ASSIGNMENT 1 FRONT SHEET

Qualification	**BTEC Level 5 HND Diploma in Computing**		
Unit number and title	Unit 9: Software Development Life Cycle		
Submission date	07/02/2021	**Date Received 1st submission**	
Re-submission Date	26/02/2021	**Date Received 2nd submission**	
Student Name	PHAM CAO NGUYEN	**Student ID**	GCC18074
Class	GCC0801	**Assessor name**	NGUYEN HUNG DUNG
Student declaration I certify that the assignment submission is entirely my own work and I fully understand the consequences of plagiarism. I understand that making a false declaration is a form of malpractice.			
		Student's signature	CAONGUYEN

Grading grid

P1	P2	P3	P4	M1	M2	D1	D2

❒ **Summative Feedback:**		❒ **Resubmission Feedback:**
Grade:	**Assessor Signature:**	**Date:**
Internal Verifier's Comments:		
Signature & Date:		

ASSIGNMENT 1 BRIEF

Qualification	**BTEC Level 5 HND Diploma in Computing**		
Unit number	Unit 9: Software Development Life Cycle		
Assignment title	Planning a software development lifecycle		
Academic Year	2019 – 2020		
Unit Tutor	Le Minh Duc		
Issue date		**Submission date**	
Name and date			

Submission Format:	
Format:	The submission is in the form of 1 document. You **must** use the *Times font* with *12pt size*, turn on *page numbering*; set *line spacing to 1.3* and *margins* to be as follows: left = 1.25cm, right = 1cm, top = 1cm, bottom = 1cm. Citation and references must follow the Harvard referencing style.
Submission:	You **must** submit the assignment **by the due date** and follow the submission method specified by the Tutor. The submission form is **soft copy**, which is to be uploaded to the following URL: http://cms.greenwich.edu.vn.
Note:	Your assignment *must* be your own work, and not copied by or from another student or from other sources, such as book etc. If you use ideas, quotes or data (such as diagrams) from books, journals or other sources, you must reference the sources, using the Harvard style. Make sure that you know how to reference properly and that you understand the plagiarism guidelines. **Plagiarism is a very serious offence**, which will result in a failing grade.
Unit Learning Outcomes:	
LO1 Describe different software development lifecycles. **LO2** Explain the importance of a feasibility study**.**	

Assignment Brief and Guidance:

System Scenario

Tune Source is a company headquartered in southern California. Tune Source is the brainchild of three entrepreneurs with ties to the music industry: John Margolis, Megan Taylor, and Phil Cooper. Originally, John and Phil partnered to open a number of brick and mortar stores in southern California specialising in hard-to-find and classic jazz, rock, country, and folk recordings. Megan soon was invited to join the partnership because of her contacts and knowledge of classical music. Tune Source quickly became known as the place to go to find rare audio recordings. Annual sales last year were $40 million with annual growth at about 3%–5% per year. Tune Source currently has a website that enables customers to search for and purchase CDs. This site was initially developed by an Internet consulting firm and is hosted by a prominent local Internet Service Provider (ISP) in Los Angeles. The IT department at Tune Source has become experienced with Internet technology as it has worked with the ISP to maintain the site.

System Request

Project Sponsor: Carly Edwards, Assistant Vice President, Marketing

Business Need: This project has been initiated to increase sales by creating the capability of selling digital music downloads to customers through kiosks in our stores, and over the Internet using our website.

Business Requirements: Using the Web or in-store kiosks, customers will be able to search for and purchase digital music downloads. The specific functionality that the system should have includes the following:

- Search for music in our digital music archive.
- Listen to music samples.
- Purchase individual downloads at a fixed fee per download.
- Establish a customer subscription account permitting unlimited downloads for a monthly fee.
- Purchase music download gift cards.

Business Value: We expect that Tune Source will increase sales by enabling existing customers to purchase specific digital music tracks and by reaching new customers who are interested in our unique archive of rare and hard-to-find music. We expect to gain a new revenue stream from customer subscriptions to our download services. We expect some increase in cross-selling, as customers who have downloaded a track or two of a CD decide to purchase the entire CD in a store or through our website. We also expect a new revenue stream from the sale of music download gift cards.

Conservative estimates of tangible value to the company include the following:

- $757,500 in sales from individual music downloads
- $950,000 in sales from customer subscriptions

- $205,000 in additional in-store or website CD sales
- $153,000 in sales from music download gift cards

Special Issues or Constraints:

- The marketing department views this as a strategic system. The ability to offer digital music downloads is critical in order to remain competitive in our market niche. Our music archive of rare and hard-to-find music is an asset that is currently underutilised.
- Many of our current loyal customers have been requesting this capability, and we need to provide this service or face the loss of these customers' business.
- Because customers have a number of music download options available to them elsewhere we need to bring this system to the market as soon as possible.

Tasks

Complete the following tasks:

Task 1 – SDLC model

You are a project manager of a company named ABC. Your company has been hired by Tune Source to carry out a project that helps them develop a software for the requirements specified in the system request. As the first step, you need to:

1. (P1) Describe the following SDLC models: waterfall, v-model, prototyping, agile and spiral. Choose one that you think suitable for the project and explain why.
    - 350 - 500 words for each model
    - Explanation: 400 – 600 words

    (M1) Discuss the suitability of each of the SDLC models for the project. For each model, specify whether it is most, moderately or least suitable.

    - Discussion and arguments: 800 - 1000 words

    (D1) Discuss the merits of applying the waterfall model to a large software development project.

