EXP1-SRS & FEASIBILITY

Technical Feasibility –

In Technical Feasibility current resources both hardware software along with required technology are analyzed/assessed to develop project. This technical feasibility study gives report whether there exists correct required resources and technologies which will be used for project development. Along with this, feasibility study also analyzes technical skills and capabilities of technical team, existing technology can be used or not, maintenance and up-gradation is easy or not for chosen technology etc.

Operational Feasibility –

In Operational Feasibility degree of providing service to requirements is analyzed along with how much easy product will be to operate and maintenance after deployment. Along with this other operational scopes are determining usability of product, Determining suggested solution by software development team is acceptable or not etc.

Economic Feasibility –

In Economic Feasibility study cost and benefit of the project is analyzed. Means under this feasibility study a detail analysis is carried out what will be cost of the project for development which includes all required cost for final development like hardware and software resource required, design and development cost and operational cost and so on. After that it is analyzed whether project will be beneficial in terms of finance for organization or not.

Difference between functional and non-functional requirements:

Functional Requirements	Non-Functional Requirements
They define a system or its component.	They define the quality attribute of a system
It specifies, “What the system should do?”	It specifies, “How should the system fulfill the functional requirements?”
User specifies functional requirement.	Non-functional requirement is specified by technical peoples e.g. Architect, Technical leaders and software developers.
It is mandatory to meet these requirements.	It is not mandatory to meet these requirements.
It is captured in use case.	It is captured as a quality attribute.
Defined at a component level.	Applied to a whole system.
Helps you to verify the functionality of the software.	Helps you to verify the performance of the software.
Functional Testing like System, Integration, End to End, API testing, etc are done.	Non-Functional Testing like Performance, Stress, Usability, Security testing, etc are done.
Usually easy to define.	Usually more difficult to define.

Examples of functional and non-functional requirements:

Below you can check the list of functional and non-functional requirements examples:

Functional Requirements Example:

1. Authentication of a user when he/she tries to log into the system.
2. System shutdown in the case of a cyber attack.
3. Verification email is sent to user whenever he/she registers for the first time on some software system.

Non-functional Requirements Example:

1. Emails should be sent with a latency of no greater than 12 hours.
2. Each request should be processed within 10 seconds.
3. The site should load in 3 seconds when the number of simultaneous users are > 10000

EXP2-USER STORY

User stories are part of an agile approach that helps shift the focus from writing about requirements to talking about them. All agile user stories include a written sentence or two and, more importantly, a series of conversations about the desired functionality.

EXP3-USECASE DIAGRAM & SPECIFICATIONS

Difference between include and extend in use case diagram?

Extend is used when a use case conditionally adds steps to another first class use case.
For example, imagine "Withdraw Cash" is a use case of an ATM machine. "Assess Fee"would extend Withdraw Cash and describe the conditional "extension point" that is represent when the ATM user doesn't bank at the ATM's owning institution. Notice that the basic "Withdraw Cash" use case stands on its own, without the extension.

Include is used to extract use case fragments that are duplicated in multiple use cases. Theincluded use case cannot stand alone and the original use case is not complete without the included one. This should be used carefully an only in cases where the duplication is significant and exists by design (rather than by coincidence).
For example, the flow of events that occurs at the beginning of every ATM use case (when the user puts in their ATM card, enters their PIN, and is shown the main menu) would be a good candidate for an include.

EXP3-USECASE DIAGRAM & SPECIFICATIONS

What is an Activity Diagram in UML?

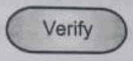
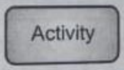
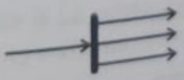
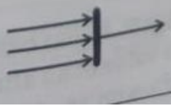
ACTIVITY DIAGRAM is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The basic purpose of activity diagrams is to capture the dynamic behavior of the system.. It is also called object-oriented flowchart.

