**SharePoint** Development Training

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**-: Day 1:-**

**Introduction to SharePoint:**

SharePoint is a platform to support collaboration and Content management system.

It is a central web-based portal where we can manage document, social activities, data and information.

**SharePoint can do:**

1.) Collaboration -> Collaboration is the strong theme of SharePoint which means bringing people together through different types of collaboration such as enterprise content management.

2.) Interoperability -> Capability to build and deploy secure and custom solutions that integrate line of business data with SharePoint and Office.

3.) Platform->SharePoint is also a platform that supports extensibility, through a rich object model, a solid set of developer tools, and a growing developer community.

**History of SharePoint:**

SharePoint first launched in 2001 By Microsoft after that some more versions launched like:

* SharePoint 2001: In 2001, Microsoft launched SharePoint Portal Server (SPS) and SharePoint Team Services (STS). SPS provided three functions i.e., Portal, Document Management & Search. STS on the other hand with FrontPage and provided team workspaces for collaboration.
* SharePoint 2003;
* SharePoint 2007;
* SharePoint 2010;
* SharePoint 2013;
* SharePoint 2016: SharePoint 2016 enhance the hybrid infrastructure (merging SharePoint On-Premises sites with SharePoint Online) and to take advantage of recent innovations in cloud technology.

**Features of SharePoint:**

* Enterprise Search Engine,
* Enterprise Content Management, Business Data Catalog,
* Excel Services, Faster Site Creation,
* List, Libraries,
* We can develop site,
* Customise site,
* Create Rest API,
* By Using Rest API, we can insert data in site.

**Prerequisite:**

* HTML
* CSS
* JavaScript
* Bootstrap
* C#
* jQuery
* React js
* CSOM
* JSOM
* SQL

**Key Feature:**

* Business connectivity Services,
* Record Management,
* Advance Searching,
* Web content Management,
* Enterprise management,
* Record Management,
* Site,
* Search,
* Branding,
* Compatibility,
* Interoperability,

Two SharePoint Premises:

SharePoint online by subscription;

SharePoint on premises by instilling on our PC;

---------------------------------------------------------------------------------------------

**-: Day 2 :-**

**Introduction of SharePoint Development platform**

SharePoint is a platform to support collaboration and content management system. It is a central web-based portal. Using SharePoint, we can manage our colleague’s and our own documents, social activities, data, and information.

* It allows groups to set up a centralized, password-protected space for document sharing.
* Documents can be stored, downloaded and edited, then uploaded for continued sharing.
* SharePoint offers such a wide array of features that it is very challenging for any one person to be an expert across all the workloads.

**Learnt with sort demo of SharePoint:**

* Default template on SharePoint and way to customise according to the user requirement.
* How to add List and libraries into the site.
* Shown how to insert different component of SharePoint into the site.

**SharePoint Master Page:**

Master Page: SharePoint master pages provide the interface and overall layout of the pages on a SharePoint site. The common elements of a page – its header, navigation links, Site Actions menu, and so forth they are placed in the same areas regardless of the page you’re viewing.

---------------------------------------------------------------------------------------------

**-: Day 3 :-**

**Difference Between List and Libraries:**

**List:** A SharePoint list is like a table in a database or a table in excel. The SharePoint list will have rows and columns that we will use to store information. For example, we may like to store Training Course details like: Training Course Name, Duration, Price etc;

* It stores homogeneous data;
* Version control is not possible.

**Libraries:** SharePoint libraries are special types of lists that are created to store documents. Each file in a SharePoint document library is like one item. It also has columns or fields or properties.

SharePoint also provides various libraries for specific proposes like picture library, form library, etc.

* Different data can be stored like video, music, website etc,
* Version control is possible.

**SharePoint Site Hierarchy**

**Three-layer architecture of SharePoint:**

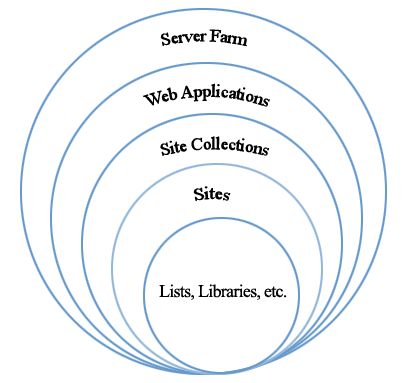


Figure 1: Diagram Shows the architectural hierarchy

**Server Farm:**  This is the top level of SharePoint; this level contains the Web Applications.

**Web Application:** This SharePoint instance contains the site collections.

**Site Collection:**  Site collections are at the third level within SharePoint hierarchy. The site collection has no parent site and as such is the first level of the hierarchy we can access. The site collection can create multiple sections underneath it which includes other sites/sub-sites and libraries.

**Site (Sub Site Collection):** The next level in the hierarchy are Sites, these are contained within the site collection, however, a site can have either a single parent or multiple parents within the site collection.

**List:** List and Libraries are the bottom level of the hierarchy. A SharePoint list is a collection of data that has some kind of structure to it

**Libraries:** SharePoint libraries are special types of lists that are created to store documents.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**-: Day 4 :-**

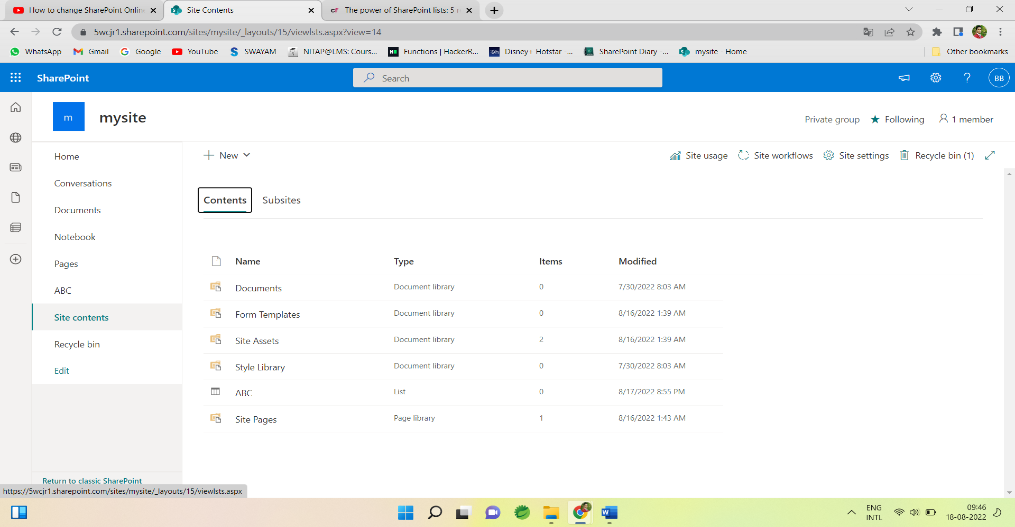
**SharePoint List**

A SharePoint list is a collection of data that has some kind of structure to it. It is like a table, a spreadsheet or a simple database. It can include many different types of information including numbers, text and even images.

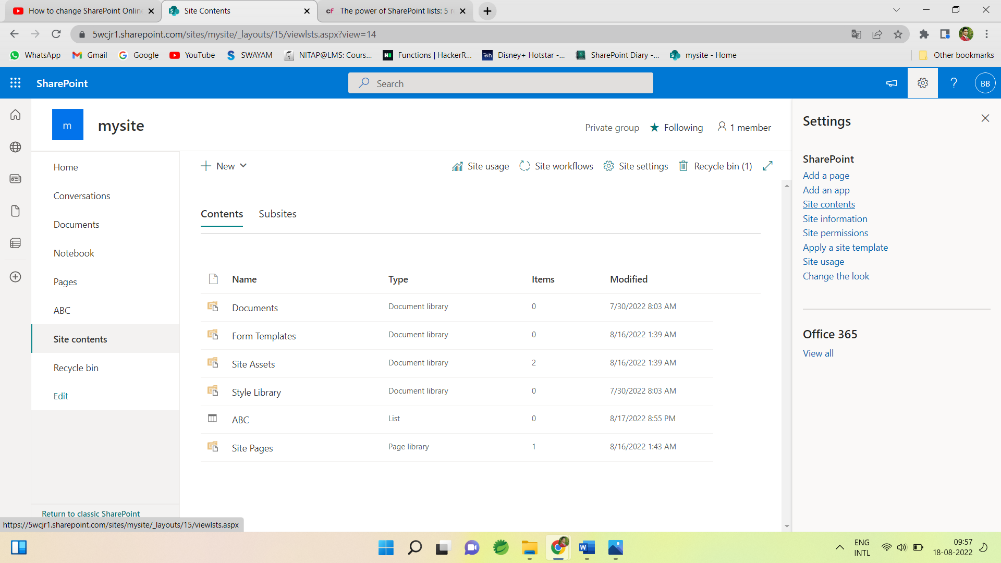
We can create and list that help to track issues, assets, routines, contacts, and more. Start from a template, Excel file, or from scratch.