    - Discussion: 800 – 1200 words
2. (P2) Identify some risks and discuss an approach to manage them.
    - You will have the present what is Risk Management process with clear illustrations and explanations
    - Then you will create a Risk Management Plan to manage risks of Tune Source project

Task 2 – Feasibility study

1. (P3) Discuss the purpose of conducting a feasibility study for the project.

- Discussion: 400 – 1600 words

2. (P4) Discuss how the three feasibility criteria (technical, economic, organisational) are applied to the project. Discuss whether the project is feasible.

Discuss alternative technical solutions using the alternative matrix.

Discussion: 1200 – 1500 words

3. (M2) Explain the components of a feasibility report.

- Discussion economic feasibility study: 350 – 500 words
- Discussion organizational feasibility study: 350 – 500 words

4. (D2) Assess the impact of each feasibility criterion on a software investigation.

- Discussion and represent as feasibility alternatives matrix for: 700 – 900 words

Learning Outcomes 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. **P2** Explain how risk is managed in the Spiral lifecycle model.	**M1** Describe, with an example, why a particular lifecycle model is selected for a development environment.	
LO2 Explain the importance of a feasibility study		**D2** Assess the impact of different feasibility criteria on a software investigation.
P3 Explain the purpose of a feasibility report. **P4** Describe how technical solutions can be compared.	**M2** Discuss the components of a feasibility report.	

Contents

Chapter 1 9

DESCRIBE DIFFERENT SOFTWARE DEVELOPMENT LIFECYCLES 9

I. Describe two iterative and two sequential software lifecycle models. 9

1. SDLC models: 9

2. I chose the Iterative SDLC Model for this project because: 19

II. Explain how risk is managed in the Spiral lifecycle model. 20

1. You will have the present what is Risk Management process with clear illustrations and explanations. 20

2. Risk Management Plan to manage risks of Tune Source project. 20

3. Risk is managed in the Spiral lifecycle model. 21

4. Risk handling in Spiral Model 21

5. There is a chance of error and it is possible to interrupt the project. 21

6. Risk handling in project. 22

Chapter 2 22

EXPLAIN THE IMPORTANCE OF A FEASIBILITY STUDY 22

III. Explain the purpose of a feasibility report. 22

1. The Project Scope. 22

2. What Is a Feasibility Study? 23

3. The Importance of Feasibility Studies. 23

4. Types of Feasibility Study. 23

IV. Describe how technical solutions can be compared. 24

1. Three feasibility criteria (technical, economic, organisational) are applied to the project. 24

2. Feasibility of the project. 26

3. Technical solutions using the alternative matrix. 27

4. The alternative matrix 27

References 28

Chapter 1

DESCRIBE DIFFERENT SOFTWARE DEVELOPMENT LIFECYCLES

Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high-quality software. ... SDLC is a framework defining tasks performed at each step in the software development process.

I. Describe two iterative and two sequential software lifecycle models.

1. SDLC models:

1.1 Waterfall Model.

a. Definition:

Waterfall – is a waterfall SDLC model in which the creation process appears like a river, going step by step through the stages of analysis, projection, realization, testing, execution, and assistance. This SDLC model fully encompasses the incremental execution of each level. This method is strictly documented and predefined with the functionality required for each step of this life-cycle software development model

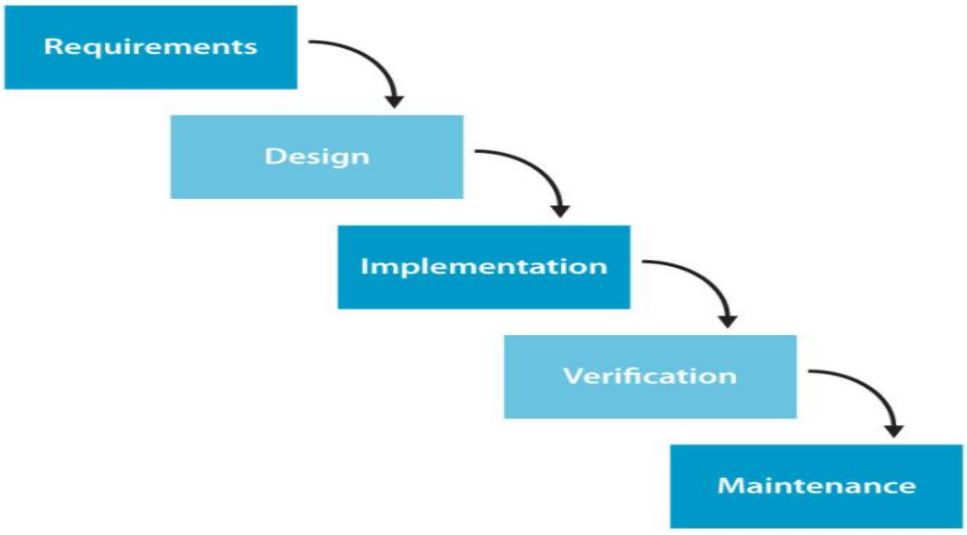

Figure 1: Waterfall SDLC Model

- **Requirement Gathering and analysis** - All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.

- **System Design** - The requirement specifications from the first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** - With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** - All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system** - Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** - There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.
- All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model, phases do not overlap.

b. Advantages:

- Before the next phase of development, each phase must be completed
- Suited for smaller projects where requirements are well defined
- They should perform quality assurance test (Verification and Validation) before completing each stage
- Elaborate documentation is done at every phase of the software's development cycle
- Project is completely dependent on project team with minimum client intervention
- Any changes in software is made during the process of the development

c. Dis-Advantages:

- Error can be fixed only during the phase
- It is not desirable for complex project where requirement changes frequently
- Testing period comes quite late in the developmental process
- Documentation occupies a lot of time of developers and testers
- Clients valuable feedback cannot be included with ongoing development phase
- Small changes or errors that arise in the completed software may cause a lot of problems

d. Use cases for the Waterfall SDLC model:

- Requirements shall be specifically recorded.
- The definition of product is stable.

