Laboratory Report

Experiment No - 01

Batch -

Date of Experiment: __________ Date of Submission: __________

Title: Prepare detailed statement of problem for the selected / allotted mini project and identify suitable process model for the same with justification

Evaluation:

1) Attendance [2]	----------------
2) Lab Performance [2]	----------------
3) Oral [1]	----------------

Overall Marks [5] ---------------

Subject In-Charge

Experiment No: - 01

TITLE: Prepare detailed statement of problem for the selected / allotted mini project and identify suitable process model for the same with justification.

PREREQUISITE:

1. Concepts of Object Oriented Programming & Methodology
2. Knowledge of developing applications with front end & back end connectivity.

HARDWARE CONFIGURATION / KIT:

Sr. No	Minimum Hardware Configuration	
1	Processor	800MHz Intel Pentium III or above versions
2	RAM	512 MB
3	HDD	1.5 GB of free disk space

SOFTWARE CONFIGURATION:

Sr. No	Minimum Software Configuration	
1	Operating System	Microsoft Windows Vista/7 or above versions
2	Editor	MS Word, Notepad

Theory: -

1. **Project Scope**

Project scope is the part of project planning that involves determining and documenting a list of specific project goals, deliverables, features, functions, tasks, deadlines, and ultimately costs. In other words, it is what needs to be achieved and the work that must be done to deliver a project.

2. **Problem Statement**

- A problem statement is a clear description of the issue(s), it includes a vision, issue statement, and method used to solve the problem.
- The 5 'W's can be used to spark the discussion about the problem.
- A problem statement expresses the words that will be used to keep the effort focused and it should represent a solvable problem.

3. How to Write a Problem Statement

A problem statement is a clear concise description of the issue(s) that need(s) to be addressed by a problem-solving team. It is used to center and focus the team at the beginning, keep the team on track during the effort, and is used to validate that the effort delivered an outcome that solves the problem statement. It has a specific form:

- Vision - what does the world look like if we solve the problem?
- Issue Statement - one or two sentences that describe the problem using specific issues. It is not a "lack of a solution" statement. For example, our problem is that we don't have an ERP system.
- Method - the process that will get followed to solve the problem. For example, DMAIC (an acronym for Define, Measure, Analyze, Improve and Control) or Kaizen (the Japanese word for "improvement").

4. Defining the problem

Before spending time trying to solve a programming problem, it is essential that your first understand the problem to be solved.

- read, read, read
- ask questions
- clarify anything you are not sure of
- try writing the problem in your own words
- produce an input processing output chart

5. How to get started

- The 5 'W's - Who, What, Where, When and Why - is a great tool that helps get pertinent information out for discussion.
- ***Who*** - Who does the problem affect? Specific groups, organizations, customers, etc.
- ***What*** - What are the boundaries of the problem, e.g. organizational, work flow, geographic, customer, segments, etc. - What is the issue? - What is the impact of the issue? - What impact is the issue causing? - What will happen when it is fixed? - What would happen if we didn’t solve the problem?
- ***When*** - When does the issue occur? - When does it need to be fixed?
- ***Where*** - Where is the issue occurring? Only in certain locations, processes, products, etc.
- ***Why*** - Why is it important that we fix the problem? - What impact does it have on the business or customer? - What impact does it have on all stakeholders, e.g. employees, suppliers, customers, shareholders, etc.? Each of the answers will help to zero in on the specific issue(s) and frame the Issue Statement. Your problem statement should be solvable. That is, it should take a reasonable amount of time to formulate, try and deploy a potential solution.

6. Selection of relevant Process Model

The software process model framework is specific to the project. Thus, it is essential to select the software process model according to the software which is to be developed. The software project is considered efficient if the process model is selected according to the requirements. It is also essential to consider time and cost while choosing a process model as cost and/ or time constraints play an important role in software development. The basic characteristics required to select the process model are project type and associated risks, requirements of the project, and the users.

One of the key features of selecting a process model is to understand the project in terms of size, complexity, funds available, and so on. In addition, the risks which are associated with the project should also be considered. Note that only a few process models emphasize risk assessment. Various other issues related to the project and the risks are listed in Table.

Table 2.1: Selections based on the Project Type and Associated Risks

Project Type and Associated Risks	Waterfall	Prototype	Spiral
Reliability requirements	No	No	Yes
Stable funds	Yes	Yes	No
Reuse components	No	Yes	Yes
Tight project schedule	No	Yes	Yes
Scarcity of resources	No	Yes	Yes

Table 2.2: Selection based on the Requirements of the Project

Requirements of the Project	Waterfall	Prototype	Spiral	RAD	Formal Methods
Requirements are defined early in SDLC	Yes	No	No	Yes	No
Requirements are easily defined and understandable	Yes	No	No	Yes	Yes
Requirements are changed frequently	No	Yes	Yes	No	Yes
Requirements indicate a complex System	No	Yes	Yes	No	No

Table 2.3: Selection based on the Users

User Involvement	**Waterfall**	**Prototype**	**Spiral**	**RAD**	**Formal Methods**
Requires Limited User Involvement	Yes	No	Yes	No	Yes
User participation in all phases	No	Yes	No	Yes	No
No experience of participating in similar projects	No	Yes	Yes	No	Yes

Exercise:

1. Write a problem statement to define the perfect title with the bounded scope of the project.
2. According to Problem Definition stated above, explain which of the Process Model you have selected and why it is selected?

Conclusion:

1. ***A problem well stated is half solved***, Wally Davis taught that one. And he's right, the better the clarity around what the team is attempting to fix, the more efficient they'll be in solving the problem, the solution will better 'fix' the issues, and the team can get back to executing the business versus fixing it.
2. Based on observation, comparison and experience the steps in best life cycle selection are:
   a. Being familiar with various models.
   b. Review and analyze the types of work performed like development, enhancement, and maintenance.
   c. Review the life cycle approach to standards required for your organization, your customer, or the type of project.
   d. Identify a set of phase and phase activities.
   e. Evaluate the effectiveness of the life cycle framework and implement improvements where needed.

Laboratory Report

Experiment No - 02

Batch -

Date of Experiment: __________ Date of Submission: __________

Title: Develop Software Requirement Specification (SRS) document in IEEE format for the project.

Evaluation:

1) Attendance [2]	----------------
2) Lab Performance [2]	----------------
3) Oral [1]	----------------
Overall Marks [5]	---------------

Subject In-Charge

Experiment No: - 02

TITLE: Develop Software Requirement Specification (SRS) document in IEEE format for the project.

PREREQUISITE:

1. Concepts of Object Oriented Programming & Methodology
2. Knowledge of developing applications with front end & back end connectivity.

HARDWARE CONFIGURATION / KIT:

Sr. No	Minimum Hardware Configuration	
1	Processor	800MHz Intel Pentium III or above versions
2	RAM	512 MB
3	HDD	1.5 GB of free disk space

SOFTWARE CONFIGURATION:

Sr. No	Minimum Software Configuration	
1	Operating System	Microsoft Windows Vista/7 or above versions
2	Editor	MS Word, Notepad

Theory: -

1. Introduction to SRS

A **software requirements specification** (**SRS**) is a description of a software system to be developed. The software requirements specification lays out functional and non-functional requirements, and it may include a set of use cases that describe user interactions that the software must provide.

Software requirements specification establishes the basis for an agreement between customers and contractors or suppliers on how the software product should function. Software requirements specification is a rigorous assessment of requirements before the more specific system design stages, and its goal is to reduce later redesign. It should also provide a realistic basis for estimating product costs, risks, and schedules. Used appropriately, software requirements specifications can help prevent software project failure.

The software requirements specification document lists sufficient and necessary requirements for the project development. To derive the requirements, the developer needs to have clear and thorough understanding of the products under development. This is achieved through detailed and continuous communications with the project team and customer throughout the software development process.

A software requirements specification (SRS) is a document that captures complete description about how the system is expected to perform. It is usually signed off at the end of requirements engineering phase.

2. Goals

The Software Requirements Specification (SRS) is a communication tool between users and software designers. The specific goals of the SRS are:

- Facilitating reviews
- Describing the scope of work
- Providing a reference to software designers (i.e. navigation aids, document structure)
- Providing a framework for testing primary and secondary use cases
- Including features to customer requirements
- Providing a platform for ongoing refinement (via incomplete specs or questions)

3. Qualities of SRS:

- Correct
- Unambiguous
- Complete
- Consistent
- Ranked for importance and/or stability
- Verifiable
- Modifiable
- Traceable

Exercise: -

Prepare SRS for the project you have selected for the Experiment No 1.

Conclusion: -

A **Software Requirements Specification (SRS)** (also known as a System Requirements Specification) is a document or set of documentation that describes the features and behavior of a system or software application. It includes a variety of elements that attempts to define the intended functionality required by the customer to satisfy their different users.

Laboratory Report

Experiment No - 03

Batch -

Date of Experiment: __________ Date of Submission: __________

Title: Identify scenarios & develop UML Use case and Class Diagram for the project

Evaluation:

1) Attendance [2]	----------------
2) Lab Performance [2]	----------------
3) Oral [1]	----------------
Overall Marks [5]	---------------

Subject In-Charge

Experiment No: - 03

TITLE: Identify scenarios & develop UML Use case and Class Diagram for the project.

PREREQUISITE:

1. Concepts of Object Oriented Programming & Methodology
2. Knowledge of developing applications with front end & back end connectivity.

HARDWARE CONFIGURATION / KIT:

Sr. No	Minimum Hardware Configuration	
1	Processor	800MHz Intel Pentium III or above versions
2	RAM	512 MB
3	HDD	1.5 GB of free disk space

SOFTWARE CONFIGURATION:

Sr. No	Minimum Software Configuration	
1	Operating System	Microsoft Windows Vista/7 or above versions
2	Editor	MS Word, Notepad

Theory: -

UML is a standardized modeling language used for software design. It provides a graphical notation for representing software designs using a variety of diagrams, such as class diagrams, activity diagrams, and sequence diagrams.

