Assignment no :- 02

Topics covered :-
- XML
- XML DTD
- JSON
- DOM
- REST API

Date of performance :- 08-08-22

Evaluation Criteria	Marks (out of 3)	Date	Signature of Instructor
Punctuality	03	08-08-22 08/8/2022	
Problem solving technique	03		
Attainment level (out of 3)	03		

Assignment No :- 02

Q.1) What is XML ? explain the XML tree structures

- **XML :-**
  - : XML stands for eXtensible markup language
  - XML is a markup language much like HTML
  - XML was designed to store and transport data
  - XML is a S/w and hardware - independent tool for storing & transporting data.
  - XML is a W3C Recommendation.

- **XML Tree Structure :**
  - An XML document has a self descriptive structure
  - it forms a tree structure which is referred as an XML tree.
  - The tree structure makes easy to describe an XML document.
  - The tree structure which refers as an XML tree.
  - A tree structure contains root element (as parent) child element and so on. it is very easy to traverse all succeeding branch and sub-branches and leaf nodes starting from the root

- Example of an XML document

```
<?xml version ='1.0'?>
<college>
    <Firstname> priyush </Firstname>
    <lastname> KHOBragade </lastname>
    <contact> 9511689393 </contact>
    <email> Rhobragadepriyush@gmail.com </email>
</ college>
```

Q.2) What is XML DTD?

DTD

DTD stands for Document Type Definition. It defines the legal building blocks of an XML document. It is used to define document structure with a list of legal element and attributes

- **Purpose of DTD**

its main purpose is to define the structure of an XML document. It contains a list of legal elements and defines the structure with help of them

Ex. of well-formed and valid XML document

```
<?XML version = "1.0"?>
<!DOCTYPE employee SYSTEM "employee.dtd">
<employee>
    <firstname> Priyanka </firstname>
    <lastname> Khairnar </lastname>
    <email> Demo@gmail.com </email>
</employee>
```

• Description of DTD :-

<!DOCTYPE employee :- it define that the root element of ty document is employee

<!ELMENT empolyee) : it define that te empolyee element contains 3 elements "firstname, lastname and email".

<!ELMENT firstname) : it define that firstname element is #PCDATA TYpe (parse-able data type)

<!ELMENT lastname) :- it define, that te lastname element contains is #PCDATA type

<!ELMENT email> : it define that te email element is #PCDATA type

• De DTD with entity Declaration

A DOCTYPE declaration can also define special strings that used in the XML file

An entity has three part :-

- An ampersand (&)
- An entity name
- A semicolon (;

| Syntax = <!ENTITY entity name "entity value"> |

Q. 3) Describe the JSON with the help of syntax & explain it with help of one example.

- **JSON**

- JSON (Javascript Object Notation) is a text based data exchange format.
- It is a collection of key value pair where the key must be string type, and the value can be of any of the following type.
  - Number
  - String
  - Boolean
  - Array
  - Object
  - Null

- **Important Rule of syntax:-**
- In the JSON data format, the key must be enclosed in double quotes
- The key & value must be separated by a colon (:) symbol.
- There can be multiple-key-value pair.
- These key-value pairs must be separated by a comma (,) symbol.
- No comments (// or /* */) are allowed in JSON data.

```
{ "name" : "pk"
  "age" : 2,
  "city" : "Mumbai"
}
```

- A collection of key-value pair enclosed by a pair of curly brackets {}
- A collection of an ordered list of key-value pairs separated by comma(,) and enclosed by a pair of square brackets [...]

EX.

```
{ "name": "Priyash",
  "id" : "5010,
  "age": 23,
   doj : "10-12-20",
  Married" : false.
}
```

The above JSON data shows the attribute of an employee The attribute are:

