***SOFTWARE ENGINEERING LAB***

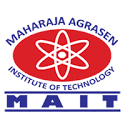
***ETCS - 353***

DR. JYOTI KAUSHIK MADHAV KHURANA

ROLL NO.: 05914802718

SEMESTER - 5TH

BATCH: C4



Maharaja Agrasen Institute of Technology, PSP Area, Sector – 22, Rohini, New Delhi – 110085

**INDEX**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO.** | **NAME OF EXPERIEMNT** | **DATE** | **SIGN.** |
| 1 | Write down the problem statement for a suggested system of relevance. | 31/08/2020 |  |
| 2 | Write down the SRS statement for a suggested system of relevance. | 07/09/2020 |  |
| 3 | To perform the function-oriented diagram: Data Flow Diagram (DFD) and Structured chart. | 14/09/2020 |  |
| 4 | To perform the user’s view analysis for the suggested system: Use case diagram. | 21/09/2020 |  |
| 5 | To draw the structural view diagram for the system: Class diagram, object diagram. | 05/10/2020 |  |
| 6 | To draw behavioral view diagram: Statechart Diagram, Activity Diagram | 19/10/2020 |  |
| 7 | To draw behavioral view diagram: Sequence Diagram, Collaboration Diagram. | 02/11/2020 |  |
| 8 | To draw implementation view diagram: Component Diagram for the system | 03/11/2020 |  |
| 9 | To draw environmental view diagram: Deployment Diagram for the system | 23/11/2020 |  |

**EXPERIMENT-1**

**AIM:** Write down the problem statement for a suggested system of relevance.

**THEORY:** Problem statement is a one to three pager non-technical document provided by the client to the software development team. It is description of the requirement of the client that has a description of various methods and constants to be added in the final software. It provides overall view of the requirement elicitation and helps in analyzing the problem. It is fact-based document that focuses on the gap between the development team and the client.

**PROJECT:**STUDENT RESULT MANAGEMENT SYSTEM

**PROBLEM STATEMENT:**

**Student Result Management System**

* Under this project, Software is to be developed for college students and teachers for displaying students results and contains the following functionality.
* For students the display board should contain the following elements:
  + - Student Name
    - Semester Number
    - Photograph and batch details
    - Subject name and marks in tabular form
    - Rank of the student in each subject
    - Overall grade and percentage
    - Previous result link button

The whole board should be in jpg form and students can download and zoom it. Students can’t access the others result

For Teachers (admins)

* Updating the marks of each subject
* Editing the data (marks)
* Student board access to teacher
* Overall class average
* No of students failed and passed
* Previous semester result and history

The teacher can access the data along with the option of sharing it. They can also download it as well.

Login:

* Teacher login – The username name will be the name of teacher & password
* Student login – username will be their enrollment number & password as their date of birth
* Password & username can be changed

**VIVA – VOCE QUESTIONS:**

1. **What is problem Statement?**

Problem statement is a one to three pager non-technical document provided by the client to the software development team

1. **What are benefits of writing problem statement?**

It can help in understanding the user requirement and make the project more guided and focused.

1. **Writing a problem statement, really important for proceeding the project?**

Yes, it is really important for proceeding the project because without it very useful information can be missed out.

1. **Explain 5 W’s can be used to spark the problem statement.**

5 W’s i.e. who, which, when, why and what are the questions that give all appropriate answer for the requirements.

1. **What are the steps that need to follow while writing problem statement?**

Describe how things should work. Explain the problem and state why it matters. Explain your problem's financial costs. Back up your claims. Propose a solution. Conclude by summarizing the problem and solution.

**EXPERIMENT – 2**

**Aim:**

Do requirement analysis and develop Software Requirement Specification Sheet (SRS) for suggested system.

**Theory:**

A software requirements specification (**SRS**) is a **document** that describes what the software will do and how it will be expected to perform. It also describes the functionality the product needs to fulfill all stakeholders (business, users) needs.

It’s used to provide critical information to multiple teams development, quality assurance, operations, and maintenance. This keeps everyone on the same page.

Using the SRS helps to ensure requirements are fulfilled. And it can also help you make decisions about your product’s lifecycle — for instance, when to retire a feature.

Writing an SRS can also minimize overall development time and costs. Embedded development teams especially benefit from using an SRS.

Software requirements specification establishes the basis for an agreement between customers and contractors or suppliers on how the software product should function (in a market-driven project, these roles may be played by the marketing and development divisions). 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.[[2]](https://en.wikipedia.org/wiki/Software_requirements_specification#cite_note-2)

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.

The SRS may be one of a contract's deliverable data item descriptions or have other forms of organizationally-mandated content.

In order to fully understand one’s project, it is very important that they come up with a SRS listing out their requirements, how are they going to meet it and how will they complete the project. It helps the team to save upon their time as they are able to comprehend how are going to go about the project. Doing this also enables the team to find out about the limitations and risks early on.

 It lays out functional and non-functional requirements, and may include a set of use cases that describe user interactions that the software must provide.

**SRS DOCUMENT FOR STUDENT RESULT MANAGEMENT SYSTEM**

1. **INTRODUCTION:** The aim of this document is to give overall requirements for building the software:” STUDENT RESULT MANAGEMENT SYSTEM” for Maharaja Agrasen Institute of Technology students and professors. The final product should be made uniquely and solely for our particular organization, the information is specified in the document, suitable assumptions can be made but developer team should not go against the requirement specified.
   1. **PURPOSE**: The purpose of this software is to assist professors in sharing, uploading and accessing, managing the student’s result and for students the software must be useful for accessing proper information.
   2. **SCOPE:** The application should only be used by students and professors and other authorized person of our college Maharaja Agrasen Institute of Technology, it should not be used/accessed by general public.
   3. **DEFINITIONS, ACRONYMS AND ABBREVATIONS:** no definitions are used.
   4. **REFERENCES:** The official website of our college: www.mait.ac.in
   5. **OVERVIEW:** The rest of SRS describes various features, interfaces and other useful information regarding the requirements of the software.
2. **OVERALL DESCRIPTION**:
   1. **PRODUCT PERSPECTIVE**: The application will be window based, self-contained and independent software product and should not be public.
   2. **PRODUCT FUNCTIONS:** The product should contain functions such as uploading, editing, displaying, sharing the results for professors’ window. Whereas for students’ window the function should be accessing and sharing the results
   3. **USER CHARACTERISTICS:** There are two types of users i.e. professors (& other management) and students.
   4. **CONSTRAINTS:** The data should be private; the professors can only share the data within the college management and staff. The students can only share their particular result, not the whole data of the class/batch.
   5. **ASSUMPTIONS AND DEPENDENCIES:** Developers can think of suitable assumptions but should not go against the requirements specified in the SRS. There are no dependencies based on the project.
   6. **APPORTIONING OF REQUIREMENTS:** If the project is delayed, the semester number and should be updated as of the time.
3. **SPECIFIC REQUIREMENTS:** The section contains the software requirements to a level of detail sufficient to enable designers to design the system, and testers to test the system.
   1. **EXTERNAL INTERFACE REQUIREMENT:** It will contain login screen, students’ marks-board, teacher uploading screen and class report screen
   2. **FUNCTIONS:** Updating the marks of each subject, Editing the data (marks), Student board access to teacher, Overall class average, displaying the marks.
   3. **PERFORMANCE REQUIREMENT**: NIL
   4. **LOGICAL DATABASE REQUIREMENT:** The teacher’s will upload database of students every time, so there should be database connectivity whereas students will also download it.
   5. **DESIGN CONSTRAINTS:** The visuals of the leaderboard must contain the photograph of the student.
   6. **SOFTWARE SYSTEM ATTRIBUTES:** The software must be compatible with windows, mac, Linux, Unix and Ubuntu.
   7. **ORGANIZING SPECIFIC REQUIREMENTS:** NIL
   8. **ADDITIONAL COMMENTS:** NIL
4. **CHANGE MANAGEMENT PROCESS**: The Stakeholder will identify a new requirement or a modification in a requirement and communicate it to the team lead via email or to the entire team in a meeting. The team will assess the feasibility of the proposed change considering the time constraints and structural constraints of the implemented modules and develop an implementation strategy and submit it to the mentor for approval.
5. **DOCUMENT APPROVAL**
6. **SUPPORTING INFORMATION**

**VIVA – VOCE QUESTIONS:**

1. **What are the objectives of requirement analysis?**

The objective is to gather, process, and track evolving stakeholder needs and requirements throughout the lifecycle of the product.