- The technology stack is predefined and does not render it complex.
- No misunderstanding of specifications.
- The project is a short one.

1.2 v-model.

a. Definition:

The V-shaped SDLC model is an extension of the classic waterfall model and is based on the associated test stage for each stage of development. This is a very strict model, and the next stage is only started after the previous stage. It is also called the "Validation and Verification" model. Each stage has the current process control to ensure that the conversion to the next stage is possible.

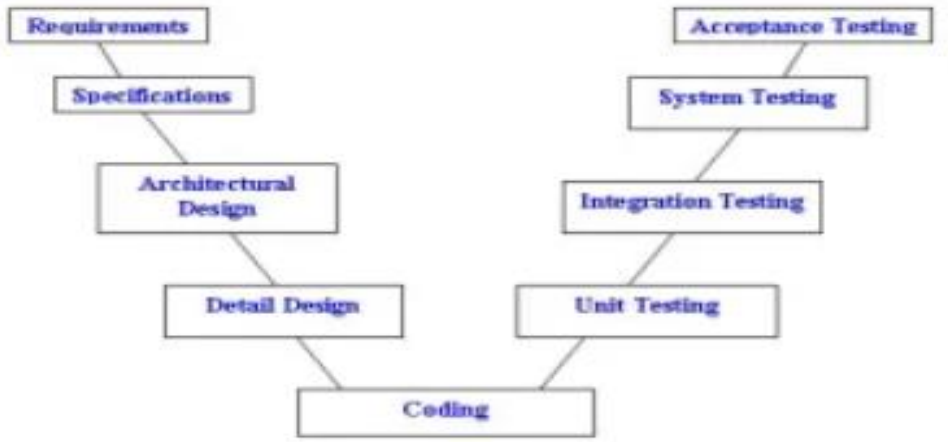

Figure 2: V-shaped SDLC Model

- **Business Requirement Analysis:** This is the first phase in the development cycle where the product requirements are understood from the customer's perspective. This phase involves detailed communication with the customer to understand his expectations and exact requirement. This is a very important activity and needs to be managed well, as most of the customers are not sure about what exactly they need. The acceptance test design planning is done at this stage as business requirements can be used as an input for acceptance testing.
- **System Design:** Once you have the clear and detailed product requirements, it is time to design the complete system. The system design will have the understanding and detailing the complete hardware and communication setup for the product under development. The system test plan is developed based on the system design. Doing this at an earlier stage leaves more time for the actual test execution later.
- **Architectural Design**
  - Architectural specifications are understood and designed in this phase. Usually more than one technical approach is proposed and based on the technical and financial feasibility the final decision is taken. The system design is broken down further into modules taking up different functionality. This is also referred to as High Level Design (HLD).

    - The data transfer and communication between the internal modules and with the outside world (other systems) is clearly understood and defined in this stage. With this information, integration tests can be designed and documented during this stage.
- **Module Design:** In this phase, the detailed internal design for all the system modules is specified, referred to as Low Level Design (LLD). It is important that the design is compatible with the other modules in the system architecture and the other external systems. The unit tests are an essential part of any development process and helps eliminate the maximum faults and errors at a very early stage. These unit tests can be designed at this stage based on the internal module designs.
- **Coding Phase**
    - The actual coding of the system modules designed in the design phase is taken up in the Coding phase. The best suitable programming language is decided based on the system and architectural requirements.
    - The coding is performed based on the coding guidelines and standards. The code goes through numerous code reviews and is optimized for best performance before the final build is checked into the repository.
- **Validation Phases**
    - **Unit Testing:** Unit tests designed in the module design phase are executed on the code during this validation phase. Unit testing is the testing at code level and helps eliminate bugs at an early stage, though all defects cannot be uncovered by unit testing.
    - **Integration Testing:** Integration testing is associated with the architectural design phase. Integration tests are performed to test the coexistence and communication of the internal modules within the system.
    - **System Testing:** System testing is directly associated with the system design phase. System tests check the entire system functionality and the communication of the system under development with external systems. Most of the software and hardware compatibility issues can be uncovered during this system test execution.
    - **Acceptance Testing:** Acceptance testing is associated with the business requirement analysis phase and involves testing the product in user environment. Acceptance tests uncover the compatibility issues with the other systems available in the user environment. It also discovers the non-functional issues such as load and performance defects in the actual user environment.

b. Advantages:

- Simple and easy to use.
- Testing activities like planning, test designing happens well before coding. This saves a lot of time. Hence higher chance of success over the waterfall model.
- Proactive defect tracking – that is defects are found at an early stage.

- Avoids the downward flow of the defects.
- Works well for small projects where requirements are easily understood.

c. Dis-Advantages:

- Very rigid and least flexible.
- Software is developed during the implementation phase, so no early prototypes of the software are produced.
- If any changes happen in midway, then the test documents along with requirement documents has to be updated.

d. Use cases for the V-shaped model:

- The V-shaped model should be used for small to medium sized projects where requirements are clearly defined and fixed.
- The V-Shaped model should be chosen when ample technical resources are available with needed technical expertise.
- High confidence of customer is required for choosing the V-Shaped model approach. Since, no prototypes are produced, there is a very high risk involved in meeting customer expectations.