**Create a list**

From SharePoint site home page

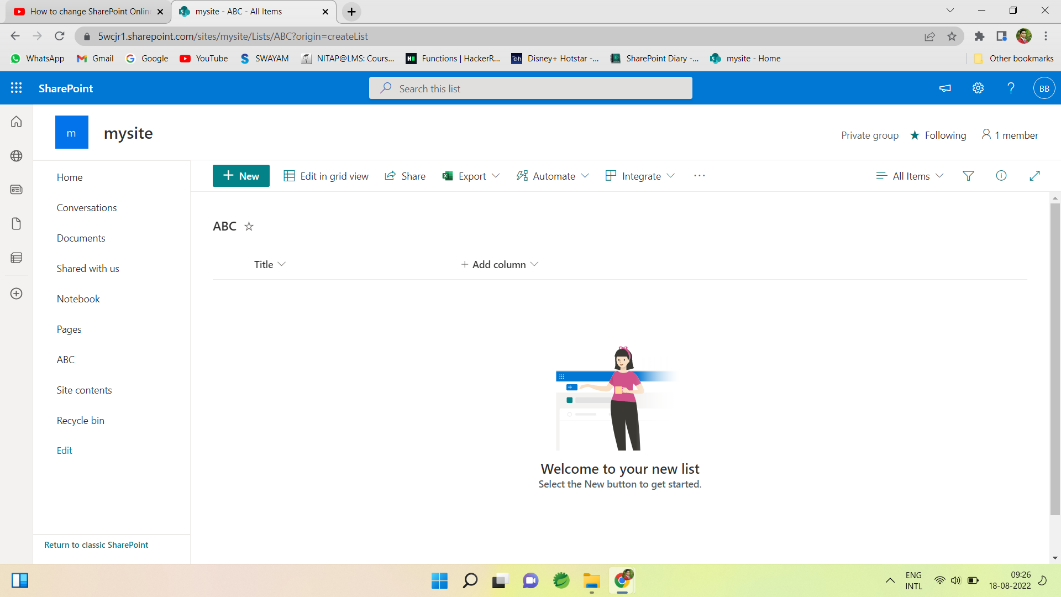


Or the Site contents page,

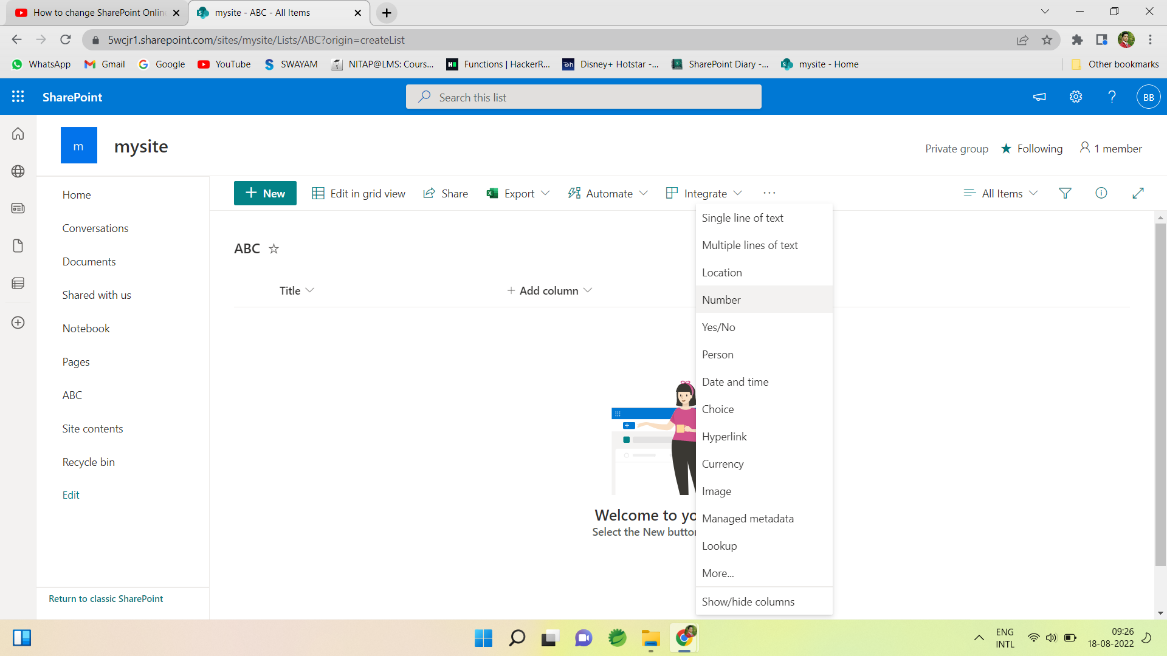


Then select -> + New -> List.

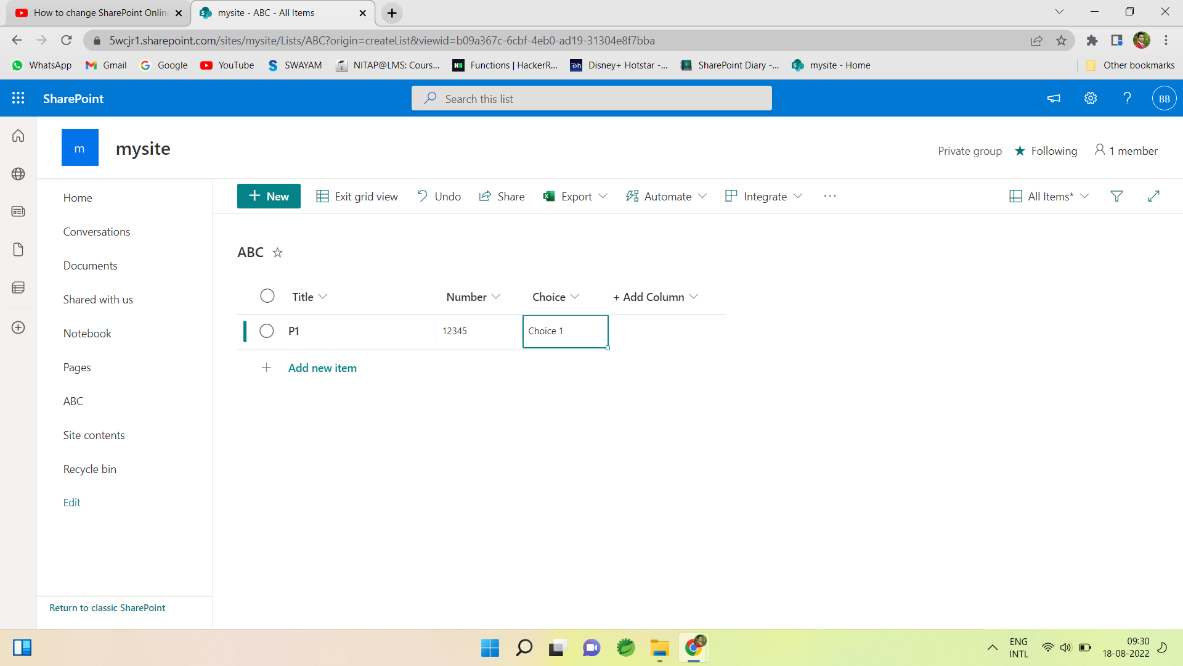
Now choose a Blank list and specify the name and description and List gets created



Now we can add column with specific type such as Number, Person, True/False, lookup, Image and many more by clicking on Add column option.



After Choosing the type of column we need to give name and description for that and column added to the list.



Similarly, we can add more column as requirement by its data type.

All created list is stored in site content which contain whole site content like list, libraries, documents etc.

After creating a list, we can edit list directly by clicking on Edit in grid view.

We can export as excel sheet by Export option.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Active Directory: Directory Which containing users, employee. Active Directory sync the users and after fetching we can extract all the necessary details.

Lookup: It is used to get the information from another table just like foreign key in SQL.

Calculated: SharePoint uses the calculated column to populate values based on some formula. The calculation can depend on other column values also, that can use other columns to calculate the values also.