1. **Define different types of requirements.**

Functional Requirements.

Performance Requirements.

System Technical Requirements.

Specifications.

1. **Outline structure of SRS documents.**

Introduction

Overall description

Change management process

Document approval

Supporting information

1. **What are the benefits of writing SRS document?**

Having a well-**written SRS** helps optimize the development process by preventing the duplication of tasks and structuring problems in a way that makes them easily solvable.

1. **Define functional and non-functional requirements.**

**Functional Requirements:** These are the requirements that the end user specifically demands as basic facilities that the system should offer.

**Non-functional requirements:** These are basically the quality constraints that the system must satisfy according to the project contract.

**EXPERIMENT – 3**

**AIM:** To perform the function-oriented diagram: Data Flow Diagram (DFD) and Structured chart.

**SOFTWARE USED**: *STAR UML*

**PROJECT:** STUDENT RESULT MANAGEMENT SYSTEM

**THEORY:**

DATA FLOW DIAGRAM

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination.

Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled. They can be used to analyze an existing system or model a new one. Like all the best diagrams and charts, a DFD can often visually “say” things that would be hard to explain in words, and they work for both technical and nontechnical audiences, from developer to CEO. That’s why DFDs remain so popular after all these years.

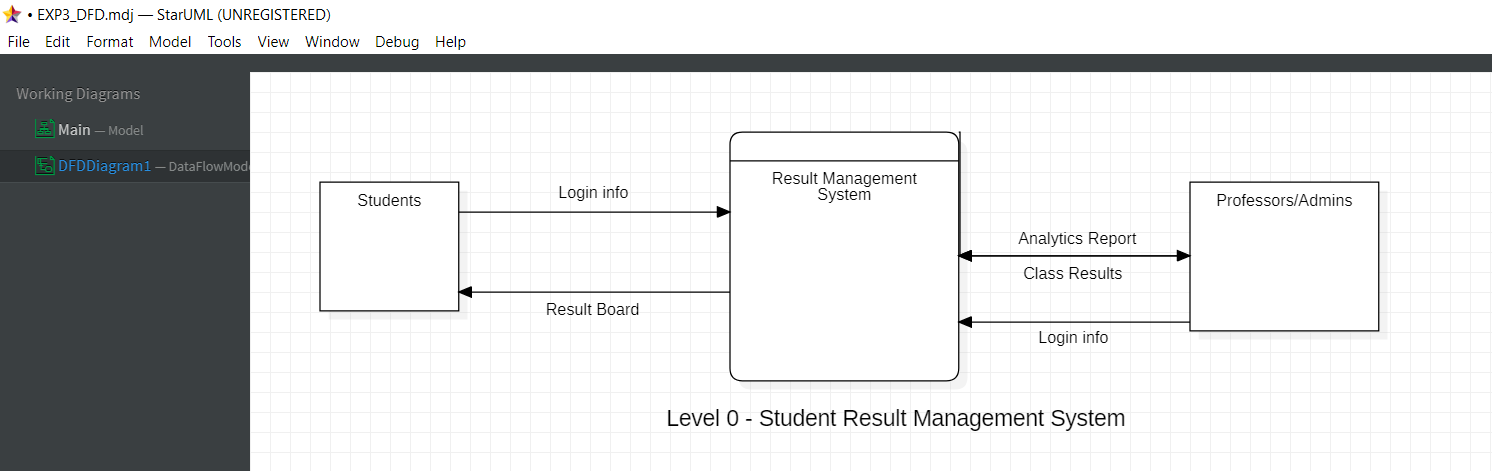
While they work well for data flow software and systems, they are less applicable nowadays to visualizing interactive, real-time or database-oriented software or systems.

STRUCTURED CHART

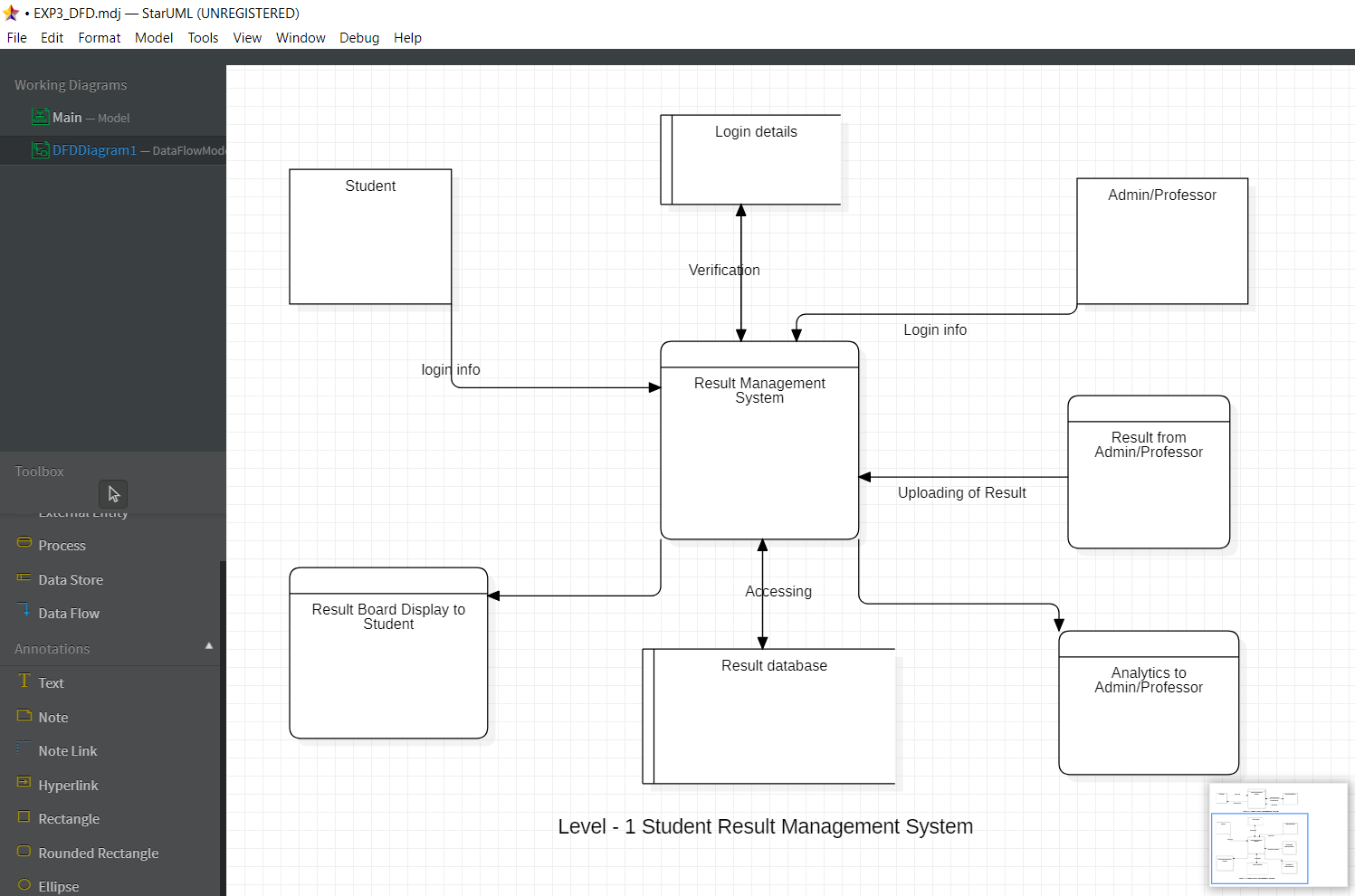
**Structure Chart** represent hierarchical structure of modules. It breaks down the entire system into lowest functional modules, describe functions and sub-functions of each module of a system to a greater detail. Structure Chart partitions the system into black boxes (functionality of the system is known to the users but inner details are unknown). Inputs are given to the black boxes and appropriate outputs are generated.