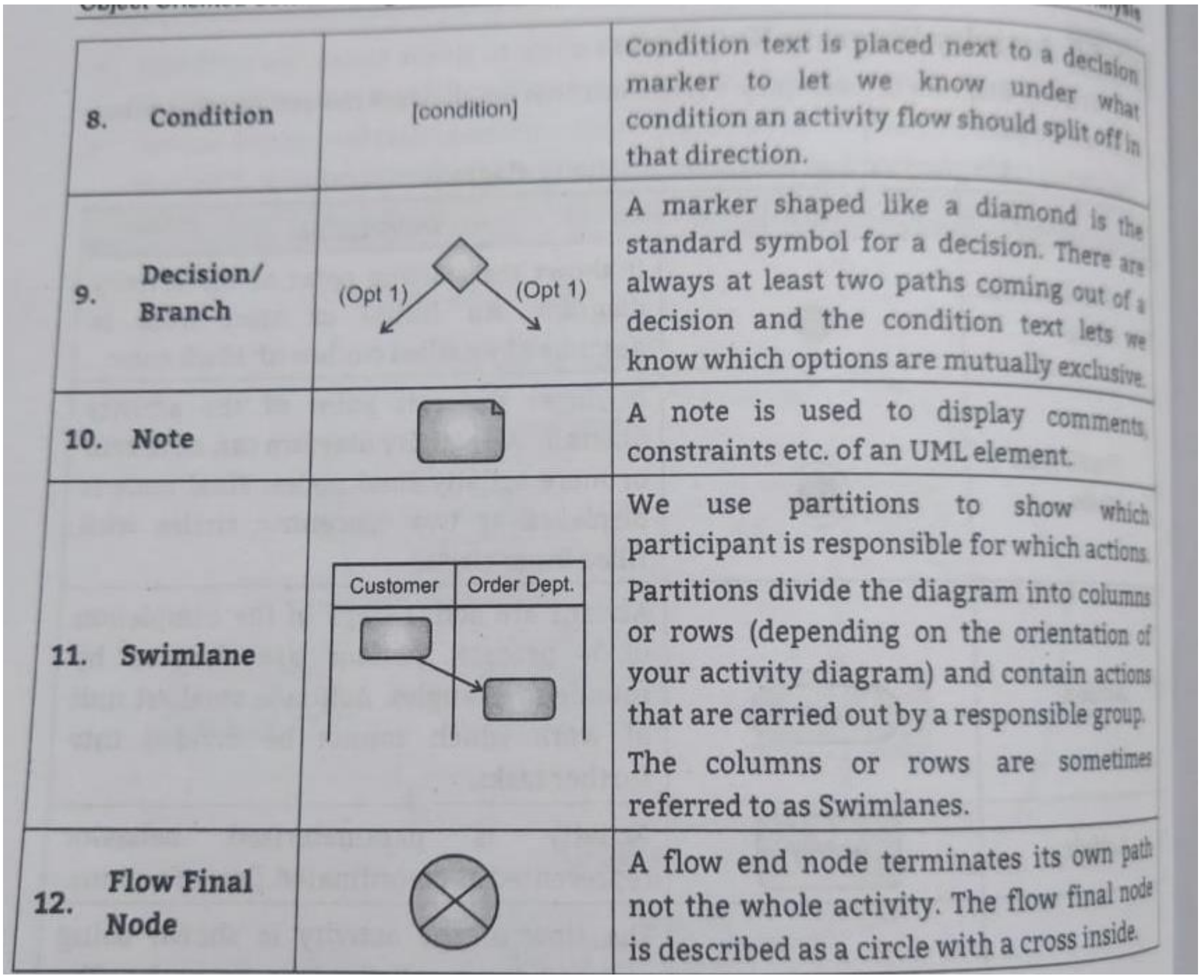
- Activity diagram is also called as **object-oriented flowcharts**.
- Activity diagrams consist of activities that are made up of smaller actions.
- Activity is a behavior that is divided into one or more actions.
- It uses action nodes, control nodes and object nodes.
- An activity partition or a swimlane is a high-level grouping of a set of related actions.
- Fork and join nodes are used to generate concurrent flows within an activity.

  Activity diagram is used to model business processes and workflows.

...izes the activity that takes

Following table gives various notations in activity diagram.

Name	Symbol	Description
Start/Initial Node	●	It shows the starting point of the activity diagram. An initial or start node is described by a filled circle with black color.
Final/Exit Node	◉	It shows the exit point of the activity diagram. An activity diagram can have zero or more activity final nodes. Final node is displayed as two concentric circles with filled inner circle,
Action		Actions are active steps in the completion of a process. Actions are denoted by rounded rectangles. Action is smallest unit of work which cannot be divided into further tasks.
Activity		Activity is parameterized behavior represented as co-ordinated flow of actions.
Transition/ Edge/Path	→	The flow of the activity is shown using arrowed lines called edges or paths. The arrowhead on an activity edge shows the direction of flow from one action to the next. A line going into a node is called an incoming edge, and a line exiting a node is called an outgoing edge.
Fork Node		It is used to show the parallel or concurrent actions. Steps that occur at the same time are said to occur concurrently or in parallel. Fork has single incoming flow and multiple outing flows.
Join Node		The join means that all incoming actions must finish before the flow can proceed past the join. Join has multiple incoming flows and single outing flow.



