

Practical No :- 06

Aim: To understand and prepare FR diagram for given problem statement.

Theory: Entity Relationship Model :

Entity relationship model is used to represents logical design of a database to be created.

In FR model real world objects are abstracted as entities and different possible associations among them are modeled as relationships.

If few e.g. Student and college, they are two entities. Students study in college. So, these two entities are associated with relation "Studies-in".

RAISONI GROUP

— a vision beyond —

* Entity Relationship set :-

An entity is a collection of all similar entities. For example, "student" is an entity set that abstracts all students.

Ram, John, are specific entities belongs to this set.

* Attribute of Entity.

Attributes are the characteristics describing any entity. Any entity belongs to an entity set. In a set can be

described by zero or more attributes

For example, any student has got a name, age, an address. At any given time a student can study only at one school.

* **Key :-** One or more attributes of an entity set can be used to define following keys:

- **Super key :-** One or more attributes which when taken together, help to uniquely identify an entity is an identity set.

RAISONNI GROUP

- **Candidate key :-** ~~which is a minimal subset of a super key. In other words, a super key might contain redundant attributes, which do not help in identifying an object uniquely.~~
When such attributes are removed, the key formed so is called a candidate key.

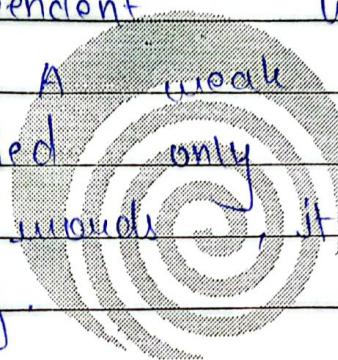
- **Primary key :-** A database might have more than one candidate key. Any candidate key chosen for a particular

implementation or primary key of the database is called primary key.

- Primary attribute :- Any attribute taking part in a super key.

* Weak Entity :-

An entity set is said to be weak if it is dependent upon another entity set. A weak entity can't be uniquely identified only by its attributes. In other words, it doesn't have a super key.



* ~~Father~~ Entity Generalization And Specialization beyond

Once we have identified the entity sets, we might find some similarities among them, for example, multiple person interact with one automobile, and just employees are other source providers.

ER model uses the "IS-A" hierarchy to depict specialization and thus, generalization.

* Mapping

Cardinalities :-

One of the main tasks of ER modeling is to associate different entity sets. Let's consider two entity sets E1 and E2 associated by a relation set R. Therefore there are four types of mappings:

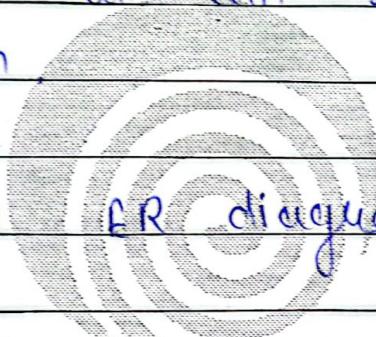
- One to One : An entity in E1 is related to at most one single entity in E2, vice versa.
- One to many :- An entity in E1 could be related to zero or more entities in E2.
- Many to One & Zero or more of entities in E1 could be associated to a single entity in E2.
- Many to Many : Any no. of entities could be related to any no. of entities in E2, including zero and vice versa.

* Importance of ER diagram :-

* ER diagram for our given problem statement.

→ Design a e-commerce website.

for making ER diagram, the first step is to identify the entities, attributes and relationships. Then we can represent them in ER diagram.



10) Entities in our ER diagram for e-commerce website :-

- User / Admin :- Represents both user and admins.
- Login :- Contains user credentials.
- Product :- A key entity, ~~product~~ available for sale.
- Category :- categorize product with attributes.
- Cart :- Shows product selected into with attributes.
- Order :- Represents customer order with attributes.
- Customer Product :- Represents product that user can customize.
- User Uploads & Manages images uploaded by user with attributes.
- Views : Tracks the no. of product is viewed.

• Attributes

- User / Admin :- - username, email, contact, role
- Login :- - username, password
- Product :- product-id, product-name, product-desc, price, color, brand, image1, image2, image3.
- Category :- cat-id, cat-name, cat-description.
- Cart :- quantity, size, custom
- Order :- order-id, address, city, state, zip-code, contact
- Custom-product :- title, image, cost, uid, cid.
- User-Upload :- uid, image.
- Views :- views.