Modules at top level called modules at low level. Components are read from top to bottom and left to right. When a module calls another, it views the called module as black box, passing required parameters and receiving results.

**DATA FLOW DIAGRAM FOR STUDENT RESULT MANAGEMENT SYSTEM:**

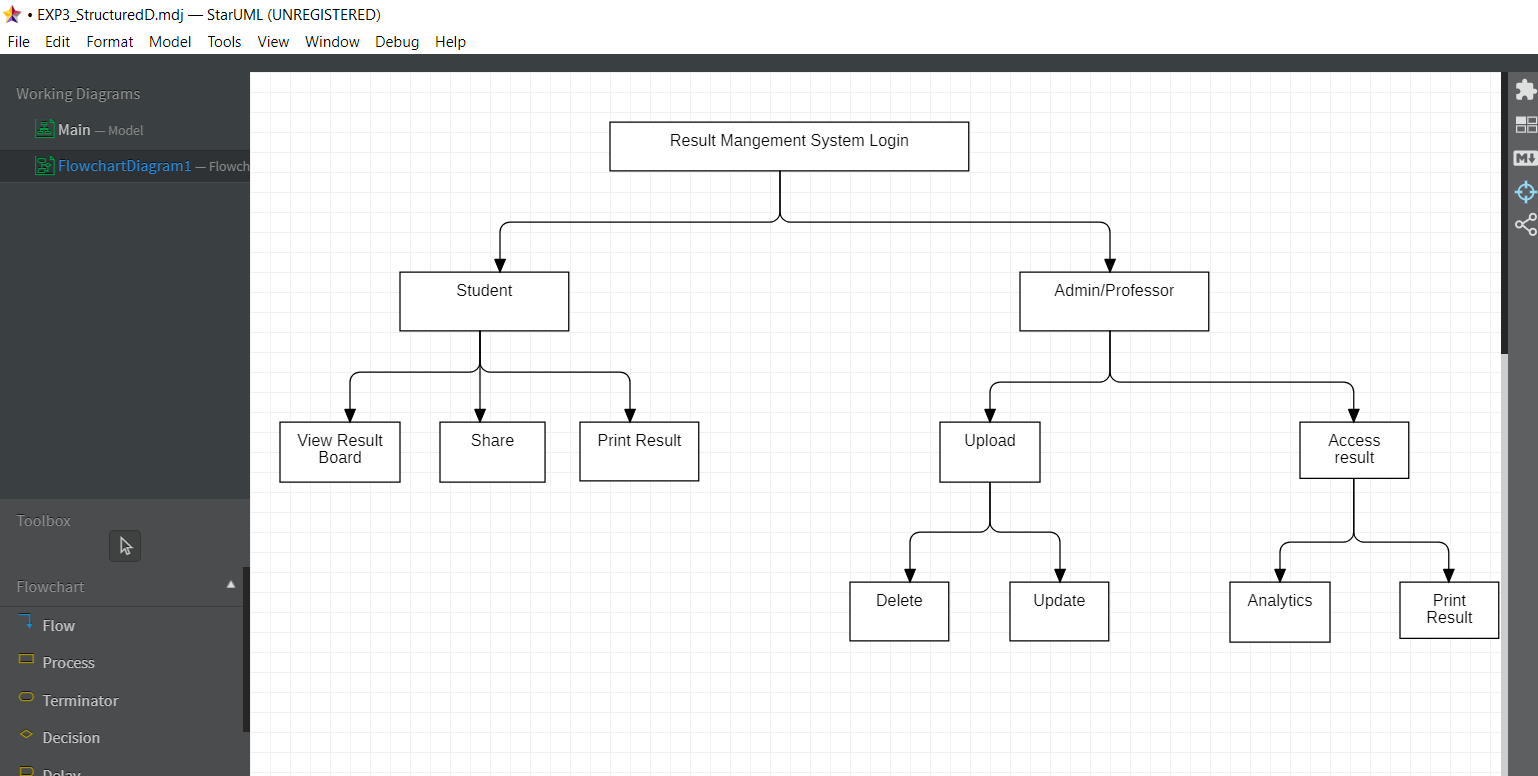


**(DFD LEVEL-0)**



**(DFD LEVEL – 1)**

**STRUCTURED CHART FOR STUDENT RESULT MANAGEMNT SYSTEM**



**VIVA VOCE QUESTIONS:**

1. **Define DFD. What are the different levels of DFD?**

A data flow diagram (DFD) maps out the flow of information for any process or system.

1. **Describe symbols used for constructing DFDs.**

It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination.

1. **Distinguish between Data Flow Diagram and a flow chart with example.**

**The main difference between DFD and Flowchart is that DFD is a graphical diagram that represents the data flow of a system while flowchart is a graphical diagram that represents the sequence of steps to solve a problem**

1. **Explain structured chart diagram.**

**Structure Chart** represent hierarchical structure of modules. It breaks down the entire system into lowest functional modules, describe functions and sub-functions of each module of a system to a greater detail.

1. **Describe symbols used for constructing structured chart diagram.**

Inputs are given to the black boxes and appropriate outputs are generated.

**EXPERIMENT – 4**

**AIM:** To perform the user’s view analysis for the suggested system: Use case diagram.

**SOFTWARE USED**: STAR UML

**PROJECT:** STUDENT RESULT MANAGEMENT SYSTEM

**THEORY:**

USE CASE DIAGRAM

A use case diagram is the primary form of system/software requirements for a new software program underdeveloped. Use cases specify the expected behavior (what), and not the exact method of making it happen (how). Use cases once specified can be denoted both textual and visual representation (i.e. use case diagram). A key concept of use case modeling is that it helps us design a system from the end user's perspective. It is an effective technique for communicating system behavior in the user's terms by specifying all externally visible system behavior.

Use case diagrams are typically developed in the early stage of development and people often apply use case modelling for the following purposes:

1. Specify the context of a system
2. Capture the requirements of a system
3. Validate a systems architecture
4. Drive implementation and generate test cases
5. Developed by analysts together with domain experts