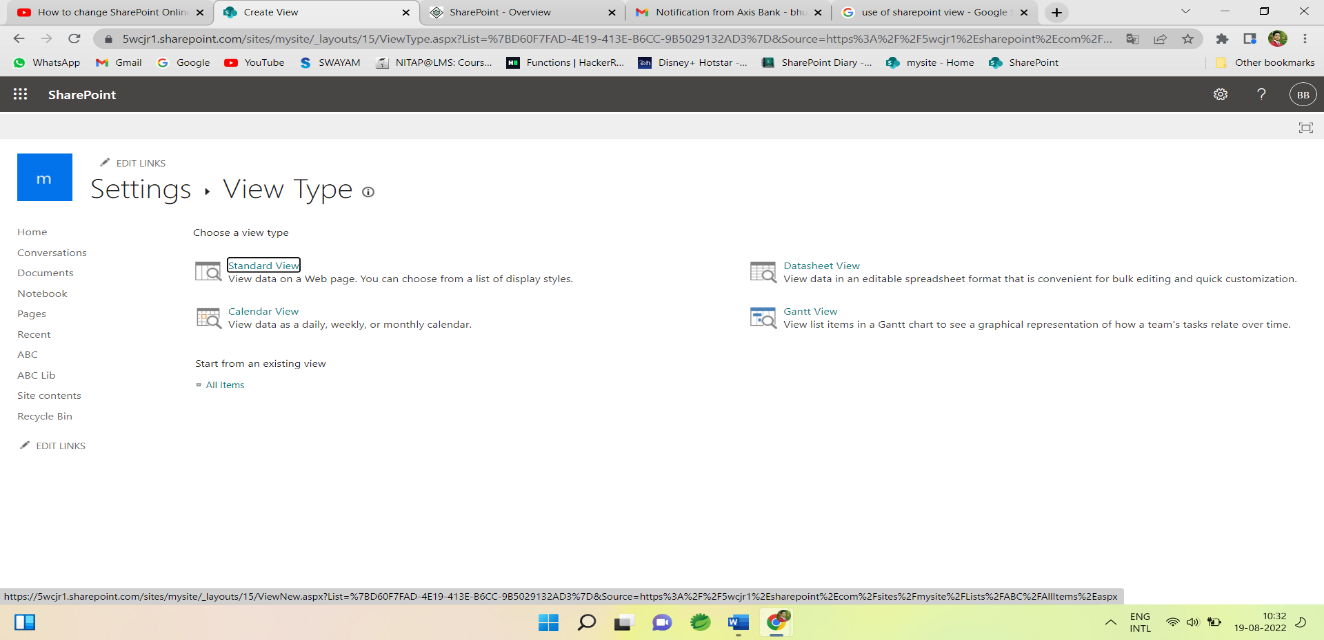
**-: Day 5 :-**

**SharePoint View**

We can create custom views of lists and libraries to organize and show items that are important (like certain columns), to add filtering or sorting, or to have a more engaging style. You can create a personal view (that only you can see) or, if you have permissions to do so, you can create a public view for everyone who uses the list to see.

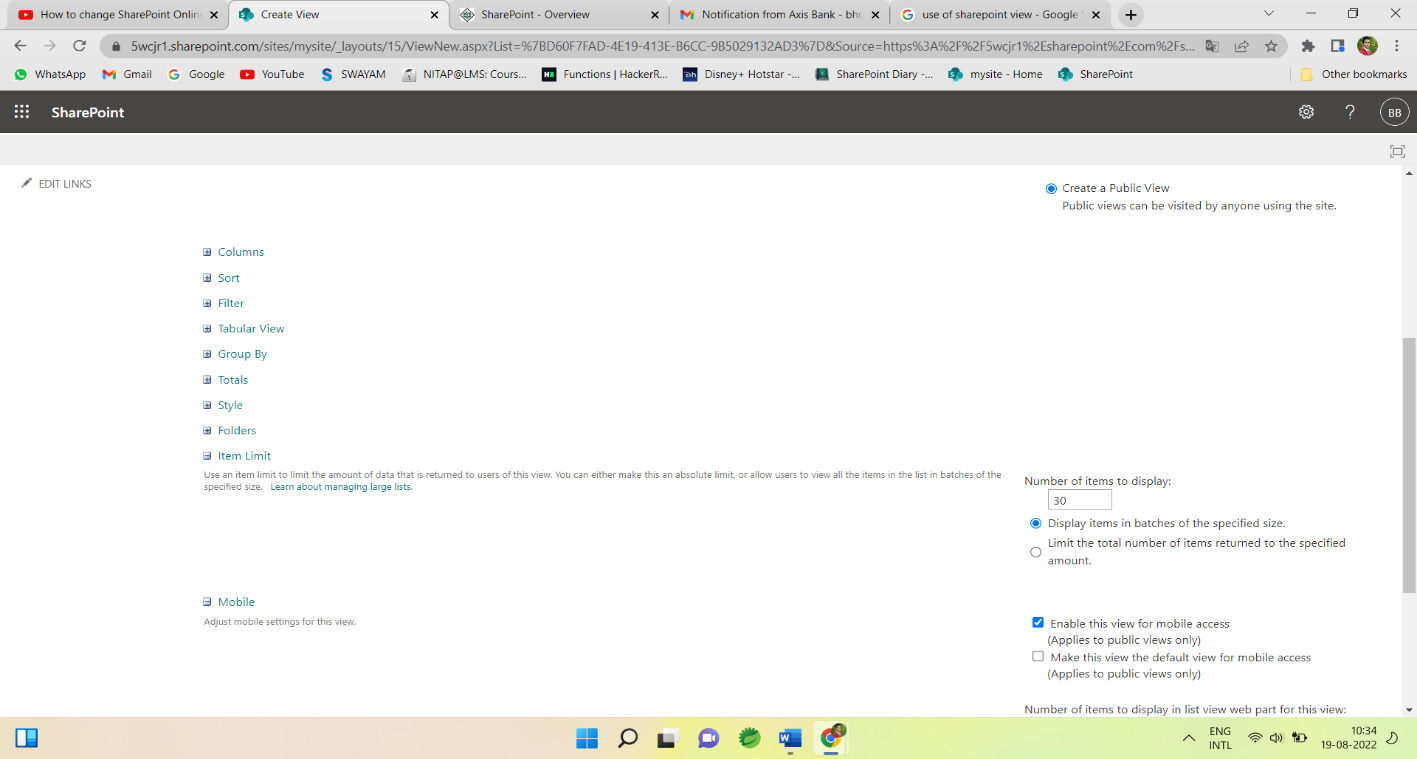
**Types of SharePoint views:**

* Standard view
* Datasheet view
* Calendar view
* Gantt view



**By using this view we can use some functionality like:**

* Columns
* Filter
* Sort
* Tabular view
* Group by
* Tables
* Style
* Folder
* Item Limit etc.



**Standard View:**  This view displays our list and library items one row following another. Standard view is the default for most types of lists and libraries. We can customize the view in many different ways, such as by adding or removing columns from the view.

**Calendar View:** This view displays our list and library in a format similar to a wall calendar. We can apply daily, weekly, or monthly views in this format. This view can be helpful if we want to see the items in the list or library chronologically. To use this view, list or library must contain columns with start dates and end dates for the calendar items.

**Datasheet View**: This view displays list and library items in a grid, similar to a spreadsheet. This view, also known as Quick Edit, can be helpful if we have to edit many items in a list or library at the same time. This view is also helpful if we want to export our data to a spreadsheet or database program. There are some limitations to Datasheet View - not all Excel functionality is available.

**Gantt View**: This view displays list and library items in bars that track progress, to see which tasks overlap each other and to visualize overall progress. To use this view, list or library must contain columns with start dates and end dates.

**SharePoint Library**

**Libraries:** SharePoint libraries are special types of lists that are created to store documents. Each file in a SharePoint document library is like one item. It also has columns or fields or properties.

SharePoint also provides various libraries for specific proposes like picture library, form library, etc.

* Different data can be stored like video, music, website, word file, pdf file etc,
* Version control is possible and important feature of SharePoint Library.

**Creation of SharePoint Library:**

Document library creation is same as List

Choose Site contents ->

Then select  + New -> Document Library.

Then Give name and Document library gets created.

In Document Library we can Store document like word file, excel file, music, video, website, link etc.