1.3 Prototyping model.

a. Definition:

The Prototyping Model is one of the most popularly used Software Development Life Cycle Models (SDLC models). This model is used when the customers do not know the exact project requirements beforehand. In this model, a prototype of the end product is first developed, tested and refined as per customer feedback repeatedly till a final acceptable prototype is achieved which forms the basis for developing the final product.

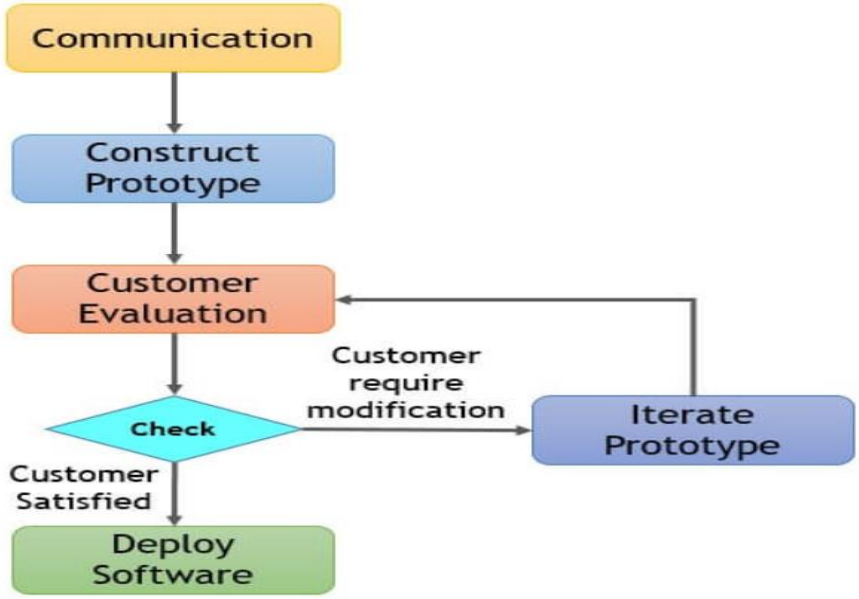

Figure 3: Prototyping SDLC Model

- **Basic Requirement Identification:** This step involves understanding the very basics product requirements especially in terms of user interface. The more intricate details of the internal design and external aspects like performance and security can be ignored at this stage.
- **Developing the initial Prototype:** The initial Prototype is developed in this stage, where the very basic requirements are showcased and user interfaces are provided. These features may not exactly work in the same manner internally in the actual software developed. While the workarounds are used to give the same look and feel to the customer in the prototype developed.
- **Review of the Prototype:** The prototype developed is then presented to the customer and the other important stakeholders in the project. The feedback is collected in an organized manner and used for further enhancements in the product under development.
- **Revise and Enhance the Prototype:** The feedback and the review comments are discussed during this stage and some negotiations happen with the customer based on factors like - time and budget constraints and technical feasibility of the actual implementation. The changes accepted are again incorporated in the new Prototype developed and the cycle repeats until the customer expectations are met.
- **Types of prototype models**
    - **Rapid throwaway:** This method involves exploring ideas by quickly developing a prototype based on preliminary requirements that is then revised through customer feedback. The name rapid throwaway refers to the fact that each prototype is completely discarded and may not be a part of the final
    - **Increment:** This technique breaks the concept for the final product into smaller pieces, and prototypes are created for each one. In the end, these prototypes are merged into the final product.
    - **Extreme:** This prototype model is used specifically for web development. All web prototypes are built in an HTML format with a services layer and are then integrated into the final product.
    - **Evolutionary:** This approach uses a continuous, working prototype that is refined after each iteration of customer feedback. Because each prototype is not started from scratch, this method saves time and effort.

b. Advantages:

- The customers get to see the partial product early in the life cycle. This ensures a greater level of customer satisfaction and comfort.
- New requirements can be easily accommodated as there is scope for refinement.
- Missing functionalities can be easily figured out.
- Errors can be detected much earlier thereby saving a lot of effort and cost, besides enhancing the quality of the software.

- The developed prototype can be reused by the developer for more complicated projects in the future.
- Flexibility in design.

c. Dis-Advantages:

- Costly time as well as money.
- There may be too much variation in requirements each time the prototype is evaluated by the customer.
- Poor Documentation due to continuously changing customer requirements.
- It is very difficult for developers to accommodate all the changes demanded by the customer.
- There is uncertainty in determining the number of iterations that would be required before the prototype is finally accepted by the customer.
- After seeing an early prototype, the customers sometimes demand the actual product to be delivered soon.
- Developers in a hurry to build prototypes may end up with sub-optimal solutions.
- The customer might lose interest in the product if he/she is not satisfied with the initial prototype.

d. Use cases for the Prototyping model:

- The Prototyping Model should be used when the requirements of the product are not clearly understood or are unstable.
- It can also be used if requirements are changing quickly.
- This model can be successfully used for developing user interfaces, high technology software-intensive systems, and systems with complex algorithms and interfaces.
- It is also a very good choice to demonstrate the technical feasibility of the product.

1.4 Agile model.

a. Definition:

- Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product. Agile Methods break the product into small incremental builds. These builds are provided in iterations. Each iteration typically lasts from about one to three weeks. Every iteration involves cross functional teams working simultaneously on various areas like: Planning, Requirements Analysis, Design, Coding, Unit Testing and Acceptance Testing.
- At the end of the iteration, a working product is displayed to the customer and important stakeholders.
- Agile is based on the adaptive software development methods, whereas the traditional SDLC models like the waterfall model is based on a predictive approach. Predictive teams in the traditional SDLC models usually work with detailed planning and have a complete forecast of the exact tasks and features to be delivered in the next few months or during the product life cycle.