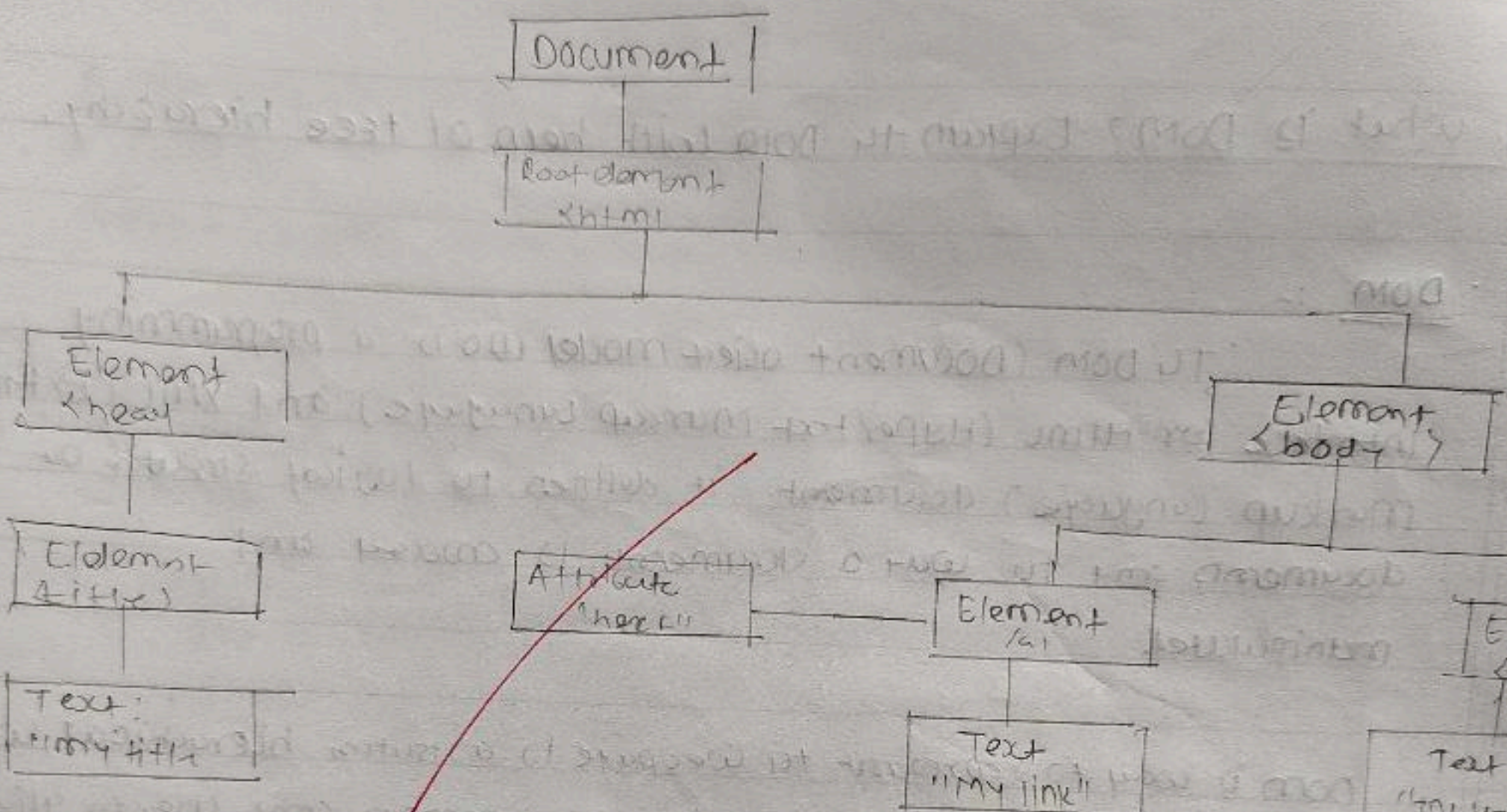
- Name : the Name of the employee. The value is of string type
- id : A unique identification of an employee it is a string type
- age : The current age of the employee, it is number.
- doj : The date of the joining employee the company it string type
- marry : The employee married? it so true or false so the value of Boolean type.

Q. 24] What is DOM? Explain the DOM with help of tree hierarchy.

DOM :-

The DOM (Document object model (DOM) is a programming interface for HTML (HyperText Markup Language) and XML (Extensible Markup Language) document. It defines the logical structure of documents and the way a document is accessed and manipulated.

DOM is way to represent the webpage in a structure hierarchical way so that it will become easier for programmers and user to glide through the document. With DOM, we can easily access and manipulate any tag, ID, Class, Attribute, or element of HTML using commands or method provided by the Document object. Using DOM, the Javascript get access to HTML as well as CSS of the web page. It can add behavior to the HTML element. So basically Document object Model is an API that represents and interact with HTML or XML document.


Document
Root element
<html>
Element
<head>
Element
<body>
Element
<title>
Attribute
"href"
Element
<a>
Text:
"My Title"
Text
"My link"