**-:Third Week :-**

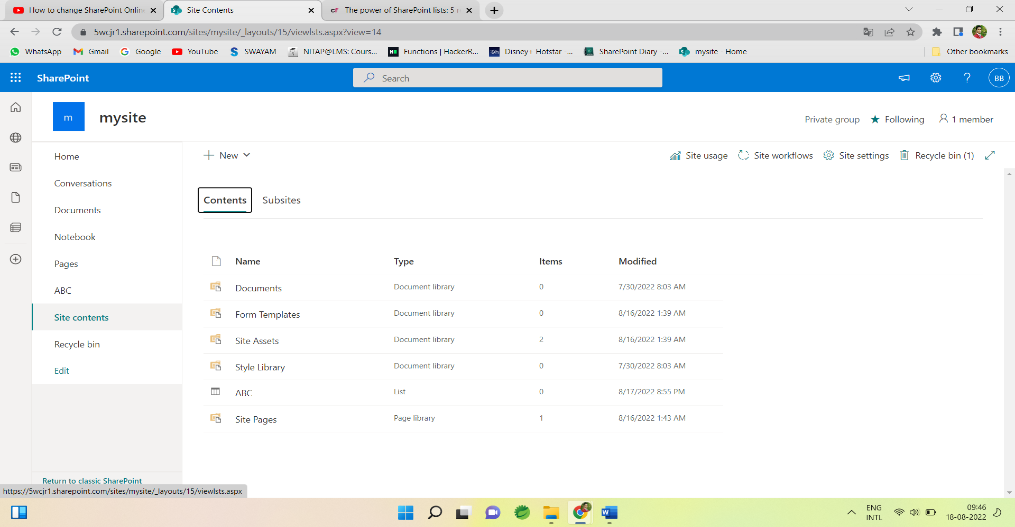
**-:Day 1st :-**

**SharePoint Predefined List Template:**

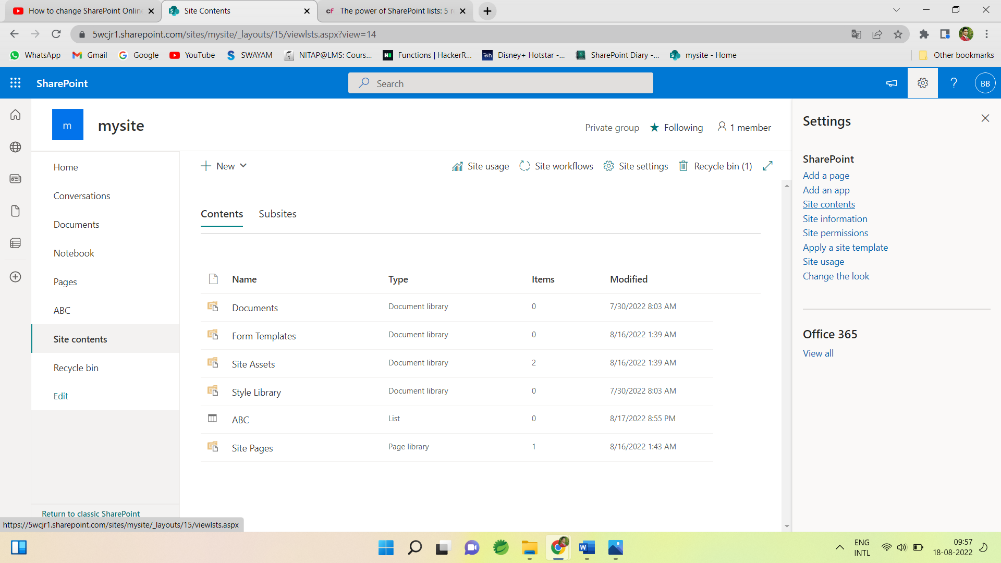
The SharePoint List Template is a great resource for developers who are looking to programmatically create new lists. Each template may represent a unique predefined set of settings, content types, columns, and views. Manually configuring a large quantity of these list settings may no longer needed when lists are created by using the appropriate SharePoint List Template.