1. **Use Case:**

A use case diagram software and system engineering term that describe how a user uses a system to accomplish particular goal. A use case act as a software modeling technique that defines the features to be implemented and the resolution of any errors that may be encountered.

In software Engineering and Project Management, a use case diagram is a type of diagram that is used to visualize the relationships and interactions between different actors (users, systems, or external entities) and the functionalities provided by a system or application.

Use case diagrams can be used to identify and analyze the different scenarios or use cases that the system or application needs to support. They can also be used to identify the different types of actors that interact with the system, and the different functionalities that the system needs to provide to these actors.

The use case is made up of a set of possible sequences of interactions between systems and users in a particular environment and related to a particular goal.

General steps to draw Use case Diagram:

1. Identify the actors: Identify the different types of users or external systems that interact with the system or application.
2. Identify the use cases: Identify the various tasks, actions, or functions that the system or application can perform.
3. Determine the relationships: Determine how the actors and use cases interact with each other. Identify the dependencies, associations, and generalizations between actors and use cases.

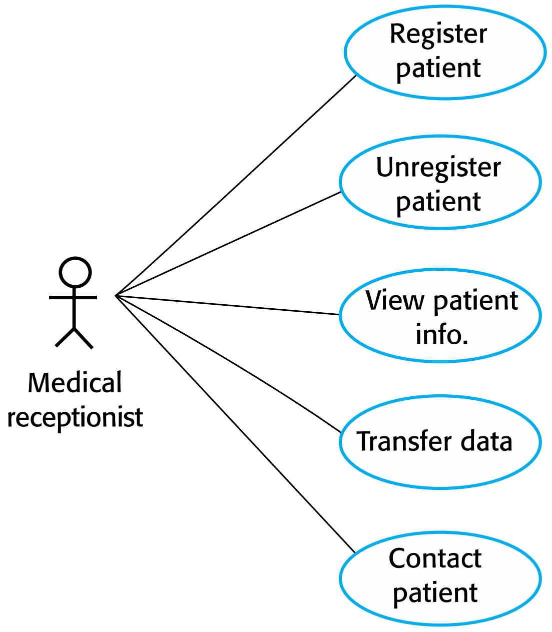


Fig: Use case Diagram for Mentcare System

2. **Class Diagram:**

A class diagram is a type of diagram in the Unified Modeling Language (UML) that is used to represent the structure of a system or software application. It is a graphical representation of classes, interfaces, associations, and other relationships between objects.

The basic elements of a class diagram are:

1. Class: A class represents a set of objects with similar properties, methods, and relationships. It is represented as a rectangular box with the class name written inside.
2. Interface: An interface is a collection of methods that define a set of operations that a class can implement. It is represented as a circle with the interface name written inside.
3. Attribute: An attribute is a property of a class that describes its state. It is represented as a name-value pair inside the class box.
4. Operation: An operation is a method or function that can be performed on a class. It is represented as a name followed by a parameter list inside the class box.

5. Association: An association represents a relationship between two or more classes. It is represented as a line connecting the classes, with optional arrows indicating the direction of the relationship.
6. Aggregation: Aggregation is a special form of association that represents a "whole-part" relationship between two classes. It is represented as a line with a diamond at the end that points to the whole class.
7. Inheritance: Inheritance is a mechanism that allows a new class to be based on an existing class, inheriting all its attributes and operations. It is represented as a line with an open arrowhead pointing to the base class.
8. Dependency: A dependency represents a relationship between two classes where one class depends on the other. It is represented as a dashed line with an arrowhead pointing to the class that is depended upon.

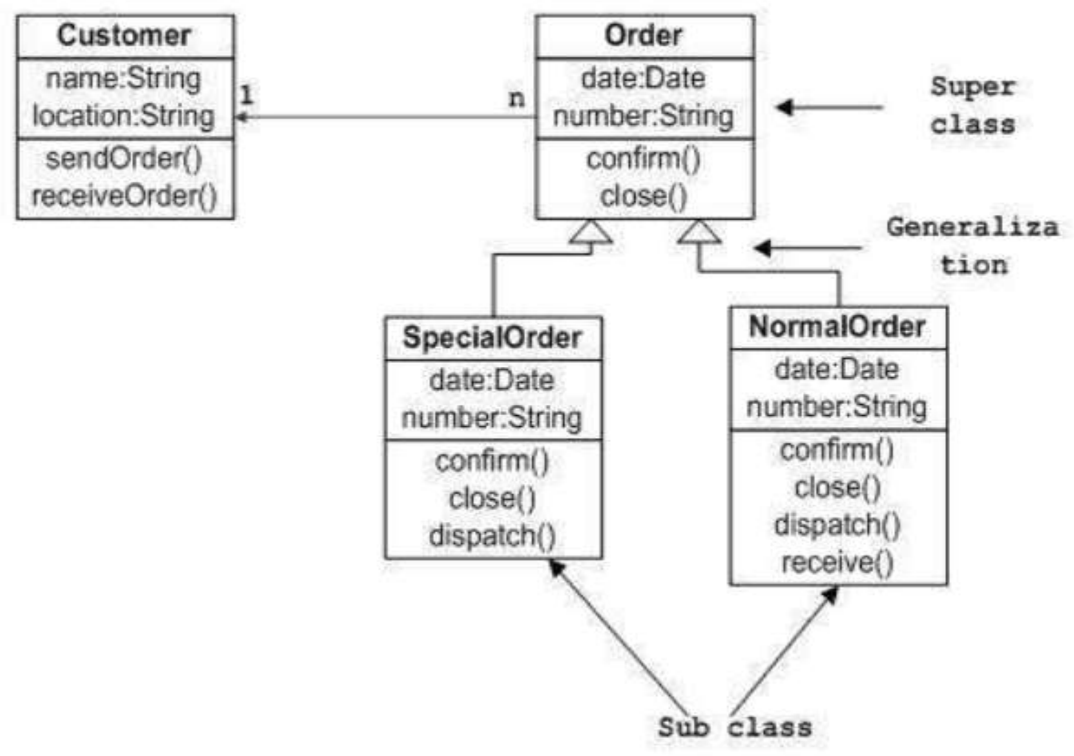


Fig: Sample Class Diagram

Exercise:

1. Identify the different user scenarios and draw Use Case diagram for the project assigned to you.
2. Draw Class diagram for the project assigned to you.

Laboratory Report

Experiment No - 04

Batch -

Date of Experiment: __________ Date of Submission: __________

Title: Draw DFD (up to 2 levels) and prepare Data Dictionary for the project.

Evaluation:

1) Attendance [2]	----------------
2) Lab Performance [2]	----------------
3) Oral [1]	----------------

Overall Marks [5] ---------------

Subject In-Charge

Experiment No: - 04

TITLE: Draw DFD (up to 2 levels) and prepare Data Dictionary for the project.

PREREQUISITE:

1. Concepts of Object Oriented Programming & Methodology
2. Knowledge of developing applications with front end & back end connectivity.

HARDWARE CONFIGURATION / KIT:

Sr. No	**Minimum Hardware Configuration**	
1	Processor	800MHz Intel Pentium III or above versions
2	RAM	512 MB
3	HDD	1.5 GB of free disk space

SOFTWARE CONFIGURATION:

Sr. No	**Minimum Software Configuration**	
1	Operating System	Microsoft Windows Vista/7 or above versions
2	Editor	MS Word, Notepad

Theory: -

Data Flow Diagram (DFD):-

A Data Flow Diagram (DFD) is a graphical representation that depicts the information flow and the processes used for transformation as the data moves from input to output. It illustrates the flow of data between various processes, data stores, and external entities. A DFD typically consists of four components: processes, data stores, data flows, and external entities.

The data flow diagram may be used to represent a system or software at any level of abstraction. DFD provides a mechanism for functional modeling as well as information flow modeling.

Processes are the activities or tasks that are performed on the data, such as data entry, processing, or storage. Data stores are the locations where data is stored, such as databases or files. Data flows represent the movement of data between processes, data stores, and external entities. External entities are the sources or destinations of data, such as users or other systems.

Steps to prepare a Data Flow Diagram:

1. Primary input and output should be carefully noted.
2. All arrows and bubbles should be labeled with meaningful names.
3. Information flow continuity must be maintained from level-to-level.
4. One bubble at a time should be refined.

5. Refinement should begin by isolating candidate processes, data objects, and data stores to be represented at the next level.

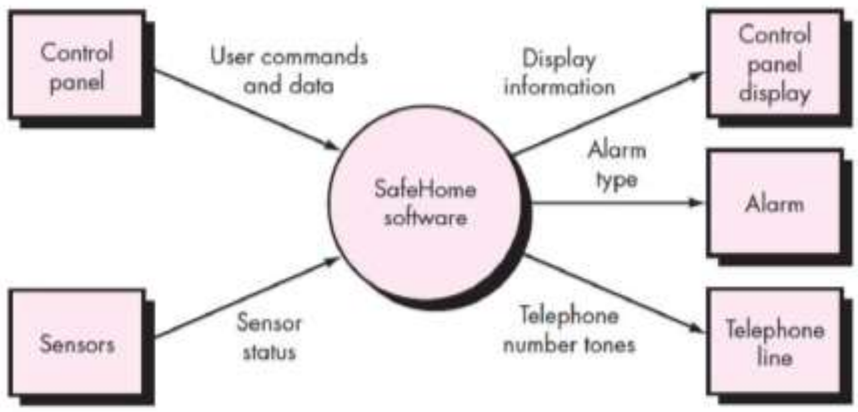


Fig: Level 0 DFD for SafeHome System.

