## References

Ambler, S. W., 2018. *agilemodeling*. [Online]

Available at: <http://agilemodeling.com/artifacts/constraint.htm>

[Accessed 10 11 2019].

Barney, N., 2022. *TechTarget*. [Online]

Available at: <https://www.techtarget.com/searchitoperations/definition/legacy-application>

[Accessed 15 December 2022].

CloudZero, 2021. *CloudZero*. [Online]

Available at: [https://www.cloudzero.com/blog/data-driven-](https://www.cloudzero.com/blog/data-driven-development#:~:text=Data%2Ddriven%20development%20is%20a,you%20can%20make%20continuous%20improvements)

[development#:~:text=Data%2Ddriven%20development%20is%20a,you%20can%20make%20continuous%20improvements.](https://www.cloudzero.com/blog/data-driven-development#:~:text=Data%2Ddriven%20development%20is%20a,you%20can%20make%20continuous%20improvements)

[Accessed 26 December 2022].

Contributor, T., 2019. *TechTarget*. [Online]

Available at: <https://www.techtarget.com/whatis/definition/finite-state-machine>

[Accessed 26 December 2022].

Contributor, T., 2019. *TechTarget*. [Online]

Available at: <https://www.techtarget.com/searchsoftwarequality/definition/spiral-model>

[Accessed 25 December 2022].

CPNI, 2022. *CPNI*. [Online]

Available at: [https://www.cpni.gov.uk/security-considerations-assessment-](https://www.cpni.gov.uk/security-considerations-assessment-sca#:~:text=The%20Security%20Considerations%20Assessment%20(SCA,Last%20Updated%2021%20February%202022)

[sca#:~:text=The%20Security%20Considerations%20Assessment%20\(SCA,Last%20Updated%2021%20February%202022](https://www.cpni.gov.uk/security-considerations-assessment-sca#:~:text=The%20Security%20Considerations%20Assessment%20(SCA,Last%20Updated%2021%20February%202022)

[Accessed 24 December 2022].

DigitalGuide, 2019. *DigitalGuide*. [Online]

Available at: [https://www.ionos.com/digitalguide/websites/web-development/waterfall-](https://www.ionos.com/digitalguide/websites/web-development/waterfall-methodology/#:~:text=The%20five%2Dstage%20waterfall%20model,implementation%2C%20testing%2C%20and%20operation)

[methodology/#:~:text=The%20five%2Dstage%20waterfall%20model,implementation%2C%20testing%2C%20and%20operation.](https://www.ionos.com/digitalguide/websites/web-development/waterfall-methodology/#:~:text=The%20five%2Dstage%20waterfall%20model,implementation%2C%20testing%2C%20and%20operation)

[Accessed 12 December 2022].

EmergentSoftware, 2021. *EmergentSoftware*. [Online]

Available at: <https://www.emergentsoftware.net/blog/the-7-stages-of-the-software-development-life-cycle-sdlc/>

[Accessed 10 December 2022].

Entrepreneurindia, 2021. *Entrepreneurindia*. [Online]

Available at: <https://www.entrepreneurindia.co/blog-description/13679/what+are+the+components+of+a+project+feasibility+report%3F>

[Accessed 15 December 2022].

Fernando, J., 2022. *Investopedia*. [Online]

Available at: <https://www.investopedia.com/terms/s/stakeholder.asp>

[Accessed 15 December 2022].

Gillis, A. S., 2019. *Techtarget*. [Online]

Available at: <https://www.techtarget.com/whatis/definition/algorithm>

[Accessed 20 December 2022].

Gillis, A. S., 2020. *TechTarget*. [Online]

Available at: <https://www.techtarget.com/searchsoftwarequality/definition/quality-assurance>

[Accessed 24 December 2022].

Grant, M., 2021. *Investopedia*. [Online]

Available at: <https://www.investopedia.com/terms/s/scope.asp>

[Accessed 25 December 2022].

GreekforGreek, 2020. *GreekforGreek*. [Online]

Available at: <https://www.geeksforgeeks.org/computer-aided-software-engineering-case/>

[Accessed 14 December 2022].

Greeksforgreeks, 2019. *Greeksforgreeks*. [Online]

Available at: <https://www.geeksforgeeks.org/dynamic-systems-development-method-dsdm/>

[Accessed 10 December 2022].

IncestopediaTeam, 2022. *Incestopedia*. [Online]

Available at: <https://www.investopedia.com/terms/f/feasibility-study.asp>

[Accessed 13 December 2022].

JavaTpoint, 2021. *JavaTpoint*. [Online]

Available at: <https://www.javatpoint.com/software-engineering-agile-model>

[Accessed 12 December 2022].

Lane, C., 2021. *Perforce*. [Online]

Available at: <https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document>

[Accessed 16 December 2022].

Martin, M., 2022. *Guru99*. [Online]

Available at: <https://www.guru99.com/software-development-life-cycle-tutorial.html>

[Accessed 12 December 2022].

Martin, M., 2022. *Guru99*. [Online]

Available at: <https://www.guru99.com/what-is-rad-rapid-software-development-model-advantages-disadvantages.html>

[Accessed 12 December 2022].

Nehra, M., 2022. *Decipherzone*. [Online]

Available at: <https://www.decipherzone.com/blog-detail/agile-development-lifecycle>

[Accessed 12 December 2022].

Nolle, T., 2021. *Tech Target*. [Online]

Available at: <https://www.techtarget.com/searchdatamanagement/definition/data-flow-diagram-DFD>

[Accessed 25 December 2022].

Peterson, R., 2022. *Guru99*. [Online]

Available at: <https://www.guru99.com/er-diagram-tutorial-dbms.html>

[Accessed 15 December 2022].

ScalerAcademy, 2022. *ScalerAcademy*. [Online]

Available at: <https://www.interviewbit.com/blog/iterative-model/>

[Accessed 12 December 2022].

Team, I. E., 2021. *Indeed*. [Online]

Available at: <https://www.indeed.com/career-advice/career-development/feasibility-report>

[Accessed 15 December 2022].

Tutorialspoint, 2021. *Tutorialspoint*. [Online]

Available at: [https://www.tutorialspoint.com/sdlc/sdlc\\_iterative\\_model.htm](https://www.tutorialspoint.com/sdlc/sdlc_iterative_model.htm)

[Accessed 12 December 2022].

Waghmare, P., 2022. *Testbook*. [Online]

Available at: <https://testbook.com/learn/software-engineering-software-analysis-design/>

[Accessed 14 December 2022].