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.

SOLUTION:

The 10 **AGILE DEVELOPMENT TOOLS** are mentioned below:

1. Jira Software
2. Infinity
3. GitScrum
4. Proggio
5. ClickUp
6. Blossom
7. Assembla
8. Planbox
9. Zoho Sprints
10. Monday.com

Case Study on JIRA TOOL:

In order to deliver a project successfully, the most important points a team needs are time and cost management and a proper co-ordination between the team members. The industry around the world is seeing a massive paradigm shift in the way teams work and co-ordinate. Initially, only cost was a major factor considered for a project but in this digital era, time and cost both are an essential components in determining the success of any project.

So, the team leaders and project managers needed a tool through which they can track, monitor, assign the tasks and also, can see the progress of each team member, and that too on a single platform. So, the answer to all the questions is JIRA.

Basically, JIRA is a software tool extensively used for project management, issue tracking.

In other words, **JIRA is an issue management platform that allows teams to easily manage their issues throughout their lifecycle.**

JIRA is mostly used by the teams that follow agile methodology for the project.

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.

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

So, we can say that agile managers embrace the changes.

Now, we start exploring the JIRA Tool.

Figure-1: This is the account creation page. You can create the account via Google or via Email.It will activate your free version.

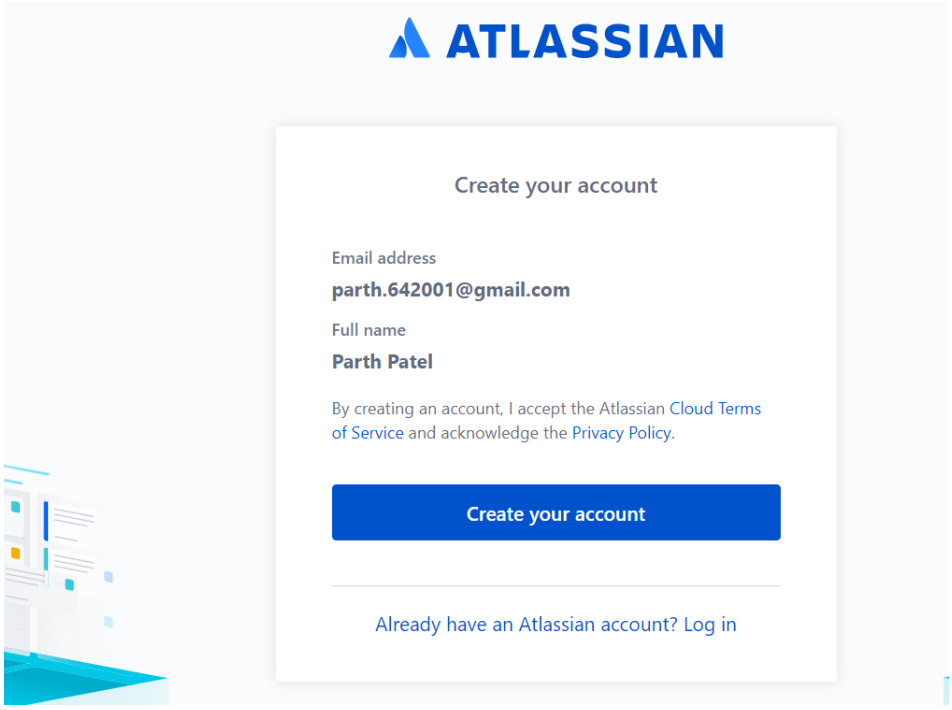


Figure-2: This is the screen where you will get prompt to create your account

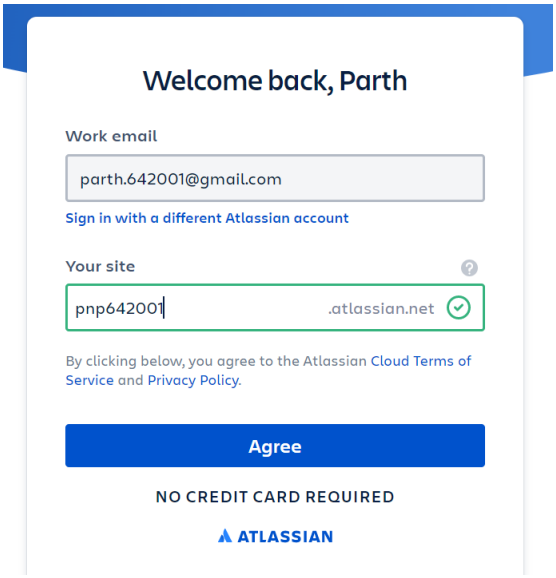


Figure-3: In order to uniquely identify you, it will ask to enter an unique user name.

Upon green tick, you need to click on Agree button.

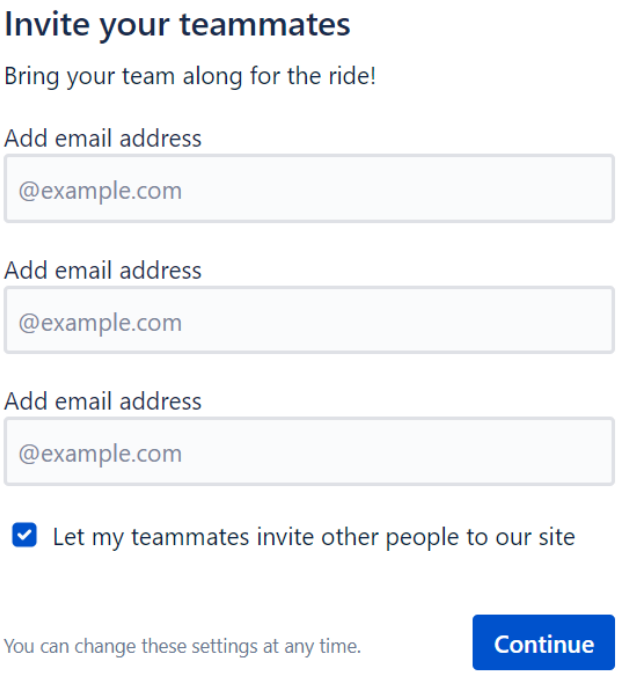


Figure-4: Once, your account is activated, you will come across the above window, where you can add your team member in your project.

NOTE: you can also add team member after creating the project.

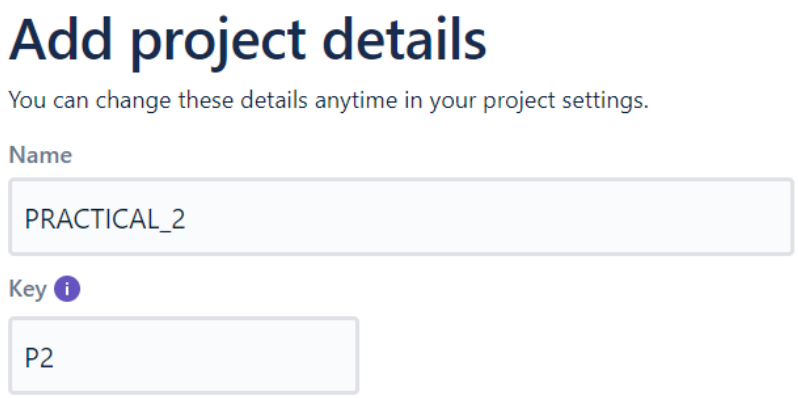


Figure-5: Then, you will be asked to create your first project.

NOTE: Key is mandatory. It is useful in identifying the project.

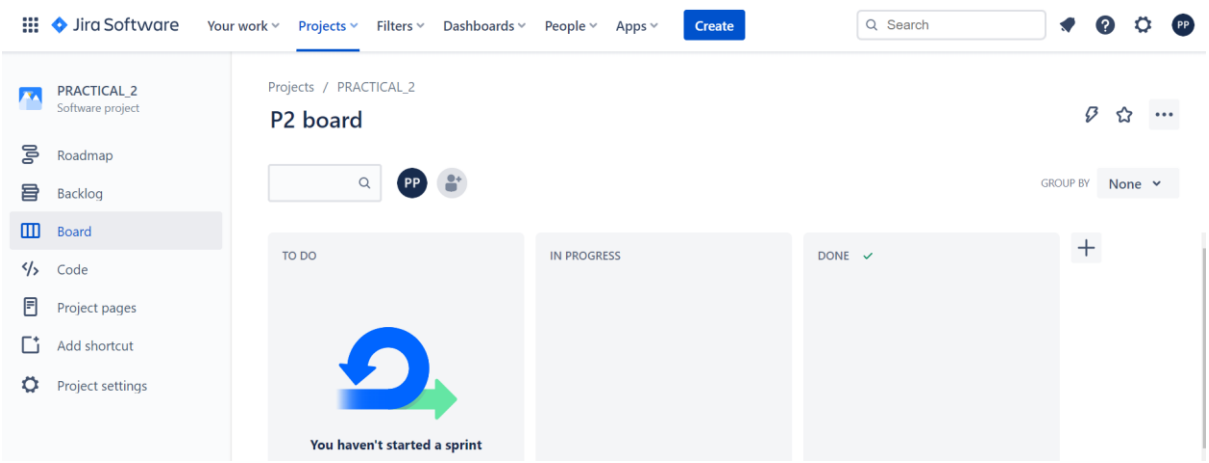


Figure-6: This is the main window that you will see once, a project is created or you are included in the project.

Now, we will explore the functionalities, one by one.