Figure 4: Agile SDLC Model

- Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product. Agile Methods break the product into small incremental builds. These builds are provided in iterations. Each iteration typically lasts from about one to three weeks. Every iteration involves cross functional teams working simultaneously on various areas like
    - Planning
    - Requirements Analysis
    - Design
    - Coding
    - Unit Testing and
    - Acceptance Testing.
- At the end of the iteration, a working product is displayed to the customer and important stakeholders.
- **Individuals and interactions** - In Agile development, self-organization and motivation are important, as are interactions like co-location and pair programming.
- **Working software** - Demo working software is considered the best means of communication with the customers to understand their requirements, instead of just depending on documentation.
- **Customer collaboration** - As the requirements cannot be gathered completely in the beginning of the project due to various factors, continuous customer interaction is very important to get proper product requirements.
- **Responding to change** - Agile Development is focused on quick responses to change and continuous development.

b. Advantages:

- Customer satisfaction by rapid, continuous delivery of useful software.
- People and interactions are emphasized rather than process and tools. Customers, developers and testers constantly interact with each other.
- Working software is delivered frequently (weeks rather than months).
- A face-to-face conversation is the best form of communication.
- Close, daily cooperation between business people and developers.
- Continuous attention to technical excellence and good design.
- Regular adaptation to changing circumstances.
- Even late changes in requirements are welcomed.
- Functional criteria corrections are introduced in the production phase to ensure competition.
- The project is split into quick and transparent iterations.
- Quick introduction of the first edition of the software.

c. Dis-Advantages:

- In case of some software deliverables, especially the large ones, it is difficult to assess the effort required at the beginning of the software development life cycle.
- There is lack of emphasis on necessary designing and documentation.
- The project can easily get taken off track if the customer representative is not clear what final outcome that they want.
- Only senior programmers are capable of taking the kind of decisions required during the development process. Hence it has no place for newbie programmers unless combined with experienced resources.
- Difficulties in calculating the ultimate expense attributable to permanent adjustments.
- The staff should be extremely competent and customer-oriented.
- New specifications could be in tension with the current architecture.

d. Use cases for the Agile model:

- When frequent changes are required.
- When a highly qualified and experienced team is available.
- When a customer is ready to have a meeting with a software team all the time.
- When project size is small.

1.5 spiral model

a. Definition:

- Spiral Model – is an SDLC model that combines architecture and stage prototyping. It is a combination of the Iterative and Waterfall SDLC models with a significant emphasis on risk analysis. The key problem of the spiral model is the idea of the correct moment to take a turn into the next point. Preliminary timeframes are proposed as a solution to this problem. The change to the next stage is done according to the schedule, even though the work on the previous stage has not yet been done.

The plan is applied on the basis of comparative evidence received during the previous projects and the expertise of the personal developer.

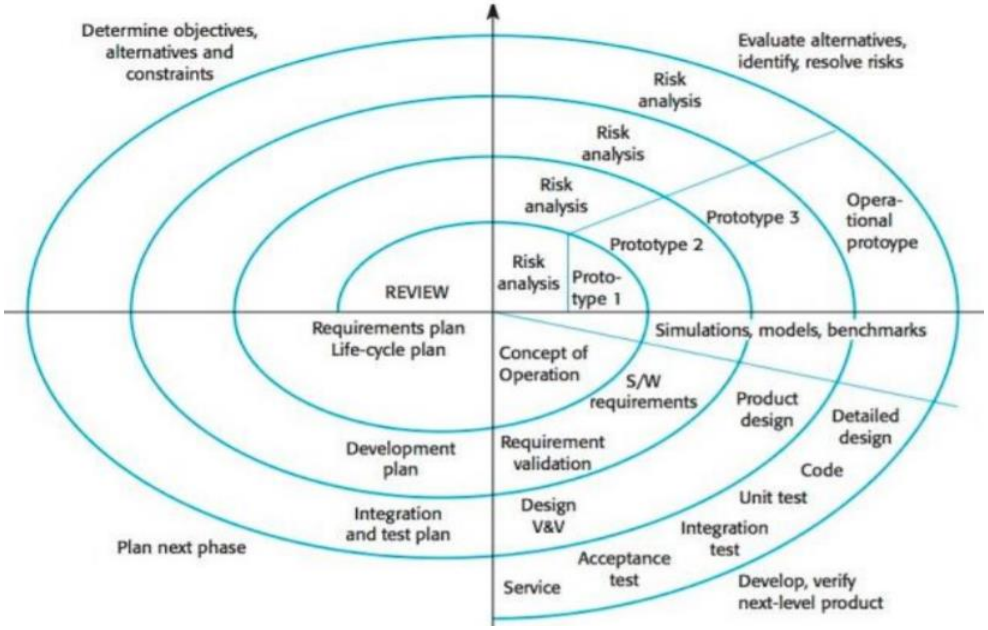

Figure 5: Spiral SDLC Model

- **Identification**
  - This phase starts with gathering the business requirements in the baseline spiral. In the subsequent spirals as the product matures, identification of system requirements, subsystem requirements and unit requirements are all done in this phase.
  - This phase also includes understanding the system requirements by continuous communication between the customer and the system analyst. At the end of the spiral, the product is deployed in the identified market.
- **Design:** The Design phase starts with the conceptual design in the baseline spiral and involves architectural design, logical design of modules, physical product design and the final design in the subsequent spirals.
- **Construct or Build**
  - The Construct phase refers to production of the actual software product at every spiral. In the baseline spiral, when the product is just thought of and the design is being developed a POC (Proof of Concept) is developed in this phase to get customer feedback.
  - Then in the subsequent spirals with higher clarity on requirements and design details a working model of the software called build is produced with a version number. These builds are sent to the customer for feedback.
- **Evaluation and Risk Analysis**