8.	Condition	[condition]	Condition text is placed next to a decision marker to let we know under what condition an activity flow should split off in that direction.
9.	Decision/ Branch	(Opt 1) (Opt 1)	A marker shaped like a diamond is the standard symbol for a decision. There are always at least two paths coming out of a decision and the condition text lets we know which options are mutually exclusive.
10.	Note		A note is used to display comments, constraints etc. of an UML element.
11.	Swimlane	Customer Order Dept.	We use partitions to show which participant is responsible for which actions. Partitions divide the diagram into columns or rows (depending on the orientation of your activity diagram) and contain actions that are carried out by a responsible group. The columns or rows are sometimes referred to as Swimlanes.
12.	Flow Final Node		A flow end node terminates its own path not the whole activity. The flow final node is described as a circle with a cross inside.

Operator	Meaning
`alt`	Alternative multiple fragments; only the one whose condition is true will execute
`opt`	Optional; the fragment executes only if the supplied condition is true. Equivalent to an `alt` with only one trace
`par`	Parallel; each fragment is run in parallel.
`loop`	Loop; the fragment may execute multiple times, and the guard indicates the basis of iteration

Operator	Meaning
`region`	Critical region; the fragment can have only one thread executing it at once.
`neg`	Negative; the fragment shows an invalid interaction.
`ref`	Reference; refers to an interaction defined on another diagram. The frame is drawn to cover the lifelines involved in the interaction. You can define parameters and a return value.
`sd`	Sequence diagram; used to surround an entire sequence diagram, if you wish.

EXP5- COMPONENT & DEPLOYMENT DIAGRAM

Deployment diagrams are used for describing the hardware components, where software components are deployed. Component diagrams and deployment diagrams are closely related.

Component diagrams are used to describe the components and deployment diagrams shows how they are deployed in hardware.

EXP6 -GANNT CHART

What Is a Gantt Chart?

A Gantt chart is a bar chart that provides a visual view of project tasks scheduled over time. A Gantt chart is used for project planning: it's a useful way of showing what work is scheduled to be done on specific days. It helps project managers and team members view the start dates, end dates and milestones of a project schedule in one simple stacked bar chart.

What Is a Program Evaluation Review Technique (PERT) Chart?

A PERT chart is a project management tool that provides a graphical representation of a project's timeline. The Program Evaluation Review Technique (PERT) breaks down the individual tasks of a project for analysis. PERT charts are considered preferable to Gantt charts because they identify task dependencies, but they're often more difficult to interpret.

What Is a Resource Chart?

Resource Chart is essentially a Gantt Chart that consists of stripes (bars) oriented along the horizontal time axis.

Each bar in Resource Charts graphically represents a certain resource being occupied by a certain activity at one moment or another. In this way, the length of each bar corresponds to the duration of the activity, whereas the ends of the bar show the start and finish times for this activity. Resource can be people, equipment, facilities, funding, etc. required for the completion of a project activity.

Differences between the two as listed in the table below:

Gantt chart	PERT chart
Gantt chart is defined as the bar chart.	PERT chart is similar to a network diagram
Gantt chart was developed by Henry L. Gantt.	PERT chart was developed by the United States navy.

Gantt chart is often used for Small Projects	PERT chart can be used for large and complex Projects
Gantt chart focuses on the time required to complete a task	PERT chart focuses on the dependency of relationships.
Gantt chart is simpler and more straightforward	PERT chart could be sometimes confusing and complex but can be used for visualizing critical path

EXP 7 -PLANNING POKER

What is Planning Poker?

Planning poker (also called Scrum poker) helps agile teams estimate the time and effort needed to complete each initiative on their product backlog. The name from this gamified technique is planning poker because participants use physical cards. These cards, which look like playing cards, estimate the number of story points for each backlog story or task up for discussion.

The design of this process was to help software organizations more accurately estimate development timeframes, build consensus among the members of the cross-functional team, and more strategically plan the team’s work.

Planning Poker is a consensus-based technique for estimating, mostly used to estimate effort or relative size of user stories in Scrum.

Planning Poker Estimation Technique

In Planning Poker Estimation Technique, estimates for the user stories are derived by playing planning poker. The entire Scrum team is involved and it results in quick but reliable estimates.

- Planning Poker is played with a deck of cards. As Fibonacci sequence is used, the cards have numbers - 1, 2, 3, 5, 8, 13, 21, 34, etc. These numbers represent the “Story Points”. Each estimator has a deck of cards. The numbers on the cards should be large enough to be visible to all the team members, when one of the team members holds up a card.
- One of the team members is selected as the Moderator. The moderator reads the description of the user story for which estimation is being made. If the estimators have any questions, product owner answers them.
- Each estimator privately selects a card representing his or her estimate. Cards are not shown until all the estimators have made a selection. At that time, all cards are simultaneously turned over and held up so that all team members can see each estimate.
- In the first round, it is very likely that the estimations vary. The high and low estimators explain the reason for their estimates. Care should be taken that all

the discussions are meant for understanding only and nothing is to be taken personally. The moderator has to ensure the same.