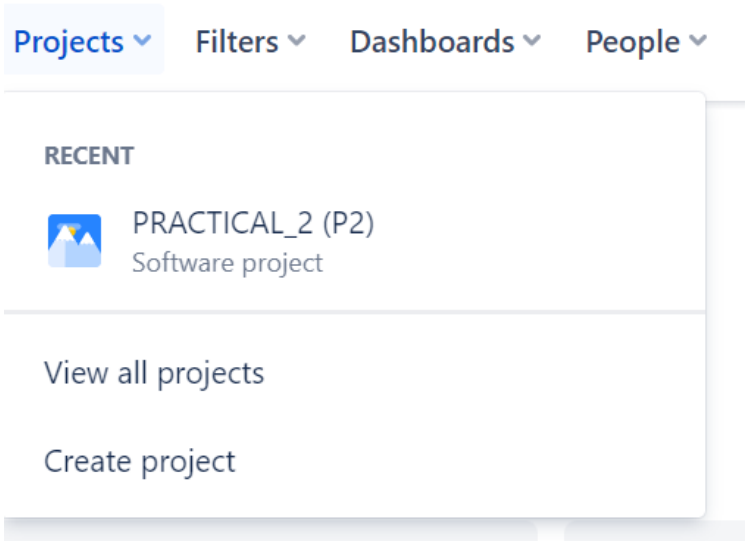


Figure-7: You can create Project and also, you can view all the projects.

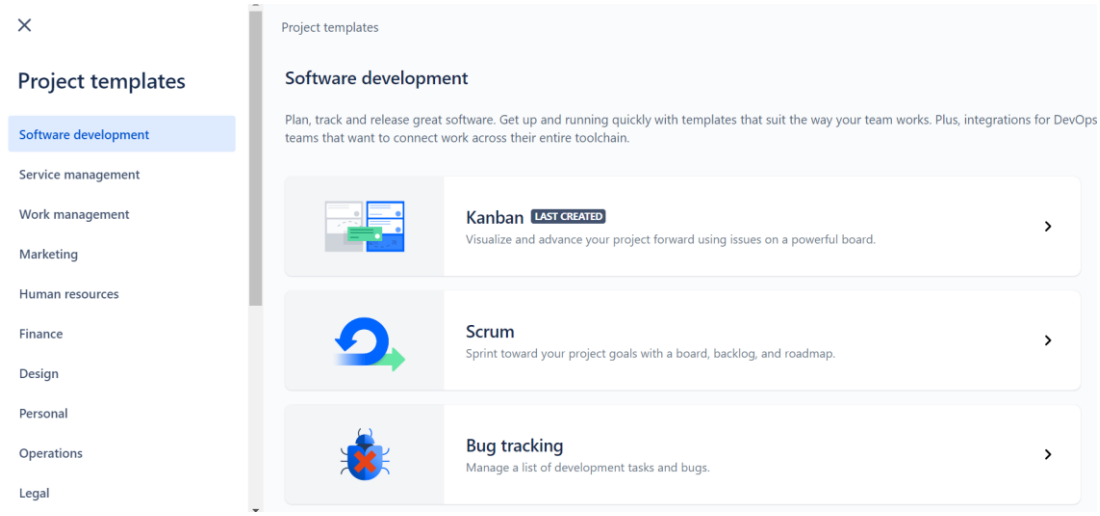


Figure-8: On clicking the create project, above window will appear, where different types of templates are available. We will select the SCRUM option.

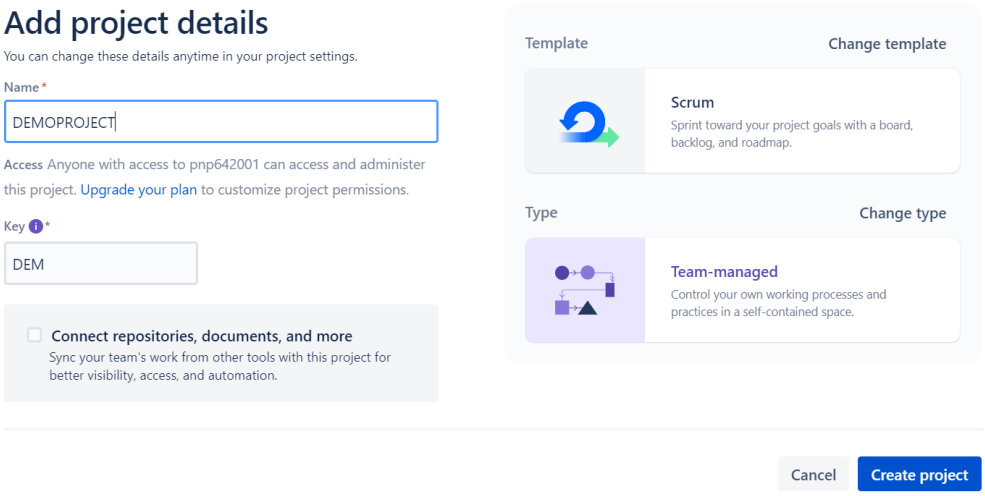


Figure-9: We will give name and key to the project and also, we will select Team Managed Project.

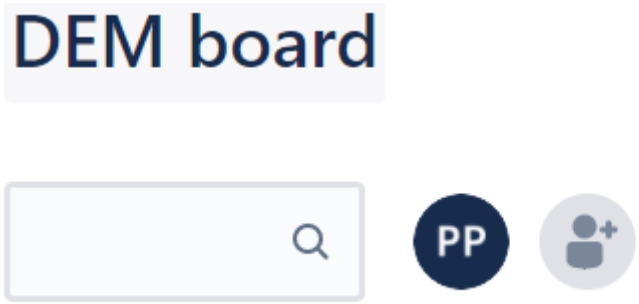


Figure-9: To add a team member, you can click the right most button shown in the image.

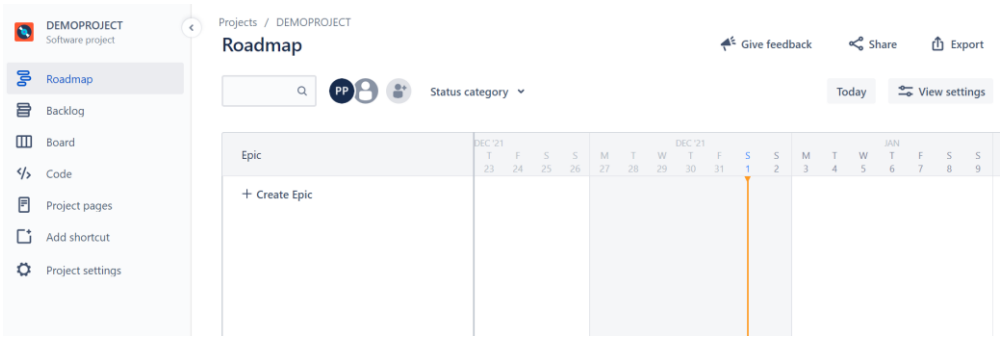


Figure-10: On clicking the roadmap option, you will see the above window, where you can create epics.

EPIC: epic is a large user story which is broken into smaller tasks (user stories) based on the customer or end-user needs

Basically, when team is discussing with the customer, then we use epics to mention the key points and features that customer wants in the project.

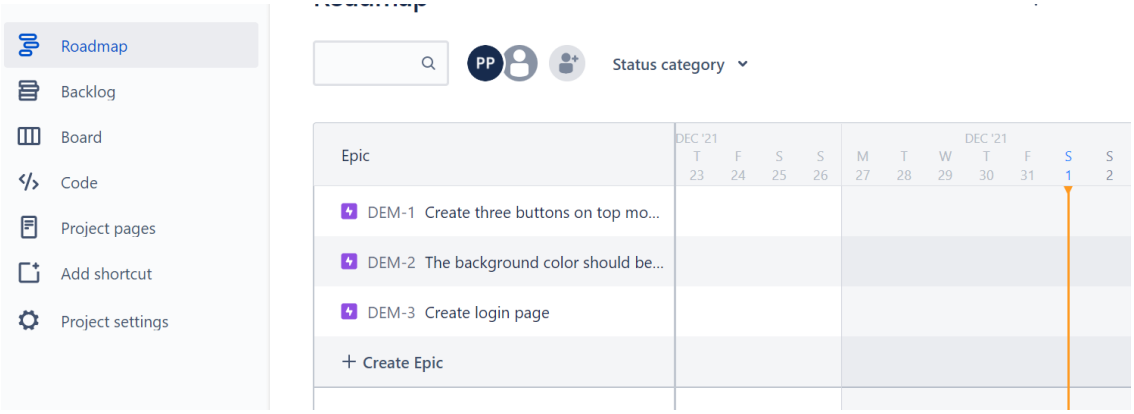


Figure-11: As you can see, we have added three epics.

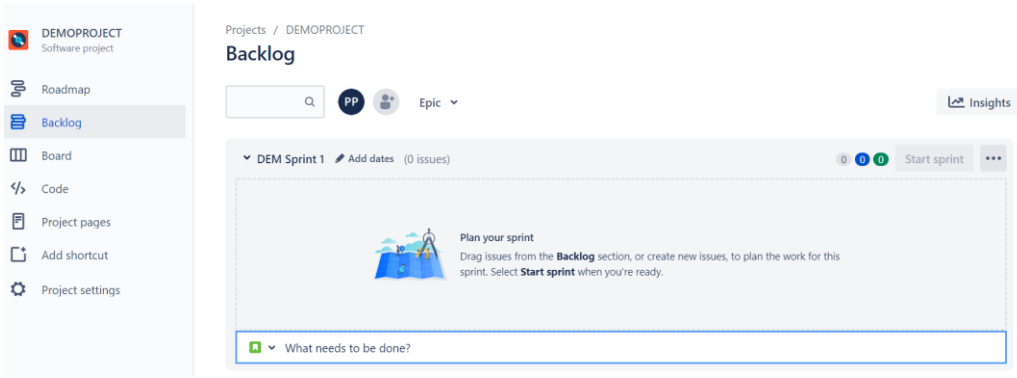


Figure-12: On clicking the backlog option, you see the above window.

