**PRACTICAL: -1**

**AIM:**

**List & draw at least 7 various Software Development Life Cycle (SDLC) models and preparing the detailed case study on “VLC Media Player” that which SDLC model is suitable to develop the “VLC Media Player” desktop application?**

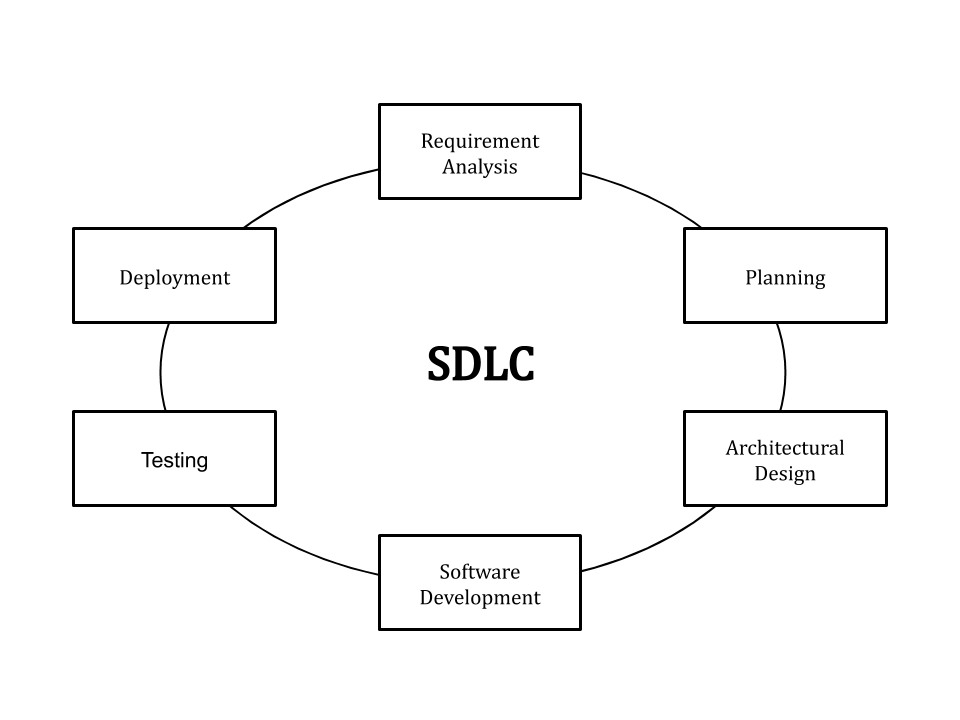
**THEORY:**

SDLC:

Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high quality softwares.  It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.

There are different software development life cycle models specify and design, which are followed during the software development phase. These models are also called "****Software Development Process Models****." Each process model follows a series of phase unique to its type to ensure success in the step of software development.

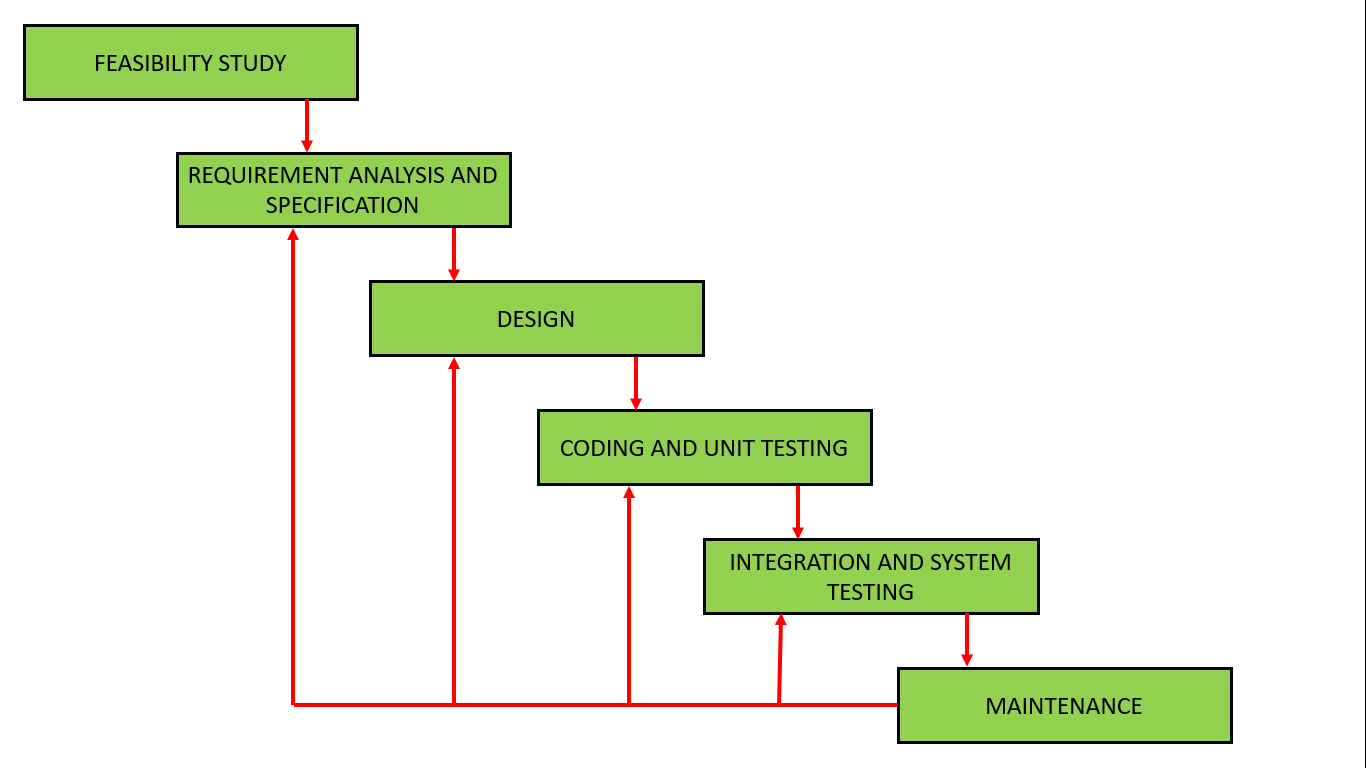
It is a process that produces software with the highest quality and lowest cost in the shortest time possible. SDLC provides a well-structured flow of phases that help an organization to quickly produce high-quality software which is well-tested and ready for production use.



1. **Iterative Model**

Iterative development is a way of breaking down the software development of a large application into smaller pieces. An iterative life cycle model does not attempt to start with a full specification of requirements. Instead, development begins by specifying and implementing just part of the software, which is then reviewed to identify further requirements.

**The iterative waterfall model provides feedback paths from every phase to its preceding phases, which is the main difference from the classical waterfall model.** Feedback paths introduced by the iterative waterfall model are shown in the figure below.



When errors are detected at some later phase, these feedback paths allow correcting errors committed by programmers during some phase. The feedback paths allow the phase to be reworked in which errors are committed and these changes are reflected in the later phases. But there is no feedback path to the stage – feasibility study, because once a project has been taken, does not give up the project easily. It is good to detect errors in the same phase in which they are committed. It reduces the effort and time required to correct the errors.

**Advantages And Disadvantages:**

**Advantages:**

* Some working functionality can be developed quickly and early in the life cycle.
* Results are obtained early and periodically.
* Parallel development can be planned.
* Progress can be measured.
* Less costly to change the scope/requirements.
* Testing and debugging during smaller iteration is easy.
* Risks are identified and resolved during iteration; and each iteration is an easily managed milestone.
* Easier to manage risk - High risk part is done first.
* With every increment, operational product is delivered.
* Issues, challenges and risks identified from each increment can be utilized/applied to the next increment.
* Risk analysis is better.
* It supports changing requirements.
* Initial Operating time is less.
* Better suited for large and mission-critical projects.
* During the life cycle, software is produced early which facilitates customer evaluation and feedback.

**Disadvantages:**

* More resources may be required.
* Although cost of change is lesser, but it is not very suitable for changing requirements.
* More management attention is required.
* System architecture or design issues may arise because not all requirements are gathered in the beginning of the entire life cycle.
* Defining increments may require definition of the complete system.
* Not suitable for smaller projects.
* Management complexity is more.
* End of project may not be known which is a risk.
* Highly skilled resources are required for risk analysis.
* Projects progress is highly dependent upon the risk analysis phase.

Applications of Iterative Model:

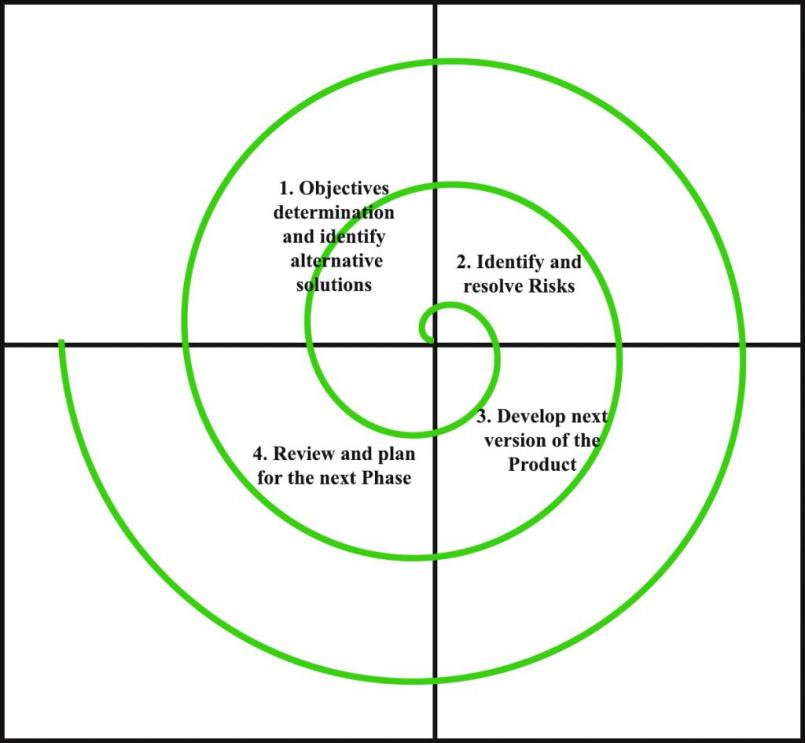
Like other SDLC models, Iterative and incremental development has some specific applications in the software industry. This model is most often used in the following scenarios.

* Major requirements must be defined; however, some functionalities or requested enhancements may evolve with time.
* Requirements of the complete system are clearly defined and understood.
* There is a time to the market constraint.
* A new technology is being used and is being learnt by the development team while working on the project.
* Resources with needed skill sets are not available and are planned to be used on contract basis for specific iterations.
* There are some high-risk features and goals which may change in the future.
* Potential defects are spotted and dealt with early.
* Functional prototypes are developed early in the project life cycle.
* Less time is spent on documenting and more on designing.
* Progress is easily measured.

1. **Spiral Model**

**Spiral model** is one of the most important Software Development Life Cycle models, which provides support for **Risk Handling**. In its diagrammatic representation, it looks like a spiral with many loops. The exact number of loops of the spiral is unknown and can vary from project to project. Each loop of the spiral is called a **Phase of the software development process.**

The exact number of phases needed to develop the product can be varied by the project manager depending upon the project risks. As the project manager dynamically determines the number of phases, so the project manager has an important role to develop a product using the spiral model. The Radius of the spiral at any point represents the expenses(cost) of the project so far, and the angular dimension represents the progress made so far in the current phase. **The below diagram shows the different phases of the Spiral Model:**



Each phase of the Spiral Model is divided into four quadrants as shown in the above figure. The functions of these four quadrants are discussed below.