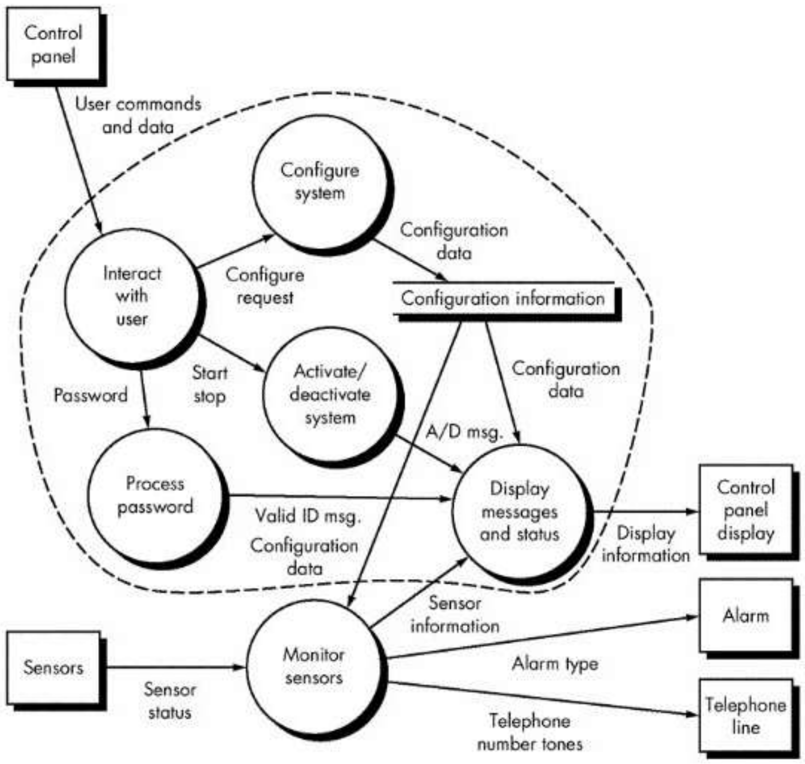


Fig: Level 1 DFD for SafeHome System.

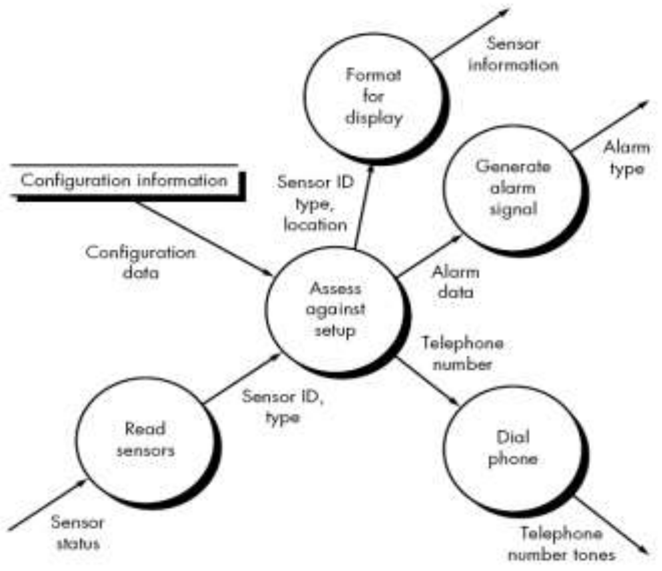


Fig: Level 2 DFD that refines the monitor sensors transform

Data Dictionary:

A data dictionary, on the other hand, is a structured document that describes the data elements and their relationships in a software project. It provides a detailed description of each data element, including its name, definition, data type, length, and other attributes. It also describes the relationships between data elements and provides a standardized vocabulary for the project.

The data dictionary serves as a reference guide for developers, testers, and other stakeholders in the project. It ensures that everyone is using consistent and accurate terminology when discussing the data elements in the project. It also helps to identify any inconsistencies or errors in the data model and facilitates communication between different teams working on the project.

A data dictionary is a file or a set of files that includes a database's metadata. The data dictionary hold records about other objects in the database, such as data ownership, data relationships to other objects, and other data. The data dictionary is an essential component of any relational database. Ironically, because of its importance, it is invisible to most database users. Typically, only database administrators interact with the data dictionary.

The data dictionary, in general, includes information about the following:

- Name of the data item
- Aliases
- Description/purpose
- Related data items
- Range of values
- Data structure definition/Forms

The name of the data item is self-explanatory.

Exercise:

1. Draw DFD (Level 0, Level 1, Level 2) for a project assigned to you.
2. Prepare a data dictionary for your project in excel.

Laboratory Report

Experiment No - 05

Batch -

Date of Experiment: __________ Date of Submission: __________

Title:

Develop Activity / State Transition diagram for the project

Evaluation:

1) Attendance [2] ----------------
2) Lab Performance [2] ----------------
3) Oral [1] ----------------

Overall Marks [5] ---------------

Subject In-Charge

Experiment No: -05

TITLE: Develop Activity / State Transition diagram for the project

PREREQUISITE:

1. Concepts of Object Oriented Programming & Methodology.

2. Knowledge of developing applications with front end & back end connectivity

HARDWARE CONFIGURATION / KIT:

Sr. No	Minimum Hardware Configuration	
1	Processor	800MHz Intel Pentium III or above versions
2	RAM	512 MB
3	HDD	1.5 GB of free disk space

SOFTWARE CONFIGURATION:

Sr. No	Minimum Software Configuration	
1	Operating System	Microsoft Windows Vista/7 or above versions
2	Editor	
3	Software	

THEORY: -

- **Activity diagrams**

  In UML, an activity diagram provides a view of the behavior of a system by describing the sequence of actions in a process. Activity diagrams are similar to flowcharts because they show the flow between the actions in an activity; however, activity diagrams can also show parallel or concurrent flows and alternate flows. In activity diagrams, you use activity nodes and activity edges to model the flow of control and data between actions.

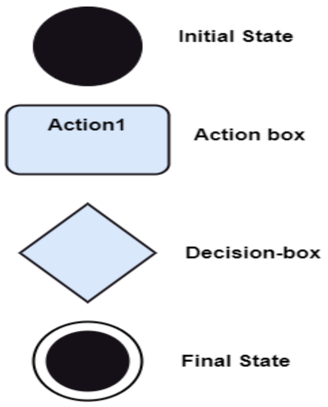
- **Activity diagrams are helpful in the following phases of a project:**

  Before starting a project, you can create activity diagrams to model the most important workflows. During the requirements phase, you can create activity diagrams to illustrate the flow of events that the use cases describe. During the analysis and design phases, you can use activity diagrams to help define the behavior of operations. As the following figure illustrates, an activity diagram belongs to an activity in the model. Before drawing an activity diagram, we should identify the following elements −

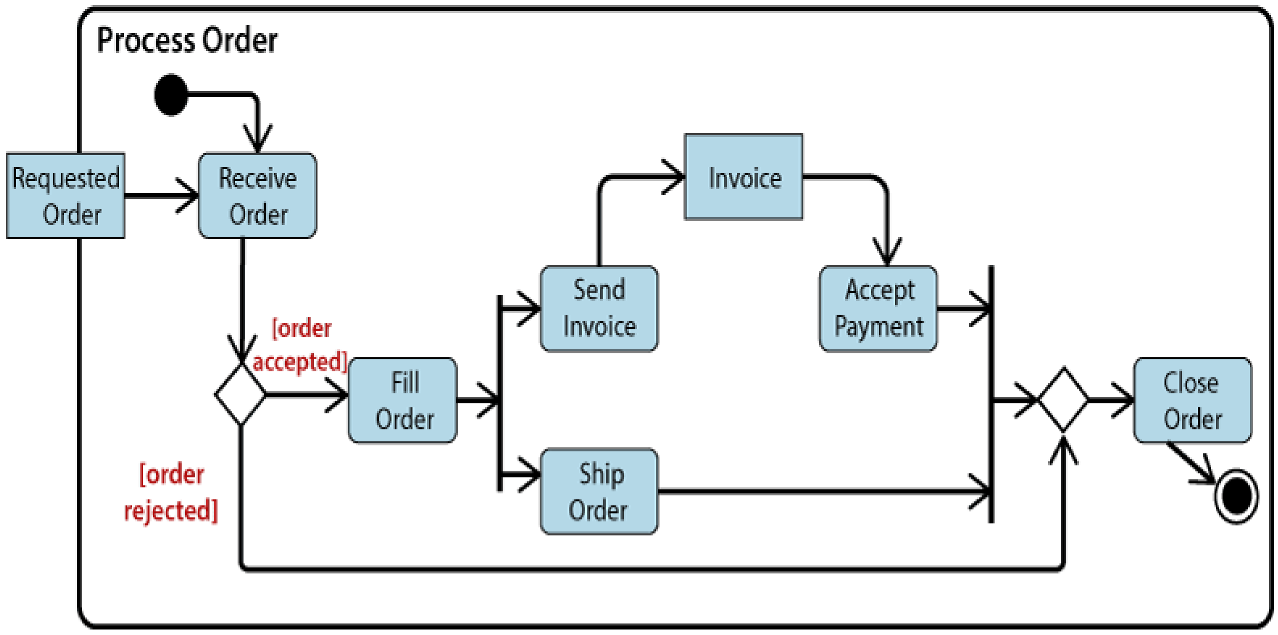
  1. Activities 2.Association 3.Conditions 4.Constraints