BACKLOG: backlog is a list of tasks that represents outstanding work in a project

SPRINT: A sprint is a fixed time period in a continuous development cycle where teams complete work from their product backlog

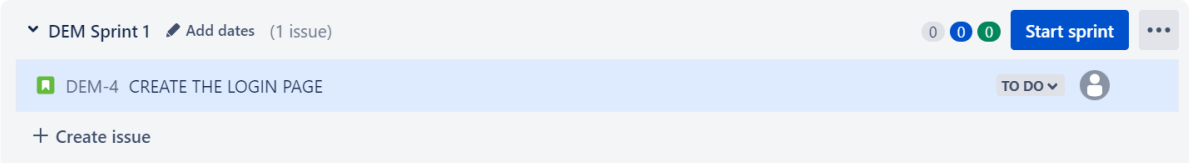


Figure-13: We have created a sprint.

Now, we will add the date and will assign to our newly added team member.

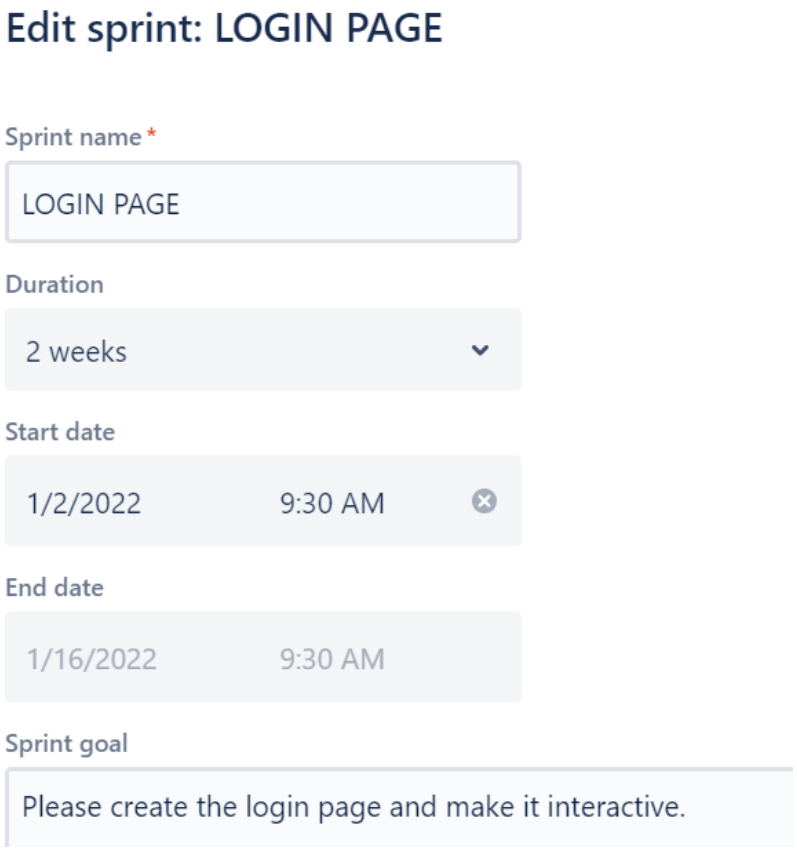


Figure-14: As you see, we have assign the deadline to the sprint along with the necessary goal.

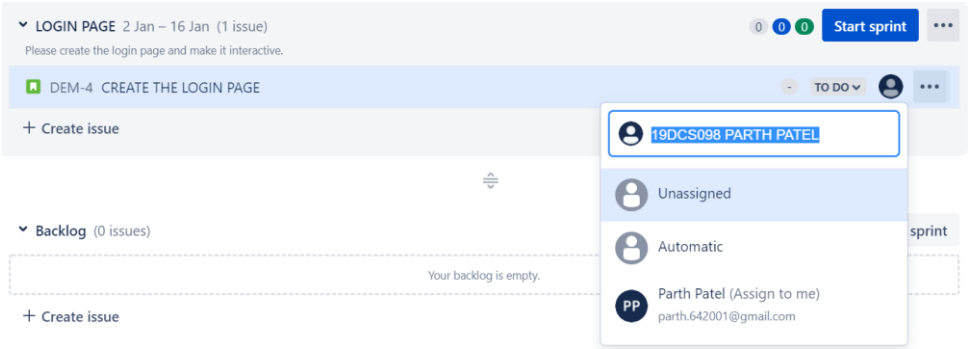


Figure-15: We have assigned the sprint to our team member.

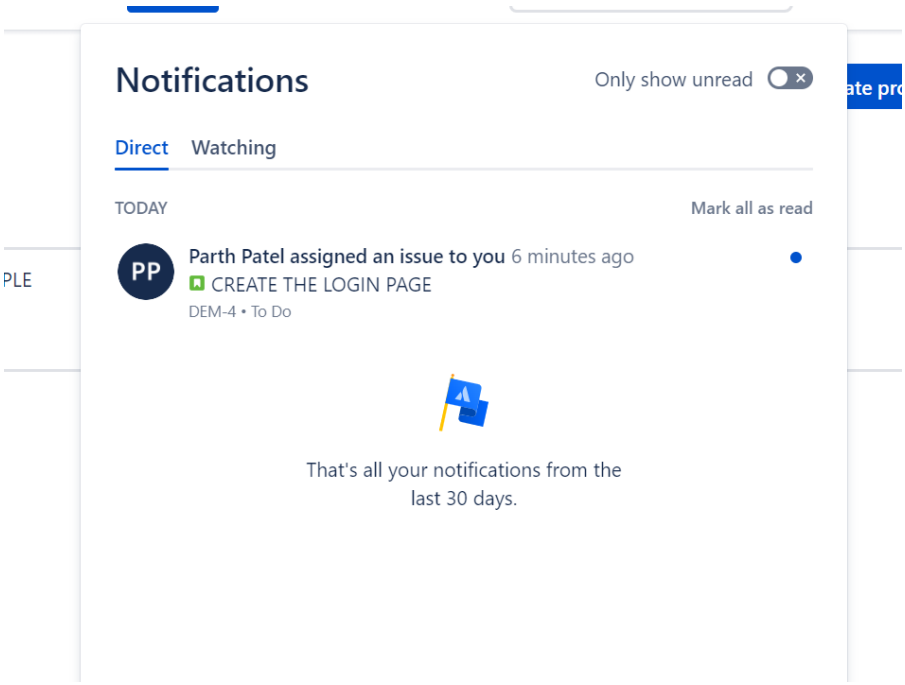


FIGURE-16: Notification arrived in the team mate's account.

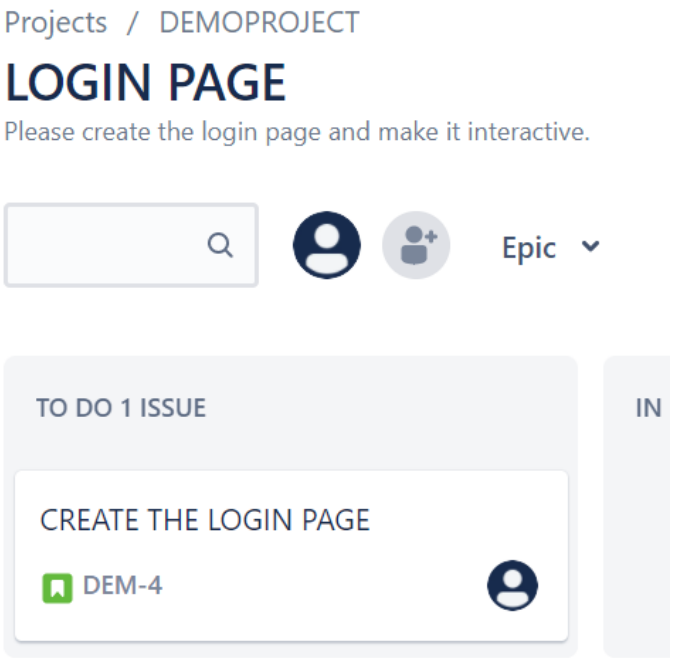


Figure-17: Once the team mate starts the sprint, it is displayed in the to-do list

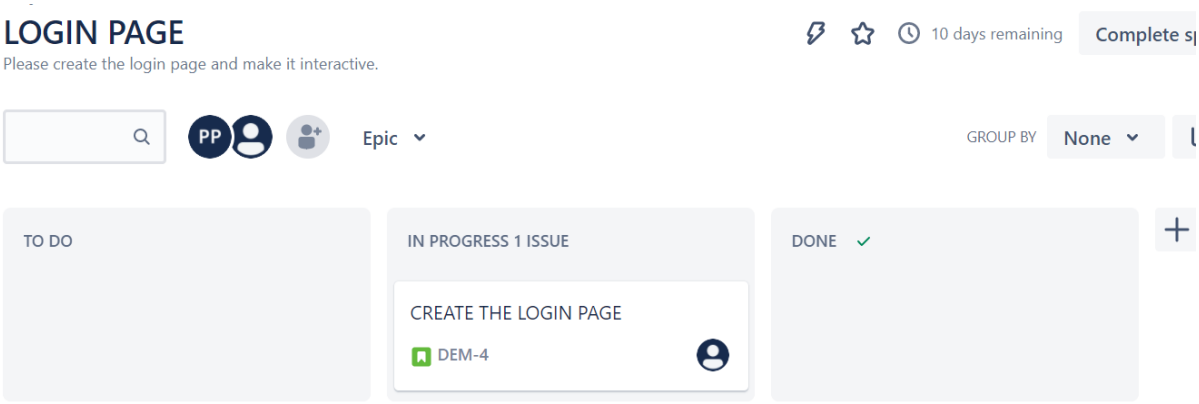


Figure-18: Once the work on sprint starts, it leader will see it in IN PROGRESS panel.