**Create a SharePoint list using predefined templates:**

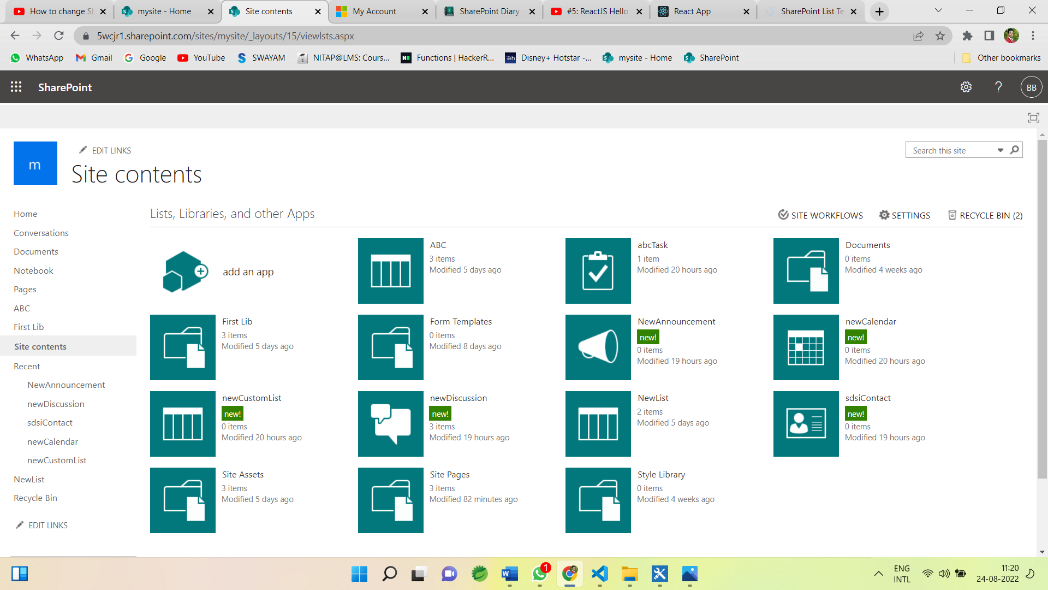
From SharePoint site home page



Or the Site contents page from gear icon,

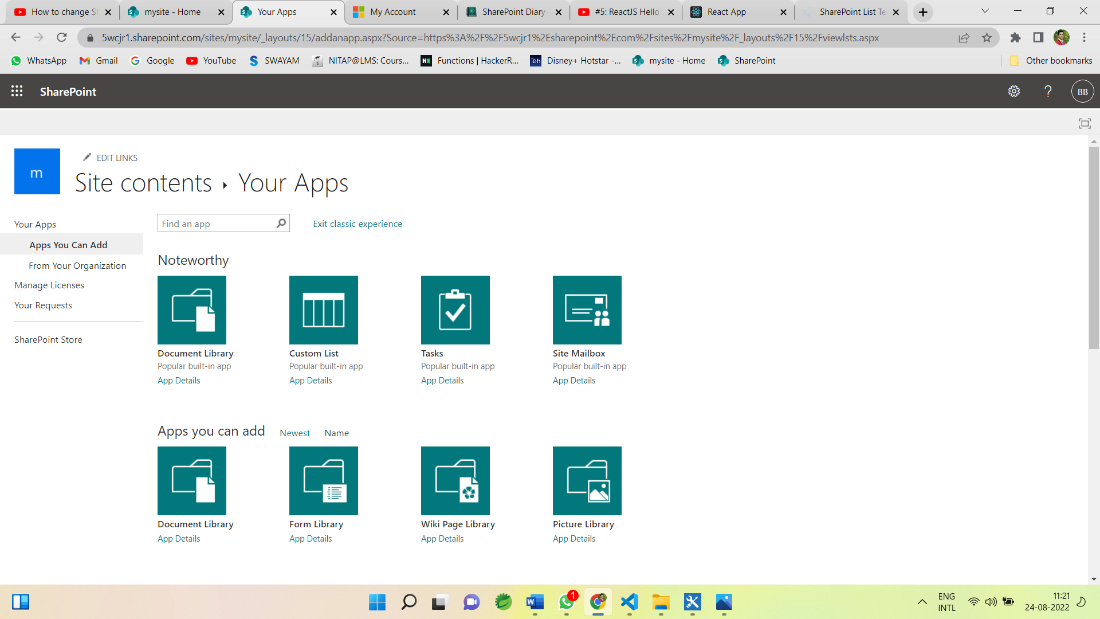


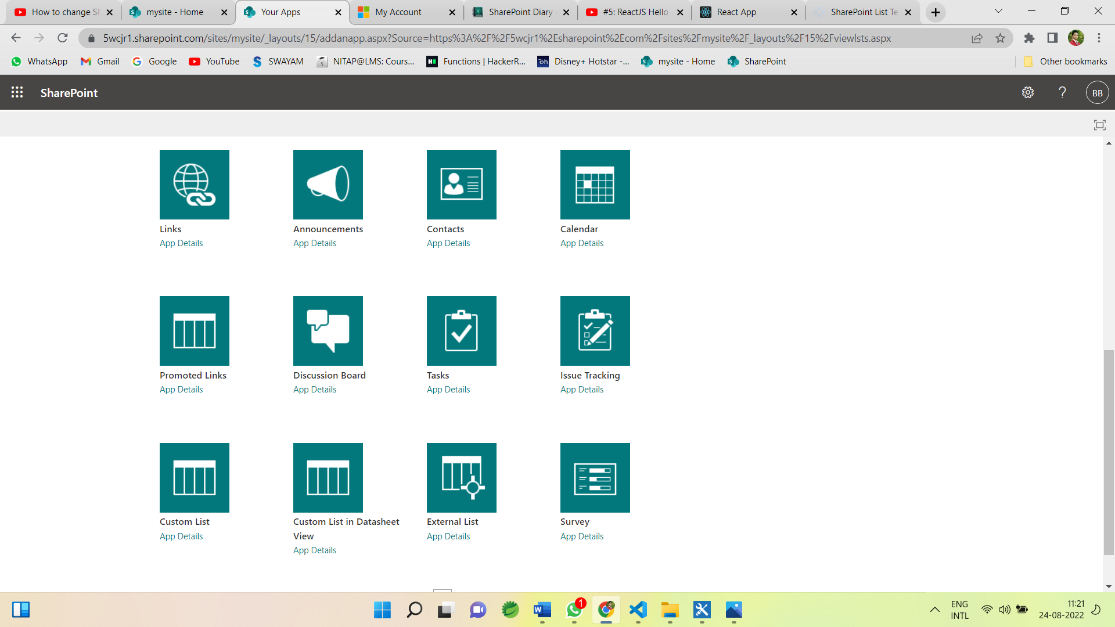
Then classic experience interface looks like



Then select ->  add an app

Now all the predefined list templates are visible



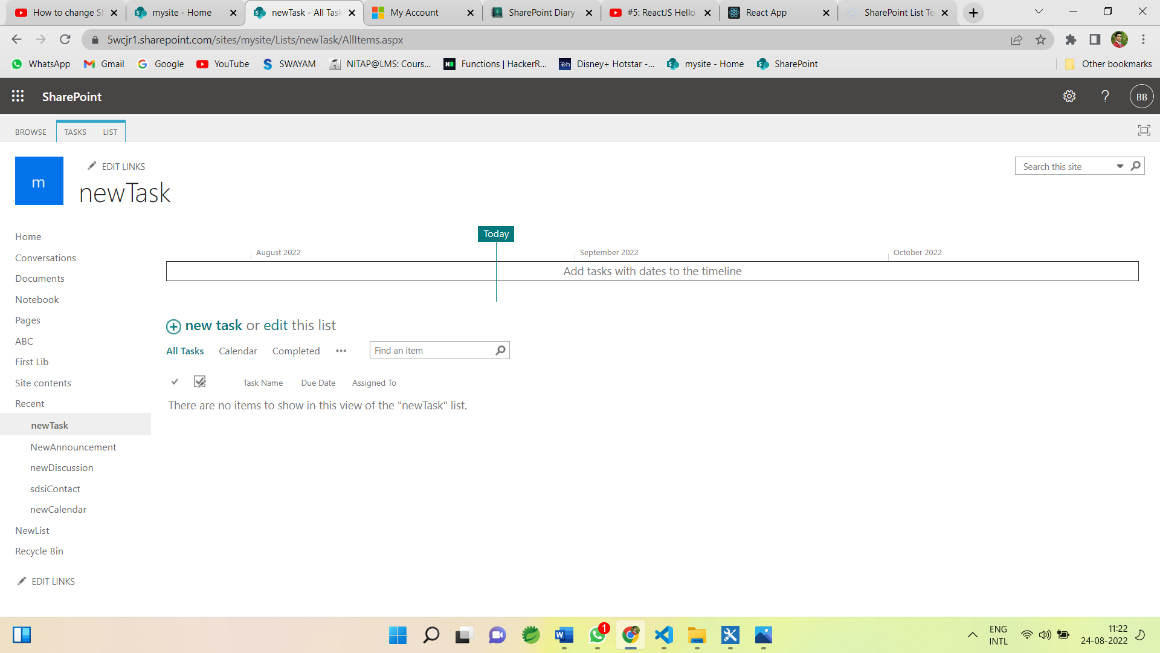


Now choose template according to the requirement.

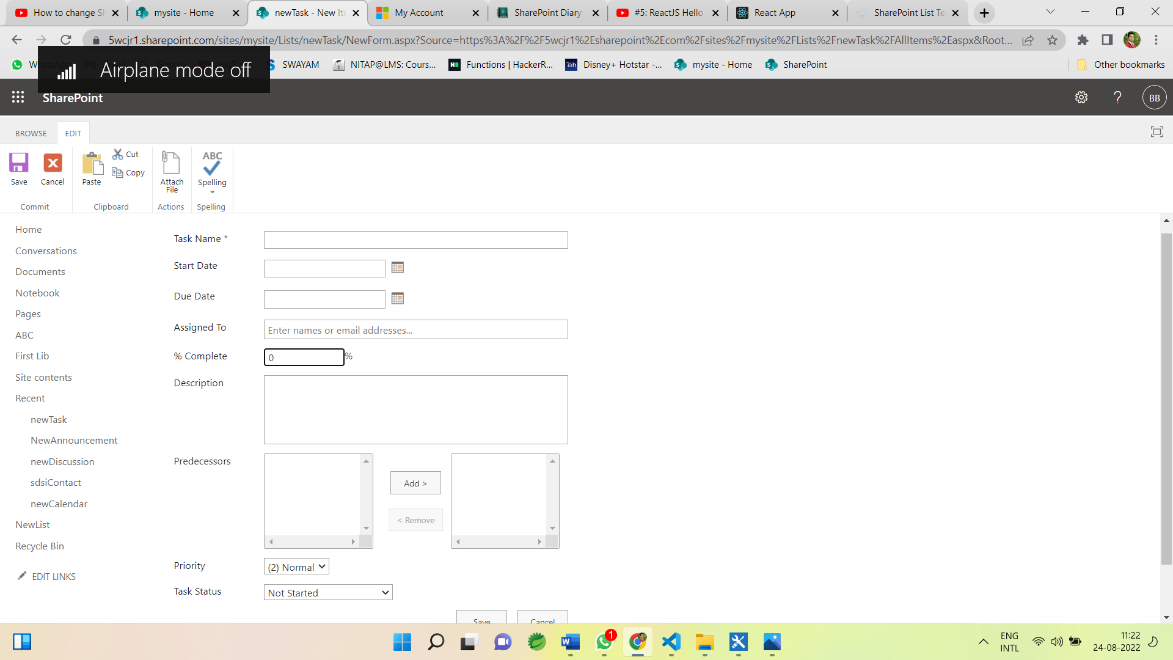
For example for Creating a task

Choose a task templete and give name

Then new task created with default field and views



Then click on + new task and its default fields open.



Now give the name and other fields.

Similarly, we can add other templates like contact, calendar, Discussion Board, Issue tracking, survey etc.

**-: Day 2nd :-**

**SharePoint Designer 2013:**

SharePoint Designer 2013 is the tool of choice for the rapid development of SharePoint applications. It is the tool of choice for the rapid development of SharePoint applications. Using SharePoint Designer, advanced users and developers alike can rapidly create SharePoint solutions in response to business needs.

Using this tool we can edit, customise the site we can write or edit code.

**SharePoint Workflow:**

* List Workflow
* Reusable Workflow
* Site Workflow

**Workflow Triger option:**

Trigger condition takes the format of an expression and must evaluate to either true or False. If the trigger condition is true then the flow will run, else it will ignore the trigger event.

Triger options are:

* Manually
* Automatic when condition satisfy.
* When an item is created and
* When item is changed.