RAISONNI GROUP

• Relationships :- a vision beyond

- Login - User / Admin
- Admin - Category
- Category - Product
- Product - Cart
- Product - Order
- Order - Customer
- Custom-product - cart
- User - User Upload
- Product - Views

P	T	D	K	Total
3M	3M	3M	6M	15M
3	3	3	4	13
Sign With Date				8/10/2023

Result :- Hence, we studied FR document for our given problem statement.

Aim :- To study and illustrate the use of class diagram on per the given problem statement.

Theory :- Class Diagram :-

It is a graphical representation for describing a system in context of its static construction.

* Elements in class Diagram :-

class diagram contains the system classes with its static members, operations and relationships between classes.

RAISONI GROUP

a vision beyond

* Class :- A set of objects containing a similar state member function is described by class. In UML syntax, class is identified by solid outline rectangle with three compartments which contain.

- class name
- Attributes
- Operations
- Generalization and Specialization.

Operation eg. :-

Course
course name : string
course ID : string
addcourse (1) : string
remove course (1) : string

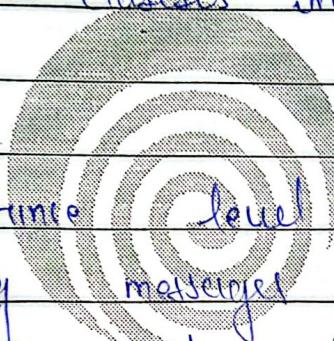
* Relationships :-

- Association Existing relationships in a system describe legitimate connections between the classes in that system.

• Association

It is an instance level relationship that allows exchanging messages among the objects of both ends of association.

A simple straight line connecting two class boxes represents basic association.



• Aggregation :-

It is a special form of association which describes a part-whole relationship between a pair of classes. It means, in a relationship, when a class holds some instances of selected classes, then that relationship can be designed as aggregation.

- Multiplicity :-

It describes how many no. of instances of one class is matched to the no. of instances of another class in an association.

Notations for different types of multiplicity :-

Single Instance - 1

zero or one Instance - 0..1

zero or more Instance - 0..*

One or more instance - 1..*

Particular Range (thin box) - 2..6

* Class Diagram for our given problem
Statement :- a vision beyond

functionalities :-

→ Customer :-

- Attributes :- customer-id, customer-name,
- Methods :- email, mobile, address.

• Method :- register(), login(), updateProfile()

→ Seller :-

- Attributes :- seller-id, address, email, mobile, seller-rating

• Method :- register(), login(), updateProfile()

→ Product :-

- Attributes :- product-id, product-name,

• qty-available , product-cost , seller-id ,
posted-date .

• Methods :- addToCart() , sellProduct() , getProductDetail()
, buyProduct() .

→ Cart :- • Attributes :- count-id , product-id , quantity .

• Methods :- addItem() , updateQuantity() , cartDetails(),
PlaceOrder() , Checkout() .

→ Order :- • Attribute e - order-id , order-date , seller-id ,
recustomer-id , product-id , total-amount ,
delivery-date , delivery-status .

• Methods :- orderDetails() .

→ Payment :- • Attribute e - id , order-id , paid , total , details .

• Methods :- sendOTP() , confirmTransaction() , getPaymentDetails() .

→ Admin :- • Attribute :- admin-id , admin-password .

• Methods :- register() , login() , updateCatalog() .

→ Review :- • Attributes : review-id , customer-id , review-content ,
rating , product-id .

• Methods :- addReview() , deleteReview() , editReview() .

Result :- Hence , we have successfully
Illustrated Class diagram
problem statement .

R	S	F	T	Total
3M	100	600	900M	15M
3	2	3	4	12
Sum				36
Sign With Date				

Practical No.: 08

Aim:- To study and illustrate use case diagram for your given problem statement.

Theory:- Use Case Diagram :-

Use case diagram belongs to the category of the behavioural diagram of UML design.

Use case diagrams aim to present a graphical overview of the functionality provided by the system. If consists of a set of actions that the concerned system can perform, one or more actions achieve dependencies among them.