Notation of an Activity diagram

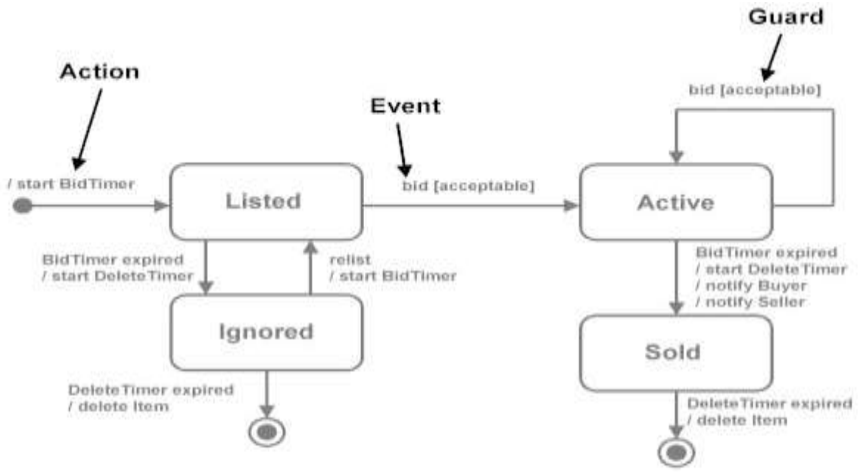
Activity diagram constitutes following notations:



Example of an Activity Diagram



- **State Transition diagram**



State-transition diagrams describe all of the states that an object can have, the events under which an object changes state (transitions), the conditions that must be fulfilled before the transition will occur (guards), and the activities undertaken during the life of an object (actions). State-transition diagrams are very useful for describing the behavior of individual objects over the full set of use cases that affect those objects. State-transition diagrams are not useful for describing the collaboration between objects that cause the transitions.
The UML notation for state-transition diagrams is shown below:
Notation
State-A condition during the life of an object in which it satisfies some condition, performs some action, or waits for some event.
Event-An occurrence that may trigger a state transition. Event types include an explicit signal from outside the system, an invocation from inside the system, the passage of a designated period of time, or a designated condition becoming true.
Guard-A boolean expression which, if true, enables an event to cause a transition.
Transition-The change of state within an object.
Action-One or more actions taken by an object in response to a state change.

Laboratory Report

Experiment No - 06

Batch -

Date of Experiment: __________ Date of Submission: __________

Title:

Develop Sequence and Collaboration diagram for the project

Evaluation:

1) Attendance [2]	----------------
2) Lab Performance [2]	----------------
3) Oral [1]	----------------
Overall Marks [5]	---------------

Subject In-Charge

Experiment No: -06

TITLE: Develop Sequence and Collaboration diagram for the project

PREREQUISITE:

1. Concepts of Object Oriented Programming & Methodology.

2. Knowledge of developing applications with front end & back end connectivity

HARDWARE CONFIGURATION / KIT:

Sr. No	Minimum Hardware Configuration	
1	Processor	800MHz Intel Pentium III or above versions
2	RAM	512 MB
3	HDD	1.5 GB of free disk space

SOFTWARE CONFIGURATION:

Sr. No	Minimum Software Configuration	
1	Operating System	Microsoft Windows Vista/7 or above versions
2	Editor	
3	Software	

THEORY: -

- **Sequence Diagram**

  The main purpose of a sequence diagram is to define event sequences that result in some desired outcome. The focus is less on messages themselves and more on the order in which messages occur; nevertheless, most sequence diagrams will communicate what messages are sent between a system's objects as well as the order in which they occur. The diagram conveys this information along the horizontal and vertical dimensions: the vertical dimension shows, top down, the time sequence of messages/calls as they occur, and the horizontal dimension shows, left to right, the object instances that the messages are sent to.

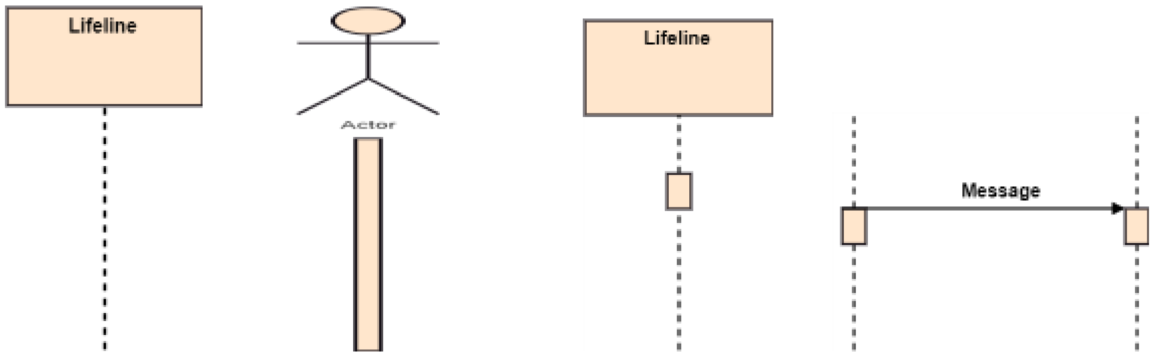
  Purpose of a Sequence Diagram

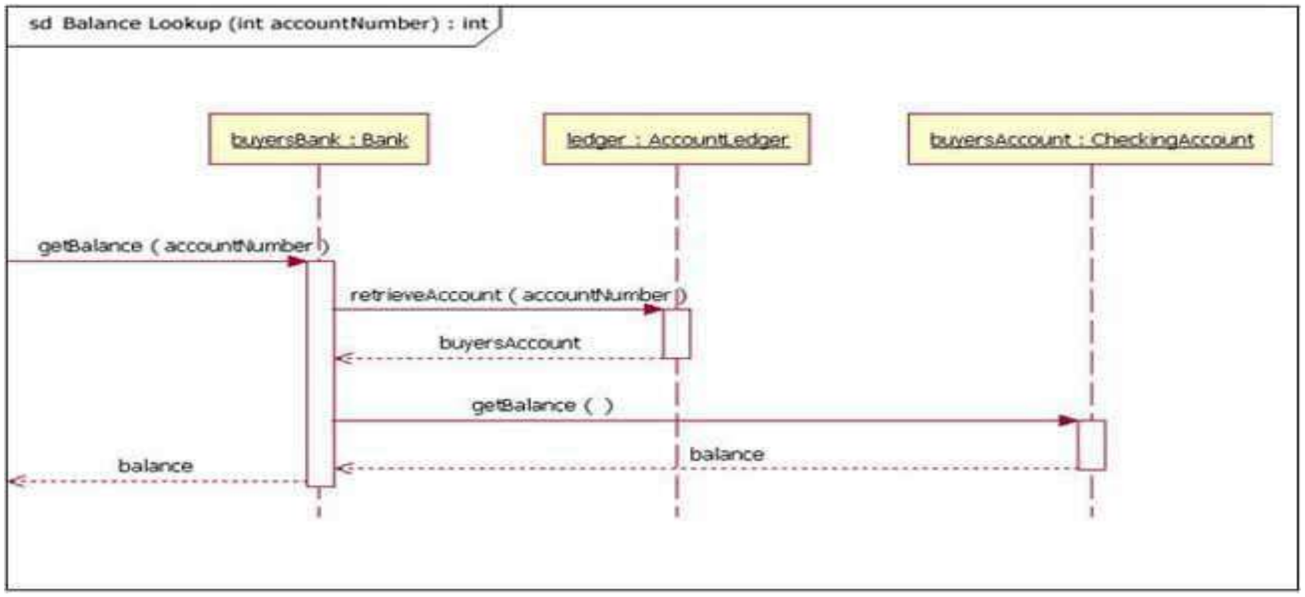
  To model high-level interaction among active objects within a system.

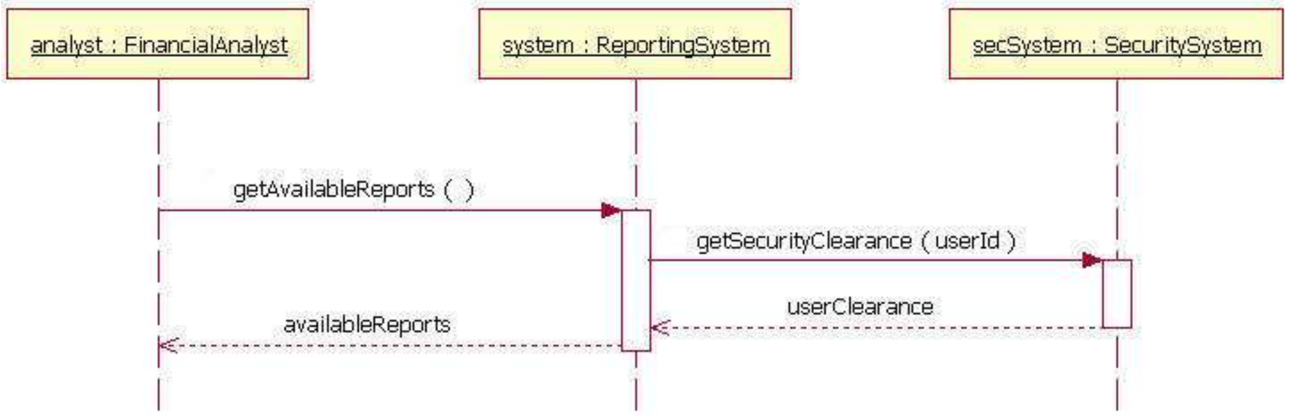
  To model interaction among objects inside a collaboration realizing a use case.