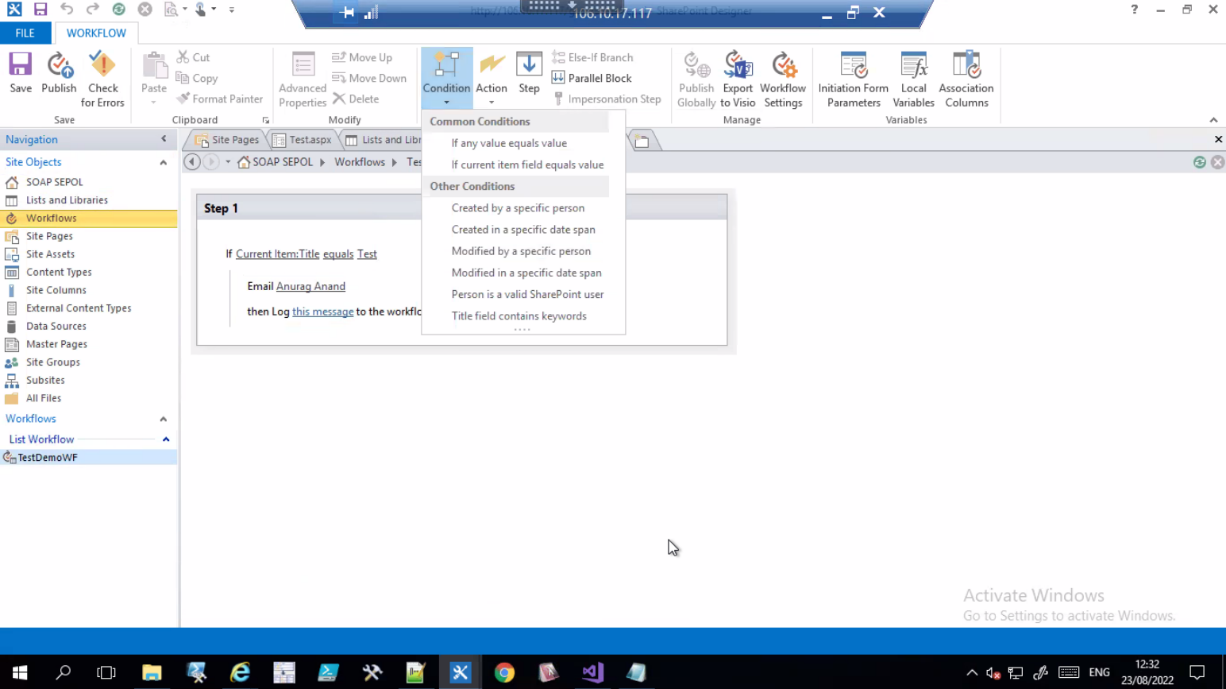
For example, if we use a trigger called ‘When an item is created or modified”, the flow will trigger with each update.

**Conditions of workflow:**

Workflow will be executed if the given condition satisfies then it executes the action inside the condition.

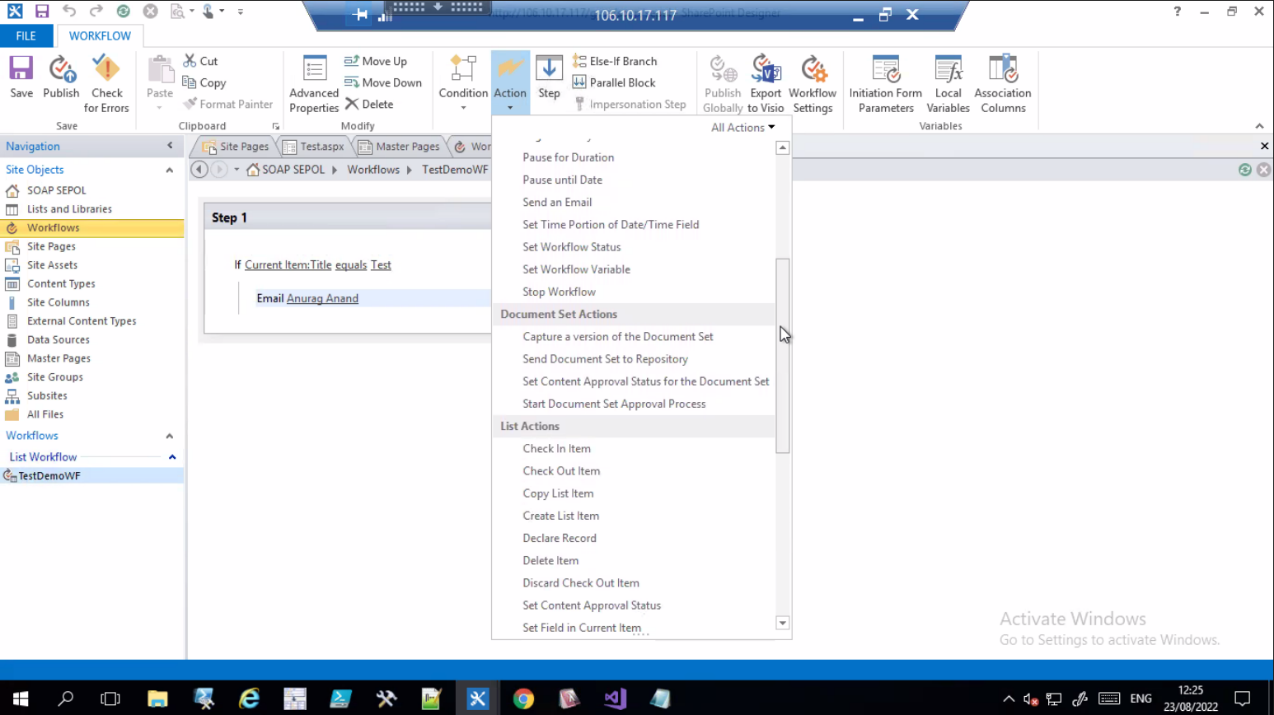
* Any value equals value
* Current item field equals value
* Created by specific person
* Created by specific date span etc.

We can apply workflow condition by choosing the suitable condition from condition tab:



Then Specify the task/action to be execute if the given condition satisfies from the action tab like:

* Pause for duration
* Pause until date
* Send an Email
* Set Workflow status
* Set Workflow variable
* Stop Workflow
* Check in Item
* Check Out Item etc.



**-: Day 3rd :-**

Today we had seen the Create operation of CRUD operation on SharePoint by coding.

**Create a HTML Form:**

Here we have a SharePoint list as “ComapnyInfoList” which has below columns:

* Username
* Password
* EmailID and
* gender

**HTML Code for form creation**

-----------------------------------------------------------------------------------------

<!DOCTYPE html>

<html lang="en" xmlns="http://www.w3.org/1999/xhtml">

<head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>

<script src="https://onlysharepoint2013.sharepoint.com/sites/Raju/SiteAssets/Preetihtml/UserInfoJs.js"></script>

<meta charset="utf-8" />

<title>My Html</title>

</head>

<body>

<h2>Information Form</h2>

Username:<br>

<input type="text" name="Username" id="usrname" />

<br><br>

Password:<br>

<input type="password" name="password" id="usrpwd" />

<br><br>

EmailID:<br>

<input type="email" name="EmailID" id="usrid" />

<br><br>

Gender:<br>

<div id="gend">

<input type="radio" name="gender" value="Male" id="gen0" /> Male

<input type="radio" name="gender" value="Female" id="gen1" /> Female

</div>

<br><br>

<input type="submit" value="Submit" id="btnSubmit" />

<br>

<p id="pTitle"></p>

</body>

</html>

**JSOM Code for inserting data**

---------------------------------------------------------------------------------------------------

$(document).ready(function () {

$("#btnSubmit").click(function () {

insertitemtolist();

});

});

function insertitemtolist() {

var username = $("#usrname").val();

var password = $("#usrpwd").val();

var emailId = $("#usrid").val();

var gender = $("input[name='gender']:checked").val();

var clientContext = new SP.ClientContext.get\_current();

var oList = clientContext.get\_web().get\_lists().getByTitle('CompanyInfoList');

var item = new SP.ListItemCreationInformation();

var oListItem = oList.addItem(item);

oListItem.set\_item('Title', username);

oListItem.set\_item('Password', password);

oListItem.set\_item('EmailID', emailId);

oListItem.set\_item('Gender', gender);

oListItem.update();

clientContext.load(oListItem);

clientContext.executeQueryAsync(Function.createDelegate(this, this.onQuerySucceeded), Function.createDelegate(this, this.onQueryFailed));

}

function onQuerySucceeded(sender, args) {

$("#pTitle").html("successfully executed");

}

function onQueryFailed(sender, args) {

alert('request failed ' + args.get\_message() + '\n' + args.get\_stackTrace());

}

----------------------------------------------------------------------------------------------------

**-: Day 4th :-**

Today we had seen the Get/Read, Update and Delete operation of CRUD operation on SharePoint by coding.