* **Objectives determination and identify alternative solutions:** Requirements are gathered from the customers and the objectives are identified, elaborated, and analyzed at the start of every phase. Then alternative solutions possible for the phase are proposed in this quadrant.
* **Identify and resolve Risks:** During the second quadrant, all the possible solutions are evaluated to select the best possible solution. Then the risks associated with that solution are identified and the risks are resolved using the best possible strategy. At the end of this quadrant, the Prototype is built for the best possible solution.
* **Develop next version of the Product:** During the third quadrant, the identified features are developed and verified through testing. At the end of the third quadrant, the next version of the software is available.
* **Review and plan for the next Phase:** In the fourth quadrant, the Customers evaluate the so far developed version of the software. In the end, planning for the next phase is started.

**Advantages and Disadvantages:**

Advantages:

* Changing requirements can be accommodated.
* Allows extensive use of prototypes.
* Requirements can be captured more accurately.
* Users see the system early.
* Development can be divided into smaller parts and the risky parts can be developed earlier which helps in better risk management.

Disadvantages:

* Management is more complex.
* End of the project may not be known early.
* Not suitable for small or low risk projects and could be expensive for small projects.
* Process is complex
* Spiral may go on indefinitely.
* Large number of intermediate stages requires excessive documentation.

Applications of spiral Model:

The following points explain the typical uses of a Spiral Model −

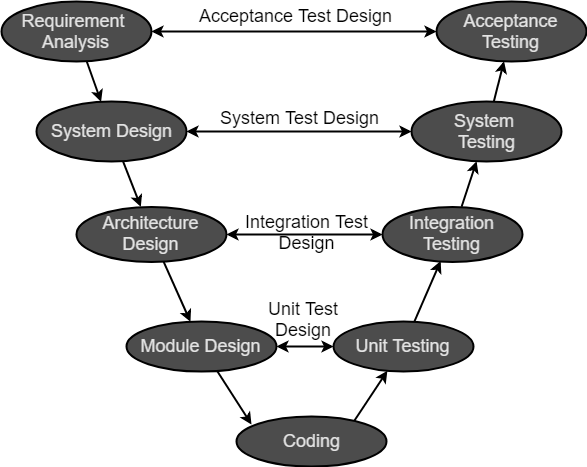
* When there is a budget constraint and risk evaluation is important.
* For medium to high-risk projects.
* Long-term project commitment because of potential changes to economic priorities as the requirements change with time.
* Customer is not sure of their requirements which is usually the case.
* Requirements are complex and need evaluation to get clarity.
* New product line which should be released in phases to get enough customer feedback.
* Significant changes are expected in the product during the development cycle.

1. **V-MODEL**

The V-model is a type of SDLC model where process executes in a sequential manner in V-shape. It is also known as Verification and Validation model. It is based on the association of a testing phase for each corresponding development stage. Development of each step directly associated with the testing phase. The next phase starts only after completion of the previous phase i.e. for each development activity, there is a testing activity corresponding to it.

Verification****:**** It involves static analysis technique (review) done without executing code. It is the process of evaluation of the product development phase to find whether specified requirements meet.

Validation****:**** It involves dynamic analysis technique (functional, non-functional), testing done by executing code. Validation is the process to evaluate the software after the completion of the development phase to determine whether software meets the customer expectations and requirements.



So V-Model contains Verification phases on one side of the Validation phases on the other side. Verification and Validation phases are joined by coding phase in V-shape. Thus, it is called V-Model.

Advantages and Disadvantages:

**Advantages:**

* This is a highly disciplined model and Phases are completed one at a time.
* V-Model is used for small projects where project requirements are clear.
* Simple and easy to understand and use.
* This model focuses on verification and validation activities early in the life cycle thereby enhancing the probability of building an error-free and good quality product.
* It enables project management to track progress accurately.

**Disadvantages:**

* High risk and uncertainty.
* It is not a good for complex and object-oriented projects.
* It is not suitable for projects where requirements are not clear and contains high risk of changing.
* This model does not support iteration of phases.
* It does not easily handle concurrent events.

Applications of V-model:

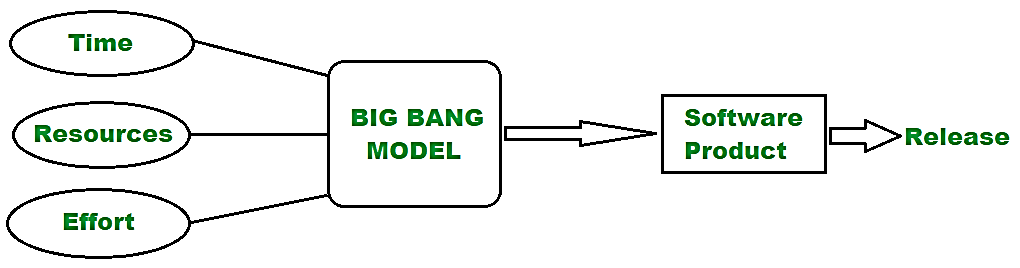
The following pointers are some of the most suitable scenarios to use the V-Model application.

* Requirements are well defined, clearly documented and fixed.
* Product definition is stable.
* Technology is not dynamic and is well understood by the project team.
* There are no ambiguous or undefined requirements.
* The project is short.

1. **BIG BANG MODEL**

The Big Bang model is an SDLC model where we do not follow any specific process. The development just starts with the required money and efforts as the input, and the output is the software developed which may or may not be as per customer requirement. This Big Bang Model does not follow a process/procedure and there is a very little planning required. Even the customer is not sure about what exactly he wants and the requirements are implemented on the fly without much analysis.

Usually, this model is followed for small projects where the development teams are very small.



Advantages and Disadvantages:

**Advantages:**

* This is a very simple model
* Little or no planning required
* Easy to manage
* Very few resources required
* Gives flexibility to developers
* It is a good learning aid for new comers or students.

**Disadvantages:**

* Very High risk and uncertainty.
* Not a good model for complex and object-oriented projects.
* Poor model for long and ongoing projects.
* Can turn out to be very expensive if requirements are misunderstood.

Applications of big bang model:

It is recommended to go for the Big Bang model only due to the following cases i.e.

* Developing a project for learning purposes or experiment purposes.
* No clarity on the requirements from the user side.
* When newer requirements need to be implemented immediately.
* Changing requirements based on the current developing product outcome.
* No strict guideline on product release or delivery date.

1. **AGILE 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.



Following are the Agile Manifesto principles −

* **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.

**Advantages and Disadvantages**

**Advantages:**

* Is a very realistic approach to software development.
* Promotes teamwork and cross training.
* Functionality can be developed rapidly and demonstrated.
* Resource requirements are minimum.
* Suitable for fixed or changing requirements
* Delivers early partial working solutions.
* Good model for environments that change steadily.
* Minimal rules, documentation easily employed.
* Enables concurrent development and delivery within an overall planned context.
* Little or no planning required.
* Easy to manage.
* Gives flexibility to developers.

**Disadvantages:**

* Not suitable for handling complex dependencies.
* More risk of sustainability, maintainability and extensibility.
* An overall plan, an agile leader and agile PM practice is a must without which it will not work.
* Strict delivery management dictates the scope, functionality to be delivered, and adjustments to meet the deadlines.
* Depends heavily on customer interaction, so if customer is not clear, team can be driven in the wrong direction.
* There is a very high individual dependency, since there is minimum documentation generated.
* Transfer of technology to new team members may be quite challenging due to lack of documentation

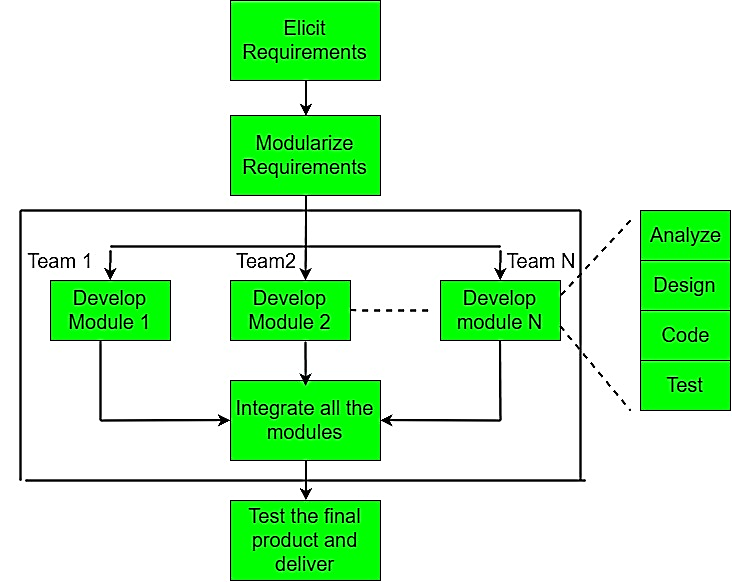
1. **RAD MODEL**

The **RAD (Rapid Application Development)** model is based on prototyping and iterative development with no specific planning involved. The process of writing the software itself involves the planning required for developing the product.

Rapid Application Development focuses on gathering customer requirements through workshops or focus groups, early testing of the prototypes by the customer using iterative concept, reuse of the existing prototypes (components), continuous integration and rapid delivery.

In the RAD model, the functional modules are developed in parallel as prototypes and are integrated to make the complete product for faster product delivery. Since there is no detailed preplanning, it makes it easier to incorporate the changes within the development process.

The most important aspect for this model to be successful is to make sure that the prototypes developed are reusable.



Advantages And Disadvantages:

**Advantages:**

* Changing requirements can be accommodated.
* Progress can be measured.
* Iteration time can be short with use of powerful RAD tools.
* Productivity with fewer people in a short time.
* Reduced development time.
* Increases reusability of components.
* Quick initial reviews occur.
* Encourages customer feedback.
* Integration from very beginning solves a lot of integration issues.

**Disadvantages:**

* Dependency on technically strong team members for identifying business requirements.
* Only system that can be modularized can be built using RAD.
* Requires highly skilled developers/designers.
* High dependency on Modelling skills.
* Inapplicable to cheaper projects as cost of Modelling and automated code generation is very high.
* Management complexity is more.
* Suitable for systems that are component based and scalable.
* Requires user involvement throughout the life cycle.
* Suitable for project requiring shorter development times.

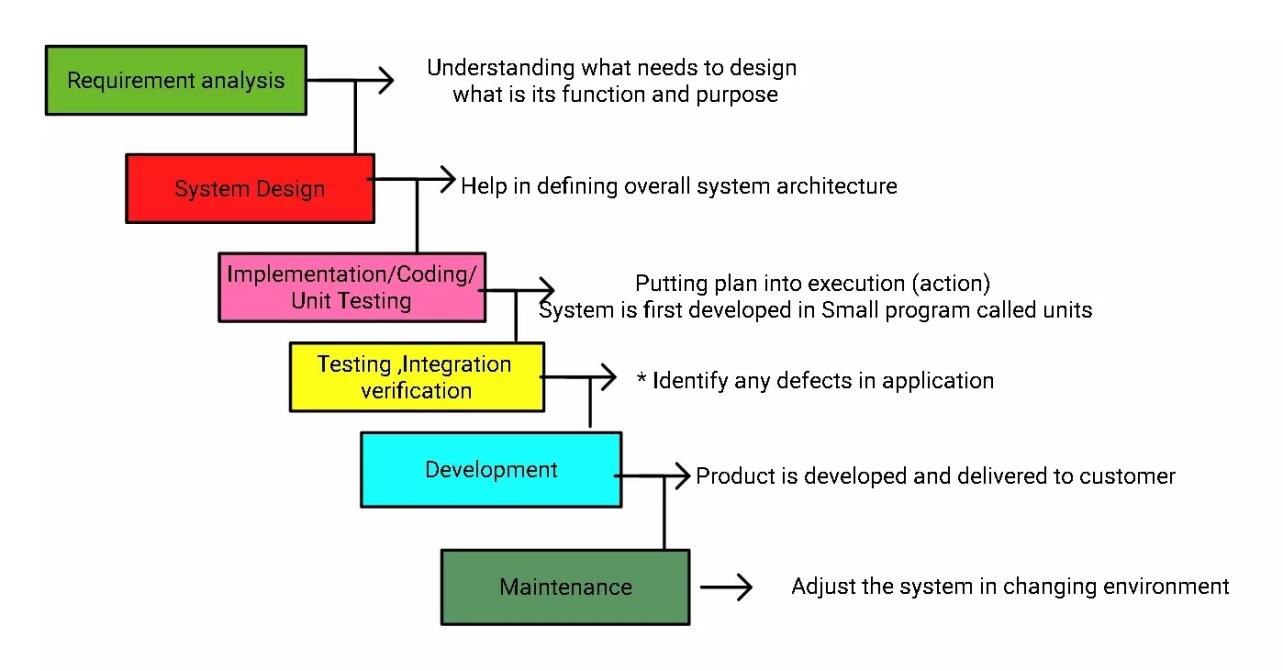
Applications of RAD model:

The following pointers describe the typical scenarios where RAD can be used

* RAD should be used only when a system can be modularized to be delivered in an incremental manner.
* It should be used if there is a high availability of designers for Modelling.
* It should be used only if the budget permits use of automated code generating tools.
* RAD SDLC model should be chosen only if domain experts are available with relevant business knowledge.
* Should be used where the requirements change during the project and working prototypes are to be presented to customer in small iterations of 2-3 months.