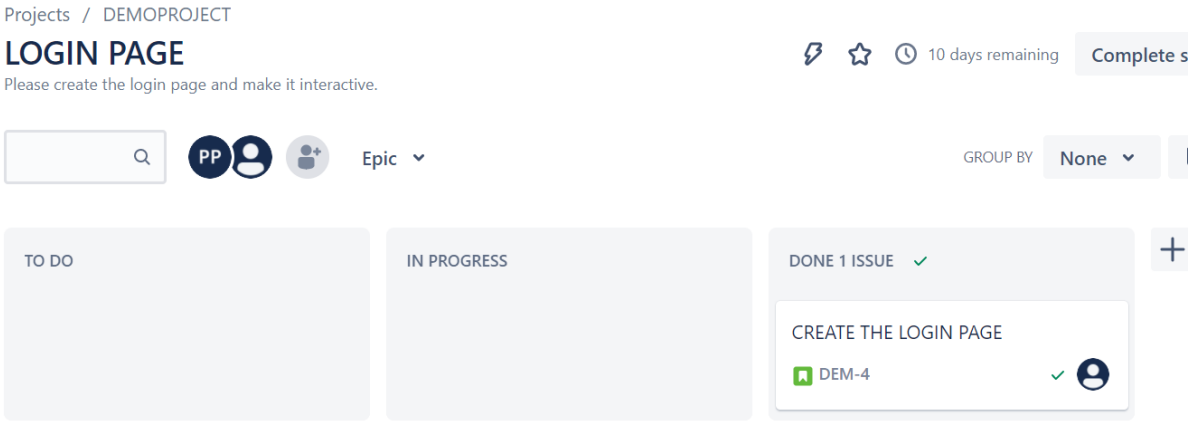


Figure-19: Once, the sprint is completed, it is displayed in the DONE PANEL.

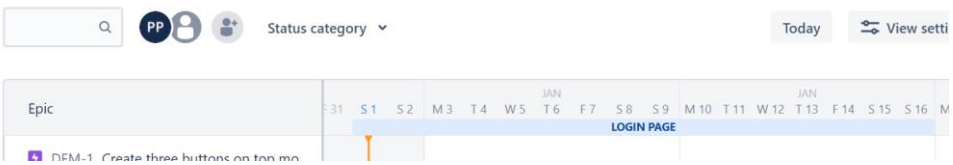


FIGURE-20: Progress updated in the roadmap

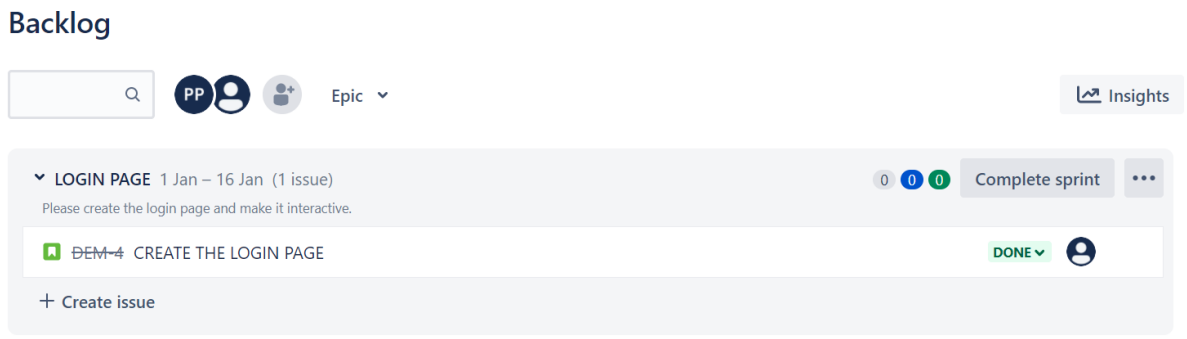


Figure-21: Progress updated in the backlog section.

NOTE: The leader can change the status of the backlog.

There are also lots of other features, which can be explored in the paid version like reports, then also certain features can be activated from the feature settings.

Some note worthy terms with their meanings to mention are as follows:

DAILY SCRUM: The **Daily *Scrum*** is a 15-minute time-boxed event for the Development Team to synchronize activities and create a plan for the next 24 hours.

SCRUM MASTER: A **scrum master** is a professional who leads a team through a project using agile project management techniques

CONCLUSION:

By performing the above practical, I learned about the JIRA TOOL which is used for agile project management and also various features of the tool that will help to manage the project in an efficient way which will help us to save both COST and TIME of the Company.

PRACTICAL-3

AIM:

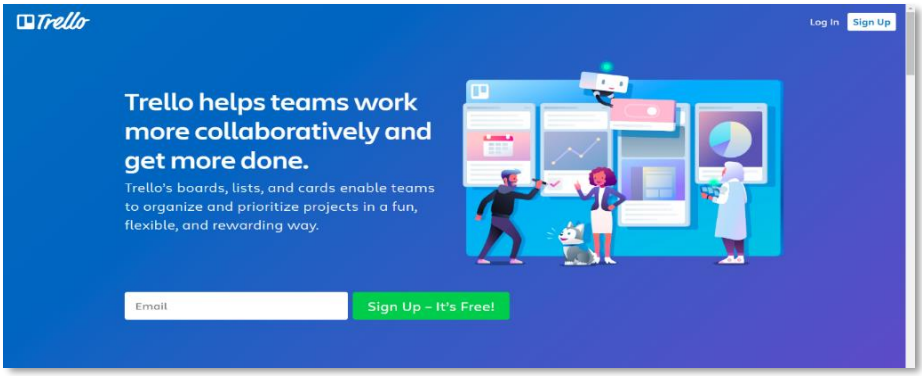
List at least 5 software development planning tools and prepare the detailed case study of Risk Analysis & Management (i.e. Risk Identification, Risk Projection, Risk Refinement, Risk Mitigation.) on “VLC Media Player” mobile application

SOLUTION:

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.

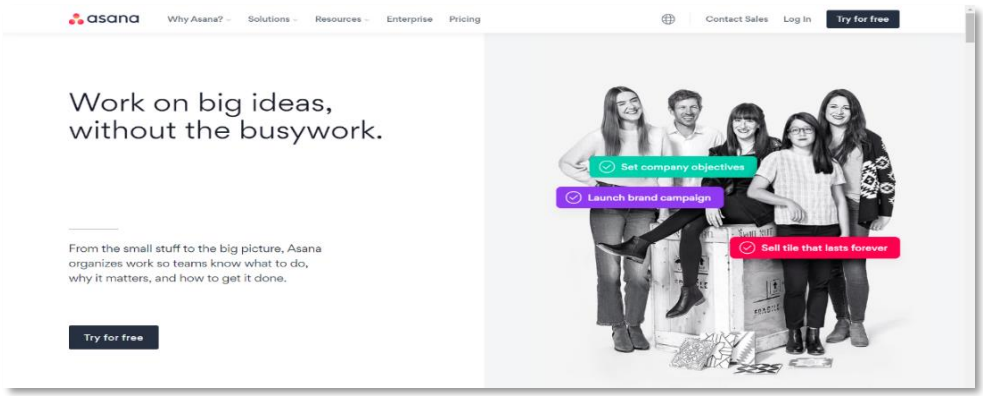
All such business and ecological imperatives bring risk in software development; hence, it is fundamental to manage software projects efficiently

1. TRELLO



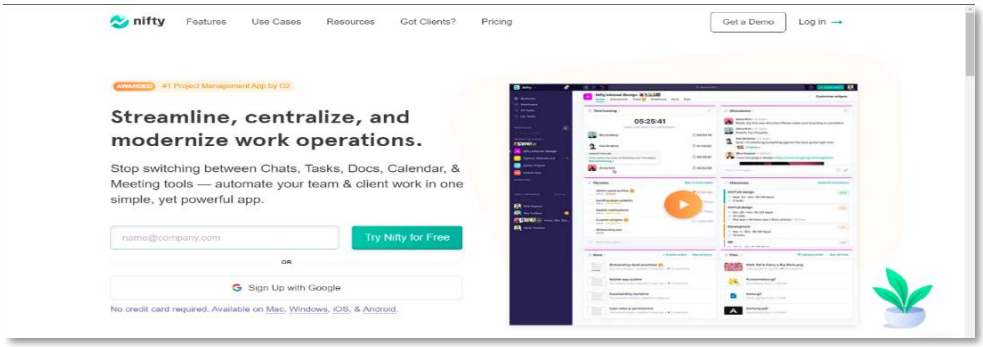
Trello allows you to create boards and fill them with different lists. I went for a simple Kanban-based board and created lists that correspond with statuses (in progress, to do etc.) Trello is a tool that would allow me to create more granular tasks but I’ve decided to use checklists that are built 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.

2. ASANA



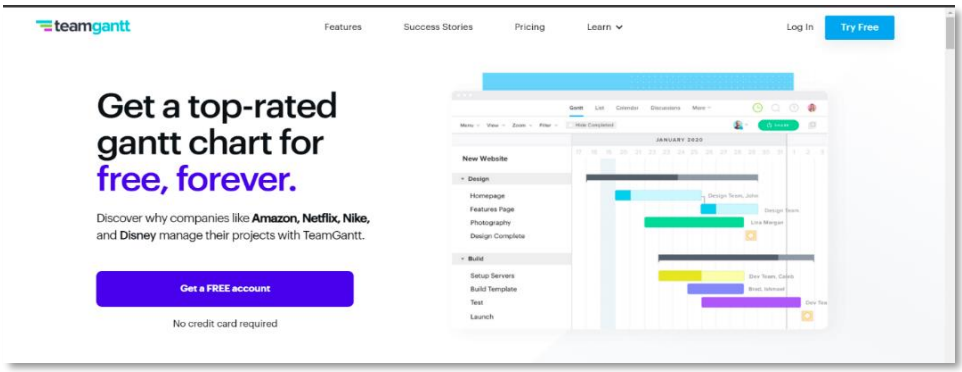
Asana is a popular tool for managing projects of various sizes, and while the free version comes with limited features, it seems to be perfectly suitable for smaller projects. The plan of your project can be displayed as a list, a board or a calendar (see below). The latter view is especially useful 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.