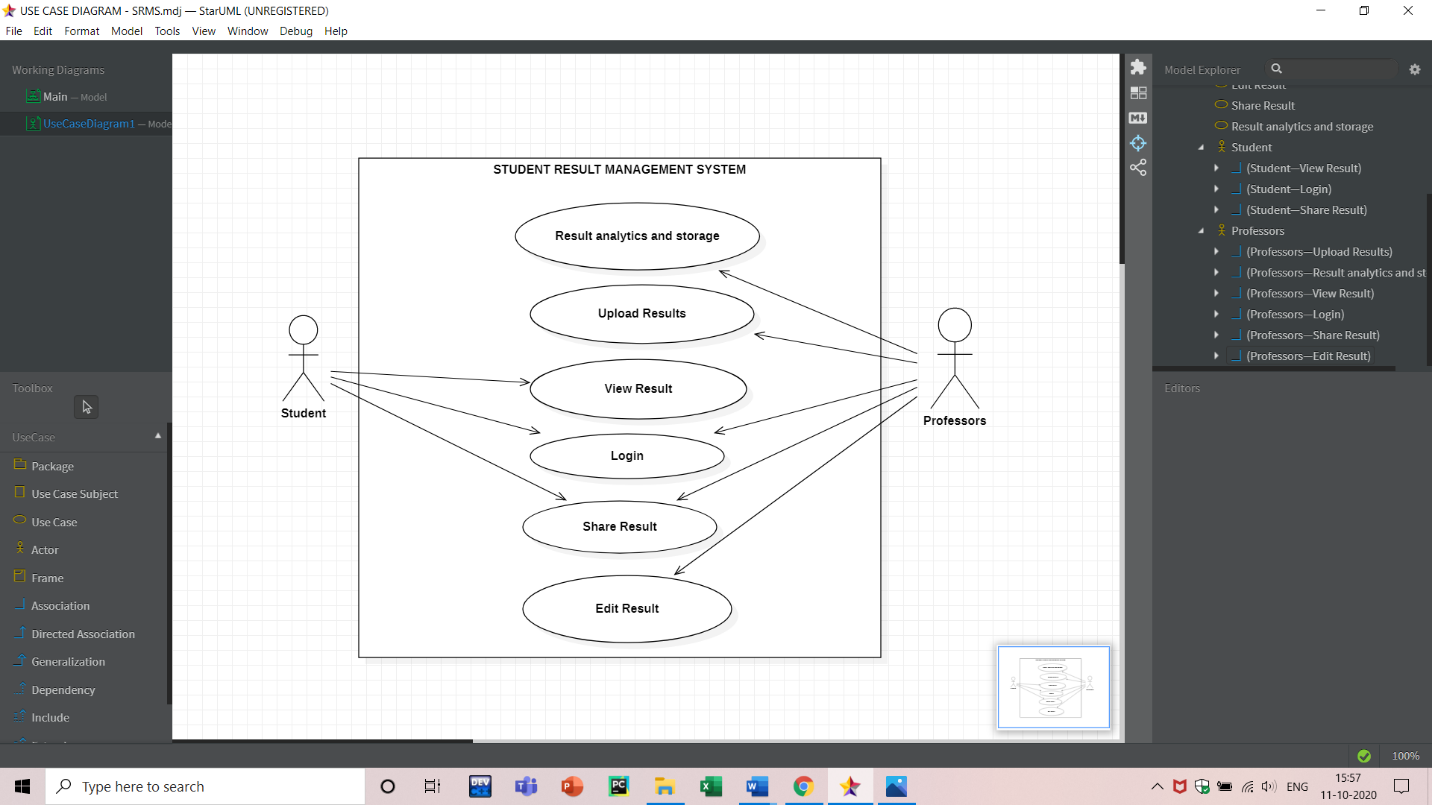
Use cases share different kinds of relationships. Defining the relationship between two use cases is the decision of the software analysts of the use case diagram. A relationship between two use cases is basically modeling the dependency between the two use cases:

* **Extends**
* **Include**
* **Generalization**

A model element used to structure the use case model to simplify analysis, communications, navigation, and planning.  If there are many use cases or actors, you can use use-case packages to further structure the use-case model in much the same manner you use folders or directories to structure the information on your hard-disk.

* To reflect the order, configuration, or delivery units in the finished system thus supporting iteration planning.
* To support parallel development by dividing the problem into bite-sized pieces.
* To simplify communication with different stakeholders by creating packages for containing use cases and actors relevant to a particular stakeholder.

**USE CASE DIAGRAM FOR STUDENT RESULT MANAGEMENT SYSTEM:**

****

**VIVA VOCE QUESTIONS:**

1. **Explain use case approach of requirement elicitation?**

Use cases are an effective and widely used technique for eliciting software requirements. The use-case approach focuses on the goals that users have with a system, rather than emphasizing system functionality

1. **Explain term: use-case, use-case scenarios, use-case diagrams?**

* USE CASE is a list of actions or event steps typically defining the interactions between a role (known in the Unified Modeling Language (UML) as an actor) and a system to achieve a goal. The actor can be a human or other external system.
* USE CASE SCENERIO represents the actions that are required to enable or abandon a goal. A use case has multiple “paths” that can be taken by any user at any one time
* USE CASE DIAGRAMS consist of actors, use cases and their relationships. The diagram is used to model the system/subsystem of an application.

1. **What are actors in use cases?**

An Actor models a type of role played by an entity that interacts with the subject (e.g., by exchanging signals and data), but which is external to the subject.

1. **Explain guidelines that should be kept in mind while creating use cases?**

Always structure and organize the use case diagram from the perspective of actors. Use cases should start off simple and at the highest view possible. Only then can they be refined and detailed further. Use case diagrams are based upon functionality and thus should focus on the “what” and not the “how”.

1. **Name the person who invented use case approach?**

Ivar Jacobson

**EXPERIMENT – 5**

**AIM:** To draw the structural view diagram for the system: Class diagram, object diagram.

**SOFTWARE USED**: STAR UML

**PROJECT:** STUDENT RESULT MANAGEMENT SYSTEM

**THEORY:**

**STRUCTURAL VIEW DIAGRAM**

The structural diagrams represent the static aspect of the system. These static aspects represent those parts of a diagram, which forms the main structure and are therefore stable. These static parts are represented by classes, interfaces, objects, components, and nodes. The four structural diagrams are − Class diagram

Structural organization consists of objects and links. The specific purpose of collaboration diagram is to visualize the organization of objects and their interaction. State chart diagram is used to represent the event driven state change of a system. It basically describes the state change of a class, interface, etc.

**CLASS DIAGRAM:**

In software engineering, a class diagram in the [Unified Modeling Language (UML)](https://en.wikipedia.org/wiki/Unified_Modeling_Language) is **a type of static structure diagram** that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

Purpose of Class Diagrams:

1. Shows static structure of classifiers in a system
2. Diagram provides a basic notation for other structure diagrams prescribed by UML
3. Helpful for developers and other team members too
4. Business Analysts can use class diagrams to model systems from a business perspective

A UML class diagram is made up of a set of classes and a set of relationships between classes.

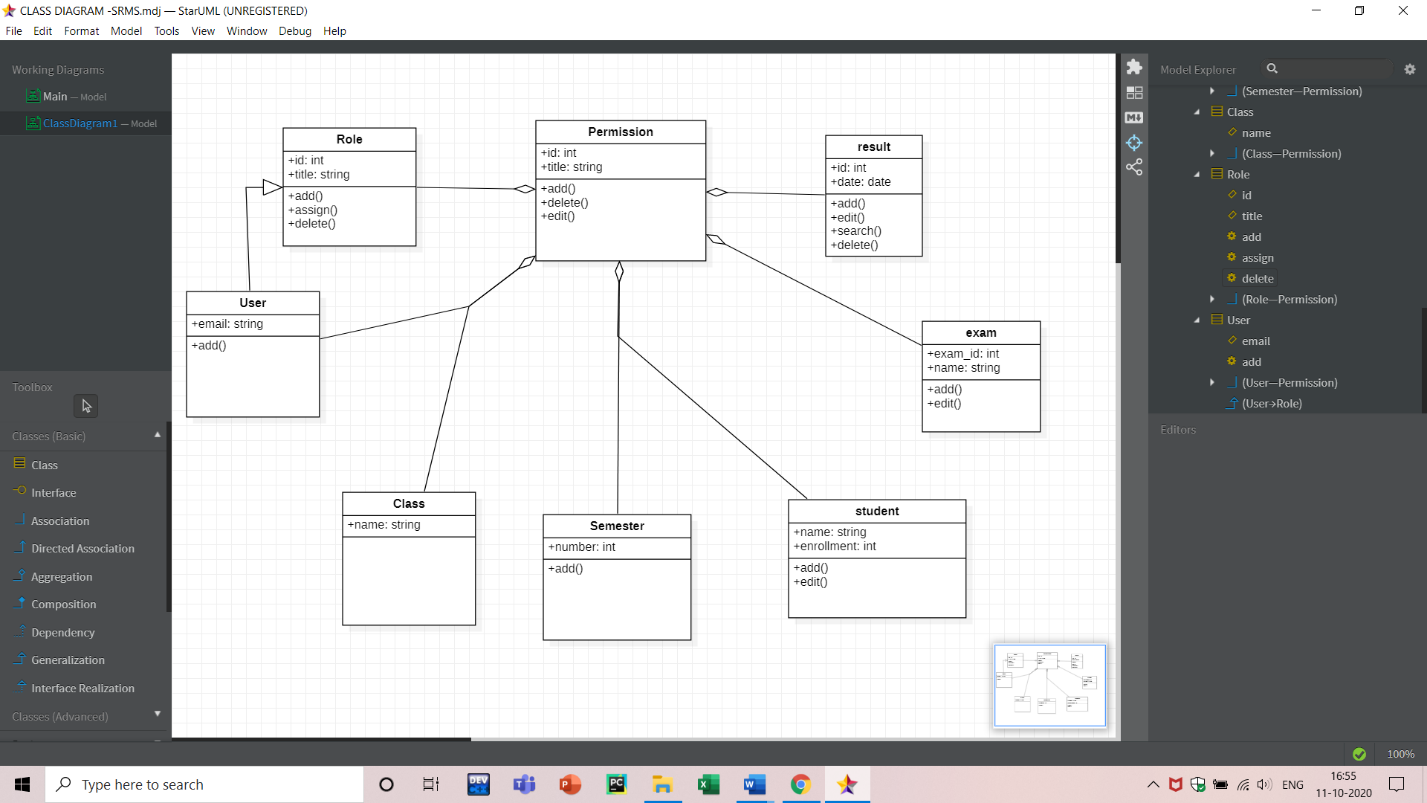
**OBJECT DIAGRAM:**

An object diagram shows this relation between the instantiated classes and the defined class, and the relation between these objects in the system. They are be useful to explain smaller portions of your system, when your system class diagram is very complex, and also sometimes modelling recursive relationship in diagram.