  It either models generic interactions or some certain instances of interaction

  Notations

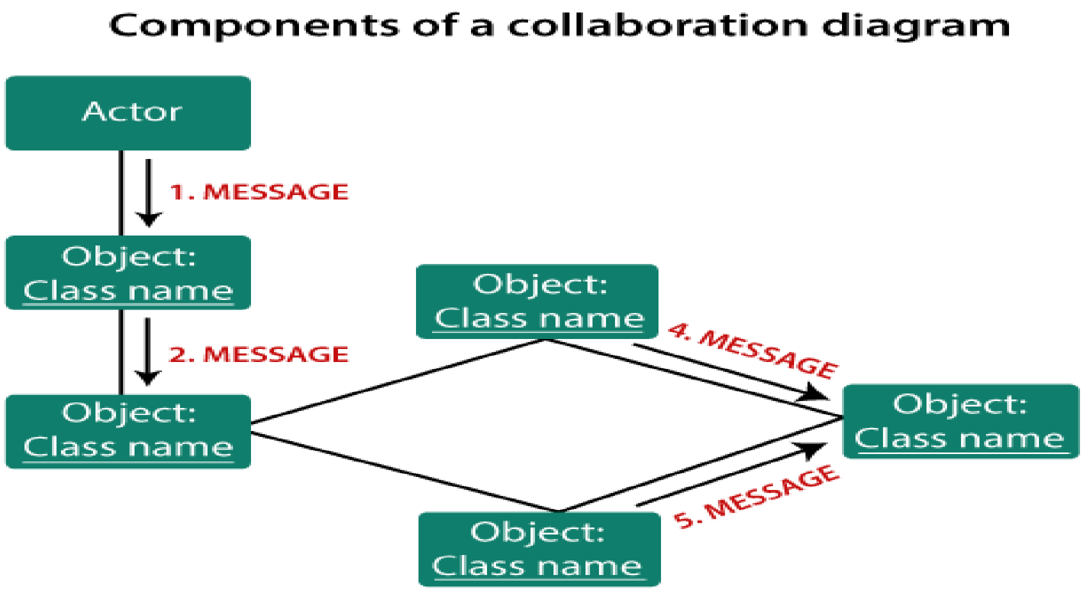

Lifeline
Actor
Lifeline
Message


sd Balance Lookup (int accountNumber) : int
buyersBank : Bank
ledger : AccountLedger
buyersAccount : CheckingAccount
getBalance (accountNumber)
retrieveAccount (accountNumber)
buyersAccount
getBalance ( )
balance
balance

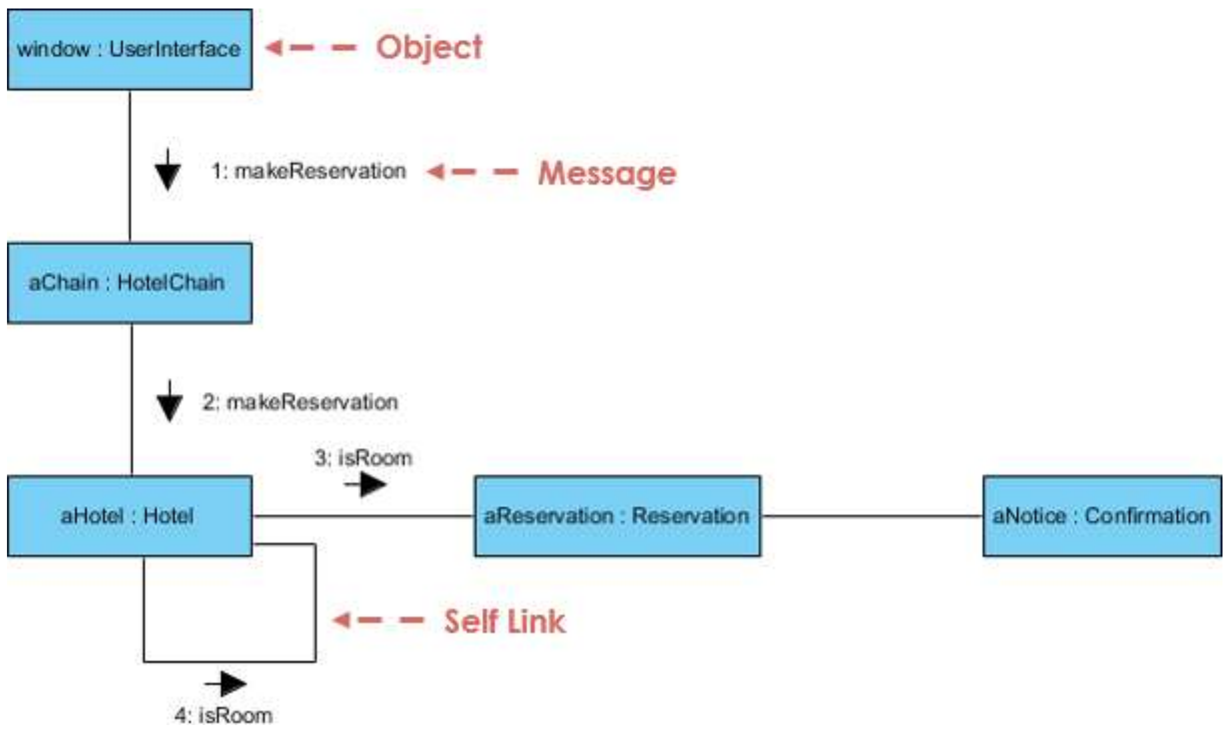

analyst : FinancialAnalyst
system : ReportingSystem
secSystem : SecuritySystem
getAvailableReports ( )
getSecurityClearance (userId)
userClearance
availableReports

- **Collaboration diagram**

The collaboration diagram is used to show the relationship between the objects in a system. Both the sequence and the collaboration diagrams represent the same information but differently. Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming. An object consists of several features. Multiple objects present in the system are connected to each other. The collaboration diagram, which is also known as a communication diagram, is used to portray the object's architecture in the system.



Example:


window : UserInterface
Object
1: makeReservation
Message
aChain : HotelChain
2: makeReservation
3: isRoom
aHotel : Hotel
aReservation : Reservation
aNotice : Confirmation
Self Link
4: isRoom

Laboratory Report

Experiment No - 07

Batch -

Date of Experiment: __________ Date of Submission: __________

Title: Use project management tool to prepare schedule for the project

Evaluation:

1) Attendance [2]	----------------
2) Lab Performance [2]	----------------
3) Oral [1]	----------------
Overall Marks [5]	---------------

Subject In-Charge

Experiment No: - 07

TITLE: Use project management tool to prepare schedule for the project.

PREREQUISITE:

1. Concepts of Object Oriented Programming & Methodology
2. Knowledge of developing applications with front end & back end connectivity.

HARDWARE CONFIGURATION / KIT:

Sr. No	Minimum Hardware Configuration	
1	Processor	800MHz Intel Pentium III or above versions
2	RAM	512 MB
3	HDD	1.5 GB of free disk space

SOFTWARE CONFIGURATION:

Sr. No	Minimum Software Configuration	
1	Operating System	Microsoft Windows Vista/7 or above versions
2	Editor	MS Word, Notepad

Theory: -

A) Project Scheduling

Software project scheduling is an action that distributes estimated effort across the planned project duration by allocating the effort to specific software Engineering and Project Management tasks.

Principles of Project Scheduling

1. **Compartmentalization**
2. **Interdependency**
3. **Time allocation**
4. **Effort Validation**
5. **Defined responsibilities**
6. **Defined outcomes**
7. **Defined milestones**

Work Breakdown Structure

Dividing complex projects to simpler and manageable tasks is the process identified as Work Breakdown Structure (WBS).

Usually, the project managers use this method for simplifying the project execution. In WBS, much larger tasks are broken down to manageable chunks of work. These chunks can be easily supervised and estimated.

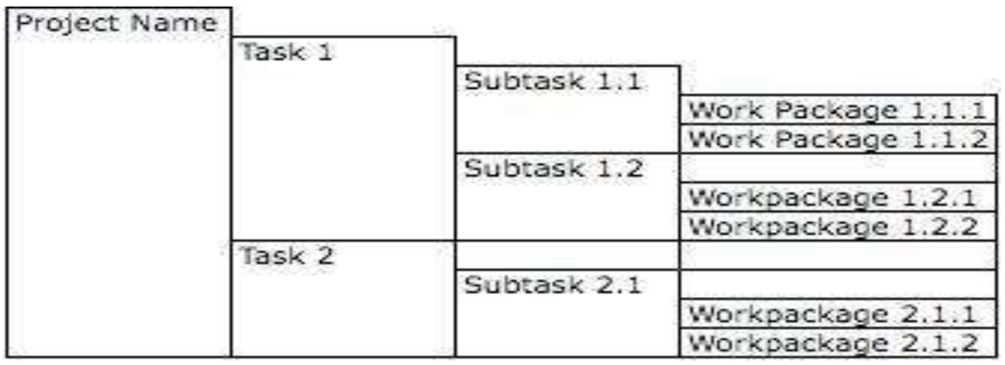


Fig: Work Breakdown Structure

Activity Network (Task Network)

A task network is also called as an activity network, is a graphic representation of the task flow for a project. It is sometimes used as the mechanism through which task sequence and dependencies are input to an automated project scheduling tool. In its simplest form, the task network depicts major software Engineering and Project Management actions.

A task set is a collection of software Engineering and Project Management work tasks, milestones, work products, and quality assurance filters that must be accomplished to complete a particular project. The task set must provide enough discipline to achieve high software quality. But, it must not burden the project team with unnecessary work.

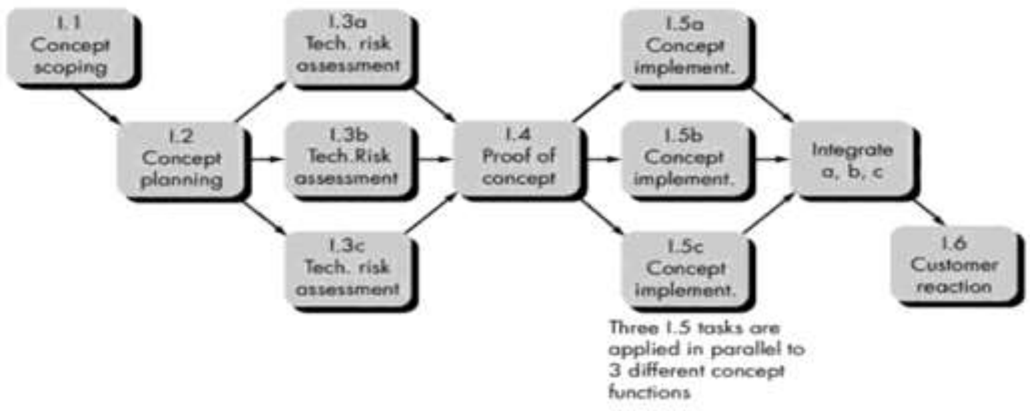


Fig: Activity Network (Task Network)

B) Project Tracking

Project scheduling is very important task. To complete project in decided timing is quite difficult. There might be reality of a technical project that there might be a hundreds of technical tasks. Some of the tasks may lie in the projects or may be some tasks lie outside the project.