1. WATERFALL MODEL:

Waterfall model is the basic ****software development life cycle**** model. In this model, the whole process of software development is divided into separate phases. The outcome of one phase acts as the input for the next phase sequentially.



The sequential phases in Waterfall model are −

* **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 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.

Advantages And Disadvantages:

**Advantages:**

* Simple and easy to understand and use
* Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
* Phases are processed and completed one at a time.
* Works well for smaller projects where requirements are very well understood.
* Clearly defined stages.
* Well understood milestones.
* Easy to arrange tasks.
* Process and results are well documented.

**Disadvantages:**

* No working software is produced until late during the life cycle.
* High amounts of risk and uncertainty.
* Not a good model for complex and object-oriented projects.
* Poor model for long and ongoing projects.
* Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
* It is difficult to measure progress within stages.
* Cannot accommodate changing requirements.
* Adjusting scope during the life cycle can end a project.
* Integration is done as a "big-bang. at the very end, which doesn't allow identifying any technological or business bottleneck or challenges early.

Applications of waterfall Model

Some situations where the use of Waterfall model is most appropriate are −

* Requirements are very well documented, clear and fixed.
* Product definition is stable.
* Technology is understood and is not dynamic.
* There are no ambiguous requirements.
* Ample resources with required expertise are available to support the product.
* The project is short.

CASE STUDY ON “VLC Media Player”

Overview:

It is open source video/media player use widely with different video format compatibility. There are lots of changes and updates happened through many years. It is platform independent because it can be operated on IOS, android, Windows, Linux and MacOS. It has very interactive interface and it support all type of high resolution videos with better audio compatibility.

Suitable model for development of VLC:

As there are many models to build VLC media player, but all are not perfect to build upon. Let’s see which one is appropriate to build.

In case of VLC media player making temporary product or making prototype was not possible because this is initially developed in collaboration with other companies to make cross platform software. So, we cannot use Prototype model. Initial goal for making this software to serve as MVP (minimum Viable Product). So, there was no significance of making module by module under incremental or Agile model. Moreover, we don’t want any model in which we can no go back to any step of building and we also we can have need of changing the requirements to make it more viable according to newly coming technologies. So, locking of requirements should not be there in our suitable model.

Agile model is good to build our media player but we don’t have any time constraint to serve as we build while there is no any risk analysis and prediction in agile model. So finally Spiral model come into play. Which fulfil almost every requirement for building VLC media player.

AS we always have to consider the risk factor for VLC media player. As it can support every type of video format so sometime unauthorized video player use it to steal data of the pc or operate in malfunction. So, risk management can be done only in Spiral model. Under Digital Millennium Copyright Act of USA decryption of CSS-encrypted DVD content or unauthorized distribution of CSS decryption is illegitimate but still VLC media player extract audio or video by decrypting CSS-encrypted DVD content. So later on, many changes can be happening. So, it is very important to make it more flexible which is supported in Spiral model.

As VLC media player is not bounded by any specific developers so any one can work on the project under open-source contribution. Even low skilled software engineer too. So Spiral model support such flexibility.

CONCLUSION

From the above theory and case study, Spiral Model suits best for the development of “VLC Media Player” desktop application.

**PRACTICAL: -2**

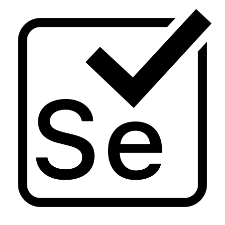
**AIM:**

**List at least 10 Agile Development tools for desktop and mobile application development in IT industry and prepare the detailed case study on “JIRA Tool” for Agile Development.**

**THEORY:**

* 1. [](https://www.atlassian.com/software/jira/free)**JIRA**JIRA is a defect tracking tool which is used for Agile testing as well as project management. This tool is not only used for recording, reporting but also integrated with code development environment.

**FEATURES:**

* JIRA Query Language helps to create quick filters with a single click
* This agile tool helps your team become more accurate and efficient
* Reporting functionality gives team critical insight into their agile process
* Extensive reporting functionality gives your team critical insight into their agile process.
* Allows creating custom workflows of any size which is helpful to build, test, and release software
  1. **SELENIUM**

Selenium is an automation agile testing tool. It aims to mimic the behaviour of a real user, and as such interacts with the HTML of the application.

**FEATURES:**

* It is a compact Object-Oriented API
* This agile tool Support for different languages like Java, Python, Ruby, Perl, PHP, and Java script
* Selenium server initializing is not required
* WebDriver finds any coordinates of any object
* It is easy tool for a WebDriver to build a keyword driven framework
  1. **ZEPHYR**

Zephyr is the #1 selling test management tool, providing end-to-end solutions for agile teams of all sizes. Get the flexibility, visibility, and insights you need to release better software FASTER.

**KEY FEATURES:**

* 1-click Integration with JIRA, Confluence, Jenkins, Bamboo, and more
* Cloud, Server, and Data Centre Deployment Options
* Advanced Analytics and DevOps Dashboards
* No Annual Commitment Required
  1. **SPRINTS**

Sprints is a tool that helps you to manage your team and product with ease. It enables you to track your progress with no hassle. This software can be used to find bottlenecks and discover ways to generate business value.

**FEATURES:**

* It is integrated with CI/CD tools.
* This tool helps you to get product feedback with ease.
* Allows you to work on any device and place.
* Enables the team to comment on code changes.
  1. **SNAGIT**

Snagit is a popular screenshot capturing tool. It provides powerful tools to edit, annotate and share screenshots. It can also be used to submit and push screenshots directly.

**FEATURES:**

* This agile testing tool offers complete features for screen capture and video recording
* Capture videos with a simple, intuitive screen recorder
* Capture a website, record an online meeting or send feedback in an email
  1. **JMETER**

Jmeter application is an open-source agile performance testing tool. It is used to load functional test behaviour and measure performance of the website.

**FEATURES:**

* Ability to load and performance test different applications/server and protocols
* Full featured Test IDE for fast Test Plan recording
* This agile tool offers complete portability and 100% Java purity
* Data analysis and visualization plugins offers great extensibility
* Functions can be used to offer dynamic input to test or provide data manipulation
* Easy Continuous Integration using third party libraries for tools like Maven, Gradle and Jenkins
  1. **APPIUM**

Appium is an open-source and free Agile tool. It is helpful for automating mobile web, iOS, and Android and hybrid applications. Native apps are those written using Android, iOS, or Windows SDKs.

**FEATURES:**

* Easy process setup process
* This best agile tool supports Safari on iOS and Chrome or the built-in 'Browser' app on Android
* It can automate Native, Hybrid, and Web mobile applications
* It supports programming languages like- Java, PHP, Ruby, Python, C#, etc.
* This agile testing tool allows native, hybrid and web application testing on physical gadgets as well as on emulator or simulator.
  1. **BACKLOG**

Backlog is an all-in-one project management tool built for developers. Agile Teams use Backlog to work with other teams for enhanced team collaboration and high-quality project delivery.

**FEATURES:**

* Easy bug tracking tool
* Project and issues with subtasks
* Git and SVN built-in
* Gantt Charts and Burndown charts
* Wikis
* Watchlists
* Native mobile apps
* Available both in cloud and on-premise
  1. **SOAP UI**

Soap UI is an agile testing tool for service-oriented architectures (SOA) and REST. Its functionality includes web service inspection, invoking, development, functional testing, and load testing.

**FEATURES:**

* It is open-source testing tool
* This agile tool offers Drag and Drop Test Creation
* It allows reusing functional test cases and security scans in just a few clicks
* It Supports Data-Driven Testing
* Multi Environment Support
* Allows service Simulation
* Static Content Mocking
  1. **USERSNAP**

Usersnap is an Agile testing tool that allows web developers to get screenshots of bugs. This tool helps testers and developers communicate bugs easily.

**FEATURES:**

* Runs on every known web browser
* This best agile tool Connects users with clients and colleagues
* Does not impede website speed
* Get visual bug reports with advanced client-side error recording
* Supports Single Page Applications

**CASE STUDY ON “AGILE DEVELOPMENT WITH JIRA”**

**Overview**:

Complaints about project management tools are an old cliché in the software development industry. They can be heard from people in many roles, in companies of all sizes, and from all countries. When listening to them, one could come to believe that, oddly, none of the more than one hundred tools existent in the market solves the problems it should.

On top of that, it's particularly interesting to note that the most popular tool in the industry, Jira Software, is notably the most infamous. Most project management tool users, both heavy and light users, have a negative opinion of Jira — weirdly, even those who have never been in direct touch with it. Several times, I have heard statements like: "I've never used Jira, but I know it sucks," and "Project management tools such as Jira are counterproductive because they slow down the team." It's almost like companies were adopting Jira just to go against the grain. Not surprisingly, though, such comments usually come from teams that follow no specific methods and processes of work and in which no one has significant experience with project management.

Agile teams use an iterative approach to break down complex projects. In the past, software development teams often failed to meet deadlines because of the sheer complexity of their work. The agile methodology for project management was developed to address the many inefficiencies present in traditional software development projects.

The agile methodology is based on four basic tenets:

* Individuals and interactions over processes and tools
* Working software over comprehensive documentation
* Customer collaboration over contract negotiation
* Responding to change over following a plan

Agile project managers embrace change. Since working software is the primary measure of progress for agile teams, the trajectory of the project is bound to change as feedback from clients, customers, and team members are integrated with each new release. While this might sound like it would slow the whole process down, it actually makes teams nimbler and ensures higher code quality.

Jira has adopted agile principles into every facet of their software. Agile teams can quickly and easily navigate charts, long-term goals (epics), and tasks. Tracking issues and spotting bugs can also be done automatically by Jira Software.

**CONCLUSION:** Jira is a powerful project management system with planning, tracking, releasing, and reporting all in one place. Teams are able to access information about tasks, productivity, bugs, and code quality in one location.