RAISONNI GROUP

• Achour :- a vision beyond

An actor can be defined as an object interacting to the system, which interacts with the system to get some meaningful work done. Actors could be human, devices, or even other systems.

• Use case :- A use case is simply a functionality provided by a system. continuing with the example of ATM, withdraw cash is functionality of that the ATM provides. Therefore, this is a use case.

The cases interact or include both successful and unsuccessful scenarios of user interactions with the system.

* **Subject :-** Subject is simply the system under consideration. Use cases apply to a subject. For example, ATM is a subject, having multiple use cases and multiple actors interacting with it.

* **Graphical Representation :-**

An actor is represented by a stick figure and name of the actor is written below.

A use case is depicted by the an ellipse and name of the use case written inside it. The object is shown by a rectangle. Label for the system could be put inside it. The cases are drawn inside the rectangle, and others are drawn outside the rectangle, as shown in figure.

* **~~Association with~~ between Actor and Use Case**

A use case is triggered by an actor.

Actions can be used cases are connected through Joins by associations indicating that the two communicates through message passing.

* An action must be associated with at least one use case.

* Use Case Relationships :-

There are three types of Relationships.

→ Include Relationship

→ Extend Relationship

→ Use Case Generalization.

* Identifying Actions

For given problem statement, the action could be identify by asking the questions:

- Who gets most of the benefit from system?
- Who keeps the system working?
- What other software / hardware does the system interact with?
- Any interface between concerned system and any other system.

* Identifying Use Cases :-

Once the primary and secondary action are being identified, we have to find out their

goals i.e. what are functioning they can obtain from the system.

* Guidelines for drawing use case diagram

Following general guidelines would be kept in mind while trying to draw a use case diagram:-

- Determine the system boundary.
- Ensure that individual actors have well-defined purpose.
- Use cases identified should let some meaningful work done by others.
- Use include relationship to encapsulate common behaviour among ~~among them~~ ~~beyond~~, if any.

* The use case diagram for our problem statement:

This use cases diagram shows :-

- 1) Actor : - • Customer
• Admin / Manager
- 2) Use Cases : - • Create Account
• Log In

- Show orders
- Add products
- Search for product
- View product details
- Update, View Cart
- Place Order
- Checkout
- Manage Accounts
- View order history

3º Relationships :- <> Includes Relationship>:-

- Log In , ◦ checkout,
- Manage Accounts , ◦ Dashed around .

RAISONNI GROUP

Result:- Hence we ~~study~~ studied and illustrate the use diagram for our given problem statement.

P	T	D	K	Total
3M	3M	3M	6M	15M
3	3	3	5	14
Sign With Date				8/8/2023

Practical 09

* Aim - To study and illustrate the various UML diagram design tools

* Theory -

What is UML

Unified Modelling Language is an essential tool in software engineering for visualising, specifying, construction and documenting software system.

This report covers several popular UML diagram design tools, their features and their applicability in SPM.

Lucid Chart

Key Features -

- Web based tools for real time collaboration
- Supports all 14 UML types
- Extensive template library
- Intuitive drag and drop interface
- Integration with popular platforms (Gsuite, Atlassian)

SPM Application. - Ideal for team projects due to its collaboration features. Useful for creating and sharing diagrams throughout the development lifecycle.

2. Visual Paradigm

Key Features -

- Comprehensive UML modelling tool.
- Code generation and reverse engineering.
- Database design and ORM.
- Requirements Management.
- Agile project management features.

SEPM Application

Suitable for complex projects requiring extensive modelling and integration with other projects management aspects.

3 Star UML

Key Features -

- Open source tool with a user friendly interface.
- Support 11 types of UML diagram.
- Model driven development approach.
- Extensible through add ons.
- Cross platform (Windows, macOS, Linux)

SEPM application

Good for academic projects and small to medium sized teams. Its open source nature makes it cost efficient for students to use.

4 Dia

Key Features

Practical No :- 10

Aim :- To study and illustrate the various automation tools like selenium.

Theory :- There are several automation tools across different domain, each used to improve efficiency, reduce human error, or streamline workflow. Below are the categories of automation tools with their usage.