There are certain tasks which fall on the critical path. If those are not considered in schedule, the project may be collapsed. The main job of project manager is to define all tasks involved in project, building a network that shows their independencies and the tasks which are critical within the network.

The major activity carried out in software project scheduling is that, it distributes estimated effects across the planned project period by allocating the effort to particular software Engineering and Project Management tasks.

Tracking the schedule

If it has been properly developed, the project schedule becomes a road map that defines the tasks and milestones to be tracked and controlled as the project proceeds.

Timeline Charts

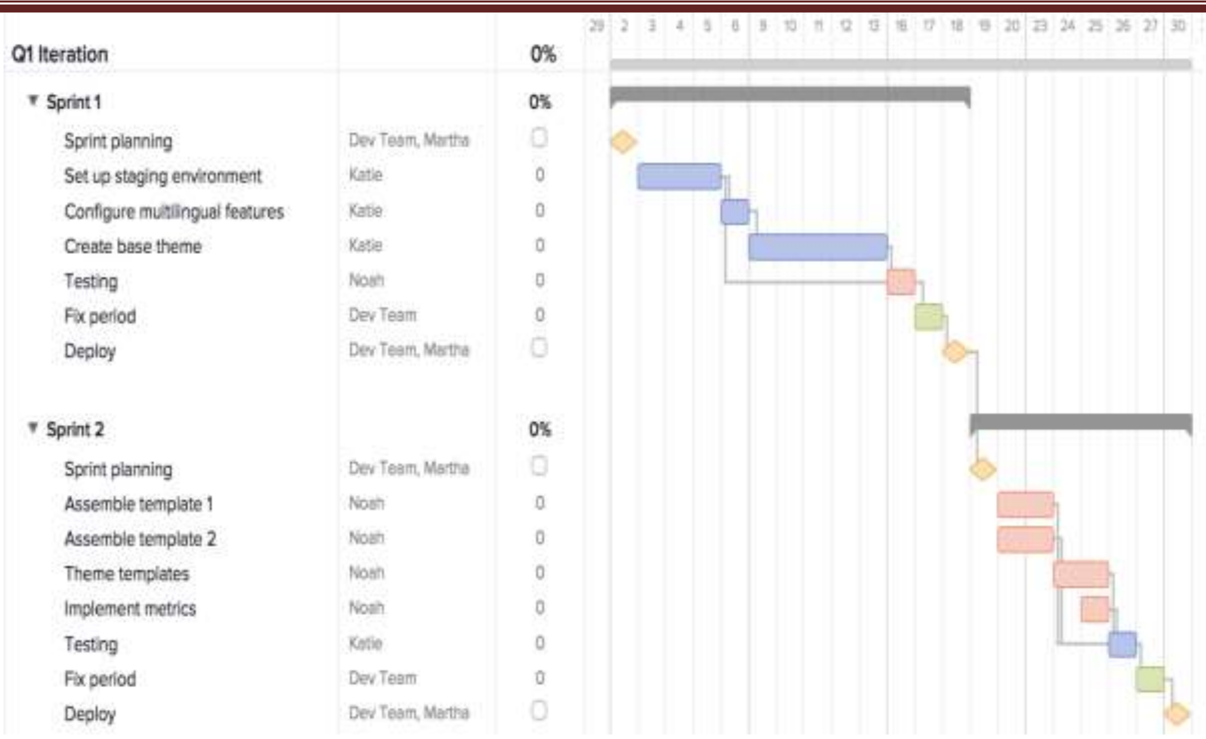
When creating software project schedule, we begin with a set of tasks. If automated tools are used, the work breakdown is input as a task network or task outline. Effort, duration and start date are then input for each task, In addition, tasks may be assigned to specific individuals.

As a consequence of this input, a time-line chart, also called a Gantt chart is generated. A time-line chart can be developed for the entire project.

A **Gantt chart** is a type of bar **chart** that illustrates a series of tasks or activities on a **timeline**. Each task is a small step that must be finished as part of completing a larger objective. The tasks are arranged in a cascading order on the **timeline**, based on their start date.

How to Create a Gantt chart:

1) Define the project settings, such as its start date, end date and scheduling mode. The most common scheduling mode is forwards from the project start date. In this mode the default is for tasks to start as soon as possible, which means that the whole project finishes at the earliest possible date.
2) Define the project calendar. This sets the number of working days in the week, the number of working hours in the day, and so on.
3) Enter or edit task names and durations.
4) Set up a global resources list and assign resources to tasks. Although you can often define the resources as you need them, it is usually quicker to start by setting up a global resources list from which you can then select resources to assign to the various project tasks.
5) Create links to specify the dependencies between the project tasks.
6) Set constraints on the tasks as necessary.
7) Make final adjustments to the project plan.
8) Once the project has actually started, inspect it at regular intervals to detect potential problems or scheduling conflicts and make any corrections required.



Q1 Iteration		0%
▾ Sprint 1		0%
Sprint planning	Dev Team, Martha	▢
Set up staging environment	Katie	0
Configure multilingual features	Katie	0
Create base theme	Katie	0
Testing	Noah	0
Fix period	Dev Team	0
Deploy	Dev Team, Martha	▢
▾ Sprint 2		0%
Sprint planning	Dev Team, Martha	▢
Assemble template 1	Noah	0
Assemble template 2	Noah	0
Theme templates	Noah	0
Implement metrics	Noah	0
Testing	Katie	0
Fix period	Dev Team	0
Deploy	Dev Team, Martha	▢

Fig. Timeline Chart / Gantt Chart

Exercise:
(Complete any 3 exercise suggested by your teacher, 4th is compulsory)

1) What are scheduling techniques?
2) Can a project have two critical paths? Justify your answer.
3) What are the Advantages of Gantt Charts?
4) Use any open source tool and perform the activity for a project given by you teacher.
    a. To identify the task set for project.
    b. To prepare the Activity Network (Network Diagram) for a project.
    c. To prepare the Gantt chart for a project.

Laboratory Report

Experiment No - 08

Batch -

Date of Experiment: __________ Date of Submission: __________

Title: Prepare RMMM plan for the project

____

____

Evaluation:

1) Attendance [2]	----------------
2) Lab Performance [2]	----------------
3) Oral [1]	----------------
Overall Marks [5]	---------------

Subject In-Charge

Experiment No: - 08

TITLE: Prepare RMMM plan for the project.

PREREQUISITE:

1. Concepts of Object Oriented Programming & Methodology
2. Knowledge of developing applications with front end & back end connectivity.

HARDWARE CONFIGURATION / KIT:

Sr. No	Minimum Hardware Configuration	
1	Processor	800MHz Intel Pentium III or above versions
2	RAM	512 MB
3	HDD	1.5 GB of free disk space

SOFTWARE CONFIGURATION:

Sr. No	Minimum Software Configuration	
1	Operating System	Microsoft Windows Vista/7 or above versions
2	Editor	MS Word, Notepad

Theory: -

Risk:

The risk denotes the uncertainty that may occur in the choices due to past actions and risk is something which causes heavy losses.

Risk is an expectation of loss, a potential problem that may or may not occur in the future. It is generally caused due to lack of information, control or time. A possibility of suffering from loss in software development process is called a software risk.

Risk management refers to the process of making decisions based on an evaluation of the factors that threats to the business. Various activities that are carried out for risk management are ---

• Risk identification
• Risk projection
• Risk refinement
• Risk mitigation, monitoring and management.

1. Risk Identification

Risk identification is a systematic attempt to specify threats to the project plan. By identifying known and predictable risks, the project manager takes a first step toward avoiding them when possible and controlling them when necessary.

There are two distinct types of risks for each of the categories: generic risks and product specific risks.

Generic risks are a potential threat to every software project. **Product- specific** risks can be identified only by those with a clear understanding of technology, the people, and the environment that is specific to the software that is to build.

2. Risk Projection/Prioritization

Risk projection also called risk estimation, attempts to rate each risk in two ways

1. The likelihood or probability that the risk is real and
2. The consequences of the problems associated with the risk

There are four risk projection steps:

1. Establish a scale that reflects the perceived likelihood of risk
2. Delineate the consequence of the risk
3. Estimate the impact of the risk on the project and the product
4. Assess the overall accuracy of the risk projection so there will be no misunderstandings.

3. Risk Analysis

The following questions are to be used for analyzing risk:

- Have top software and customer manager formally committed to support the project?
- Are end users enthusiastically committed to the project and the system/product to be built?
- Are requirements fully understood by the software Engineering and Project Management team and its customers?
- Have customers been involved fully in the definition of requirements?
- Do end users have realistic expectations?
- Is the project scope stable?

And so on….

4.RMMM Strategy

The goal of the risk mitigation, monitoring and management plan is to identify as many potential risks as possible. Risk analysis support the project team in constructing a strategy to deal with risks.

There are three important issues considered in developing an effective strategy:

- **Risk avoidance or mitigation -** It is the primary strategy which is fulfilled through a plan.
- **Risk monitoring -** The project manager monitors the factors and gives an indication whether the risk is becoming more or less.
- **Risk management and planning -** It assumes that the mitigation effort failed, and the risk is a reality.