**PRACTICAL: – 3**

**AIM**:

**List at least 5 software development planning tools and prepare the detailed case study**

**of** **Risk Analysis & Management (i.e. Risk Identification, Risk Projection, Risk Refinement, Risk Mitigation.) on “VLC Media Player” mobile application**

**THEORY**

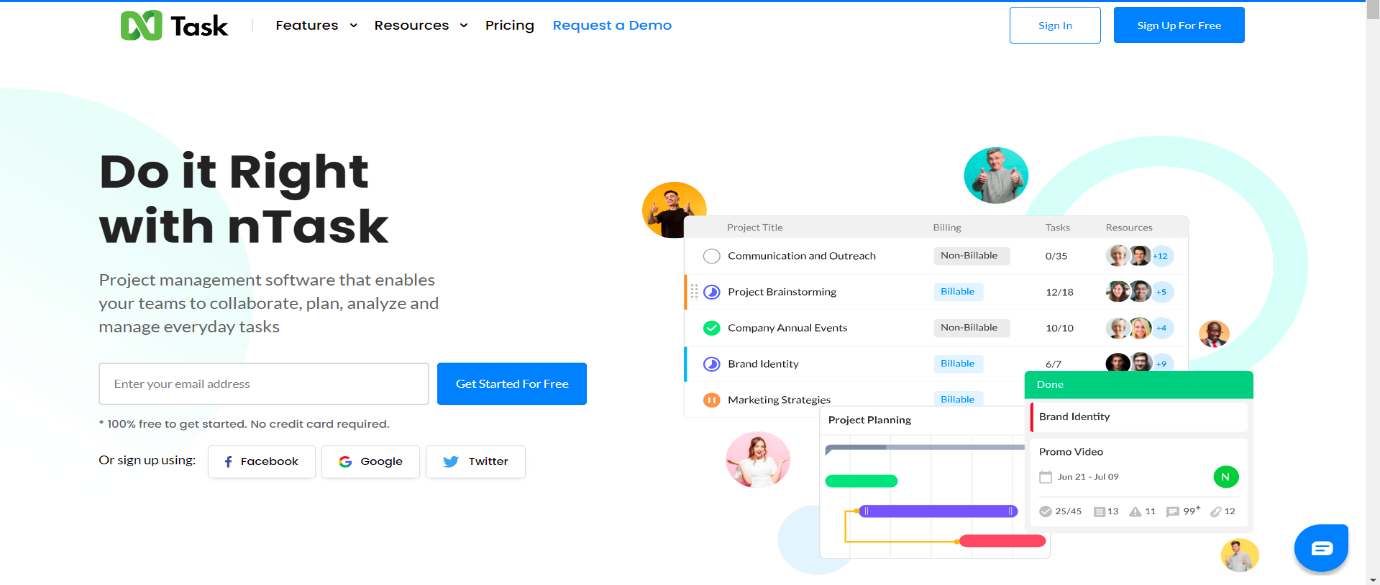
**Need of Project Planning:**

Software development is a sort of all new streams in world business, and there's next to no involvement in structure programming items.

Most programming items are customized to accommodate customer's necessities. The most significant is that the underlying technology changes and advances so generally and rapidly that experience of one element may not be connected to the other one.

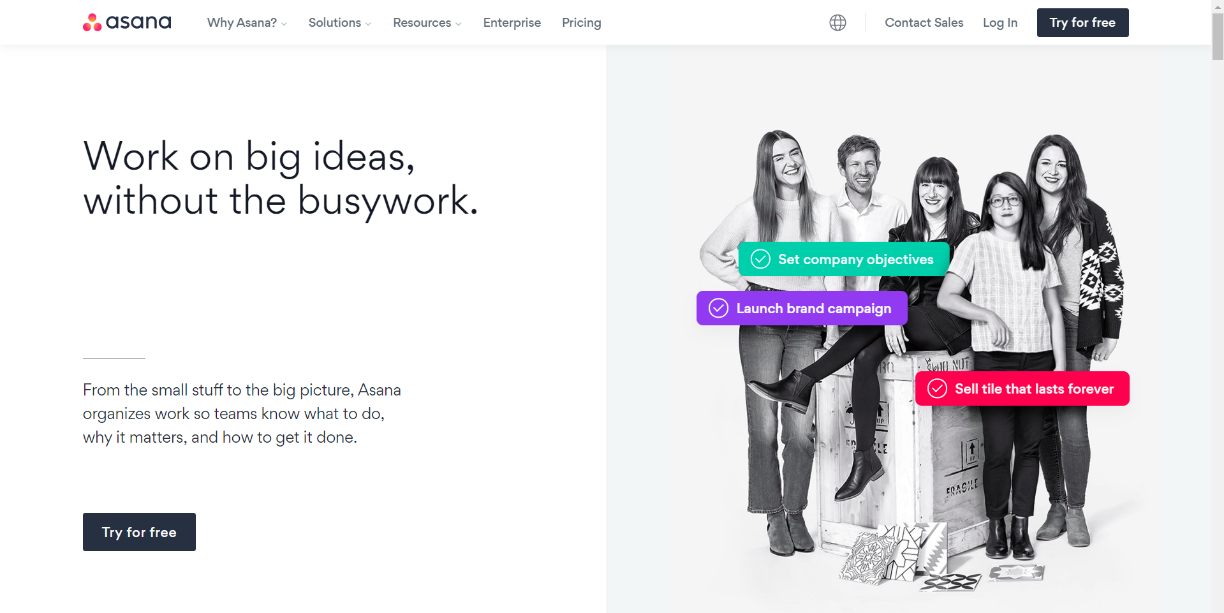
All such business and ecological imperatives bring risk in software development; hence, it is fundamental to manage software projects efficiently

1. **NTASK**



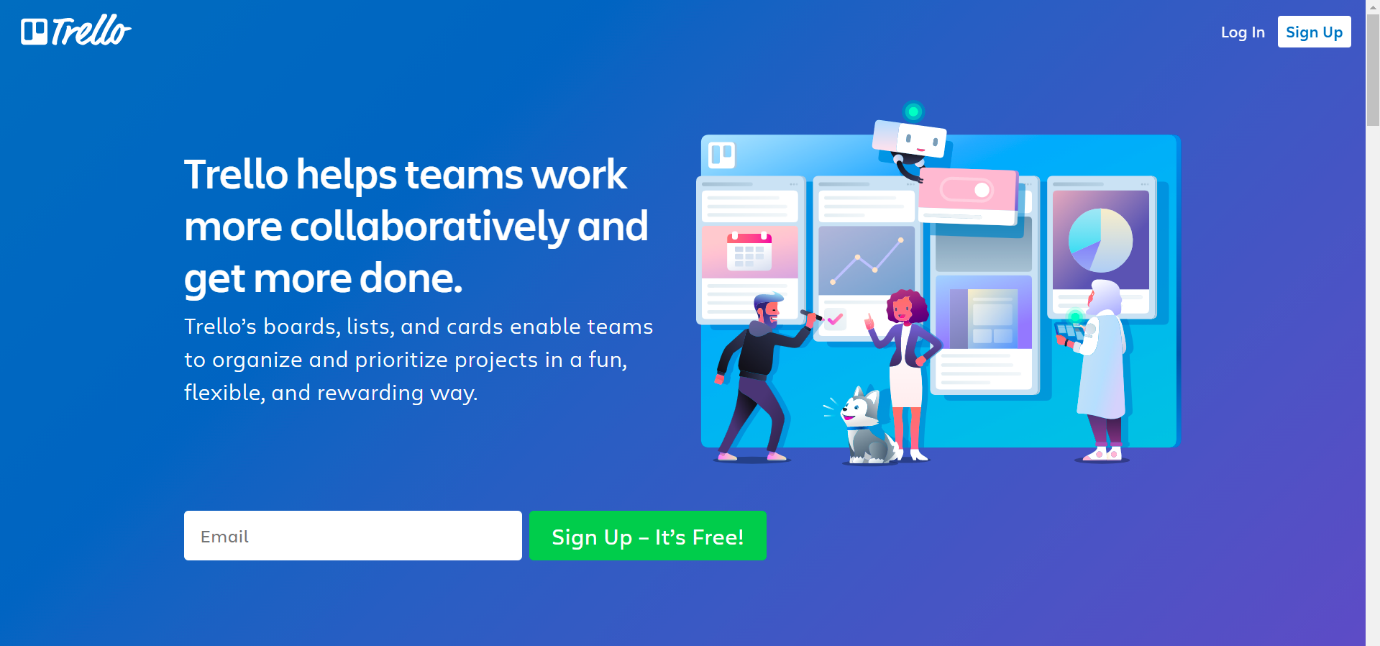
If you’re looking for a tool for planning and managing a series of projects, nTask could just be right for you. Of course, it will also be suitable for single project purposes. It’s just that nTask is really good at providing you with the big picture of all your ongoing projects. When it comes to planning a single project, you can create a list of tasks or a simple Gantt chart. The assignments can also be viewed as a grid.

1. **ASANA**



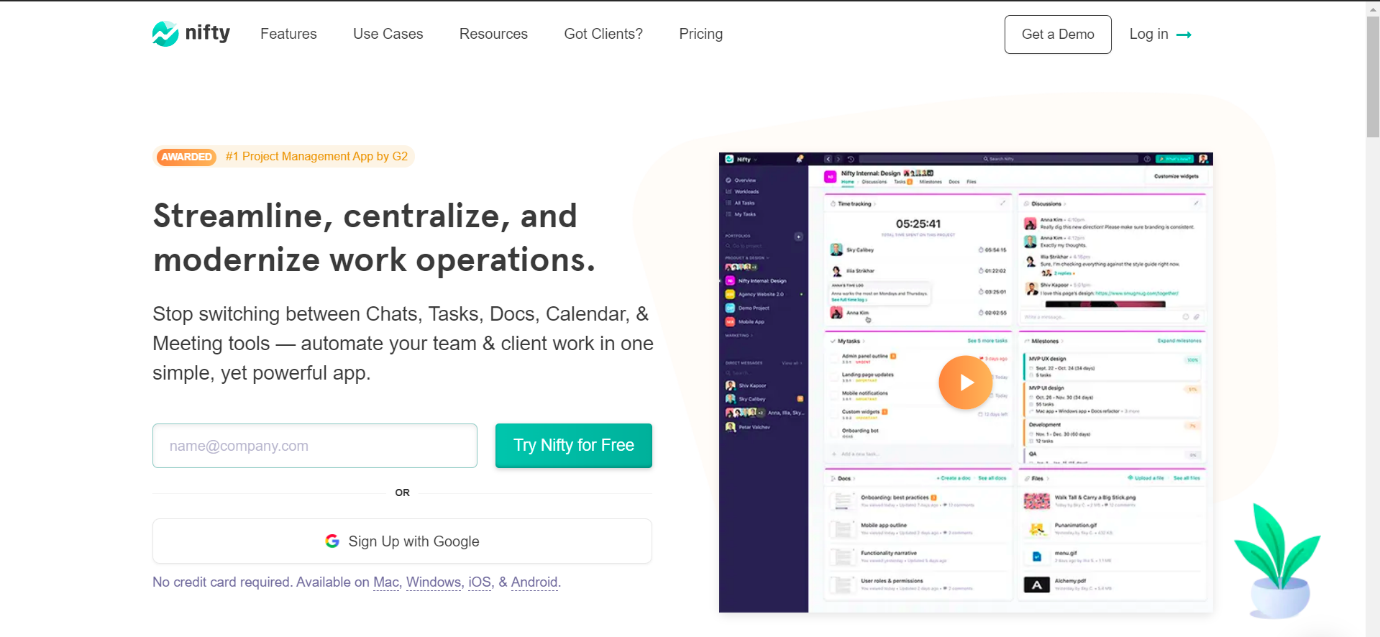
Asana is a popular tool for managing projects of various sizes, and while the free version comes with limited features, it seems to be perfectly suitable for smaller projects. The plan of your project can be displayed as a list, a board or a calendar (see below). The latter view is especially useful when planning milestones and deadlines. You can see how much time there actually is between different due dates. Keep in mind, however, that setting start dates for tasks is not available in the free plan.