- Risk Analysis includes identifying, estimating and monitoring the technical feasibility and management risks, such as schedule slippage and cost overrun. After testing the build, at the end of first iteration, the customer evaluates the software and provides feedback.
- Based on the customer evaluation, the software development process enters the next iteration and subsequently follows the linear approach to implement the feedback suggested by the customer. The process of iterations along the spiral continues throughout the life of the software.

b. Advantages:

- Additional functionality or changes can be done at a later stage
- Cost estimation becomes easy as the prototype building is done in small fragments
- Continuous or repeated development helps in risk management
- Development is fast and features are added in a systematic way in Spiral development
- There is always a space for customer feedback

c. Dis-Advantages:

- Risk of not meeting the schedule or budget
- Spiral development works best for large projects only also demands risk assessment expertise
- For its smooth operation spiral model protocol needs to be followed strictly
- Documentation is more as it has intermediate phases
- Spiral software development is not advisable for smaller project, it might cost them a lot

d. Use cases for the spiral model:

- The customer is not sure of the requirements.
- Major changes are expected during the development cycle
- Mid-or high-risk projects where it is important to avoid such risks
- The new product is expected to be released in a few stages to have enough feedback from customers

2. I chose the Iterative SDLC Model for this project because:

Tune Source can be suitable for the waterfall model because it is simple and easy to understand and use, it is easy to manage due to the rigidity of the model - each stage has specific distributions and review process. In this model, the stages are processed and completed at a time. The stages do not overlap. Suitable for small and medium projects and projects that have not changed much. Waterfall projects are divided into discrete and easily understandable phase. As a result, project management is straightforward and the process is easily understandable even to non-developers. Clients can know in advance the cost and timeline of the project so they can plan their business activities and manage cash flow according to the plan. Because waterfall projects are simple, it's much easier to measure your progress by quickly looking at a Gantt chart.

II. Explain how risk is managed in the Spiral lifecycle model.

1. You will have the present what is Risk Management process with clear illustrations and explanations.

- Project risk management is the process of identifying, analyzing and then responding to any risk that arises over the life cycle of a project to help the project remain on track and meet its goal. Risk management isn't reactive only; it should be part of the planning process to figure out risk that might happen in the project and how to control that risk if it in fact occurs.
- A risk is anything that could potentially impact your project's timeline, performance or budget. Risks are potentialities, and in a project management context, if they become realities, they then become classified as "issues" that must be addressed. So risk management, then, is the process of identifying, categorizing, prioritizing and planning for risks before they become issues.
- Risk management can mean different things on different types of projects. On large-scale projects, risk management strategies might include extensive detailed planning for each risk to ensure mitigation strategies are in place if issues arise. For smaller projects, risk management might mean a simple, prioritized list of high, medium and low priority risks.

Figure 6: Risk Management process

2. Risk Management Plan to manage risks of Tune Source project.

- Evolutionary processes that disregard risks from scalability problems include the scope of this risk, as well as growing investment in an engineering architecture process that needs to be revamped or replaced. But that's necessary for the future production of goods. Since a time of quick change is the framework of the spiral

model, it is easy for the organization to build random risks. Simple development of unexpected problems.

- Tune Source needs to provide a contingency fund for the risk analysis phase and take timely action for unexpected incidents. At the same time, predictions of problems that may occur unexpectedly in time to address these events and have appropriate solutions (the solution for this is wise allocation of finance and at the very beginning placed the project in tight frames.)

3. Risk is managed in the Spiral lifecycle model.

- For any project activity (e.g., requirements analysis, design, prototyping, testing), the project team must decide how much effort is enough. In authentic spiral process cycles, these decisions are made by minimizing overall risk.
- "Hazardous spiral look-alikes" that violate this invariant include evolutionary processes that ignore risk due to scalability issues, and incremental processes that invest heavily in a technical architecture that must be redesigned or replaced to accommodate future increments of the product.
- Since risk monitoring requires additional resources, this model can be pretty costly to use. Each spiral requires specific expertise, which makes the management process more complex. That's why this SDLC model is not suitable for small projects
- A large number of intermediate stages. As a result, a vast amount of documentation
- Time management may be difficult. Usually, the end date of a project is not known in the first stages

4. Risk handling in Spiral Model

- The danger is any adverse circumstance that may hinder the successful execution of a software project. The most critical aspect of the spiral model is the management of these unforeseen threats after the launch of the project
- Such hazard resolution is made easier by designing a prototype. The spiral model promotes risk assessment by having the scope to create a prototype at any level of software development.
- The prototyping model still supports risk management, but before the start of the project development work, risks must be thoroughly established. But after construction work starts, real-world project risk will occur, in which case we can't use the Prototyping Model.
- The properties of the product are dated and evaluated in each phase of the Spiral Model, and the threats are identified and addressed by prototyping at that time. This model is also more robust than other versions of SDLC as well.

5. There is a chance of error and it is possible to interrupt the project.

- The system is very complex, so it is difficult to run and prone to errors, so the project might not be finished early due to error correction, because the software specifications are complex and wide and may increase customer demand.
- The solution to this problem is to be careful in each stage of the model that minimizes errors during implementation.

6. Risk handling in project.

a. Create a risk management plan

- The risk management strategy should describe project methodology, project risk tolerance, to identify and prioritize risk. It takes time and effort to create such a plan, but engaging in the planning process also pays off by developing a roadmap that will direct our team during the project execution phase.

b. Keep up to date on the project risk registry.