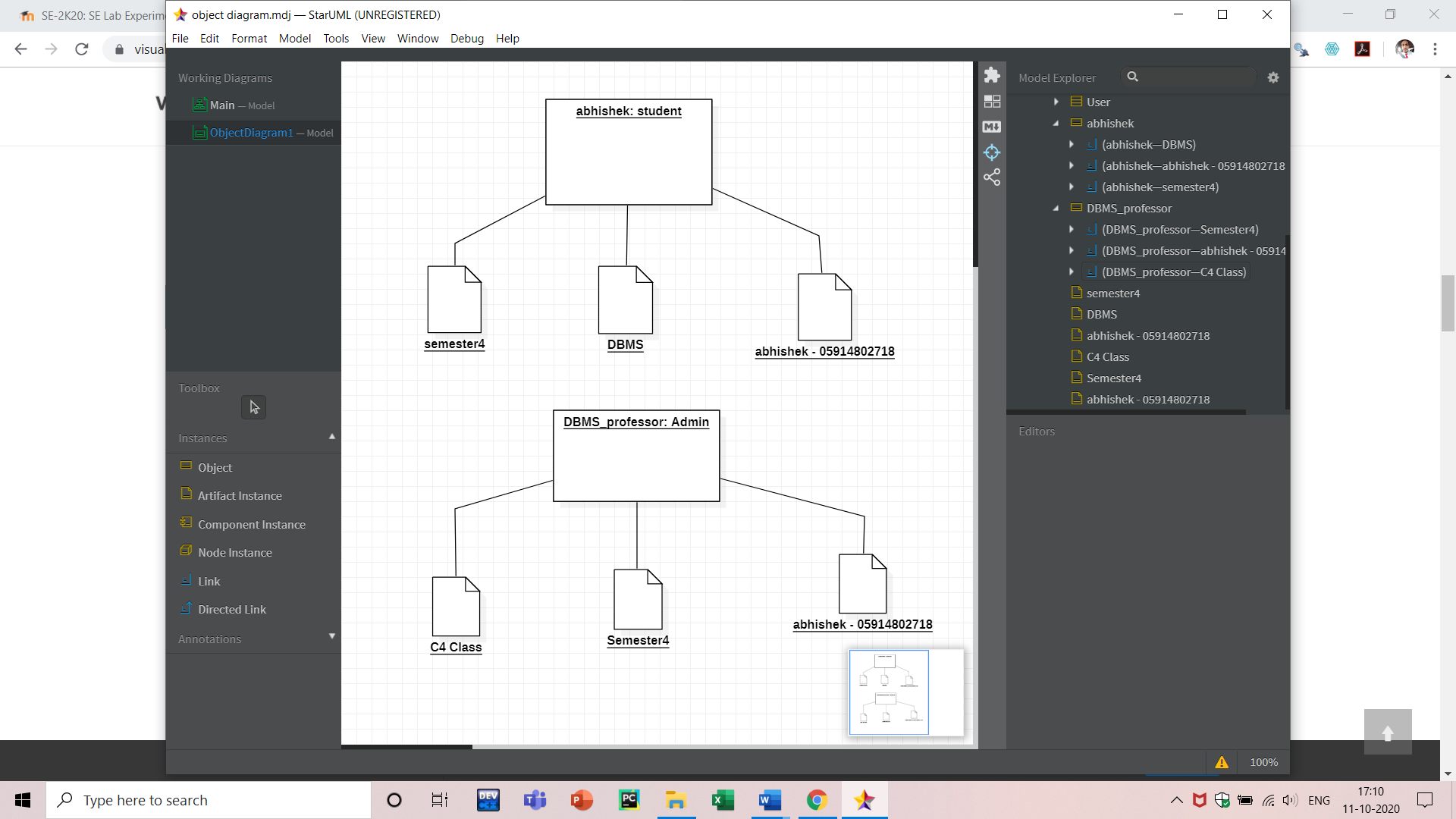
The best way to illustrate what an object diagram look like is to show the object diagram derived from the corresponding class diagram.

The following Order Management System shows their relationships. This small class diagram shows that a university Department can contain lots of other Departments and the object diagram below instantiates the class diagram, replacing it by a concrete example.

**CLASS AND OBJECT DIAGRAM FOR STUDENT RESULT MANAGEMENT SYSTEM:**

****

**(CLASS DIAGRAM)**

****

**(OBJECT DIAGRAM)**

**VIVA VOCE QUESTIONS:**

1. **Explain class diagram?**

A class diagram in the Unified Modeling Language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations, and the relationships among objects.

1. **Explain symbols used in it?**

Pre-drawn UML class diagram symbols represent class, template class, object, item, package, interface, dependency, composition, and association, etc. These symbols help create accurate diagrams and documentation. UML class diagram templates offer you many useful shapes.

1. **Explain four types of relationship used in class diagram?**

Type of UML Relationship are Association, Dependency, Generalization, and Realization. It is a set of links that connects elements of the UML model. It also defines how many objects are taking part in that relation.

1. **Explain terms classes, interfaces, collaborations and dependency?**

Classes are used to represent objects. Objects can be anything having properties and responsibility, As the object is an actual implementation of a class, which is known as the instance of a class.

Interface is just like a template where you define different functions, not the implementation. When a class implements the interface, it also implements the functionality as per requirement.

Collaboration is represented by a dotted eclipse as shown in the following figure. It has a name written inside the eclipse. Collaboration represents responsibilities. Generally, responsibilities are in a group.

1. **Explain object diagram?**

An object diagram is a graph of instances, including objects and data values. A static object diagram is an instance of a class diagram; it shows a snapshot of the detailed state of a system at a point in time.

**EXPERIMENT-6**

**AIM:** To draw behavioral view diagram: Statechart Diagram, Activity Diagram

**THEORY:** Statechart diagram is one of the five UML diagrams used to model the dynamic nature of a system. They define different states of an object during its lifetime and these states are changed by events. Statechart diagrams are useful to model the reactive systems. Reactive systems can be defined as a system that responds to external or internal events.

Statechart diagram describes the flow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is triggered. The most important purpose of Statechart diagram is to model lifetime of an object from creation to termination.

Statechart diagrams are also used for forward and reverse engineering of a system. However, the main purpose is to model the reactive system.

Following are the main purposes of using Statechart diagrams −

* To model the dynamic aspect of a system.
* To model the life time of a reactive system.
* To describe different states of an object during its life time.
* Define a state machine to model the states of an object.

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system.

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 purposes of activity diagrams is similar to other four diagrams. It captures the dynamic behaviour of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part.