1. **TRELLO**



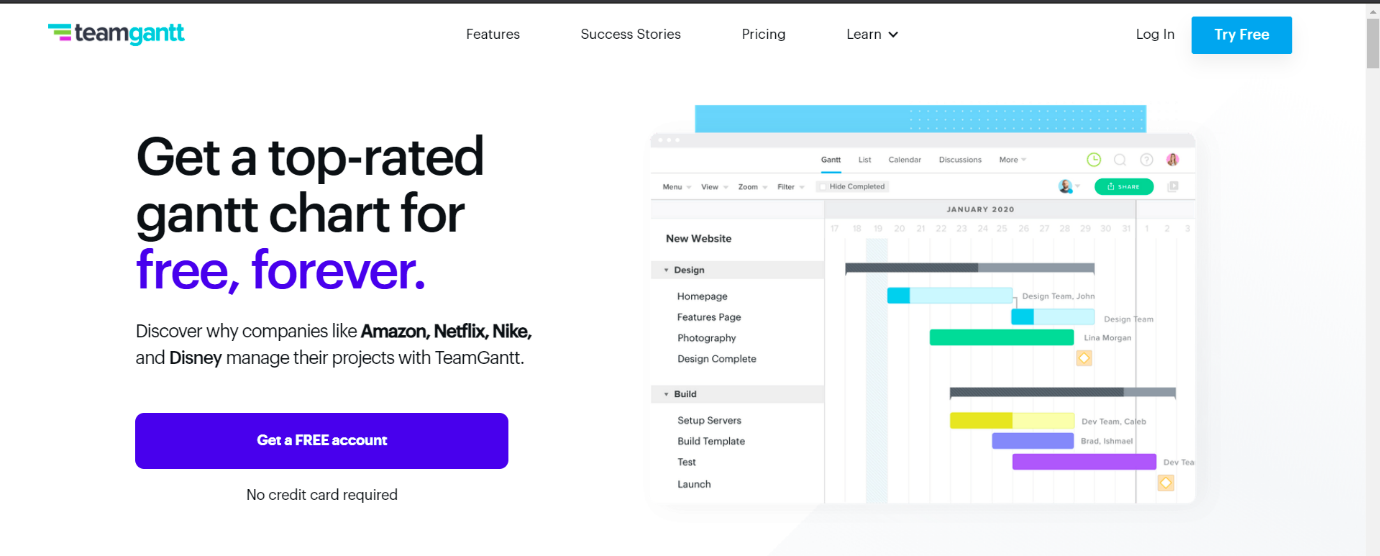
Trello allows you to create boards and fill them with different lists. I went for a simple Kanban-based board and created lists that correspond with statuses (in progress, to do etc.) Trello is a tool that would allow me to create more granular tasks but I’ve decided to use checklists that are built inside the cards to track smaller tasks. You can also add attachments to cards, making it easier to communicate new iterations or getting feedback. Adding due dates to particular cards is also handy when you want to plan a project.

1. **NIFTY**



The creators of Nifty pride themselves on building an app that covers project management and team collaboration. The collaboration aspect is supported by the fact that a team chat is a part of this app. What about the project management part? You can plan your project starting with a list of tasks, or map your milestones on a Gantt chart.

1. **TEAMGANTT**



Out of the box you’ll notice that TeamGantt will not leave you hanging. The onboarding experience will help you tremendously, especially if you’re new to project planning or gantt charts in general. As the name of this software would suggest, creating Gantt charts is the core functionality here. The process of building and editing charts is very straightforward, you can also assign people to particular tasks

**CASE STUDY ON “RISK ANALYSIS & MANAGEMENT IN VLC Player”**

**Overview:**

 VideoLan software, or simply VLC media player, is an academic project in 1996.  It was intended to consist of a client and server to stream videos from satellite dishes across a campus network. The cone icon used in VLC is a reference to the traffic cones collected by École Centrale's Networking Students' Association. Since development of such great app includes high risk factors which we will analyse here.

Risk management is concerned with identifying risks and drawing up plans to minimise their effect on a project.

A risk is a probability that some adverse circumstance will occur.

• Project risks affect schedule or resources

• Product risks affect the quality or performance of the software being developed

• Business risks affect the organisation developing or procuring the software

A risk management process contains 4 major steps.

• Risk identification - Identify project, product and business risks

• Risk analysis - Assess the likelihood and consequences of these risks

• Risk planning - Draw up plans to avoid or minimise the effects of the risk

• Risk monitoring - Monitor the risks throughout the project.

**RISK ANALISIS AND MANAGEMENT ON VLC MEDIA PLAYER**

1. Risk Identification

From my perspective, in Risk Identification, there is some risk that can affect VLC media player and that are:

* Business Impact: there is some similar kind of App. that are in Competition of VLC media player.
* Technology to the limit: the Technology is latest or not.
* Development environment: If the Development Environment Changes the Application may get to the Failure.

2. Risk Projection

|  |  |  |  |
| --- | --- | --- | --- |
| RISK | CATEGORY | PROBABILITY | IMPACT |
| Other Applications that are similar to VLC player like MXplayer | BU | 60% | 4 |
| Lack of Technology tools | DE | 40% | 2 |
| Application not meet expectations | TE | 20% | 1 |

3. Risk Refinement

We can use the risk Identification list that are made on the basis of the risk item checklist that is build on risk analysis questions. So further that question will be get sub question like for example we have risk in Lack of technology tools so there is some sub question like what tools and what is the cost? and risk impact is on level 2 so risk has to be solved.

4. Risk Mitigation

from the Risks that we have discussed above in in risk projection:

* Other Applications that are similar to VLC media player like MX player, Xplayer , Video player so we can Avoid this Risk for Risk Mitigation.
* Lack of Technology tools so we can Accept the Risk and find the best employee that can work on any environment.
* Application not meet expectations so for this Risk we can Reduce the Risk.

**CONCLUSION**

In this practical, we learned about different software development planning tools. Then we explored risk analysis and risk management. We performed case study for VLC media player on same topic.

**PRACTICAL: -4**

**AIM:**

**Design the Software Requirement Specification (SRS) document on VLCMedia Player desktop application in IEEE format only. (Excluding UML Diagram)**

**THEORY:**

**1. Introduction**

**1.1 Purpose:**

The main purpose of our product MP4 player is to make a portable media player that comply with certain standards and formats. The scope of this product includes supporting a VI format, Photo Bank (JPEG), movie play (AMV files). It also includes video recorder, MP3 Player, FM tuner, A-B segment replay Function.

**1.2 Document Conventions**

The SRS document is written in Regular Font style of size 11. For stating terms having high significance and to state high level requirements, we are using different font with size 12 and Bold.

**1.3 Intended Audience and Reading Suggestions**

This document is intended for, such as developers, project managers, marketing staff, users, testers and documentation writers. The sequence for reading the document is beginning with the overview sections and proceeding through the sections.

**1.4 Project Scope**

The scope of the project includes adding new features to existing players. It will be a new version over MP3 player. It will be having functionalities ofMP3 player. Also, it will include JPEG image browser, image interface, document management for movable disk, Solid radio with full segment FM reception, support for AVI formats.

**1.5 References**

Readers can take reference from websitewww.made-in-china.com and www.chinavasion.com.

**2. Overall Description**

**2.1 Product Perspective**

This Product is a follow-on member of product family of Media Players. It is a new version overMP3Player. It contains all functionalities of MP3 player also having certain additional features like supporting various audio, video formats, Image file formats JPEG. In this way, it has become replacement for existing media players like MP3 player.

**2.2 Product features**

The major features include voice recording, FM tuning, support to multiple formats such as MP1, MP2, MP3, WMA, WMV, ASF and WAV. It is also having feature of USB 2.0 full speed interface, movie play with Aviles, Multilanguage support, Built-in microphone, A-B segment replay, Photo Bank and JPEG image browse, Image interface, document management of movable disk, High capacity: 1GB/2GB/4GB available, Equalizer with 7 different quality, Built-in Lithium rechargeable battery.

**2.3 User classes and Characteristics**

One end user class will be of naïve users, who will have no knowledge of internal architecture of MP4 player, they will follow by a menu driven program. Other user class can be of sophisticated users having some knowledge of software. They have some knowledge how to insert, delete, update software and data. Other user class can be of Experts which will have full knowledge of hardware and software as well as their interfaces.

**2.4 Operating Environment**

**2.4.1 Hardware**

Requirement of storage device either flash or hard disk storage. It also includes memory card slots. For interface, it may use touch and wheel mechanism. Screen Sizes range all the way up to 7 inches. As well, resolutions also vary. Most screens come with a color depth of 16-bit, but higher quality video-oriented devices may range all the way to 24-bit. For FM, it will include a radio receiver also. Electronic video processing also. Electronic AV format.

**2.4.2 Operating System**

It supports Microsoft Windows 98/2000/ME/XP OS, Mac OS 10. It can run on Microsoft windows like Window 7, Window XP, Window vista.

**2.5 Design and Implementation constraints**

The major constraint is on Operating system. We can run it directly on Microsoft windows, but cannot run it on Linux directly. For this task, we will need an MP4 Convertor. Also, it should be designed in a way that it can support multiple formats like MP1, MP2, MP3, WMA, WMV, ASF and WAV.

**2.6 User Documentation**

It will include user manuals to guide user how to use MP4 Player. Manuals will provide information to user about various functionalities of MP4 player like Movie play, FM tuning, voice recording etc.

**3. System Features**

* + Movie player using Amile format, MP3 Player, Voice recorder, FM Turner, Flash disk, A-B Segment replay function, Photo bank (JPEG)
  + Multi-Language supported: English, French, German, Italian, Dutch, Portuguese, Spanish, Chinese, Simple Chinese, Korean, Japanese, Denmark, Polish, Russian, Czech, Turkish.
  + It provides 1.8"/260K color display screen. It supports double screen display in-phase.
  + It will support JPEG image browse. It also provides Image interface, document management of movable disk.
  + It supports multiple formats, such as MP1, MP2, MP3, WMA, WMV, ASF and Wait can support all AVI formats.
  + It provides USB2.0 full speed interface

**4. External User Interfaces**

**4.1 User Interfaces**

For user interface, it provides1.8"/260K color display screen. It supports double screen display in-phase. It can support Touch and wheel mechanism for interface. It also supports shortcut key startup.

**4.2 Hardware Interfaces**

It provides USB2.0 full speed interface. It includes memory card slots. It provides high capacity: 1GB/2GB/4GB.

**4.3 Software Interfaces**

It supports Microsoft Windows 98/2000/ME/XP OS, Mac OS 10. It also provides Image interface, document management of movable disk. It also provides interface for multiple audio and video formats.

**4.4 Communication Interfaces**

To communicate with radio (FM) channels it provides solid radio interface with full segment FM reception. It also provides FM tuner to communicate with different radio channels.

**5. Other Non-Functional Requirements**

**5.1 Performance Requirements**

The main requirement for performance is good audio quality. It should provide equalizer with different qualities. It should provide a sophisticated volume control system.

The other main main requirement for performance is good video quality. It should provide fine resolution for videos.

**5.2 Safety Requirements**

The main safety requirement is use of good quality hardware, so that there should be minimum wear out of hardware. The screen should be made of appropriate material, so that it can work for long time. For safety concerns, we can make it water proof.

**5.3 Security Requirements**

The main security requirement is including of GPS system, so that if it gets misplaced somewhere, we can easily find it out.

**5.4 Software Quality Attributes**

It includes adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability of Software. The important quality attribute is that it should support all audio and video formats.

**6. Other Requirements**

**6.1 Glossary: -**

* AVI: - It is Audio Video Interleave. It is a multimedia container format. AVI files can contain both audio and video data in a file container that allows synchronous audio-with-video playback.
* AMV: - AMV is a proprietary video file format, produced for MP4 players. The container is a modified version of AVI.
* JPEG: - It is Joint Photographic Experts Group. In multimedia, Motion JPEG (M-JPEG) is an informal name for a class of video formats where each video frame or interfaced field of a digital video sequence is separately compressed as a JPEG image.
* WMV: - It is Windows Media Video. It is a compressed video compression format for several proprietary codecs developed by Microsoft.
* WMA: - Windows Media Audio (WMA) is an audio data compression technology developed by Microsoft. The name can be used to refer to its audio file format or its audio codecs. It is a proprietary technology that forms part of the Windows Media framework.
* ASF: - It is Advanced Systems Format, a Microsoft streaming format associated with Windows Media Player.
* WAV: - Waveform Audio File Format, is a Microsoft and IBM audio file format standard for storing an audio bitstream.
* Proprietary: - Proprietary software is computer software licensed under exclusive legal right of its owner.
* Codec: - A codec is a device or computer program capable of encoding and/or decoding a digital data stream or signal.

**PRACTICAL – 5**

**AIM**: List at least 10 software design principles & online/offline tools for software development process and Draw the UML diagram for VLC media player desktop application.

**THEORY** Here we will be discussing about a number of software design principles and also some of the tools for software development process.

**Software Design Principles:-** The first five principles form the basics of the software development principles and are also known as “S.O.L.I.D principles”.