3. NIFTY



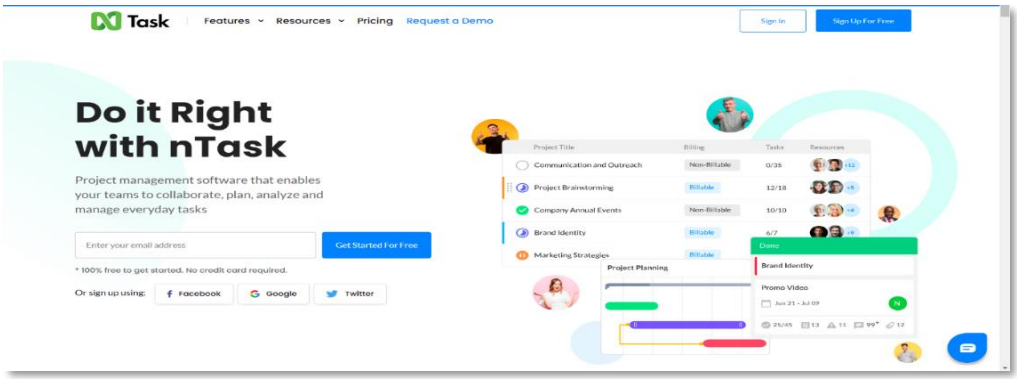
The creators of Nifty pride themselves on building an app that covers project management and team collaboration. The collaboration aspect is supported by the fact that a team chat is a part of this app. What about the project management part? You can plan your project starting with a list of tasks, or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

4. 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

5. 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.

CASE STUDY ON "RISK ANALYSIS & MANAGEMENT"

IN

"EDA APPLICATION"

OVERVIEW:

- **EXPLORATORY DATA ANALYSIS** application aka **EDA app** is an application which gives an generalized overview of an data set by following the EDA principles.

- This application aims at saving the precious time of data analysts by automating their basic tasks that they need to perform in order to determine whether the data set is relevant or not.

- The application will be available as a web application

- We know that development of such large application is time consuming process and also the project may involve high risk factors that can be fatal for the project. As a result, the risk analysis and proper management of the project is necessary.

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 INFORMATION:

Project	EDA Application for CHARUSAT
Risk Type	Operational Risk
Priority	3
Risk Factor	Further Project Execution will depend on the functioning of the application. The application may not work for certain datasets
Probability	30%
Impact	Project completion will be delayed if certain datasets that needs to be processed will cause the stoppage error.
Monitoring Approach	To check the population set of the data sets to be used in the project in the testing phase in order to find and rectify the problem.
Contingency Plan	Modification in the testing strategy and to change the deadlines of the project in order to get sufficient time for the damage control
Estimated Resources	2 additional data analysts to be added by next week for further guidance in both development and testing.

RISK IDENTIFICATION:

According to my point of view, in the risk identification process, certain risk that our application has are as follows:

- **Technical Disadvantage:**

The technology that is being used/provided by the application may become obsolete before launching it or in a short time after the launch.

- **Development Environment:**

There is high probability that, change in development flow or method due to some miscellaneous reasons can affect the application.

For example, if a developer leaves the project and back up is not ready then, the development of the application might get delayed.

The situation may get worse if lead developer leaves the project.

- **Business Impact:**

The market already have the similar applications which can give tremendous competition and impact the brand value.

- **Application not meeting the expectation**

This is the most common risk that every project possesses, that whether their application will satisfy the needs of the end user or will it fail once launched.

Risk Projection

RISK	CATEGORY	PROBABILITY	IMPACT
Technical Disadvantage	TR	30%	4
Development Environment	PR	40%	8
Business Impact	BR	30%	5
Application not meeting the expectation	BR	35%	9

TR- Technical Risk

PR- Project Risk

BR- Business Risk

Risk Planning:

- An effective strategy for dealing with risk must consider three issues
    - Risk mitigation (i.e., avoidance)
    - Risk monitoring
    - Risk management and contingency planning

Risk mitigation (avoidance) is the primary strategy and is achieved through a plan

As we have identified the risks, so, we will be planning to either avoid, monitor, or will try to eliminate the risk. Here, we can divide the risk into multiple sub lists. For example, for the risk of technical disadvantage, we can further more go into details and can try to avoid the risk by segregating the technologies that can cause the risk.

For Development risk, we can have a back up plan so risk can be managed and the impact can be reduced.

Similarly, for the business impact, we can avoid the risk by adding some unique features in our application which the rivals are not offering.

Risk monitoring

We can monitor the development risk by following a specific SDLC model like agile.
For the technical disadvantage, we can hire some versatile developers who can adapt very quickly to the changing tech and can help in reducing the risk.

CONCLUSION:

By performing the above practical, which included the case study for the risk analysis and management, I learned how to estimate and identify the risks involved in the projects, how to measure their impact and how to avoid, monitor and rectify the risks.

PRACTICAL-4

AIM:

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

THEORY:

What is SRS?

- A software requirements specification (SRS) is a document that details how and what the software or system will accomplish.
- It specifies the features and functionality that the product must have in order to meet the needs of all stakeholders (business and users).
- A standard SRS includes:
    - Goal
    - Summary
    - Specific Requirements

Why we need SRS Document?

- An SRS provides you with a comprehensive overview of your entire project. It establishes a single source of truth that all development teams will adhere to.
- It's your game plan, and it keeps all of your teams on the same page, from development to maintenance.

IMPLEMENTATION:

SOFTWARE REQUIREMENT SPECIFICATION FOR EXPLORATORY DATA ANALYSIS APPLICATION

VERSION 1.0 APPROVED

PREPARED BY

PARTH N PATEL (19DCS098)

Devang Patel Institute of Advance Technology and Research

18 January 2022

1. Introduction

1.1 Purpose

Exploratory Data Analysis App is a web application which is intended to serve the primary purpose of data analysts. The application aims at giving a generalized view of data set which is used for data analytics. The purpose of the application is to automate the basic tasks that analysts need to perform for all the data sets before determining whether the data set is relevant for analytics or not.

1.2 Document Conventions

When you read this manual, certain words are represented in different fonts, typefaces, sizes, and weights. This highlighting is systematic; different words are represented in the same style to indicate their inclusion in a specific category. The types of words that are represented this way include the following:

o Regular text conventions are followed as per the IEEE Recommended Practice.

Software Requirements Specifications standard 830-1998.

o This style indicates that the program is an end-user application (as opposed to system software). For example: Use Chrome to browse the Web.

o This style indicates that the Products that uses windows For example: HoloLens,phones

o This style indicates that Keys: Power, Volume up

o For Reference Links : https://windows10.org

1.3 Intended Audience and Reading Suggestions

This document is intended and can be referred by data analysts, data scientists, machine learning engineers, big data engineers, business analysts, software developers, interns, project managers and team leaders.

1.4 Product Scope

The product is mainly intended to solve the issues and hassles experienced by the team while finalizing the data sets that will be used in the project. The product automates the basic steps and gives the team a sense of relief by providing the insights and basic details of the data set through the medium of visualization. The product also provides ease of use with simple interface and user-friendly functioning. The product also reduces approx. 100-120 lines of coding that the team needs to do with all the datasets so in the macro saving lots and lots of hours and energy.

1.5 References

- *https://towardsdatascience.com/exploratory-data-analysis-8fc1cb20fd15*
- *https://www.geeksforgeeks.org/what-is-exploratory-data-analysis/#:~:text=Exploratory%20Data%20Analysis%20(EDA)%20is,statistical%20summary%20and%20graphical%20representations.*
- *https://en.wikipedia.org/wiki/Exploratory_data_analysis*
- *https://r4ds.had.co.nz/exploratory-data-analysis.htm*

2. Overall Description

2.1 Product Perspective

The EDA app is a first-generation application i.e., it has no predecessors. It is a new self-contained product. The product is supposed to be licensed software which will be free to use. It is a web-based implementing system which implements and adhere client-server model. The product has a simple easy to use interface and has potentially no learning curve for its usage. The EDA of the uploaded data set will be shown in the same window and that too in simple manner so as to avoid the complexity that might arise during the process.

2.2 Product Functions

The EDA app codenamed as “project 6” is the first release. The build number is 1.0. One thing to point out is that build number and version is same for now but it will be changed as the product will undergo scheduled updates.

- The functionalities provided in the application:
- Support to wide range of files
- Easy upload process
- Support to upload from google drive
- Clean and clear visualization options
- Coloured graphs as a part of output for better analysis

2.3 User Classes and Characteristics

The following is not applicable to the project.

2.4 Operating Environment

The basic hardware requirement is to have a processor of 2.1 GHz and recommended RAM of 4 GB for smooth processing. RAM size less than 4 GB will also do the job but patience can be tested. For storage options, less than 1 GB of hard disk space will be required in case if one opts to download the report. The app is compatible with all the major operating systems like Windows, Linus, Chrome OS, MAC OS etc.

- For windows, it is recommended to have at least windows 8.1 for enhanced experience.
- For Linux, ubuntu linux or kali linux is recommended.
- Web browser like Chrome or Edge will be required.
- The app is compatible with all the web browsers except the internet explorer.

1.5 Design and Implementation Constraints

In order for software to function properly, only .csv files or excel files need to be uploaded.

The file should not be more than 15 GB in size.

2.6 USER ASSUMPTIONS AND DEPENDENCIES

- The product needs the following access for operation:
- Download access
- Access to browse Files

- Access to upload Files via web browser

3 EXTERNAL INTERFACE REQUIREMENTS

3.1 USER INTERFACES

UPLOAD SECTION

- The upload section plays an important role in the product as the .csv file will be uploaded

DISPLAY SECTION

- The display section is the place where the results of the EDA will be displayed.