- The risk register, which can be paired with either our risk management plan or a separate document, is a list of all potential risk incidents that may have an impact on the project. Having this paper would allow us to stay on top of potential problems, but keeping the project updated is crucial so that we still have an accurate snapshot to refer to.

c. Gantt chart for Risk Management Plan.

- In order to prevent risk from becoming a problem, use the Gantt chart to create systematic risk management strategies. Schedule, assign and track the project's activities with optimum visibility. Participants in the crew will also add comments and files to their assigned assignments to ensure that all interaction takes place in real-time at the project level.

Chapter 2

EXPLAIN THE IMPORTANCE OF A FEASIBILITY STUDY

III. Explain the purpose of a feasibility report.

1. The Project Scope.

- The market question and/or opportunity to be addressed is used to describe it. The reach should be definitive and to the point; no intent serves a rambling narrative and can potentially confuse project participants. The sections of the company affected, either directly or indirectly, including project participants and end-user areas affected by the project, also need to be identified.

- In the business world, there are so many ventures that have begun without a well-defined project scope. As a result, projects have drifted in and out of their bounds, allowing them to generate much too much or far too little than what is really required.

2. What Is a Feasibility Study?

A feasibility study is an investigation that takes into account all pertinent aspects of the project-including fiscal, technological, legal, and scheduling considerations-to assess the probability of successful completion of the project. Project managers use feasibility analyses to determine the pros and disadvantages of conducting a project before spending a lot of time and resources on it.

3. The Importance of Feasibility Studies.

For business development, feasibility studies are important. They can make it easy for a business to address where it will operate and how it can work. They can also identify potential challenges that may hinder their operations and realize the amount of funds needed to keep the business up and running. Feasibility studies are targeted at communication tactics that may help to reassure customers or banks that investing in a specific project or company is a good decision.

4. Types of Feasibility Study.

- A feasibility review measures the potential for profitability of the project; thus, assumed objectivity is a crucial factor in the reliability of the report for potential borrowers and lending institutions. There are five types of feasibility studies-separate fields that the feasibility study explores, as listed below.
    - **Technical Feasibility**

      The assessment relies on the organization's available technical resources. It enables companies to determine whether the potential is fulfilled by technological capabilities and whether the technical staff is capable of translating ideas into working processes. Technical feasibility also requires the assessment of the proposed device's hardware, software, and other technical requirements.
    - **Operating viability**

      This evaluation includes conducting an analysis to evaluate and decide whether-and how well-the needs of the company will be fulfilled by finishing the project. Analyses of operational feasibility often evaluate how the project proposal meets the requirements specified in the device development design review process.
    - **Scheduling Feasibility**

      This assessment is the most important for the project's progress; after all, if it is not completed on schedule, the project will fail. The organization estimates how long the plan will take to complete while preparing the feasibility.

- **Legal Feasibility**

  This review examines whether some aspect of the planned project is in conflict with legal standards such as zoning regulations, data privacy legislation, or social media legislation. Let's say the organization wants to construct a new office building in a certain area. A feasibility study could show that the company's ideal position is not for that type of business. The organization has only saved a tremendous amount of time and money by learning that their proposal was not viable right from the outset.

- **Economic Feasibility**

  This assessment usually requires a cost/benefit analysis of the project, helping organizations decide the feasibility, cost, and benefits of a project before allocating financial resources. It often serves as an impartial project assessment and improves the credibility of the project, encouraging decision-makers to consider the positive economic benefits of the company.

IV. Describe how technical solutions can be compared.

1. Three feasibility criteria (technical, economic, organisational) are applied to the project.

a. Technical.

- The analyst must find out whether it is feasible to introduce the new approach, given current technical resources. If not, is it possible to update or add it to the system in a manner that meets the request under consideration? If existing technologies cannot be implemented or upgraded, the next question is whether there is technology in existence that meets the requirements.
- A large part of the resource determination has to do with the evaluation of technical feasibility. It considers the technical requirements of the project being proposed. Then, the technical requirements are compared to the organization's technical capability. If the internal technical capacity is sufficient to support the project requirements, the system project is considered technically feasible.
- In technical feasibility the following issues are taken into consideration.
  - Whether the required technology is available or not
  - Whether the required resources are available - Manpower- programmers, testers & debuggers - Software and hardware
  - Is the proposed technology or solution practical?
  - Do we currently possess the necessary technology?
  - Do we possess the necessary technical expertise and is the schedule reasonable for this team?

  - Is relevant technology mature enough to be easily applied to our problem?
  - What kinds of technology will we need?
  - Is the required technology available “in house”?
- Once the technical feasibility is established, it is important to consider the monetary factors also. Since it might happen that developing a particular system may be technically possible but it may require huge investments and benefits may be less. For evaluating this, economic feasibility of the proposed system is carried out.

b. Economic.

- Economic viability is the second component of resource determination. The basic resources to consider are your time and that of the system testing staff, the cost of performing a full system study, the cost of the company worker's time, the estimated hardware cost, and the estimated cost of developing software or software.
- The organization concerned must be able to see the value of the investment it is pondering before committing to an entire-systems review. The system is not economically feasible and no further progress can be made on the project if short-term losses are not overshadowed by long-term benefits or do not result in an immediate reduction in operating costs.
- For any system if the expected benefits equal or exceed the expected costs, the system can be judged to be economically feasible. In economic feasibility, cost benefit analysis is done in which expected costs and benefits are evaluated. Economic analysis is used for evaluating the effectiveness of the proposed system.
- In economic feasibility, the most important is cost-benefit analysis. As the name suggests, it is an analysis of the costs to be incurred in the system and benefits derivable out of the system. Click on the link below which will get you to the page that explains what cost benefit analysis is and how you can perform a cost benefit analysis.

c. Organisational.