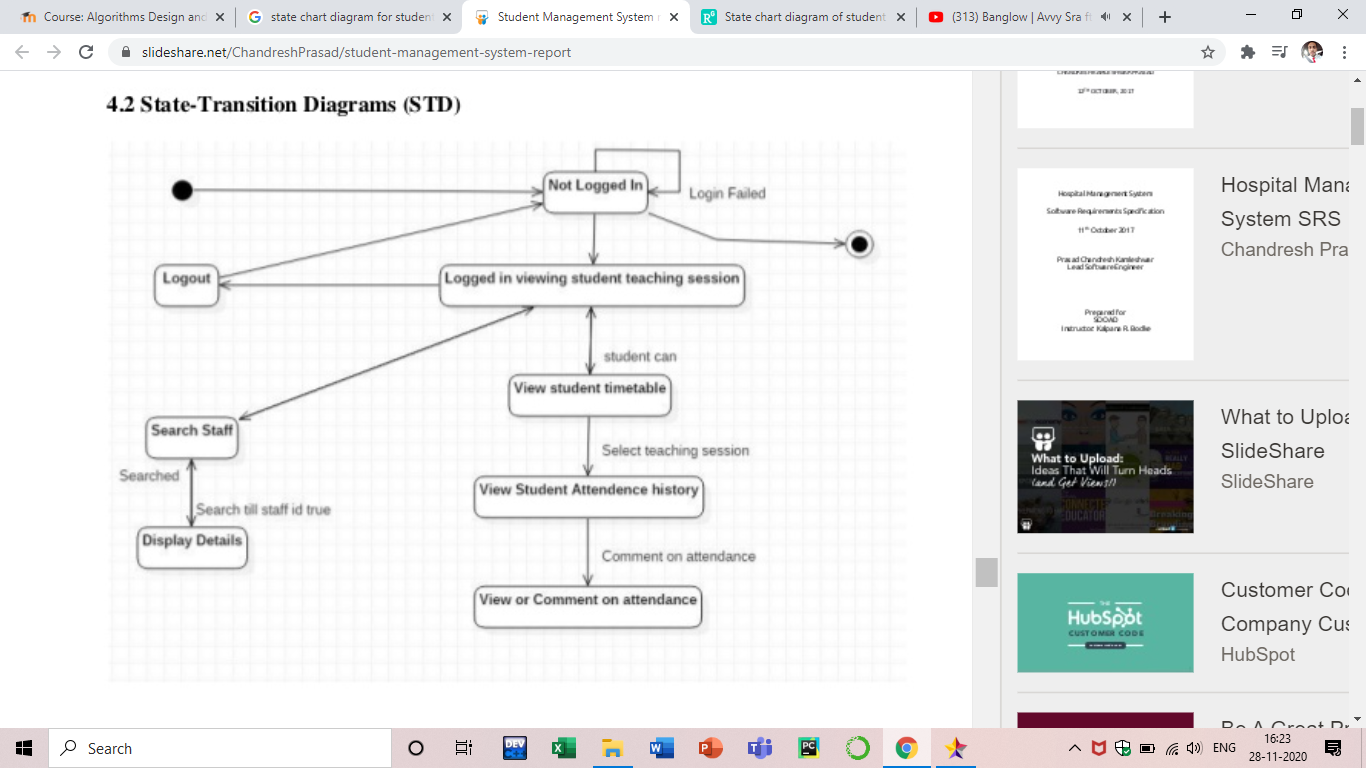
It does not show any message flow from one activity to another. Activity diagram is sometimes considered as the flowchart. Although the diagrams look like a flowchart, they are not. It shows different flows such as parallel, branched, concurrent, and single.

The purpose of an activity diagram can be described as −

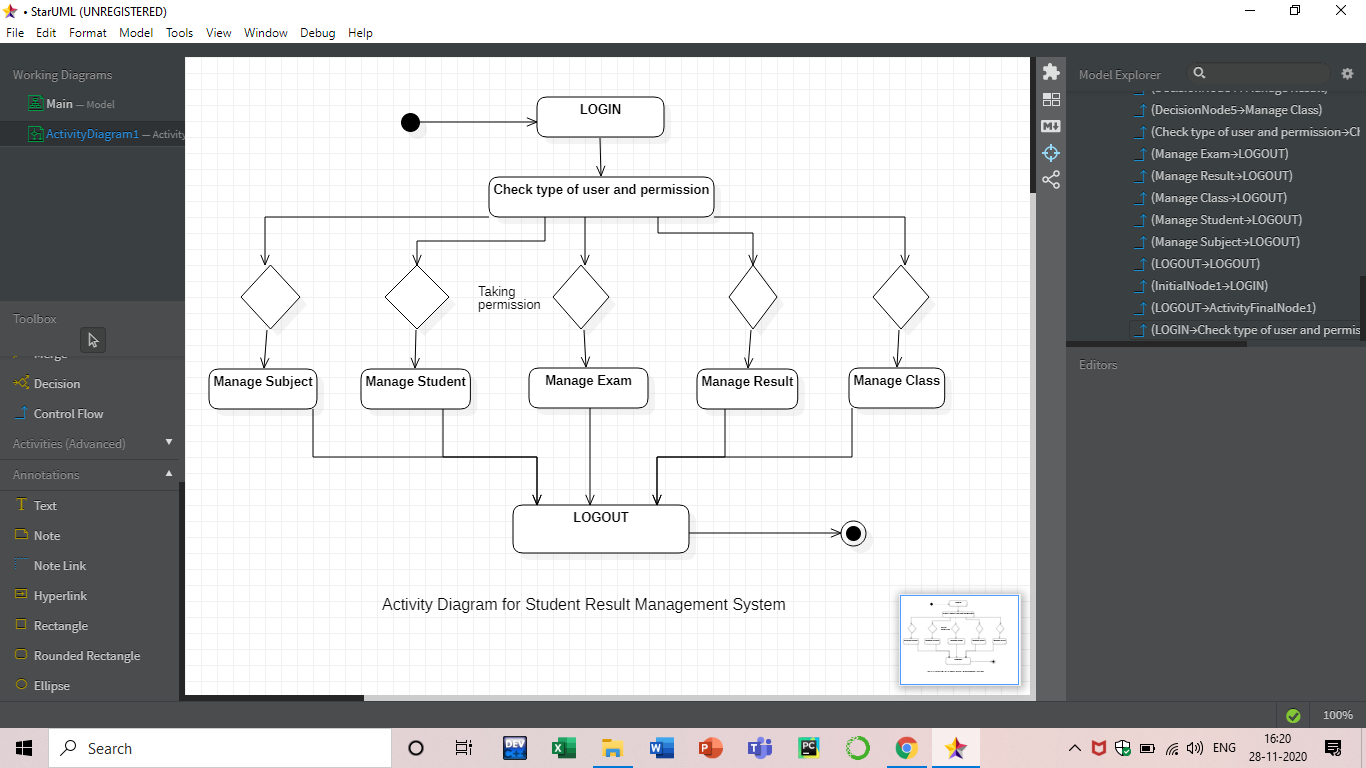
* Draw the activity flow of a system.
* Describe the sequence from one activity to another.
* Describe the parallel, branched and concurrent flow of the system.

**PROJECT:**STUDENT RESULT MANAGEMENT SYSTEM

Statechart Diagram:

****

Activity Diagram:



**VIVA – VOCE QUESTIONS:**

**Question 1. What Are The Elements In State Chart Diagrams?**

* + Initial State: This state shows the first activity of the flow.
  + State: A state represents the state of an object at a particular given point of time.
  + Transition: The transition from one state to another state of objects is represented by an arrow.
  + Event and Action: A trigger that causes a transition to occur.
  + Signal: When a message or a trigger caused by an event to a state, which causes a transition, this message is called as a signal.
  + Final State: The state diagram ends with a diagram that depicts a bulls eye is known as Final State.

**Question 2. Explain All Elements Of A State-chart Diagram?**

* + Initial State: The first or the default state the object is in. It is denoted by a solid circle.
  + State: All the states an object can go in are mentioned in this. It is represented by a rectangle with rounded edges.

**Question 3. What Are The Different Elements Of A Collaboration Diagram?**

* + Object: The interaction between objects takes place in a system. An object is depicted by a rectangle with the name of the object, preceded by a colon and underline.
  + Relation/Association: Association among objects is linked by connecting them. The cardinality can be depicted by placing qualifiers on either ends.
  + Messages: An arrow that commencing from one object to the destination object. This depicts the interaction between objects. The sequence or order of the interaction is depicted by the number.