RMMM Plan

- It is a part of the software development plan or a separate document.
- The RMMM plan documents all work executed as a part of risk analysis and used by the project manager as a part of the overall project plan.
- The risk mitigation and monitoring start after the project is started and the documentation of RMMM is completed.

Procedure:-

Prepare Risk Table & RMMM Plan for a project assigned to you by your teacher

➢ Template for Risk table:

No.	List of Risk	Category	Probability	Impact RMMM

➢ Risk Table Construction

• List all risks in the first column of the table.
• Classify each risk and enter the category label in column two.
• Determine a probability for each risk (Rare, Unlikely, Moderate Likely, Very likely) and enter it into column three.
• Enter the severity of each risk (negligible, marginal, critical, and catastrophic) in column four.

Exercise:-

1. What are the software Risks?
2. What is risk management in software development?
3. Explain RMMM and RMMM plan.

Laboratory Report

Experiment No - 08

Batch -

Date of Experiment: __________ Date of Submission: __________

Title: Prepare RMMM plan for the project

____

____

Evaluation:

1) Attendance [2]	----------------
2) Lab Performance [2]	----------------
3) Oral [1]	----------------

Overall Marks [5] ---------------

Subject In-Charge

Experiment No: -09

TITLE: Change specification and make different versions using any SCM Tool

PREREQUISITE:

1. Concepts of Object Oriented Programming & Methodology.
2. Knowledge of developing applications with front end & back end connectivity

HARDWARE CONFIGURATION / KIT:

Sr. No	Minimum Hardware Configuration	
1	Processor	800MHz Intel Pentium III or above versions
2	RAM	512 MB
3	HDD	1.5 GB of free disk space

SOFTWARE CONFIGURATION:

Sr. No	Minimum Software Configuration	
1	Operating System	Microsoft Windows Vista/7 or above versions
2	Editor	
3	Software	GIT, Git Bash , GitHub

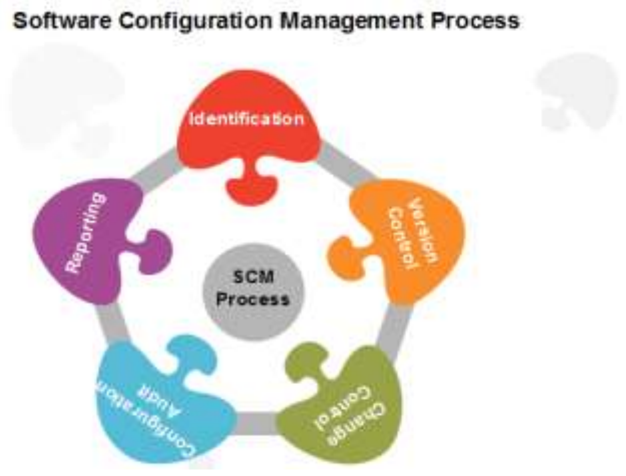
THEORY: -

- **Software Configuration Management (SCM)**

Whenever a software is build, there is always scope for improvement and those improvements brings changes in picture. Changes may be required to modify or update any existing solution or to create a new solution for a problem. Requirements keeps on changing on daily basis and so we need to keep on upgrading our systems based on the current requirements and needs to meet desired outputs. Changes should be analyzed before they are made to the existing system, recorded before they are implemented, reported to have details of before and after, and controlled in a manner that will improve quality and reduce error. This is where the need of System Configuration Management comes.

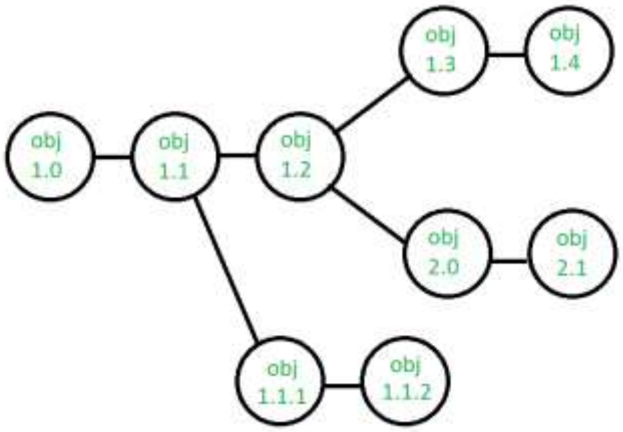
System Configuration Management (SCM) is an arrangement of exercises which controls change by recognizing the items for change, setting up connections between those things, making/characterizing instruments for overseeing diverse variants, controlling the changes being executed in the current framework, inspecting and revealing/reporting on the changes made. It is essential to control the changes in light of the fact that if the changes are not checked legitimately then they may wind up undermining a well-run programming. In this way, SCM is a fundamental piece of all project management activities.

Processes involved in SCM –



Configuration management provides a disciplined environment for smooth control of work products. It involves the following activities:

- ✓ **Identification and Establishment** – Identifying the configuration items from products that compose baselines at given points in time (a baseline is a set of mutually consistent Configuration Items, which has been formally reviewed and agreed upon, and serves as the basis of further development). Establishing relationship among items, creating a mechanism to manage multiple level of control and procedure for change management system.
- ✓ **Version control** – Creating versions/specifications of the existing product to build new products from the help of SCM system. A description of version is given below:



Suppose after some changes, the version of configuration object changes from 1.0 to 1.1. Minor corrections and changes result in versions 1.1.1 and 1.1.2, which is followed by a major update that is object 1.2. The development of object 1.0 continues through 1.3 and 1.4, but finally, a noteworthy change to the object results in a new evolutionary path, version 2.0. Both versions are currently supported.

- ✓ **Change control** – Controlling changes to Configuration items (CI). A change request (CR) is submitted and evaluated to assess technical merit, potential side effects, overall impact on other configuration objects and system functions, and the projected cost of the change. The results of the evaluation are presented as a change report, which is used by a change control board (CCB) —a person or group who makes a final decision on the status and priority of the change. An engineering change Request (ECR) is generated for each approved change.

- ✓ **Configuration auditing –** A software configuration audit complements the formal technical review of the process and product. It focuses on the technical correctness of the configuration object that has been modified. The audit confirms the completeness, correctness and consistency of items in the SCM system and track action items from the audit to closure.
- ✓ **Reporting –** Providing accurate status and current configuration data to developers, tester, end users, customers and stakeholders through admin guides, user guides, FAQs, Release notes, Memos, Installation Guide, Configuration guide etc .

- **SCM Tools –**
  Different tools are available in market for SCM like: **Git,*CFEngine, Bcfg2 server, Vagrant, SmartFrog, CLEAR CASETOOL (CC), SaltStack, CLEAR QUEST TOOL, Puppet, SVN-Subversion, Perforce, TortoiseSVN, IBM Rational team concert, IBM Configuration management version management, Razor, Ansible, etc.*** It is recommended that before

selecting any configuration management tool, have a proper understanding of the features and select the tool which best suits your project needs and be clear with the benefits and drawbacks of each before you choose one to use.

Open Source SCM Tools:

CFEngine, Puppet ,CHEF, JUJU

Laboratory Report

Experiment No - 10

Batch -

Date of Experiment: __________ Date of Submission: __________

Title: Develop test cases for the project using testing techniques

Evaluation:

1) Attendance [2] ----------------

2) Lab Performance [2] ----------------

3) Oral [1] ----------------

Overall Marks [5] ---------------

Subject In-Charge

Experiment No: -10

TITLE: Develop test cases for the project using testing techniques.

PREREQUISITE:

1. Concepts of Object Oriented Programming & Methodology.
2. Knowledge of developing applications with front end & back end connectivity

HARDWARE CONFIGURATION / KIT:

Sr. No	Minimum Hardware Configuration	
1	Processor	800MHz Intel Pentium III or above versions
2	RAM	512 MB
3	HDD	1.5 GB of free disk space

SOFTWARE CONFIGURATION:

Sr. No	Minimum Software Configuration	
1	Operating System	Microsoft Windows Vista/7 or above versions
2	Editor	
3	Software	

THEORY: -

What is Software Testing Technique?

Software Testing Techniques helps to design better test cases. Since exhaustive testing is not possible; Manual Testing Techniques help reduce the number of test cases to be executed while increasing test coverage. They help identify test conditions that are otherwise difficult to recognize.

Test case Design Technique

What is a Test case?

A test case has components that describe input, action, and an expected response, in order to determine if a feature of an application works correctly. A test case is a set of instructions on "HOW" to validate a particular test objective/target, which, when followed will tell us if the expected behavior of the system is satisfied or not.

Following are the typical design techniques in software Engineering and Project Management:

1. Deriving test cases directly from a requirement specification or black box test design technique. The Techniques include:

- Boundary Value Analysis (BVA)
- Equivalence Partitioning (EP)
- Decision Table Testing
- State Transition Diagrams
- Use Case Testing

2. Deriving test cases directly from the structure of a component or system:
- Statement Coverage
- Branch Coverage
- Path Coverage
- LCSAJ Testing

3. Deriving test cases based on tester's experience on similar systems or testers intuition:
- Error Guessing
- Exploratory Testing

Sample Test Case Template with Test Case Examples

Test case formats may vary from one organization to another. However, using a standard test case format for writing test cases is one step closer to setting up a testing process for your project. It also minimizes Ad-hoc testing that is done without proper test case documentation. But even if you use standard templates, you need to set up test cases writing, review & approve, test execution and most importantly test report preparation process, etc. by using manual methods. Also, if you have a process to review the test cases by the business team, then you must format these test cases in a template that is agreed by both the parties.

Recommended Tools

#1) TestRail
#2) Katalon Platform
#3) Testiny

Several standard fields for a sample Test Case template are listed below.

Test case ID: Unique ID is required for each test case. Follow some conventions to indicate the types of the test. **For Example,** ‘TC_UI_1’ indicating ‘user interface test case #1’.