3.2 HARDWARE INTERFACES

- Only the recommended configuration (basic requirements of a computer system)
- No other specific hardware is required for the product.

3.3 SOFTWARE INTERFACES

- Browser to load and view the web pages
- Operating System

4 SYSTEM FEATURE

4.1 TIME SAVER

- The product saves the crucial time of analysts as the general EDA required in order to determine whether the dataset is relevant or not is saved.

4.2 DETAILED ANALYTICS

The product gives generalized but detailed report of the analytics of the uploaded dataset.

4.3 CRISP VISUALIZATION

The product gives crisp and clear visualization of the data in the form of graphs and other modes of visualization.

4.4 DETAILED EXPLANATION

The product gives a detailed explanation of the data in a generalized manner.

5 OTHER NON-FUNCTIONAL REQUIREMENTS

5.1 Error Handling

The product shall handle the expected and unexpected error in ways that prevent loss in the quality of analytics.

5.2 Performance Requirements

The system shall accommodate and withstand the max capacity without any crash and degradation of the quality of the analytics

6 OTHER REQUIREMENTS

Not applicable

Appendix A: Glossary

EDA

- exploratory data analysis is an approach of analyzing data sets to summarize their main characteristics, often using statistical graphics and other data visualization methods.

Constraint

- In controls that involve user input, such as text boxes, input constraints are a valuable way to prevent errors. For example, if the only valid input for a particular control is numeric, the control can use appropriate value constraints to enforce this requirement.

Error

- A state in which a problem has occurred.

APPENDIX C: TO BE DETERMINED LIST

- Planning and adding of data cleaning features.
- Improvision of User Interface
- Upgrading the data capacity
- Adding a new panel for twitter sentiment analysis

CONCLUSION:

By performing the practical, I learnt about the basics of srs document, it's importance, why we need it and how to make it.

PRACTICAL-5

AIM:

List at least 10 software design principals & online/offline tools for the software development process and Draw the UML diagram.

THEORY:

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”

Single Responsibility Principle (SRP)

- It's a software engineering theory that says a class should only change for one reason. In other words, it must only be responsible for one thing. We're talking about cohesion here. All elements in a given class hierarchy or module should have the same functional affinity. You can improve your class's cohesiveness by clearly identifying its responsibilities.
- It is denoted by “S”

Open/Closed Principle (OCP)

- The notion states that you should be able to change a class's behaviour without having to edit it. As a result, you can expand the behaviour of the class using composition, interface, and inheritance. You cannot, however, open it to make minor changes.
- It is denoted by “O”

Liskov Substitution Principle (LSP)

- The LSP principle is primarily concerned with the extent to which a software uses inheritance. While inheritance is advantageous, it is best to apply it in a context and in moderation. The principle aims to avoid situations in which classes are only extended through common things. Before conducting inheritance, you must evaluate the class's pre- and post-conditions.
- It is denoted by “L”

Interface Segregation Principle (ISP)

- Many particular interfaces are preferred by ISP over a broad interface. The goal is to create client-specific interfaces that are tightly grained. Interface cohesiveness must be improved, and modules with limited behaviours should be developed. It's difficult to maintain and evolve interfaces with multiple behaviours. As a result, they must be avoided entirely.
- It is denoted by “I”

Dependency Inversion Principle (DIP)

- The Dependency Inversion Principle, symbolised by the letter "D," is the fifth and final principle of SOLID. Programmers should rely on abstractions rather than real classes, according to this notion. This can be divided into two parts: We can divide it into two parts:
- High-level modules must be self-contained from low-level modules.
- Abstractions should be used in both cases.
- Abstractions should be devoid of specifics.
- Abstractions should determine the details.

We'll now look at some of the other fundamental principles besides SOLID.

Keep It Simple

- It's critical to make sure that the program's coding is simple and straightforward. The code should not be difficult to understand for a newcomer. Methods and functions should be as succinct as possible while yet being clear.
- Only one or two problems should be solved by each of them.
- There should also be few conditions in the project code (simple and nested conditions).
- The QA team would benefit from optimising the circumstances because it would be easier to understand and detect faults.

You Aren't Gonna Need It (YAGNI)

- Most programmers fall into the trap of attempting to implement all of the features at the same time, right from the start.
- In the end, some or all of these features will be rendered obsolete. Begin by creating a class with only a few methods.
- After that, as your product takes shape and new requirements emerge, you can add more features. You'll be able to create a lean development software in this manner.
- YAGNI saves you time, effort, and money that you would have spent trying to debug or comprehend the code.

Measure Twice and Cut Once

- If not done correctly, the requirement step of the development life cycle frequently causes more than 50% code errors.
- As a result, a methodical methodology should be created. It's critical to double-check all of the project requirements to ensure that no crucial sections (features) are overlooked or that nothing is added inadvertently.
- After that, create blueprints to guide the entire process and ensure high-quality code all the way through. Always test your project from the ground up to ensure everything is in working order.
- This method produces far more predictable results, particularly when the project's cost is already high. You'll avoid the hassles of having to delete or add code lines to meet requirements.

Don’t Repeat Yourself (DRY)

- Don't make the same mistake twice when coding your code. To put it another way, don't copy-paste your code into different places. Future maintenance will be tough if this is not done.
- The reason for this is that you'll need to update the coding in a few different places. Adjustments in the tests will be required as a result of these changes in order for the results to turn green.
- All of this will necessitate extra time, effort, and financial resources. You can avoid this issue by extracting common logic into functions. Also, if there are any manual tasks that you can automate to keep your code lean, do so.

- The aforementioned methods will aid in the reusability of code in software development without the need to duplicate it.

Least Astonishment

- According to the principle of least astonishment, it's best to avoid designing a feature with a high level of surprise. End-users should be able to assume certain behaviours from your system's components.
- As a result, only clear, predictable, and consistent project outputs will be lucrative. Users will be hesitant to employ features or structures that surprise, surprise, or confound them if they aren't familiar with them.
- You're a software developer who creates products that people can utilise. Thus, creating user-friendly features will pay off handsomely. Make an effort to match the mental models, experiences, and expectations of human beings. Keep in mind that you need to grab the user's attention as soon as feasible. The attention span of today's users has dwindled, as we all know.

SOFTWARE DEVELOPMENT PROCESS TOOLS:

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.

2. 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.

3. 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.

4. 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.

5. 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

6. 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.

7. 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."

8. 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.

9. 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.

10. NetBeans:

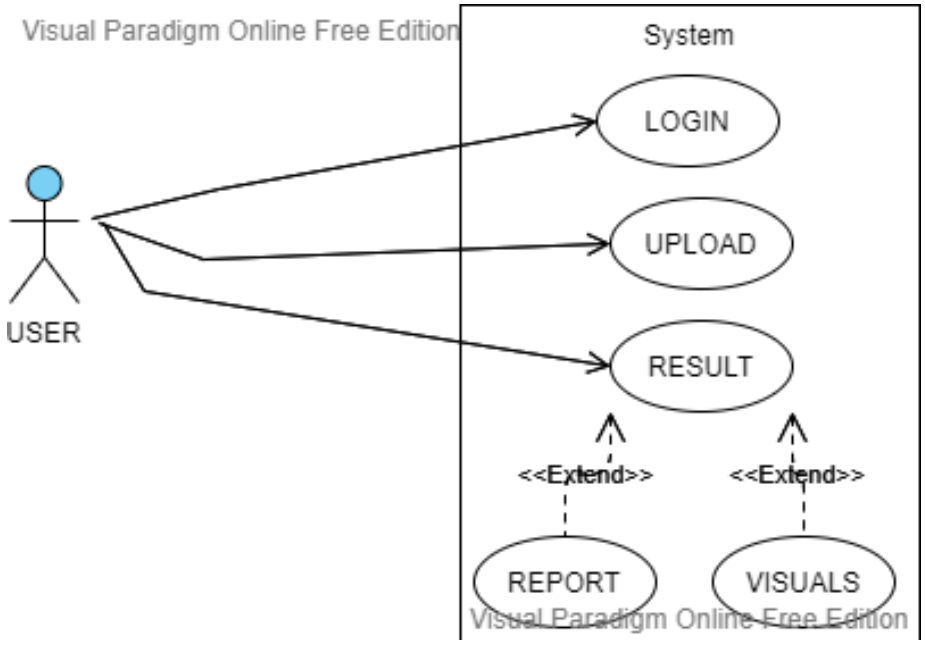
NetBeans is an open source and a free software development tool written in Java that develops world-class web, mobile, and desktop applications easily and quickly. It uses C / C++, PHP, JavaScript, Java etc.