1. **Liskov Substitution Principle (LSP):-**The LSP principle mainly focusses on the degree of use of the Inheritance in a software. While inheritance is beneficial, it is advisable to use it contextually and moderately. The principle strives to prevent cases where classes are extended only through common things. You need to consider the pre-conditions and post-conditions of a class before performing inheritance. This is the third principle denoted by **“L”.**
2. **Keep It Simple :-** Ensuring that the program coding is simple and very easy to understand is very much important. The code shouldn’t give a new person hard time to figure out how things are working. Methods and functions should be as brief as possible but also easily understandable. Each of them should solve only one problem or two at max. Also the project code should not have a lot of conditions (simple and nested conditions). Optimizing the conditions would help understanding and finding bugs easily for the QA team.
3. **Open/Closed Principle (OCP):-** The principle says that you should be able to change the behavior of a class without modifying it. Therefore, you can extend the class’s behavior through composition, interface, and inheritance. However, you cannot open it for minor modifications. This is the second principle denoted by **“O”.**
4. **Single Responsibility Principle (SRP):-** It is a software engineering principle that states that a class should have only one reason to change. In other words, it must have only one responsibility. Here, we are talking about cohesion. All elements in given class structures or modules should have a functional affinity to one another. By clearly defining your class’s responsibility, you increase its cohesiveness. This is the first principle denoted by **“S”.**
5. **Interface Segregation Principle (ISP):-** ISP prefers many specific interfaces to a general interface. The goal is to have finely grained and client-specific interfaces. You need to enhance cohesion in interfaces and develop modules with few behaviors. Interfaces that have many behaviors are hard to maintain and evolve. So, they should be completely avoided. This is the fourth principle denoted by **“I”.**
6. **Dependency Inversion Principle (DIP):-** The fifth and the final principle of SOLID is Dependency Inversion Principle and is denoted by **“D”.** The principle asserts that programmers should depend on abstractions and not on concrete classes. This can be broken down to 2 parts as: We can break it into two:

* High-level modules need to be independent of low-level ones. Both should depend on abstractions
* Abstractions should be independent of details. Details should depend on abstractions.

Now we look at some of the important principles other than SOLID.

1. **You Aren't Gonna Need It (YAGNI):-** Most programmers fall into the pit of trying to implement all the functionalities at once, from the word go. In the end, some or most of these functionalities become useless. Start by adding just a few methods to a class. After that, as your project starts taking shape and new demands arise, you can add more functionalities. That way, you’ll achieve a lean development software. YAGNI saves time, efforts, and costs that you would have wasted in trying to understand or debug the code.
2. **Measure Twice and Cut Once:-** The development life cycle’s requirement stage usually introduces more than 50% coding issues if not done well. Therefore, a systematic approach should be developed. It is very much important to double check all the project requirements so that one does not miss any important section(feature) OR does not accidentally add too much. After that, make blueprints that will guide the whole process to achieve high-quality coding throughout. Always test your project from basics to ensure everything is fine. This principle gives much more predictable outcomes, especially if the project’s cost is already high. You’ll save yourself headaches that come with deleting or adding code lines to meet requirements.
3. **Don’t Repeat Yourself (DRY):-** When writing your code, don’t repeat yourself. That is, avoid copy-pasting your code in different places. Otherwise, future maintenance will be difficult. The reason is that you will have to make changes to the coding in those various places. Those changes will further necessitate changes in the tests to make the results click green. All of that will need more time, effort, and money. To avoid such a pitfall, you can extract a common logic into functions. Additionally, if there are any manual works that you can automate, do so to keep your code lean. For software development, the above steps will help in the code re-usability without having to repeat it.
4. **Least Astonishment:-** The principle of least astonishment says that it is advisable to design a feature that doesn’t have a high-astonishment factor. Your system’s components should behave in a way that end-users expect. Therefore, your project’s outcomes will be profitable only if they are obvious, predictable, and consistent. Otherwise, users will shy from using features or structures that astonish, surprise, or confuse them. You are making software products for people to use. Thus, you’ll reap a lot by designing user-friendly features. Strive to match human beings’ mental models, experience, and expectations. Remember, you have to capture the user’s attention as quickly as possible. As we know, the current users’ attention span has plummeted.

**SOFTWARE DEVELOPMENT PROCESS TOOLS:**

1. **GitHub:-** GitHub is a powerful collaboration tool and development platform for code review and code management. With this GitHub, the users can build applications and software, manage the projects, host the code, review the code etc.

**Features:-**

* With GitHub, developers can easily document their code and can host the same from the repositories.
* GitHub’s project management tools help its users to stay aligned, co-ordinate easily and get their task done accordingly.
* Few features of GitHub that make it a useful tool are its code security, access control among the team members, integration with other tools etc.
* Few developers use GitHub for experimenting new programming languages in their personal projects.
* GitHub can be hosted on servers and on a cloud platform. It runs on Windows and Mac OS.
* GitHub is free for open source projects and public use. For developers it is charged based on different criteria and services requested.

1. **Studio 3T:-** Studio 3T for MongoDB helps you to build queries fast, generate instant code, import/export in multiple formats, and much more.

**Features:-**

* Query MongoDB faster with our Visual Query Builder, IntelliShell, or SQL Query tool.
* Our Data Masking tool enables data compliance and bolsters security with powerful field-level data obfuscation.
* Import to MongoDB from JSON, CSV, BSON/mongodump, and SQL, and get a preview of your output documents as you make changes.
* Migration from MongoDB to SQL (or vice versa) has never been easier with our Migration tools."

1. **Embold:-** Fixing bugs before deployment saves a lot of time and energy in the long run. Embold is a software analytics platform that analyses source code and uncovers issues that impact stability, robustness, security, and maintainability.

**Features:**

* With the Embold plugins, you can pick up code smells and vulnerabilities as you code, before making commits.
* Unique anti-pattern detection prevents the compounding of unmaintainable code.
* Integrate seamlessly with Github, Bitbucket, Azure, and Git and plugins available for Eclipse and IntelliJ IDEA.
* Get deeper and faster checks than standard code editors, for over 10 languages.

1. **Linx:** Linx is a low code IDE and server. IT pros use Linx to quickly create custom automated business processes, integrate applications, expose web services and to efficiently handle high workloads.

**Features:-**

* Easy-to-use, drag-and-drop interface
* Over 100 pre-built functions and services for rapid development
* One-click deployment to any local or remote Linx Server directly from the IDE
* Input and outputs include nearly any SQL & NoSQL databases, numerous file formats (text and binary) or REST and SOAP Web services
* Live debugging with step through logic
* Automate backend processes via timer, directory events or message queue or expose web services, and call APIs via HTTP requests

1. **Kite:-** Kite is IDE for Software Development that automatically completes multiple line codes. This editor supports more than 16 languages. It helps you to code faster with no hassle.

**Features:-**

* It offers Software Development documentation.
* This editor provides a function signature as you type.
* You will get a tooltip on mouse hover.
* Provides support in email.
* Uses machine learning models for Software Development language.
* Also it is a free to use open source tool.

1. **NetBeans:-** NetBeans is an open source and a free software development tool written in Java that develops world-class web, mobile, and desktop applications easily and quickly. It uses C / C++, PHP, JavaScript, Java etc.

**Features:-**

* Support for fast & smart code editing.
* Easy & Efficient Project Management process.
* Rapid User Interface Development.
* Helps to write bug-free code.
* NetBeans IDE offers superior support for C/C++ and PHP developers.

1. **Cloud9 IDE:-** Cloud9 IDE is an online integrated software development environment. It supports many programming languages like C, C++, PHP, Ruby, Perl, Python, JavaScript and Node.js.

**Features:-**

* Allows to clone entire development environment.
* Built-In Terminal for command-line wizard.
* Code Completion suggestions helps software developers to code faster and avoid typos.
* The Debugger helps developers to set breakpoints, and inspect variables of any JS/Node.js app.
* Simply drag any file or Terminal to create multiple split views.
* Developers can select an extensive set of default Runners to execute app, such as Ruby, Python, PHP/Apache.

1. **Atom:-**Atom is a solid all-around text-editor. It is fully free and open source. It can be customized to do anything but without a need of modifying the config file.

**Features:-**

* Atom works across many popular operating systems like OS X, Windows, or Linux.
* It helps developers to write code faster with a smart, flexible autocomplete.
* Easily browse and open whole project or multiple projects in one window.
* It is possible to split Atom interface into multiple panes to compare and edit code across files.
* Find, preview, and replace text type in a file or across the entire project.

1. **Bitbucket:-** Bitbucket is a distributed, web-based version control system that is used for collaboration between software development teams (code and code review). It is used as a repository for source code and development projects.

**Features:-**

* Useful features of Bitbucket that makes it a powerful tool are its flexible deployment models, unlimited private repositories, code collaboration on steroids etc.
* Bitbucket supports few services like code search, issue tracking, Git large file storage, bitbucket pipelines, integrations, smart mirroring etc.
* Using Bitbucket, one can organize the repositories into the projects with which they can focus easily on their goal, process or product.
* To rationalize the development process of any software it can integrate into the prevailing workflow.
* Bitbucket offers a free plan for 5 users with unlimited private repositories.

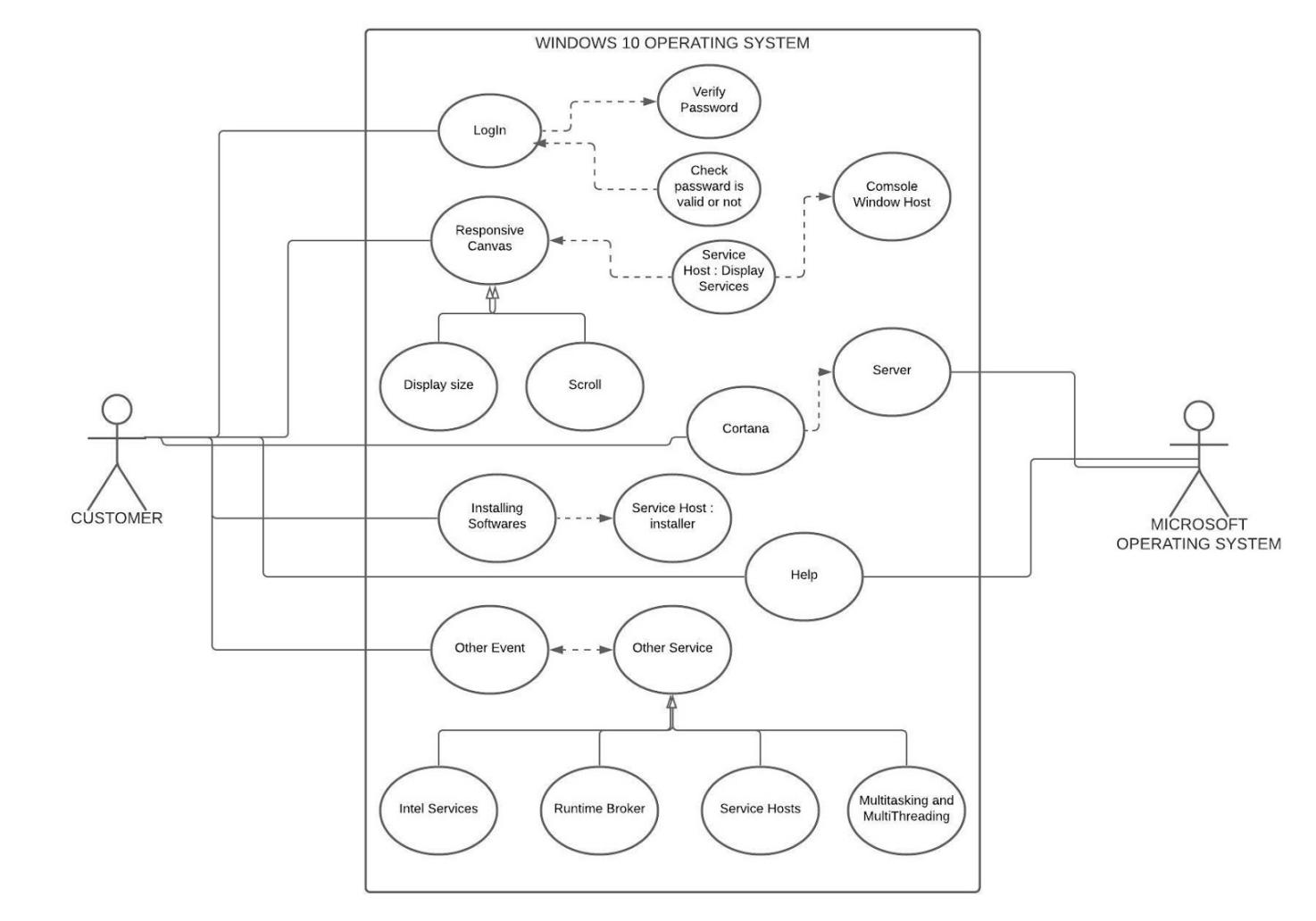
1. **JIRA:-**Jira is the most popular software development tool that is used by agile teams for planning, tracking and releasing the software.