**Question 4. Brief Explanation Of All Elements In Activity Diagrams?**

* + Activities: An activity indicates an action that performed in the system.
  + Transitions: Transitions are represented by open arrow heads. Transitions are used to indicate the flow among elements in the diagram.
  + Decision Points: The logical branching is depicted by the decision points.
  + States: A state is shown in a rounded rectangle. States are indicated to mention the mile stones of processing in the activity diagrams.

**EXPERIMENT-7**

**AIM:** To draw behavioral view diagram: Sequence Diagram, Collaboration Diagram.

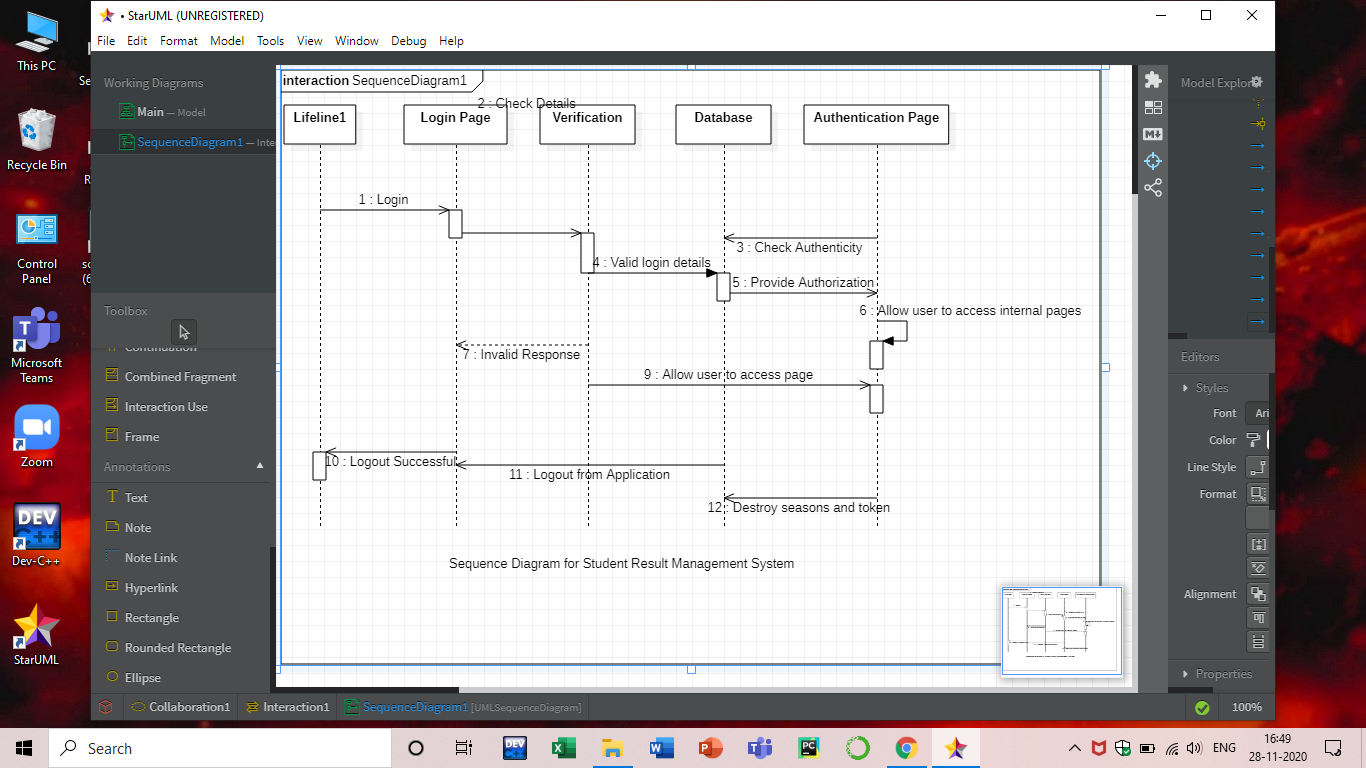
**THEORY:** A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

Purpose of Sequence Diagram

* Model high-level interaction between active objects in a system
* Model the interaction between object instances within a collaboration that realizes a use case
* Model the interaction between objects within a collaboration that realizes an operation
* Either model generic interactions (showing all possible paths through the interaction) or specific instances of a interaction (showing just one path through the interaction)

**PROJECT:**STUDENT RESULT MANAGEMENT SYSTEM

Sequence Diagram:

****

**VIVA – VOCE QUESTIONS:**

**Question 1. Explain About Aggregation?**

Aggregation gives a much more detail than association. In aggregation you can name it and it can have same adornments. It may not be involved with more than two classes. It can have a collection of classes but its classes are not dependent on the life cycle. It's contents are not destroyed even when its classes are destroyed.

**Question 2. Detail The Meaning Of Association?**

There are five types of association but importance is given to only two they are Bidirectional and unidirectional. It represents a family of links. Binary associations are represented by two ends and they are connected to class box. Higher order associations can have more than two ends.

**Question 3. Write Down The Main Phases Of Problem Solving Model?**

The main phases of problem solving model are given below:  
• Data gathering.  
• Problem redefining.  
• Finding ideas.  
• Finding solutions.  
• Implementation.

**Question 4. What Is Prototype?**

A prototype is a system or partially complete system that is built quickly to explore some aspects of the system requirements. It is not intended as the final system.

**EXPERIMENT-8**

**AIM:** To draw implementation view diagram: Component Diagram for the system

**THEORY:** Component diagrams are different in terms of nature and behaviour. Component diagrams are used to model the physical aspects of a system. Now the question is, what are these physical aspects? Physical aspects are the elements such as executables, libraries, files, documents, etc. which reside in a node.

Component diagrams are used to visualize the organization and relationships among components in a system. These diagrams are also used to make executable systems.

## Purpose of Component Diagrams:

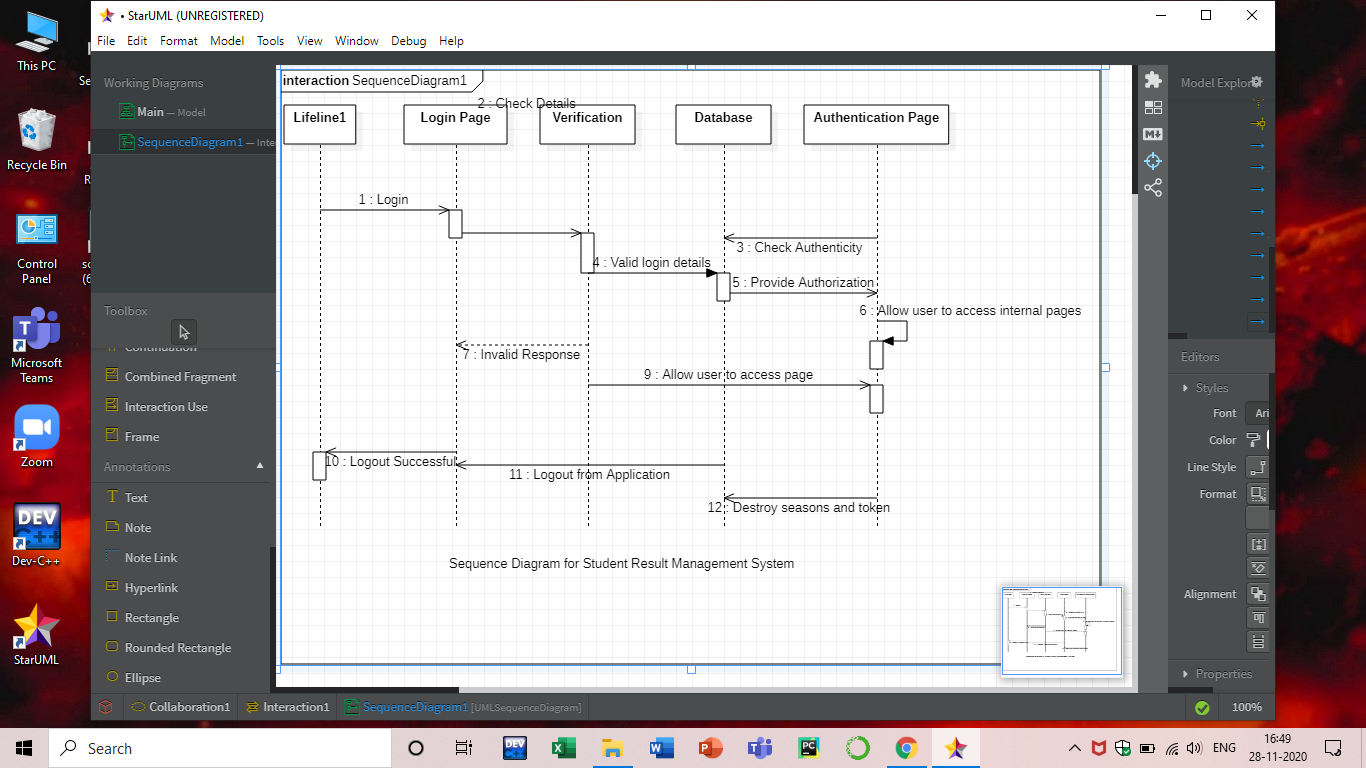
Component diagram is a special kind of diagram in UML. The purpose is also different from all other diagrams discussed so far. It does not describe the functionality of the system but it describes the components used to make those functionalities. Thus from that point of view, component diagrams are used to visualize the physical components in a system. These components are libraries, packages, files, etc. Component diagrams can also be described as a static implementation view of a system. Static implementation represents the organization of the components at a particular moment. A single component diagram cannot represent the entire system but a collection of diagrams is used to represent the whole.