Test priority (Low/Medium/High): This is very useful during test execution. Test priorities for business rules and functional test cases can be medium or higher, whereas minor user interface cases can be of a low priority. Testing priorities should always be set by the reviewer.

Module Name: Mention the name of the main module or the sub-module.

Test Designed By Name of the Tester.

Test Designed Date: Date when it was written.

Test Executed By Name of the Tester who executed this test. To be filled only after test execution.

Test Execution Date: Date when the test was executed.

Test Title/Name: Test case title. **For example,** verify the login page with a valid username and password.

Test Summary/Description: Describe the test objective in brief.

Pre-conditions: Any prerequisite that must be fulfilled before the execution of this test case. List all the pre-conditions in order to execute this test case successfully.

Dependencies: Mention any dependencies on other test cases or test requirements.

Test Steps: List all the test execution steps in detail. Write test steps in the order in which they should be executed. Make sure to provide as many details as you can.

Test Data: Use of test data as an input for this test case. You can provide different data sets with exact values to be used as an input.

Expected Result: What should be the system output after test execution? Describe the expected result in detail including the message/error that should be displayed on the screen.

Post-condition: What should be the state of the system after executing this test case?

Actual result: The actual test result should be filled after test execution. Describe the system behavior after test execution.

Status (Pass/Fail): If the actual result is not as per the expected result, then mark this test as **failed**. Otherwise, update it as **passed**.

Notes/Comments/Questions: If there are any special conditions to support the above fields, which can't be described above or if there are any questions related to expected or actual results then mention them here.
Add the following fields if necessary:

Defect ID/Link: If the test status **fails**, then include the link to the defect log or mention the defect number.

Test Type/Keywords: This field can be used to classify tests based on test types.
For Example, functional, usability, business rules, etc.

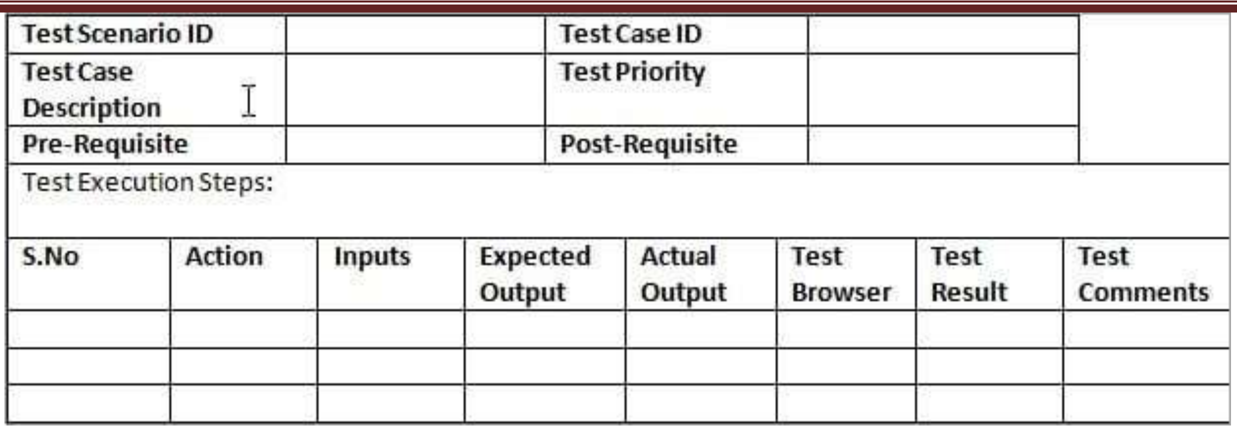
Requirements: Requirements for which this test case is being written for. Preferably the exact section number in the requirement doc.

Attachments/References: This field is useful for complex test scenarios in order to explain the test steps or expected results using a Visio diagram as a reference. Provide a link or location to the actual path of the diagram or document.

Automation? (Yes/No): Whether this test case is automated or not. It is useful to track automation status when test cases are automated.

- **Test Case Formats**

1.



Test Scenario ID		Test Case ID	
Test Case Description		Test Priority	
Pre-Requisite		Post-Requisite	

Test Execution Steps:

S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result	Test Comments

2.

Project Name:	
Test Case Template	
Test Case ID: Fun_10	**Test Designed by:** <Name>
Test Priority (Low/Medium/High): Med	**Test Designed date:** <Date>
Module Name: Google login screen	**Test Executed by:** <Name>
Test Title: Verify login with valid username and password	**Test Execution date:** <Date>
Description: Test the Google login page	
Pre-conditions: User has valid username and password	
Dependencies:	

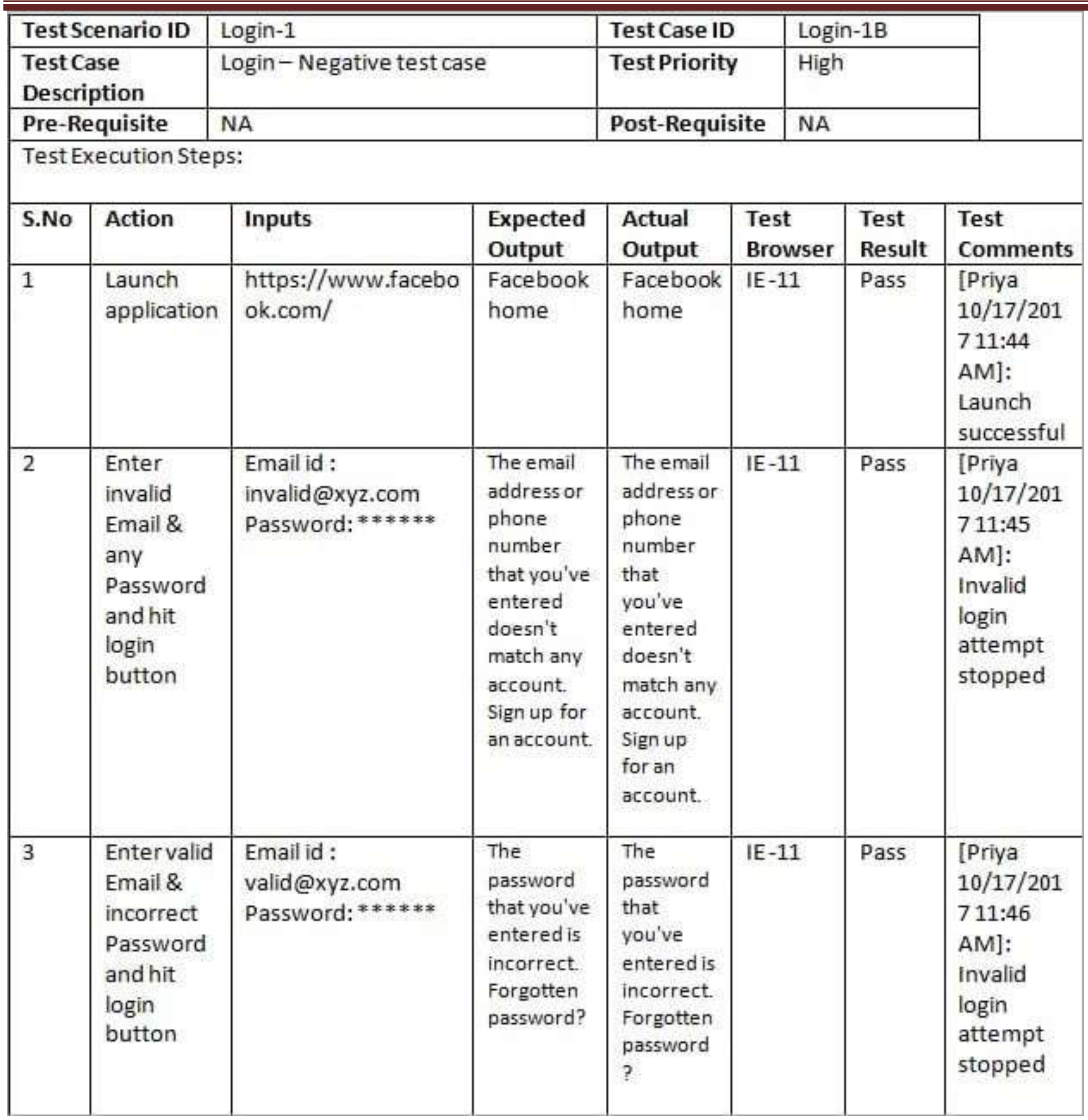
Step	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)	Notes
1	Navigate to login page	User= example@gmail.com	User should be able to login	User is navigated to	Pass	
2	Provide valid username	Password: 1234		dashboard with successful		

				login		
3	Provide valid password					
4	Click on Login button					

Post-conditions:

User is validated with database and successfully login to account. The account session details are logged in database.

3.



Test Scenario ID	Login-1	**Test Case ID**	Login-1B
Test Case Description	Login – Negative test case	**Test Priority**	High
Pre-Requisite	NA	**Post-Requisite**	NA

Test Execution Steps:

S.No	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Launch application	https://www.facebook.com/	Facebook home	Facebook home	IE-11	Pass	[Priya 10/17/2017 11:44 AM]: Launch successful
2	Enter invalid Email & any Password and hit login button	Email id : invalid@xyz.com Password: ******	The email address or phone number that you've entered doesn't match any account. Sign up for an account.	The email address or phone number that you've entered doesn't match any account. Sign up for an account.	IE-11	Pass	[Priya 10/17/2017 11:45 AM]: Invalid login attempt stopped
3	Enter valid Email & incorrect Password and hit login button	Email id : valid@xyz.com Password: ******	The password that you've entered is incorrect. Forgotten password?	The password that you've entered is incorrect. Forgotten password?	IE-11	Pass	[Priya 10/17/2017 11:46 AM]: Invalid login attempt stopped