**Features:**-

* This tool is customizable and also has some prevailing features that are used in every development phase.
* Using Jira, we can accomplish the work in progress, generate reports, backlogs etc.
* Few other important features of Jira software are Scrum boards, Kanban boards, GitHub integration, Disaster recovery, Code Integration, Portfolio Management, Sprint Planning, Project Management etc.
* Jira works for Windows and Linux/Solaris operating systems.

**UML DIAGRAM FOR VLC MEDIA PLAYER DESKTOP APPLICATION**

UML, short for Unified Modeling Language, is a standardized modeling language consisting of an integrated set of diagrams, developed to help system and software developers for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling and other non-software systems. The UML uses mostly graphical notations to express the design of software projects. Using the UML helps project teams communicate, explore potential designs, and validate the architectural design of the software. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing object oriented software and the software development process.



**CONCLUSION:**

In this practical, we learned about different software development principles to keep in mind during a software development planning. Also we have seen some of the most popularly used software development tools which are available online as well as application software. Also at the end we have learnt about UML diagrams, how to prepare them and also seen the UML diagram of VLC media player desktop application.

**PRACTICAL – 6**

**AIM**: List at least 10 type of Testing for software development life cycle in IT industry

and Design Test Case, Test Suites & Testing Strategy for the “VLC Media

Player” Mobile Application

**THEORY**:

**Types Of Testing**

1. **Regression Testing:**

Every time a new module is added leads to changes in the program. This type of testing makes sure that the whole component works properly even after adding components to the complete program.

Example:

In school record suppose we have module staff, students and finance combining these modules and checking if on integration these module works fine is regression testing

1. **Integration Testing**

The objective is to take unit tested components and build a program structure that has been dictated by design. Integration testing is testing in which a group of components is combined to produce output.

Integration testing is of four types:

(i) Top-down (ii) Bottom-up (iii) Sandwich (iv) Big-Bang

Example

(a) Black Box testing:- It is used for validation. In this we ignore internal working mechanism and focuses on what is the output?.

(b) White Box testing:- It is used for verification. In this we focus on internal mechanism i.e., how the output is achieved?

1. **Unit Testing**

It focuses on the smallest unit of software design. In this, we test an individual unit or group of interrelated units. It is often done by the programmer by using sample input and observing its corresponding outputs.

Example:

a) In a program we are checking if loop, method or

function is working fine

b) Misunderstood or incorrect, arithmetic precedence.

c) Incorrect initialization

1. **Alpha Testing**

This is a type of validation testing. It is a type of acceptance testing which is done before the product is released to customers. It is typically done by QA people.

Example:

When software testing is performed internally within the organization

1. **Beta Testing**

The beta test is conducted at one or more customer sites by the end-user of the software. This version is released for a limited number of users for testing in a real-time environment

Example:

When software testing is performed for the limited number of people

1. **Smoke Testing:**

This test is done to make sure that software under testing is ready or stable for further testing. It is called a smoke test as the testing an initial pass is done to check if it did not catch the fire or smoke in the initial switch on.

Example:

If project has 2 modules so before going to module II make sure that module 1 works properly

1. **System Testing:**

This software is tested such that it works fine for the different operating systems. It is covered under the black box testing technique. In this, we just focus on the required input and output without focusing on internal working. In this, we have security testing, recovery testing, stress testing, and performance testing

Example:

This include functional as well as non-functional testing

1. **Stress Testing:**

In this, we give unfavorable conditions to the system and check how they perform in those conditions.

Example:

(a) Test cases that require maximum memory or other

resources are executed

(b) Test cases that may cause thrashing in a virtual

operating system

(c) Test cases that may cause excessive disk requirement

1. **Performance Testing:**

It is designed to test the run-time performance of software within the context of an integrated system. It is used to test the speed and effectiveness of the program. It is also called load testing. In it we check, what is the performance of the system in the given load.

Example:

Checking number of processor cycles.

1. **Object-Oriented Testing:**

This testing is a combination of various testing techniques that help to verify and validate object-oriented software. This testing is done in the following manner:

* 1. Testing of Requirements,
  2. Design and Analysis of Testing,
  3. Testing of Code,
  4. Integration testing,
  5. System testing,
  6. User Testing.

**Test Case, Test suite and Testing strategy for VLC Media Player mobile application**

1. Verify that Media Player should be installed in Mobile.
2. Verify that Media Player should support the Mobile OS.
3. Verify that Media Player should evoke effortlessly with one action.
4. Verify that Media Player should be close effortlessly.
5. Verify that Media Player should have controllers like Play, Pause, Next, Previous, Stop, Close, Volume increase and decrease button, Contrast increase, and decrease button.
6. Verify that Media Player should be able to work and display in both Portrait and Landscape view.
7. Verify that Media Player should maintain its consistency of controllers in both Portrait and Landscape view.
8. Verify that Media Player should be able to run Video file of any format like AVI, MKV etc.
9. Verify that Media Player should be able to run an audio file of any format like mp3, WAV etc.
10. Verify that Media Player is able or not able to run Video file of any quality like 3GP, MP4, Low Definition, High Definition, 144p, 240p, 360p, 480p, 720p, 1080p etc.
11. Verify that Media Player is able or not able to run Audio file of any quality, bit rate, bandwidth.
12. Verify that Media Player is able or not able to change the Aspect ratio.
13. Verify that Media Player is able or not able to change the Crop.
14. Verify that Media Player is able or not able to update from the internet.
15. Verify that Media Player should be able to run the Audio or Video files from SD / Memory Card.
16. Verify that Media Player should be able to run the Audio or Video files from Phone Memory / Internal Memory.
17. Verify that Media Player is able or not able to run the Audio or Video files from the internet.
18. Verify that Media Player should continue to run the Video file if Media Player go to backgrounds or minimizes.
19. Verify that Media Player should continue to run the Audio file if Media Player go to backgrounds or minimizes.
20. Verify that Media Player should be able to Stop or Minimize if pressing Back button one or two times.
21. Verify that Media Player should have Media Library files.
22. Verify that Media files in Media Player’s Media Library should be a shuffle, Add, Remove Media files.
23. Verify that Media files in Media Player’s Media Library should be Add Media files.
24. Verify that Media files in Media Player’s Media Library should be Remove Media files.
25. Verify that Audio and Video files both can be Add, Shuffle and Remove from Media Library of Media Player.
26. Verify that Video files of different quality and size should work correctly on Play- Pause-Play action, that image of videos should not be a break, the voice should not be break, lag or lead with the video.
27. Verify that Audio files of different quality and size should work correctly on Play- Pause-Play action, that audio should not be break, lag or lead.
28. Verify that Media Player is able to keep on the run or not any Video or Audio file if tap on the Back button and minimize the app.
29. Verify that Media Player is able to resume playing if any paused Video or Audio file in-app in minimizing condition then evoke.
30. Verify that two Video files can able to run in two different Media Players simultaneously in a device.
31. Verify that one Video and one Audio file can able to run in two different Media Players simultaneously in a device.
32. Verify that one Video and one Audio file can able to run in two different Media Players simultaneously in a device.

**PRACTICAL – 7**

**AIM**: Prepare the details Case Study on Design coding standards and guidelines for

your respective SGP project definition and justify which Software Quality

Standards & Testing Tool will be suitable for your SGP project.

**THEORY**:

**Case study: Face recognition system**

**Abstract**

A face recognition system is one of the biometric information processes, its applicability is easier and working range is larger than others, i.e.; fingerprint, iris scanning, signature, etc. The system uses a combination of techniques in two topics; face detection and recognition. The face detection is performed on live acquired images without any application field in mind. Processes utilized in the system are white balance correction, skin like region segmentation, facial feature extraction and face image extraction on a face candidate. Then a face classification method that uses FeedForward Neural Network is integrated in the system. The tested system has acceptable performance to recognize faces within intended limits. System is also capable of detecting and recognizing multiple faces in live acquired images.

**Introduction**

Face recognition systems are part of facial image processing applications and their significance

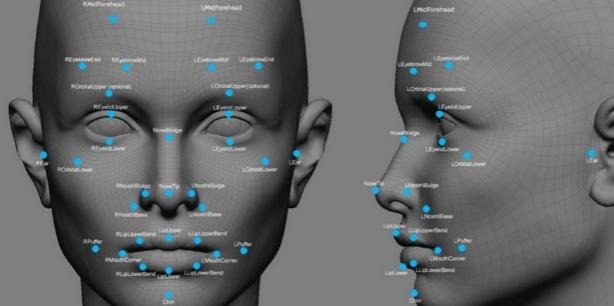
as a research area are increasing recently. They use biometric information of humans and are

applicable easily instead of fingerprint, iris, signature etc., because these types of biometrics are not much suitable for non-collaborative people. Face recognition systems are usually applied and preferred for people and security cameras in metropolitan life. These systems can be used for crime prevention, video surveillance, person verification, and similar security activities.

Face recognition system is a complex image-processing problem in real world applications with complex effects of illumination, occlusion, and imaging condition on the live images. It is a combination of face detection and recognition techniques in image analyzes. Detection application is used to find position of the faces in a given image. Recognition algorithm is used to classify given images with known structured properties, which are used commonly in most of the computer vision applications. Recognition applications use standard images, and detection algorithms detect the faces and extract face images which include eyes, eyebrows, nose, and mouth. That makes the algorithm more complicated than single detection or recognition algorithm.

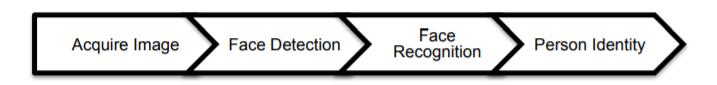
**Working**

The working of facial recognition system varies largely on the technology used. However, here are the basic steps:

* + - 1. A picture of your face is captured from a photo or video. Your face might appear alone or in a crowd. Your image may show you looking straight ahead or nearly in profile
      2. Facial recognition software reads the geometry of your face. Key factors include:
         1. the distance between your eyes
         2. the distance from forehead to chin.
         3. facial landmarks — one system identifies 68 of them — that are key to distinguishing your face.

The result: your facial signature.

* + - 1. Your facial signature — a mathematical formula — is compared to a database of known faces. And consider this: at least million people have images of their faces in one or more databases.
      2. A determination is made. Your faceprint may match that of an image in a facial recognition system database.



**Design Coding Standards and Guidelines:**

The following coding standards and guidelines have been followed throughout the project:

* + - 1. Limited use of globals
      2. Standard headers for different modules:

These headers include the following information:

* Name of the module
* Date of module creation
* Author of the module
* Modification history
* Synopsis of the module about what the module does
* Different functions supported in the module along with their input output parameters
* Global variables accessed or modified by the module
  + - 1. Naming conventions for local variables, global variables, constants and functions
         1. Local variables: camelCase
         2. Global variables: PascalCase
         3. Constants: UPPERCASE
         4. Functions: camelCase
      2. Indentation

**Software Quality Standards**

We have used ISO/IEC 9126 Software quality standard for our project

This standard deals with the following aspects to determine the quality of a software application:

* Quality model
* External metrics
* Internal metrics
* Quality in use metrics

This standard presents some set of quality attributes for any software such as −

* Functionality
* Reliability
* Usability
* Efficiency
* Maintainability
* Portability

**Testing**

We have used the following testing methods:

1. Stress testing:

In this, we give unfavourable conditions to the system and check how they perform in those conditions, i.e., blurry images, more than one people, background noise, etc.