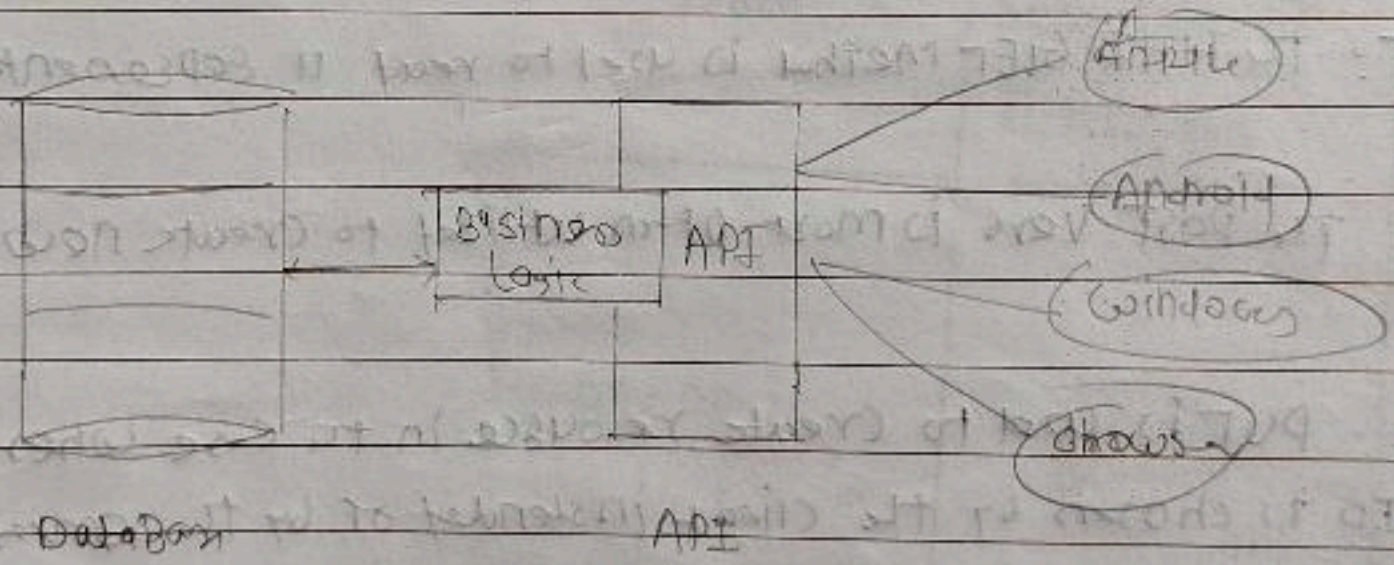
Page No. : 18
Date :

57 Explain · REST API.

* API :

Application programming interface (API) is software interface that allow two Applications to interact with each other without any user intervention.

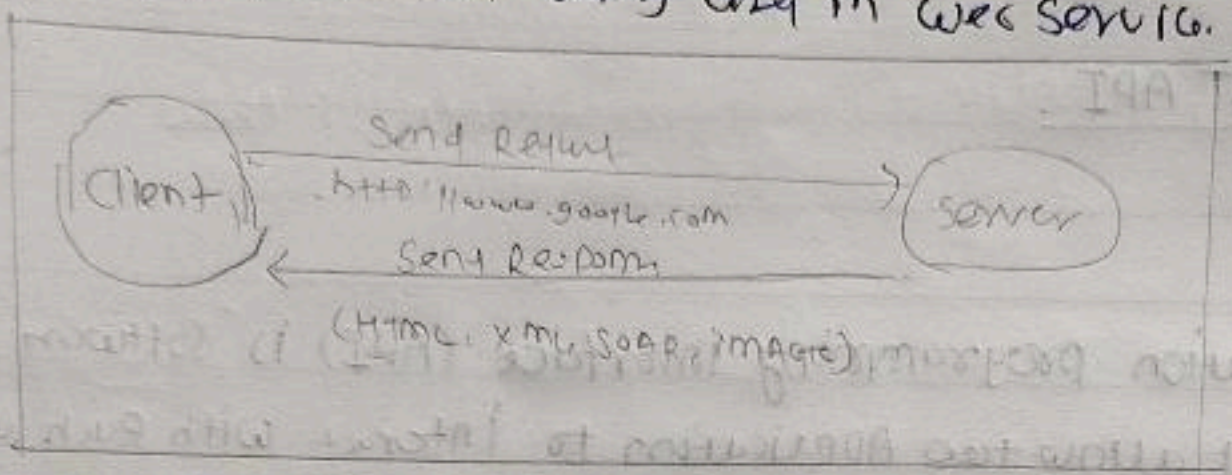
API means software code that can be accessed or executed. API define as a code that can be that helps two different Softwares to communicate and exchange data with each other.



* REST API

Representational state transfer REST in an architectural style that define a set of constraints to be used for creating web services. REST API is way of accessing web service in simple and flexible way without having any processing.

* **Working** :- A request is sent from client to server in the form of web URL as HTTP GET or POST or PUT or DELETE request. After that, a response come back from the server in the form of a resource which can be anything like HTML, XML, Image or JSON. But Now JSON is most popular format being used in web service.



In HTTP there we five method that are commonly used in REST based Architecture i.e POST, GET, PUT, PATCH & DELETE. These correspond to create, read, update & delete operations respectively.

- **GET**: The HTTP GET Method is used to read a representation of a resource.
- **POST**: The POST Verb is most often utilized to create new resource.
- **PUT** : PUT is used to create resource in the case where the resource ID is chosen by the client instended of by the server.
- **PATCH** :- it is used to modify capabilities. The PATCH request only needs to contain the changes to the resource.
- **DELETE** :- it is used to delete a resource identified by a URL.

28/08/2022