**JSOM Code for updating data in CompanyInfoList**

---------------------------------------------------------------------------------------------------

$(document).ready(function () {

ExecuteOrDelayUntilScriptLoaded(retrieveListItems, "sp.js");

$("#btnUpdate").click(function () {

updateListItemByID();

});

});

var masterListItem;

function retrieveListItems() {

var id = GetParameterValues('MyID');

getitemsbyID(id);

}

function getitemsbyID(itemID) {

var clientContext = new SP.ClientContext.get\_current();

var masterlist = clientContext.get\_web().get\_lists().getByTitle('CmpanyInfoList');

masterListItem = masterlist.getItemById(itemID);

clientContext.load(masterListItem);

clientContext.executeQueryAsync(Function.createDelegate(this, this.onQuerySucceeded),

Function.createDelegate(this, this.onQueryFailed));

}

function onQuerySucceeded() {

$("#usrname").val(masterListItem.get\_item('Title'));

$("#usrpwd").val(masterListItem.get\_item('Password'));

$("#usrid").val(masterListItem.get\_item('EmailID'));

if (masterListItem.get\_item('Gender') == "Male") {

$("#gen0").attr('checked', 'checked');

}

else if (masterListItem.get\_item('Gender') == "Female") {

$("#gen1").attr('checked', 'checked');

}

}

function onQueryFailed(sender, args) {

alert('Request failed. \nError: ' + args.get\_message() + '\nStackTrace: ' + args.get\_stackTrace());

}

function GetParameterValues(param) {

var url = window.location.href.slice(window.location.href.indexOf('?') + 1).split('&');

for (var i = 0; i < url.length; i++) {

var urlparam = url[i].split('=');

if (urlparam[0] == param) {

return urlparam[1];

}

}

}

function updateListItemByID() {

var id = GetParameterValues('MyID');

var username = $("#usrname").val();

var password = $("#usrpwd").val();

var emailId = $("#usrid").val();

var gender = $("input[name='gender']:checked").val();

var clientContext = new SP.ClientContext.get\_current();

var oList = clientContext.get\_web().get\_lists().getByTitle('CmpanyInfoList');

var oListItem = oList.getItemById(id);

oListItem.set\_item('Title', username);

oListItem.set\_item('Password', password);

oListItem.set\_item('EmailID', emailId);

oListItem.set\_item('Gender', gender);

oListItem.update();

clientContext.executeQueryAsync(Function.createDelegate(this, this.success), Function.createDelegate(this, this.failue));

}

function success() {

alert('Item updated!');

}

function failue(sender, args) {

alert('Request failed. ' + args.get\_message() + '\n' + args.get\_stackTrace());

}

**Create HTML File for displaying the data:**

**-----------------------------------------------------------------------------------------**

<!DOCTYPE html>

<html lang="en" xmlns="http://www.w3.org/1999/xhtml">

<head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script> ----> // $ in Jsom

<script src="https://onlysharepoint2013.sharepoint.com/sites/Raju/SiteAssets/Preetihtml/UserInfoJs.js"></script> ---> SPServices in Jsom

<meta charset="utf-8" />

<title>ALL LIST ITEM</title>

</head>

<body>

<p id="disp"></p>

</body>

</html>

**JSOM Code for Reading/Getting the data From list “CompanyInfoList”**

**----------------------------------------------------------------------------------------**

$(document).ready(function () {

getdatafromlist();

});

function getdatafromlist(){

var clientContext = new SP.ClientContext.get\_current();

var oList = clientContext.get\_web().get\_lists().getByTitle('CompanyInfoList');

var oListItems=oList.getItems(SP.CamlQuery.createAllItemsQuery()); ------>to etrieve all list items we use CamlQuery

clientContext.load(oListItem);

clientContext.executeQueryAsync(Function.createDelegate(this, this.onQuerySucceeded),

Function.createDelegate(this, this.onQueryFailed));

}

function onQuerySucceeded(sender, args) {

var newtitle='';

var listItemEnum=oListItems.getEnumerator(); -----> data is in the list enumerator will fetch one by one

while(listItemEnum.moveNext()){ ------>As long as the Enumerator will find next item this is will return true

var oListItem=listItemEnum.get\_current().get\_item('title');

newtitle+=oListItem;

}

$("#disp").append(newtitle);

}

function onQueryFailed(sender, args) {

alert('request failed ' + args.get\_message() + '\n' + args.get\_stackTrace());

}

**JSOM Code for Deleting data From List “CompanyInfoList”**

**----------------------------------------------------------------------------------------**

$(document).ready(function () {

deleteitem();

});

function deleteitem() {

var dltItem=$("#dlttext").val();

var clientContext = new SP.ClientContext.get\_current();

var oList = clientContext.get\_web().get\_lists().getByTitle('CompanyInfoList');

oListItem=oList.getItemById(dltItem); -------->Get the element by given ID

oListItem.deleteObject();

clientContext.executeQueryAsync(Function.createDelegate(this, this.onQuerySucceeded),

Function.createDelegate(this, this.onQueryFailed));

}

function onQuerySucceeded(sender, args) {

$("#dltmsg").html("successfully deleted");

}

function onQueryFailed(sender, args) {

alert('request failed ' + args.get\_message() + '\n' + args.get\_stackTrace());

}

**SharePoint Branding:**

SharePoint Branding is the process of changing the look and feel of the SharePoint user interface using custom master pages, style sheets, images, JavaScript, jQuery and so on.  
  
  
  
A few of the branded sites are shown below.  
  
  
  
**Advantages of Branding**  
  
A good branding leads to:

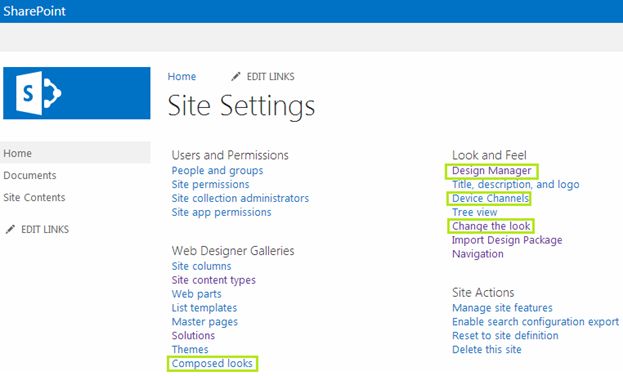
1. Better content presentation
2. Better navigation
3. Better look and feel
4. Enable responsive web design
5. Hide SharePoint default look

**Skills needed for Branding**

* HTML,
* CSS
* Basic Scripting skills and
* SharePoint Branding knowledge is an essential skill to integrate the custom branding artifacts.

**SharePoint Branding knowledge includes:**

* Master Page location
* Master Page elements like Site Settings, Quick Launch and so on.
* Composed looks

We can access these pages from Site Actions > Site Settings menu.  
  
  
  
**Note:**The Publishing feature needs to be activated for getting the Design Manager feature.  
  
**Branding Elements in SharePoint 2013**  
  
The following are the branding elements in SharePoint 2013:

1. Master Pages
2. Preview Files
3. CSS Files
4. Color Palettes
5. Font Palettes
6. Images

**-:Fourth Week :-**

**-:Day 1st :-**

**Security – User Group and Permissions:**

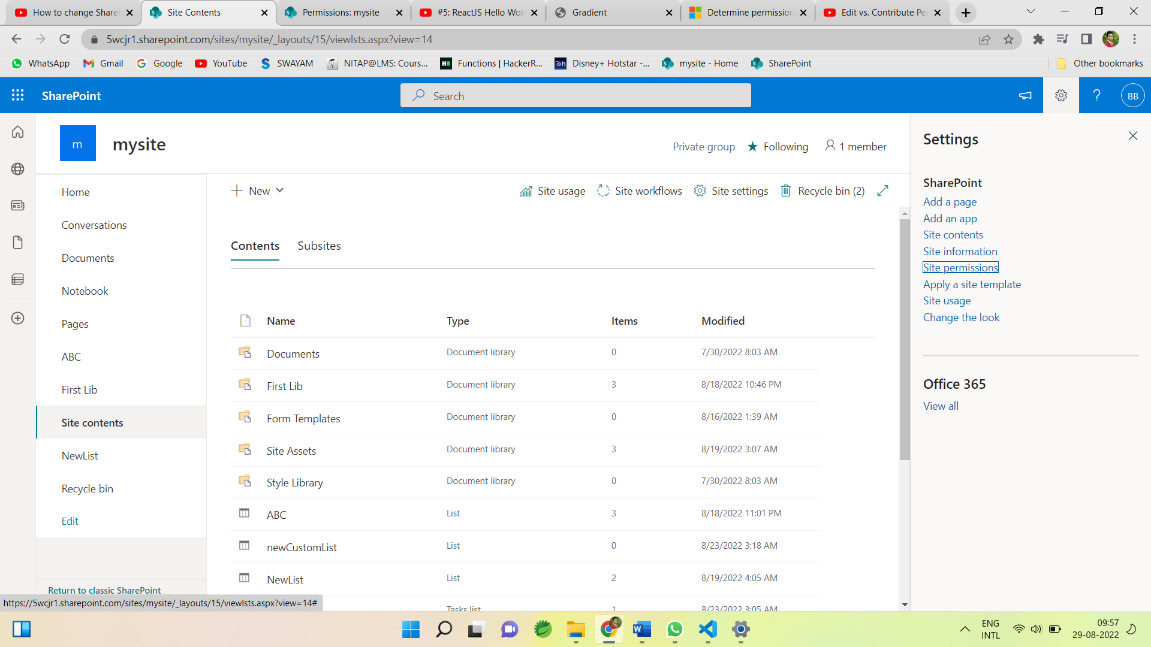
Some sites in an enterprise contain content that should not be available to all users. For example, proprietary technical information should be accessible only on a need-to-know basis. An intranet portal of company should be available to employees only, whereas the home page of an Internet Web site is accessible by anonymous clients.