- For a moment, imagine that all technological and economic capital are deemed sufficient. The operational viability of the requested project must also be considered by the systems analyst. Operational viability relies on the human capital available for the project and includes planning whether, once built, the device can work and be used. Resistance to introducing the new system would be high if users are practically married to the current system, see no issues with it, and usually are not interested in demanding a new system. There are poor chances of it ever being operational.
- Organizational feasibility analysis is conducted to determine whether a proposed business has sufficient management expertise, organizational competence, and resources to successfully launch its business.
- It describes the corporate and legal framework of a corporation. It will also provide information on the educational history of the owners, the principals of the firm, and

the expertise they would apply to the company's functions. An overview of organizational viability should include:

  - The business structure's description
  - The organizational structure's description
  - The internal and external policies and values of the company
  - Resumes and professional skills.

- Management prowess: A proposed company may determine its original management team's prowess or ability, whether it is a single entrepreneur or a wider community. In their self-assessments, this challenge allows the persons launching the business to be truthful and candid. The solo entrepreneur or management team's passion for the business idea and the degree to which the management team or solo entrepreneur knows the markets in which the organization can compete are two of the most critical factors in this field. There are no feasible solutions to benefits in these sectors.
- Resource sufficiency: The second field of corporate viability research is to assess if the planned venture has adequate resources to go forward, or is capable of acquiring them. The aim is to recognize the most relevant non-financial tools and determine if they are available. A start-up that would need workers with advanced abilities is an example. Another primary resource sufficiency problem is the need to defend key facets of the organization from intellectual property.

2. Feasibility of the project.

- Once the number of projects has been reduced following the previously discussed criteria, it is still necessary to determine if the chosen projects are feasible. As the feasibility for system projects is assessed in three main ways: operationally, scientifically, and economically, our concept of feasibility goes much deeper than the standard use of the term. The feasibility study is not a study of full-blown systems. Instead, the feasibility study is used to obtain large data for management members, which in turn helps them to determine whether to continue with a system study.
- Budget Analysis: An assessment of the project's cash flow and resource allocation estimations.
- Economic Modeling: A short- and long-term forecast of the project's ability to create economic value for the organization and its stakeholders.
- Financial Analysis: A comprehensive financial breakdown of the project's anticipated funding and revenue generation opportunities (including rates of return, capital investments, payback periods, etc.) over the lifespan of the project.
- Citizen Impact Analysis: For government agencies, we conduct a thorough study of the projected impacts of the project on the physical, social, cultural, economic and political environments.
- Market Research: Deep-dive research and analysis for specific industries and verticals relevant to the project and the proposed project plan.

- Needs Assessment: A study of the true need for the project, the beneficiaries of the project and an evaluation of how organizational needs will be addressed by the project, or if the needs would be better addressed by an alternative approach.
- The Technical Feasibility Analysis takes into account the organization's current state, as well as existing resource capabilities and the availability/sustainability of new technological offerings.
- The Economic Feasibility Analysis examines the long-term revenue generation potential of the project, including both tangible and intangible elements. Additional analyses - including cultural feasibility, social feasibility, political feasibility, and environmental feasibility - may be deployed depending on client needs and objectives.

3. Technical solutions using the alternative matrix.

- An Alternatives Evaluation Matrix can be used to compare alternatives for numerous requirements including hardware, software, databases, operating systems, or languages.
- A Weighted Alternatives Evaluation Matrix, or Weighted Matrix, assigns weighting factors to criteria when comparing alternatives.
- Before constructing an Alternatives Evaluation Matrix, identify the evaluation criteria and a list of alternatives to be considered and evaluated.
- The project alternative is sometimes not a suitable substitution for the original preference, but it may often provide similar alternatives, albeit with a different mix and interpretation. The essence of the alternative to the project makes the two alternatives mutually exclusive. Meanwhile, a combination of two or more alternatives to the project produces a single alternative project.

4. The alternative matrix

Criteria	**Weight**	**Alternative 1**	**Alternative 2**	**Alternative 3**	**Alternative 4**
Cost	.30	5	3	5	2
Response Time	.17	3	4	5	3
Training Time	.17	5	5	3	3
Ease of use	.17	4	3	5	4
Strong Team	.10	3	4	3	4
Team Experience	.10	5	5	5	5
Total	1.0	25	24	26	21
Weighted Total		4.3	3.8	4.5	3.2

References

Alagar, V.S. and Periyasamy, K., 2011. *Specification of software systems*. Springer Science & Business Media.

Cervone, H.F., 2006. Project risk management. *OCLC Systems & Services: International digital library perspectives*.

Al-Bahar, J.F. and Crandall, K.C., 1990. Systematic risk management approach for construction projects. *Journal of construction engineering and management*, *116*(3), pp.533-546.

Fan, M., Lin, N.P. and Sheu, C., 2008. Choosing a project risk-handling strategy: An analytical model. *International Journal of Production Economics*, *112*(2), pp.700-713.

Fleming, W., 2020. 7.7 Feasibility Reports. *Technical Writing at LBCC*.

Ruparelia, N.B., 2010. Software development lifecycle models. *ACM SIGSOFT Software Engineering Notes*, *35*(3), pp.8-13.

Al-Sebie, M. and Irani, Z., 2005. Technical and organisational challenges facing transactional e-government systems: an empirical study. *Electronic Government, an International Journal*, *2*(3), pp.247-276.