1. Performance Testing:

It is designed to test the run-time performance of software within the context of an integrated system. It is used to test the speed and effectiveness of the program. It is also called load testing. In it we check, what is the performance of the system in the given load. We give our system images of different sizes to check the performance.

**Conclusion**

Face recognition systems are part of facial image processing applications and their significance as a research area are increasing recently. Implementations of system are crime prevention, video surveillance, person verification, and similar security activities. The face recognition system implementation will be part of humanoid robot project at Atılım University. The goal is reached by face detection and recognition methods. Knowledge-Based face detection methods are used to find, locate and extract faces in acquired images. Implemented methods are skin color and facial features. Neural network is used for face recognition

**PRACTICAL – 8**

**AIM**: List at least 5 online or offline Software Maintenance tools & prepare the detail

case study on “VLC Media Player” Maintenance document as a VLC Media

Player Developer.

**THEORY**:

* 1. **File comparator:** Compares two files or systems and maintains the record of the differences in the files. In addition, it determines whether the two files or the systems are identical.
  2. **Compiler and linker:** Compilers are used to check syntax errors and in some cases, locate the type of errors. When the code is compiled, the linker is used to link the code with other components, which are required for the program execution. Linkers sometimes are used to track the version numbers of the components so that appropriate versions are linked together.
  3. **Debugger:** Allows tracing the logic of the program and examines the contents of the registers and memory areas.
  4. **Cross-reference generator**: Assures that the changes in code are in compliance with the existing code. When a change to a requirement is requested, this tool enables to know which other requirements, design, and code components will be affected.
  5. **Static code analyzer**: Measures information about the code attributes such as the number of lines of code, number of spanning paths, and so on. This can be calculated when the new versions of the system are developed.

**Practical 9**

**Aim:- Prepare the detailed case study on Puppet DevOps tool in the area of cloud business application.**

**Abstract:**

Puppet is a management that

manages infrastructure physical

as well as virtual machines. It is

an open – source software

configuration developed by

Ruby. The one who wants to

know Puppet must have

knowledge of system

administration, infrastructure

and network protocol

communication. One should

know how to command on

Ruby script writing and the

system in order to use Puppet.

**What is** [DevOps](https://puppet.com/blog/what-is-devops)**?**

[DevOps](https://puppet.com/blog/what-is-devops) is a way to release better software. It is not just technical tools or workflows. DevOps is also a cultural practice. DevOps produces better software, faster, by aligning development, staging, and deployment.

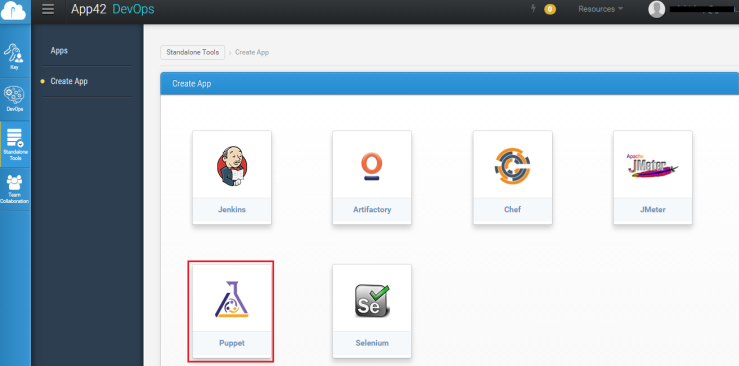
DevOps engineers usually have a more comprehensive range of functions, and they connect all the phases into the whole by applying process automation. They gather code, describe infrastructure, deploy testing environments promptly, and quickly update production without losing quality.

**What is Puppet?**

Puppet is developed by Puppet Labs to automate infrastructure management and configuration. Puppet is a very powerful tool which helps in the concept of Infrastructure as code.

[Puppet](https://www.liquidweb.com/kb/install-puppet-ubuntu-18-04/" \t "_blank) is a cross-platform client-server based application used for configuration management. It handles software configurations on multiple servers. There are two versions available. One is open-source, the other is a commercial version.

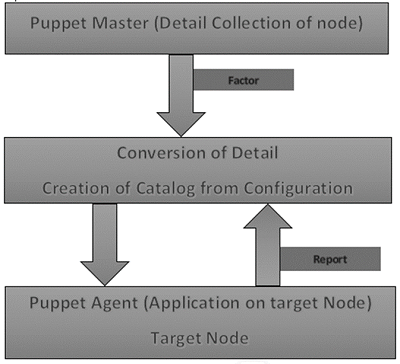
It works on both Linux and Windows platforms. It uses a declarative approach to automate updates, installations, and other tasks.



Puppet is one of the great DevOps programs for managing multiple servers.

**Puppet − Workflow**

Puppet uses the following workflow to apply configuration on the system.



## Puppet Master

Puppet Master is mechanism which handles all the configuration. And this applies the configuration to nodes using the Puppet agent.

## Puppet Agent

Puppet Agents are managed by the Puppet master. They have the Puppet agent daemon service running inside them.

## Config Repository

This is the repo where all nodes and server related configurations are saved and pulled when required.

## Facts

Facts are the details related to the master machine. which are

basically used for analyzing the current status of any node. On the basis of facts all changes are done on any target machine. There are so many pre-defined and custom facts in Puppet.

## Catalog

All the configuration which are written in Puppet are 1st converted into a compiled format called catalog and later those catalogs are applied on the target machine.

### Puppet Installation

Puppet works on the client server architecture, where we call the server as the Puppet master and the client as the Puppet node. This setup is achieved by installing Puppet on both the client and well as on all the server machines.

## Prerequisites

Factor is the only pre-requisite.

### Installing Puppet and Facter Using Ruby Gem

# Installing Facter

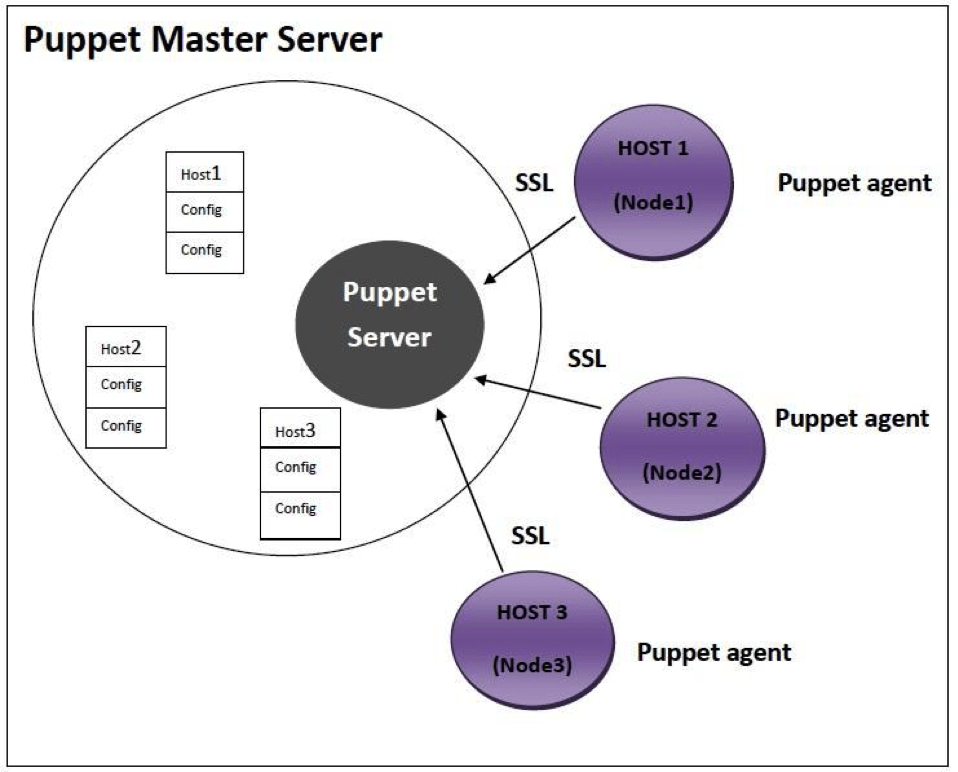
$ wget http://puppetlabs.com/downloads/gems/facter-1.5.7.gem

$ sudo gem install facter-1.5.7.gem

# Installing Puppet

$ wget http://puppetlabs.com/downloads/gems/puppet-0.25.1.gem

$ sudo gem install puppet-0.25.1.gem



### Why do we use Puppet?

It is one of the most popular configuration management tools in the IT world now a days. It is being extensively used by Fortune 500 companies as well. Most of the businesses using DevOps methodologies are getting inclined toward using Puppet.

**Advantages of Puppet**

* It has better documentation
* Unlike other configuration management tools, Puppet, after deploying a configuration on any system, keeps verifying those configurations in certain intervals. These intervals can be modified as well
* Puppet is being used by some of the major organizations in the industry, namely, Google, Red Hat, etc.
* Puppet helps DevOps professionals and System Administrators work faster and smarter and grants DevOps admins the ability to create unique configurations for each host and configuration designs for multiple hosts.
* It easily integrated with other tools.
* It saves time by increasing software development speed.

**Conclusion**

* Puppet is on its way to becoming famous for configuration management. More than 75 percent of the Fortune 500 companies are using Puppet these days. With Puppet, System Administrators are now able to do all those tasks that were once considered tedious to do.

**References**

1. [https://intellipaat.com/blog/tutorial/devops-tutorial/puppet-tutorial/#\_why\_do\_puppet](https://intellipaat.com/blog/tutorial/devops-tutorial/puppet-tutorial/" \l "_why_do_puppet)
2. <https://www.tutorialspoint.com/puppet/index.htm>

**Practical-10**

**Aim: Prepare the detailed case study on Design Thinking Methodology in the area of Mobile application development (Android or iOS).**

**What is Design Thinking?**

Design thinking is a method used by designers in ideation as well as [mobile app development](https://www.thesunflowerlab.com/mobile-app-development-company-columbus-ohio/" \t "_blank).

It is very useful when complex problem is given by client.

It also helps in tackling problems which are unknown.

Design Thinking is not only limited to IT field it also helpful in Engineering, medicine, business, architecture, sports, research, etc.

**Design Thinking Process:**

**Research**: Research the requirements of the users or target users.

**Define**: Define and understand the problems using research data.

**Ideation**: Find the solution for defined problems and ideas.

**Prototype**: Develop a prototype for solution.

**Testing**: Test the prototype to check that whether it is developed as per the requirements or not.

These 5 stages are not always sequential, they can occur simultaneously as well as repetitively.

**Importance of Design Thinking:**

* As Design Thinking has a human-centred approach, it focuses on the end-users and how to make the user experience better and more fulfilling.
* Since it involves multidisciplinary teams, the collective advantage of wisdom, expertise and experience is available while creating solutions.
* It also involves finding solutions in an unconventional way. So, while solving real problems this creates & delivers value to the end-users.
* Going to the market with a MVP (minimum viable product) is totally perfect for design driven firms. In such cases, they learn from the feedback of users, incorporate the same in their build and release an upgraded version of the product. Eg., Facebook, Instagram, Whatsapp and the likes.
* It helps create successful brands and generate ROI from these brands.

**Benefits of using design thinking in mobile app development:**

* **Increase in sales**: If the UX for end-user is seamless, the end-user will repeatedly use your application. This repeated use suggests brand loyalty which can further result in increase in sales.
* **Current and future marketing trend**: Mobile app is the need of the hour and is also the future of various applications and businesses. Hence, it becomes vital for businesses to develop mobile applications.
* **Marketing on the go**: Marketers can now focus on promoting their business via web and mobile platforms.
* **Ease of connecting to customers**: Customers can provide feedback effortlessly and so companies can learn and improve on their setbacks directly from the customers’ feedback.