- The team can discuss the story and their estimates for a few more minutes.
- The moderator can take notes on the discussion that will be helpful when the specific story is developed. After the discussion, each estimator re-estimates by again selecting a card. Cards are once again kept private until everyone has estimated, at which point they are turned over at the same time.

EX - 8

What is Jira Software?

Jira Software is an agile project management tool that supports any agile methodology, be it scrum, kanban, or your own unique flavor. It is a Bug tracking system, project management software. From agile boards, backlogs, roadmaps, reports, to integrations and add-ons you can plan, track, and manage all your agile software development projects from a single tool.

What is JIRA?

JIRA is an issue tracking product or a software tool developed by Atlassian, commonly used for bug tracking, project management, and issue tracking; it is entirely based on these three aspects. It is widely used in software development and software testing.

Why use JIRA software?

The reason behind using JIRA is :

1. Upfront and fair licensing policy
2. Features that is not available elsewhere
3. Get the latest update on the progress of projects
4. It runs anywhere and recognized with many famous companies
5. Easily extensible and customizable

What is scrum

Scrum is a lightweight framework that helps people, teams and organizations generate value through adaptive solutions for complex problems.

diff between Jenkins and github

Github is a web platform where you can host your files using what's called Repositories either private or public ones, and you can interact with it via using the control versioning such GIT.

In the other hand, Jenkins is a server for automating your deployment and continuous integration, it's a pipeline where your build, tests and deployment are going through.

diff between gitlab and Jenkins

Jenkins is for Continuous Integration whereas Gitlab CI/CD is for Code Collaboration and Version Control.

What are stories, epics, initiatives and tasks?

- Stories, also called "user stories," are short requirements or requests written from the perspective of an end user.
- Epics are large bodies of work that can be broken down into a number of smaller tasks (called stories).
- Initiatives are collections of epics that drive toward a common goal.
- Scrum tasks are detailed pieces of work that are necessary to complete a story.
- Tasks can range from a few hours to several hours (usually up to 12) and are assigned to team members who have the skills or expertise to do them.

What is a Backlog?

A backlog is a list of tasks required to support a larger strategic plan. For example, a product development context contains a prioritized list of items that the team has agreed to work on next. Typical items on a product backlog include user stories, changes to existing functionality, and bug fixes.

One key component that gives a backlog meaning is that its items are ordered by priority. Therefore, the items ranked highest on the list represent the team's most important or urgent items to complete.

What are sprints?

A sprint is a short, time-boxed period when a scrum team works to complete a set amount of work. Sprints are at the very heart of scrum and agile methodologies, and getting sprints right will help your agile team ship better software with fewer headaches.

what is the difference between issues and Bugs ?

Issue types are a JIRA concept and are the underlying objects for request types.Keep track of different types of issues, such as bugs or tasks. Each issue type can be configured differently.

Bug is a problem which impairs or prevents the functions of the product.

What is a Test Plan?

A Test Plan refers to a detailed document that catalogs the test strategy, objectives, schedule, estimations, deadlines, and the resources required for completing that particular project. Think of it as a blueprint for running the tests needed to ensure the software is working properly – controlled by test managers.

EX - 9

What is risk analysis?

Risk analysis is the process of identifying and analyzing potential issues that could negatively impact key business initiatives or projects. This process is done in order to help organizations avoid or mitigate those risks.

EX - 10

What is Jenkins?

Jenkins is a self-contained, open-source automation server that can be used to automate all sorts of tasks related to building, testing, and delivering or deploying software. Jenkins can be installed through native system packages, Docker, or even run standalone by any machine with a Java Runtime Environment (JRE) installed.

Continuous Integration, Continuous Delivery, and Continuous Deployment?

Continuous Integration: A software development process where the changes made to software are integrated into the main code as and when a patch is ready so that the software will be always ready to be - built, tested, deployed, monitored - continuously.

Continuous Delivery: This is a Software Development Process where the continuously integrated (CI) changes will be tested & deployed continuously into a specific environment, generally through a manual release process, after all the quality checks are successful.

Continuous Deployment: A Software Development practice where the continuously integrated (CI) changes are deployed automatically into the target environment after all the quality checks are successful

What are the common use cases Jenkins is used for?

Jenkins being open-source automation can be used for any kind of software-based automation. Some of the common use-cases include but not limited to –

- Software build jobs
- Sanity/Smoke/CI/Regression test jobs
- Web/Data Scraping related jobs
- Code coverage measurement jobs
- General-purpose automation
- Reverse Engineering jobs 2
- Key Decoding jobs & many other jobs where software automation will be applicable.