The purpose of the component diagram can be summarized as −

* Visualize the components of a system.
* Construct executables by using forward and reverse engineering.
* Describe the organization and relationships of the components.

**PROJECT:**STUDENT RESULT MANAGEMENT SYSTEM

Component Diagram:

****

**VIVA – VOCE QUESTIONS:**

**Question 1. What Is Component Diagrams In Uml?**

A component diagram is particularly useful with teams of larger size. UML components are great to perform architectural landscape for a specific system. The component diagram allows to model high level software components and interfaces to those components.

**Question 2. What Is An Operation Signature?**

determined by the operation’s name, the number and type of its parameters and the type of the return value if any. Polymorphically redefined operations have the same signature.

**Question 3. What Are Subclass And Supper Class?**

**Subclass:**a specialized class that acquires general features from its ancestor super-classes in a generalization hierarchy, but that also adds one or more specialized characteristics of its own.  
**Superclass:** a generalized class that is an abstraction of the common characteristics of its subclasses in a generalization hierarchy.

**Question 4. What Is Usability Requirement?**

Usability requirement are those that will enable us to ensure that there is a good match between the system that is developed and both the users of that system and the tasks that they will undertake when using it.

**EXPERIMENT-9**

**AIM:** To draw environmental view diagram: Deployment Diagram for the system

**THEORY:** Deployment diagrams are used to visualize the topology of the physical components of a system, where the software components are deployed.

Deployment diagrams are used to describe the static deployment view of a system. Deployment diagrams consist of nodes and their relationships.

## **Purpose of Deployment Diagrams**

The term Deployment itself describes the purpose of the 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.

UML is mainly designed to focus on the software artifacts of a system. However, these two diagrams are special diagrams used to focus on software and hardware components.

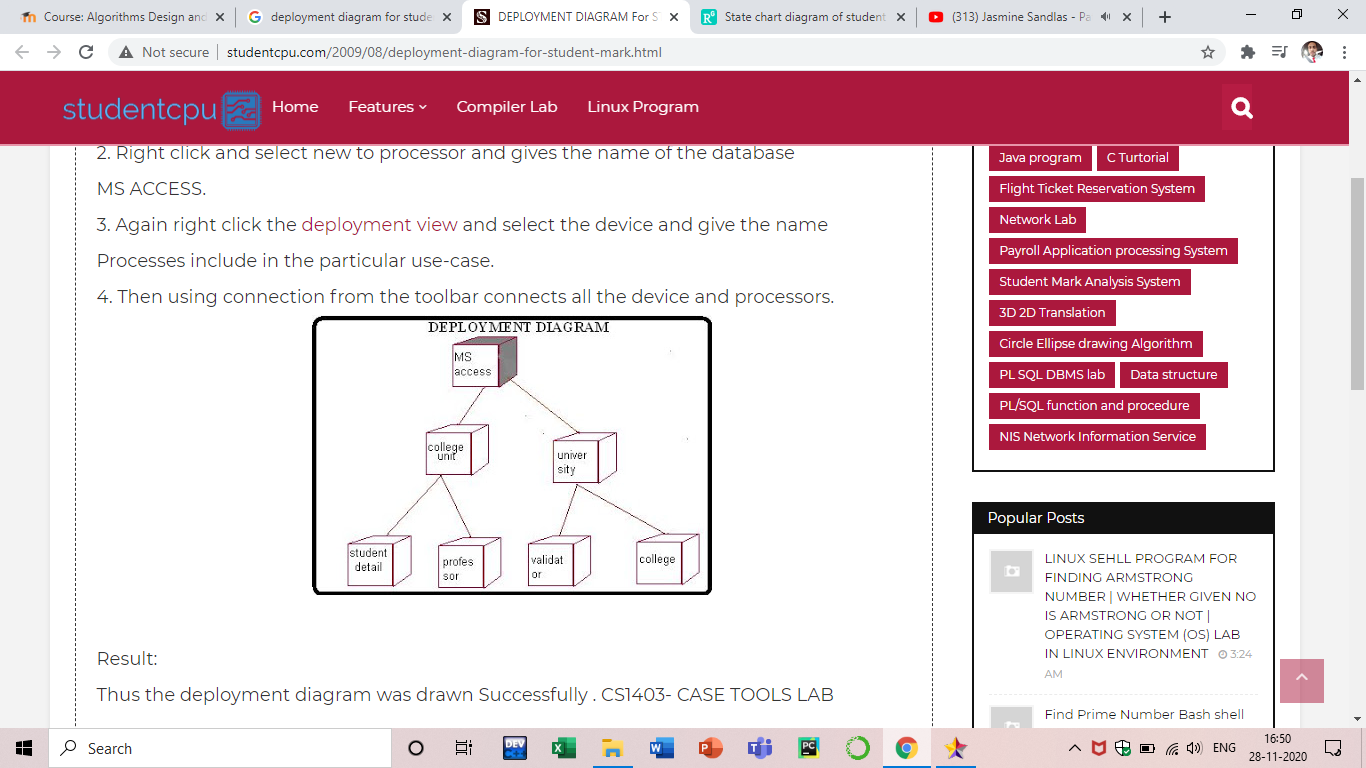
Most of the UML diagrams are used to handle logical components but deployment diagrams are made to focus on the hardware topology of a system. Deployment diagrams are used by the system engineers.

The purpose of deployment diagrams can be described as −

* Visualize the hardware topology of a system.
* Describe the hardware components used to deploy software components.
* Describe the runtime processing nodes

**PROJECT:**STUDENT RESULT MANAGEMENT SYSTEM

Deployment Diagram:

****

**VIVA – VOCE QUESTIONS:**

**Question 1. What is a Deployment Diagram?**

A diagram that shows the run-time configuration of processing nodes (q.v.) and the components, processes and objects that are located on them. One of two UML implementation diagrams (q.v.).

**Question 2. What are Referential Integrity, Dependency Constraint And Domain Integrity?**

**Referential Integrity:**Referential integrity ensures that an object identifier in one object actually refers to an object that exists.  
**Dependency constraints:**Dependency constraints ensure that attribute dependencies, values are maintained consistently, where the value of one attribute is calculated from other attributes, are maintained consistently.  
**Domain integrity:**Domain integrity ensures that attributes only hold permissible values.

**Question 3. What Are Components Of An Mvc Architecture?**

The components of MVC architecture are:  
• Model  
• View  
• Controller

**Question 4. What Is Integrity Constraint?**

A constraint that has to be enforced to ensure that the information system holds data that is mutually consistent and is manipulated correctly.

**Question 5. What Is Interface?**

That part of the boundary between two interacting systems across which they communicate, the set of all signatures for the public operations of a class or package.