- ★ Testing automation tools :-
Testing automation tools helps automate the testing of software application. They ensure that software works as expected by running predefined test scripts without manual intervention.
- Selenium :- A widely used open-source for automating web browser actions.
 - Use Case :- Automating browser testing across different browsers like chrome, firefox and safari.
 - Example :- Automatically performs login and form submission tests on a website.

- JUnit :- JUnit :- A unit testing framework for Java application.
 - Use case :- Automating the testing of individual units of source code in Java programs
 - Example :- Running automated tests for Java methods in an application.

- Appium :- A mobile application testing tool that supports automation of both android and ios platforms.
 - Use case :- Automating mobile app testing across different devices.
 - Example :- Testing mobile app functionalities such as navigation, swipe gestures, and login processes.

* Selenium Testing :-

It is an open-source automation testing tool primarily used for automating web applications. It allows testers to write scripts in multiple programming languages to automate web browsers.

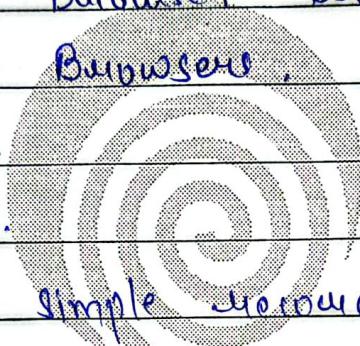
Selenium supports multiple browsers such as chrome, firefox, safari and Edge.

* Components of Selenium :-

1. Selenium Web Driver :-

→ Purpose :- Automates the browser and interacts with web elements.

- Architecture :-
- Client Libraries
 - JSON Wire Protocol.
 - Browser Drivers.
 - Browsers.



2. Selenium IDE :-

→ Purpose :- A simple record and playback tool used for creating quick test scripts without coding.

DAISCOM GROUP

a vision beyond

→ Features :-

- Allows recording interactions with a web-page.

• Test steps are recorded as commands can be played back.

→ Limitations :- It has limited control flow, cannot handle complex scenarios, and is less flexible than Web Driver.

3.) Selenium Grid :-

→ Purpose :- Used for Parallel test execution across multiple environments.

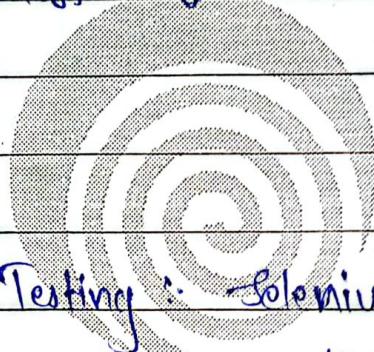
→ Architecture :-

- Hub
- Nodes

→ Use Cases :-

- Cross Browser Testing.

★ Use Cases :-



→ Cross browser Testing :- Selenium web-driver allows tests to run on different browsers, ensuring compatibility across chrome, firefox, safari, etc.

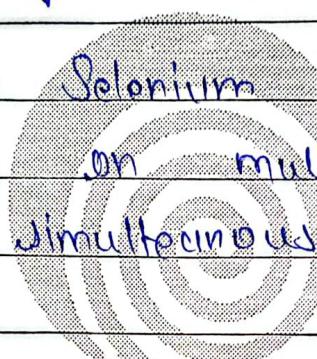
RATIONAL GROUP

→ Regression Testing :- Automating repetitive test cases to check if new code breaks existing functionality.

→ Continuous Integration / Continuous Deployment (CI/CD) :- Automated selenium test can be integrated with tool like jenkins, allowing automatic test execution during the build process.

* Advantages of Selenium :-

- Open source and free to use.
- Support multiple languages and browsers.
- Flexible : can be integrated with various testing frameworks and CI/CD tools.
- Parallel execution :- Selenium grid allows you to run tests on multiple machines and browsers simultaneously.



Limitation of Selenium :-

RAISONI GROUP

No desktop or mobile browser automation.

Learning curve.

Limited support for dynamic web elements.

~~Result :- Hence, we studied and practiced automation tools like automation selenium.~~

P	T	D	K	Total
3M	3M	3M	6M	15M
3	3	3	9	12
Sign With				
Date				8/8/2023