Features:

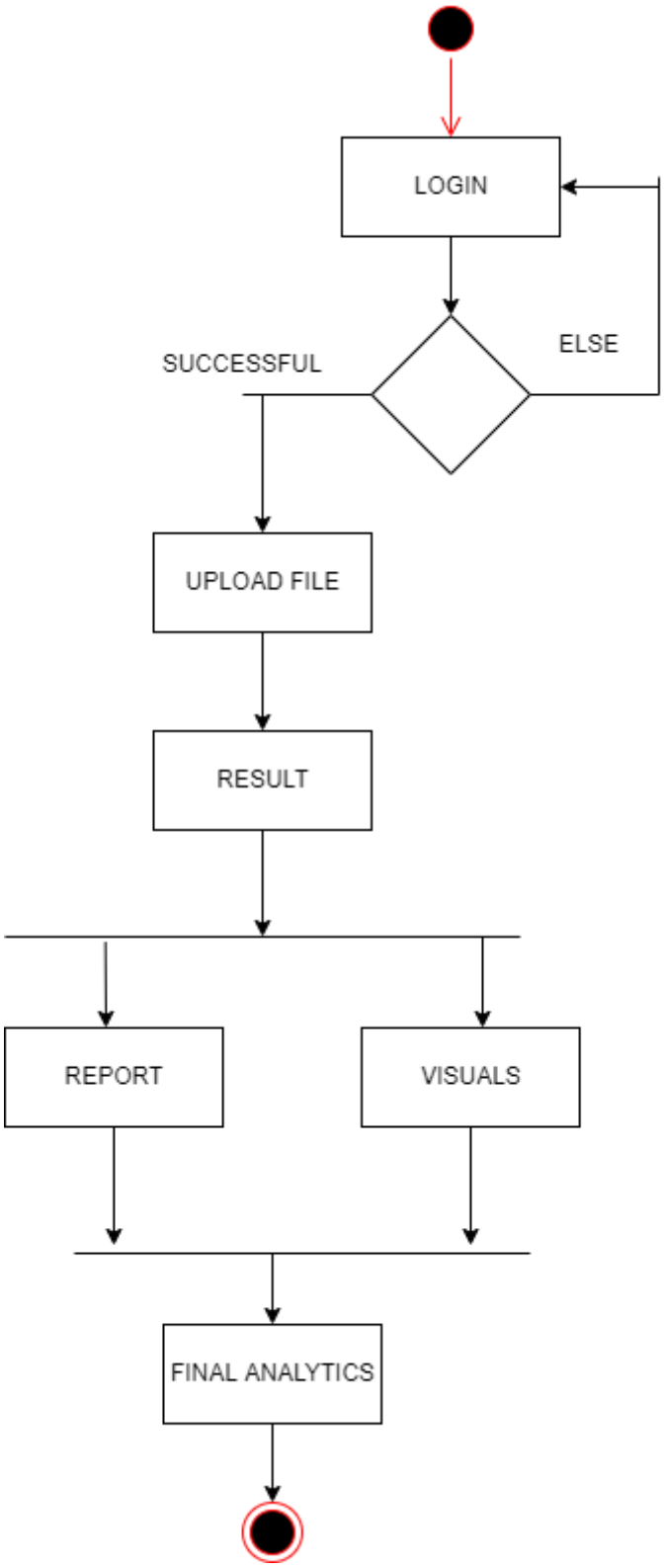
- Support for fast & smart code editing.
- Easy & Efficient Project Management process.
- Rapid User Interface Development.
- Helps to write bug-free code.
- NetBeans IDE offers superior support for C/C++ and PHP developers.

UML DIAGRAM:

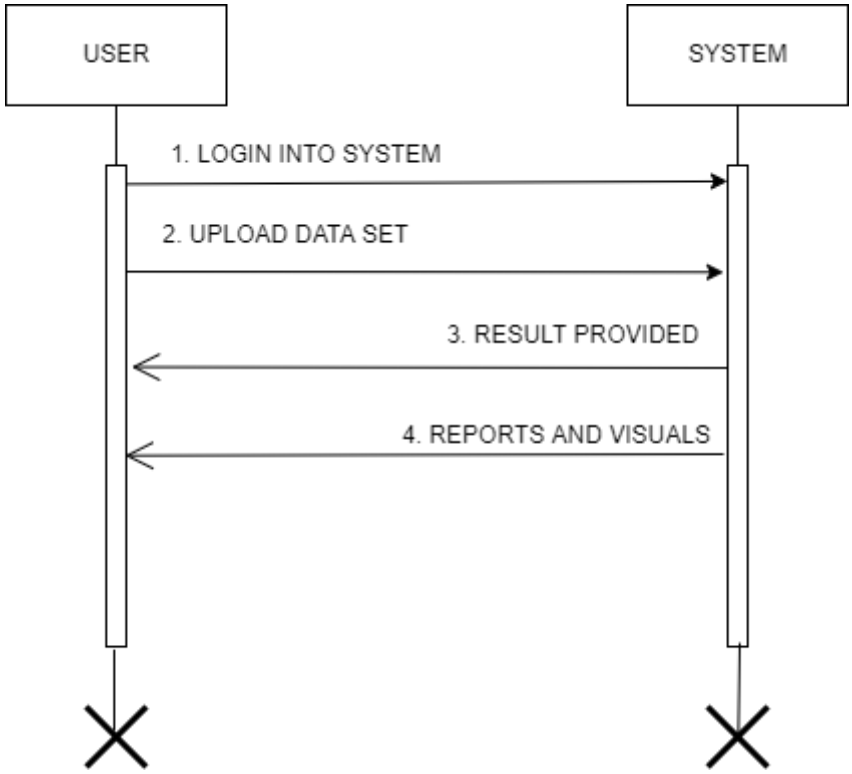
USE-CASE DIAGRAM:



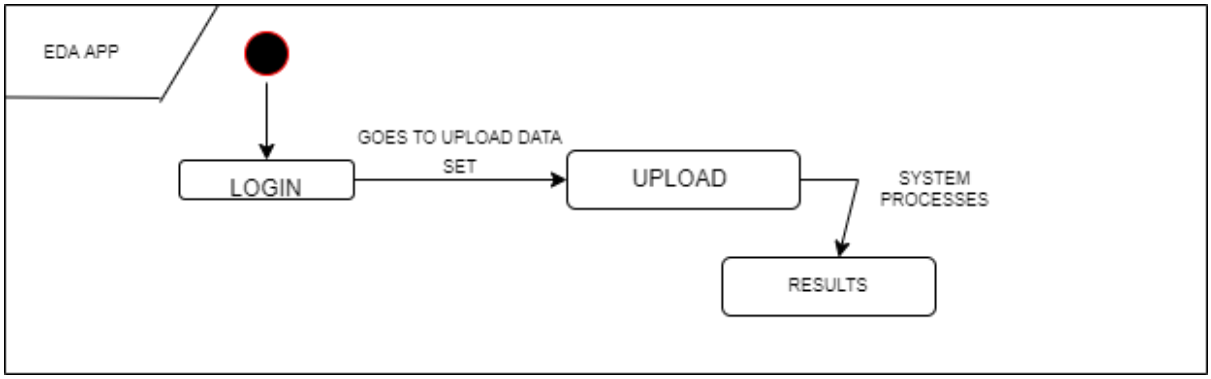
ACTIVITY DIAGRAM:



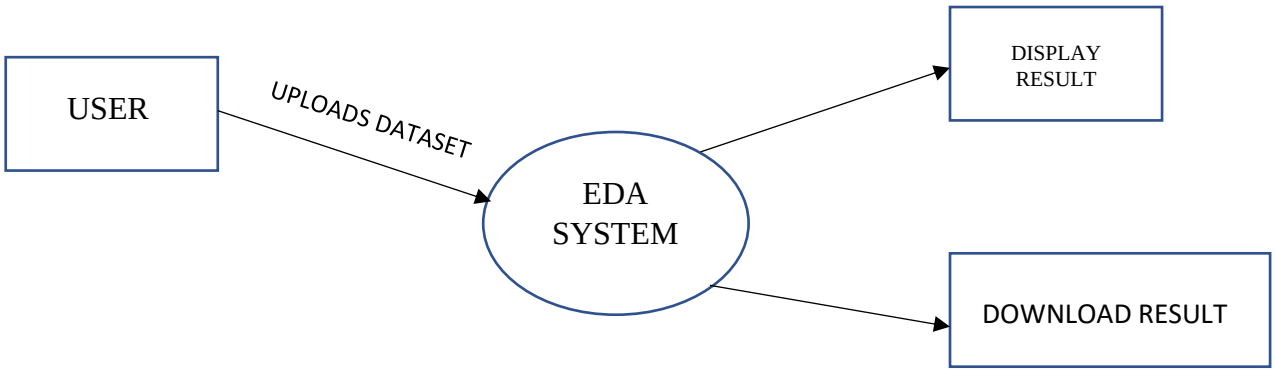
SEQUENCE DIAGRAM:



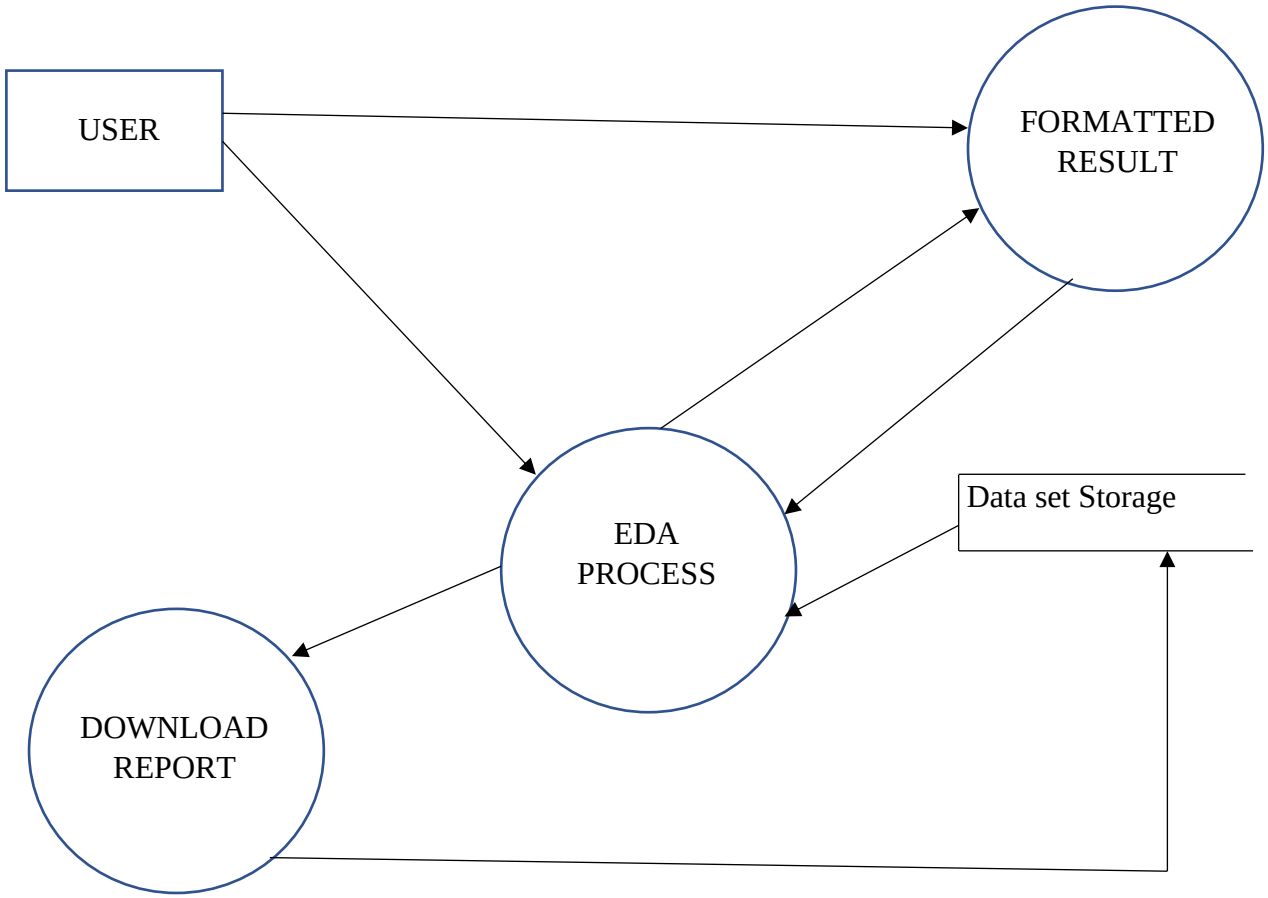
STATE DIAGRAM:



LEVEL 0 DFD:



LEVEL 1 DFD:



CONCLUSION:

By performing the practical, I learnt the basics of software design principles and software development process and UML diagram.

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.

IMPLEMENTATION:

INTRODUCTION:

- Testing is the process of running a software and looking for flaws. Our software must be error-free in order to perform well. If the testing is completed successfully, the software will be free of all errors for majority if not for all.

Principles of Testing:

- All the test should meet the customer requirements
- To make our software testing should be performed by a third party
- Exhaustive testing is not possible. As we need the optimal amount of testing based on the risk assessment of the application.
- All the test to be conducted should be planned before implementing it
- It follows the Pareto rule (80/20 rule) which states that 80% of errors come from 20% of program components.
- Start testing with small parts and extend it to large parts.

10 Types of Common Testing:

- Unit Testing
- Integration Testing
- System Testing
- Sanity Testing
- Smoke Testing
- Interface Testing
- Regression Testing
- Beta/Acceptance Testing

Unit Testing

- It concentrates on the tiniest aspect of software development. This is where we test a single unit.
- The programmer frequently does this task by using sample input and seeing the accompanying results.

Example of it our SGP:

- We will be doing Unit testing, once development of each module is completed, we will perform unit testing. It will be performed by the respective developer.
- Checking if dataframes are working correctly
- Correct precedence

Integration Testing

- The goal is to take unit-tested components and use them to create a software program that is dictated by design.
- Integration testing involves integrating a number of components to achieve a result.

Example of it our SGP:

a**) Black Box testing**: - It is used for validation. In this we ignore internal working mechanism

In our project, if we give the testing to a third party like Capgemini then, they will perform black box testing as they will only have .exe file and srs.

(b) **White Box testing**:- It is used for verification. In this we focus on internal mechanism

If we will perform the testing, then we will have access to both source code and srs, so it will be perfect example of white testing.

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

Our Example:

- While developing the project, we will have bugs and thus, after identifying and resolving the bug, we will perform the regression to make sure that our program is running rightly.

Smoke Testing

- This test is done to make sure that software under testing is ready or stable for further testing.

Our Example:

- As we will develop the project in phased manner and each phase will be divided into modules and each module will be developed by different developers and once, module is completed, we will smoke testing it in order to determine that module 1 is working perfectly before moving to next module.

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

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.

Our Example:

- After performing the alpha testing, we will give the software to our professors and peers in order to take their feedback. Thus, it is beta testing.

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

Stress Testing

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

Example:

- We will stress test our product to determine whether it can process big datasets or not and how the RAM is performing.

Reliability Testing

- Reliability Testing is a software testing process that checks whether the software can perform a failure-free operation for a specified time period in a particular environment.

Example:

- We will need to perform it as our product deals with data of all types, so it is important for it to be reliable.

Security Testing

- Security testing is a process intended to reveal flaws in the security mechanisms of an information system that protect data and maintain functionality as intended

Example:

- As data is involved, so we will need to determine that security is sufficient so it can be securely processed.

TEST CASES:

- A test case is a document, which has a set of test data, preconditions, expected results and postconditions, developed for a particular test scenario in order to verify compliance against a specific requirement.
- For us, we will have to create multiple test cases, in such a way that it will have minimal inputs in the form of the data sets and maximal level of inputs.
- Each Test case, will have corresponding inputs, state of testing and expected output.

Example:

- We will give the relevant data set as an input.
- For state, we will give description of the input and also certain specialities of the input data.
- For output, as expected, the desired output data will be given in order to determine whether the product is giving right result for the given input.

TEST SUITES:

- We can simply say that test suite is the collection of relevant test cases for the product.
- Test suite normally covers wide range of test cases to get the relevant result.

TESTING STRATEGY:

- Test Strategy is a set of guidelines that explain the test design and determine how testing needs to be done.

Our approach:

- As far as our testing strategy is concerned, we have a very straight forward approach.
- Firstly, unit testing will be performed by the developers
- The test cases will be designed in module specific manner
- Then, smoke testing will be undertaken before moving to next module.
- After that, integration testing will take place in order to determine the running of the product during the integration process.
- Then, white box testing will take place by the team.
- As we are inexperienced in the testing, we will also undertake black box testing which will be done by the trusted third party.
- Then, once the product is ready, we will do alpha testing, where all the fellow team members will use the product.
- Then, beta testing will be done, where, the peers and professors will use the product in their day-to-day use.

CONCLUSION:

By performing the above practical, I learnt the basics of software testing and the methodology followed and different testing happening in the industry

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.

IMPLEMENTATION:

In the Coding phase, different modules specified in the design document are coded according to the module specification. The major purpose of the coding phase is to use a high-level language to code from the design document created during the design phase, and then to unit test this code.

Code standards are a well-defined and standard style of coding that good software development companies expect their programmers to follow. They frequently create their own coding standards and rules based on what works best for their company and the types of software they create. Maintaining coding standards is critical for programmers; else, code will be rejected during code review.

PURPOSE:

- A coding standard offers the programmes created by different engineers a consistent appearance.
- It increases the code's readability and maintainability while simultaneously reducing its complexity.
- It aids in the reuse of code and the detection of errors.
- It encourages good programming habits and boosts programmers' productivity.

Coding Guidelines Have the Following Benefits:

- Coding rules improve the software's efficiency while also reducing development time.
- Coding rules aid in the early detection of problems, lowering the extra costs spent by the software project.
- When coding rules are followed correctly, the software code becomes more readable and understandable, reducing the code's complexity.
- It lowers the software development's hidden costs.

CODING STANDARD FOR OUR SOFTWARE GROUP PROJECT:

Limited use of Global Variable

- **Avoid the use of globals as much as possible during the development of the project.**
- If required to use, please discuss with the team and find an alternate solution.

Standard header for different modules

- For better understanding and maintenance of the code, the header of different modules should follow some standard format and information. The header format must contain below things that is being used in various companies:
- 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

Naming conventions for local variables, global variables, constants and functions:

- Local variables should be named in snake case.
- Global Variables should contain the word “global” and upper case should be used.
- Constant names should be in upper case with “const” word as its suffix
- Function name should follow camel case with underscore(_) for separating two words.

Indentation:

- Proper indentation is very important to increase the readability of the code. For making the code readable, programmers should use White spaces properly. Some of the spacing conventions are given below:
- There must be a space after giving a comma between two function arguments.
- Each nested block should be properly indented and spaced.
- Proper Indentation should be there at the beginning and at the end of each block in the program.
- All braces should start from a new line and the code following the end of braces also start from a new line.

Use a coding style that isn't too difficult to comprehend.

- The code should be simple to comprehend. Maintenance and debugging are difficult and expensive due to the sophisticated code.

Avoid using the same identifier for different purposes.

- Each variable should have a descriptive and meaningful name that explains why it is being used. When one identifier is used for multiple purposes, this is not conceivable, and the reader may become confused. Furthermore, it makes future enhancements more difficult.

Try not to use GOTO statement

- GOTO statement makes the program unstructured, thus it reduces the understandability of the program and also debugging becomes difficult.

Quality Standard:

- Functionality - We have tried to make it as user friendly as possible. We have added the functionalities such that user has to do the minimum work.
- Reliability - We have tried to make it as reliable as possible. We have also used the best formulae to predict the probability and implemented heatmap for better understanding of the output.
- Usability - It is very easy to use so any user can easily access and use it.

SOFTWARE TESTING:

Selenium:

- Selenium is a portable framework for testing web applications.
- Selenium provides a playback tool for authoring functional tests without the need to learn a test scripting language (Selenium IDE).
- It also provides a test domain-specific language (Selenese) to write tests in a number of popular programming languages, including C#, Groovy, Java, Perl, PHP, Python, Ruby and Scala.
- The tests can then run against most modern web browsers. Selenium runs on Windows, Linux, and macOS. It is open-source software released under the Apache License 2.0.

CONCLUSION:

- By performing the above practical, I learnt how to design the coding standard for the project.