By permissions control access to sites and site content we can manage permissions by using SharePoint groups, which control membership. Permissions also help to secure content at the item and document level.

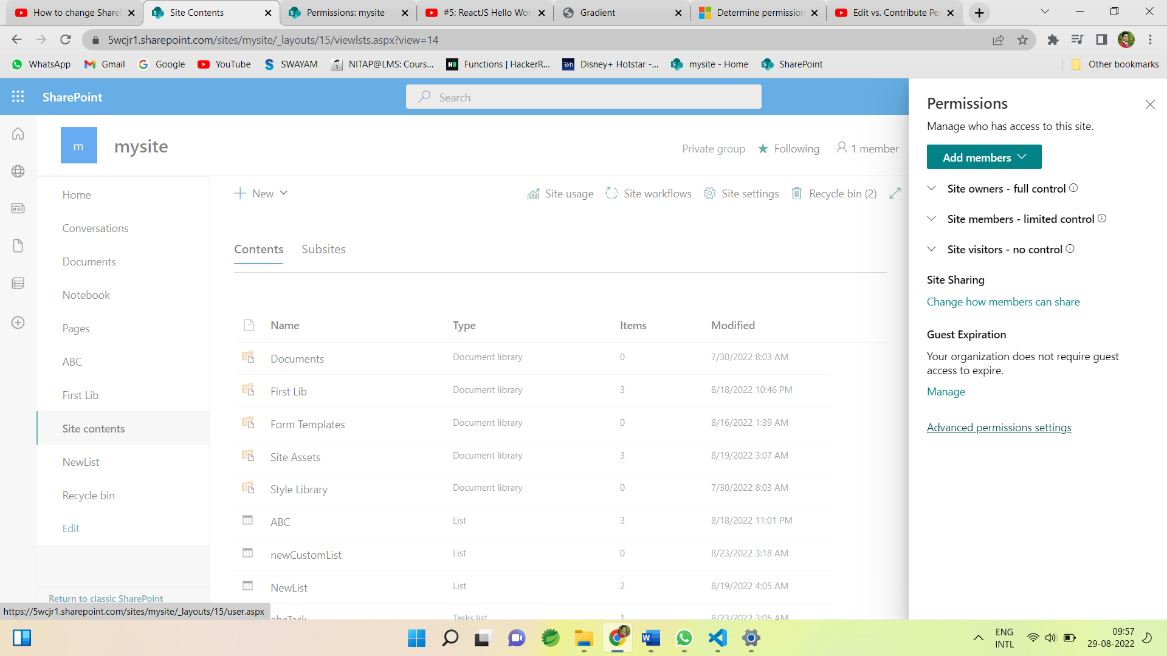
In SharePoint we can give permission by default permission levels of SharePoint or we can create our own Permission and specify the accessibility.

**SharePoint default permissions:**

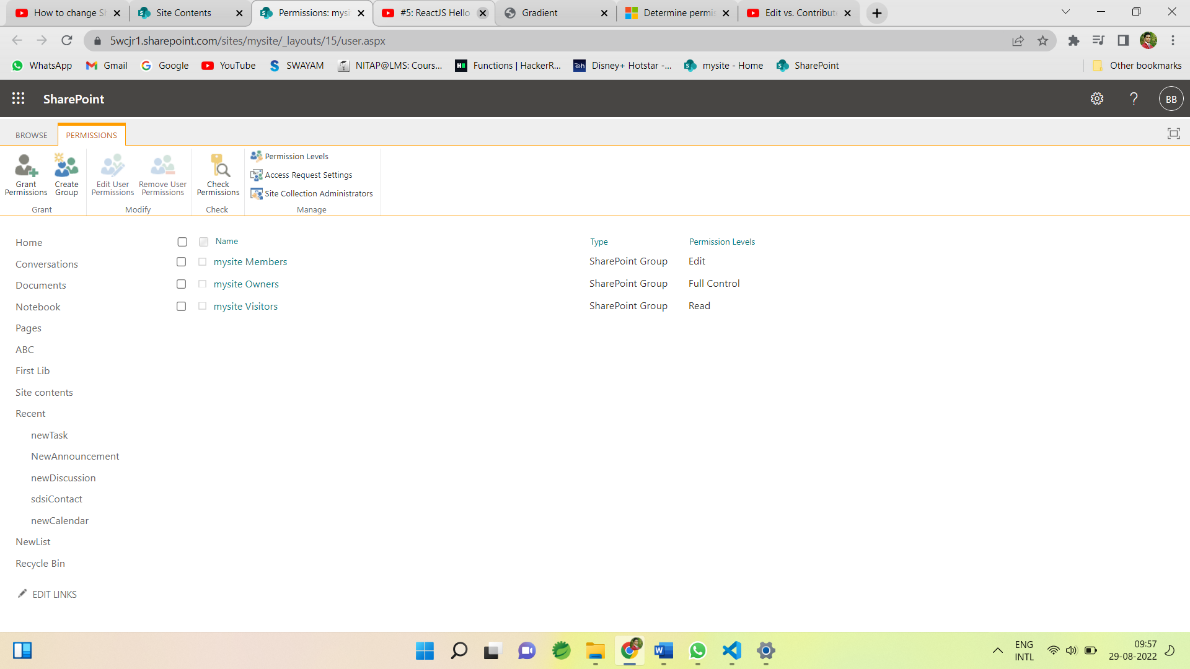
From SharePoint site home page clicking on the gear icon, we get an interface having 🡪 site permission



Clicking on Site permissions we get another tab of Permissions having Advance Permission setting



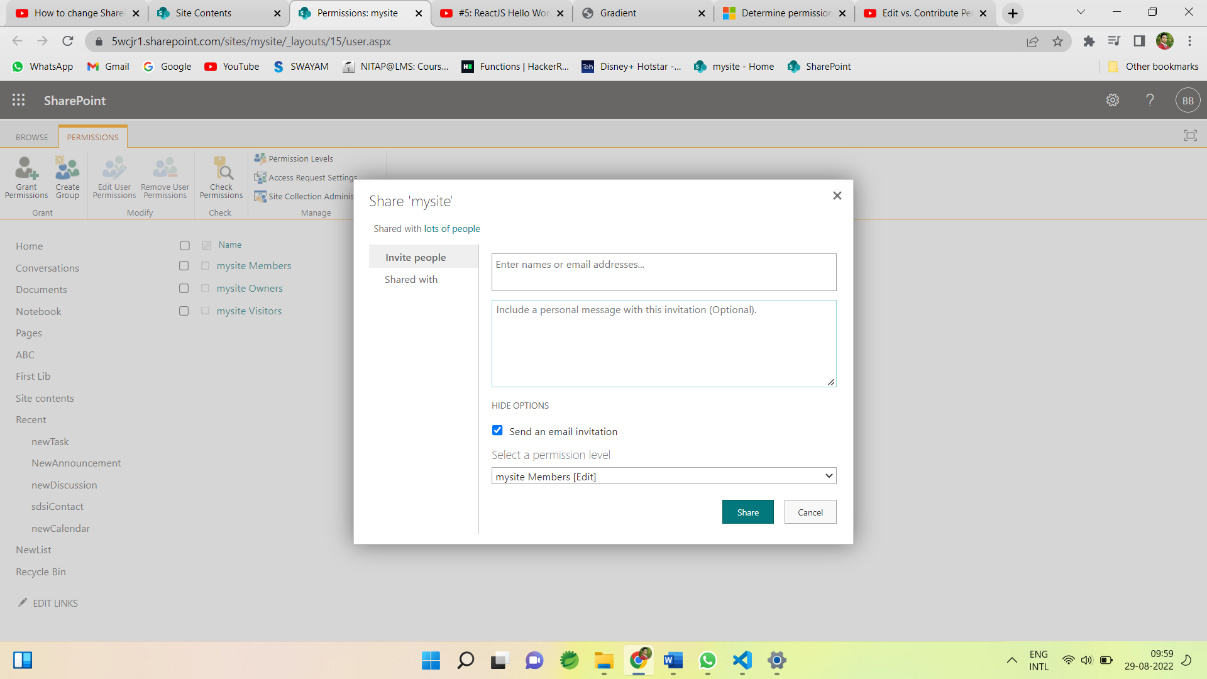
Click on Advance Permission setting we get all the Permissions Settings.



Here We can Create groups, Grant permissions, Edit user permissions, Remove user permissions, Check permissions, we can see permissions levels etc.

**Grant permissions :**

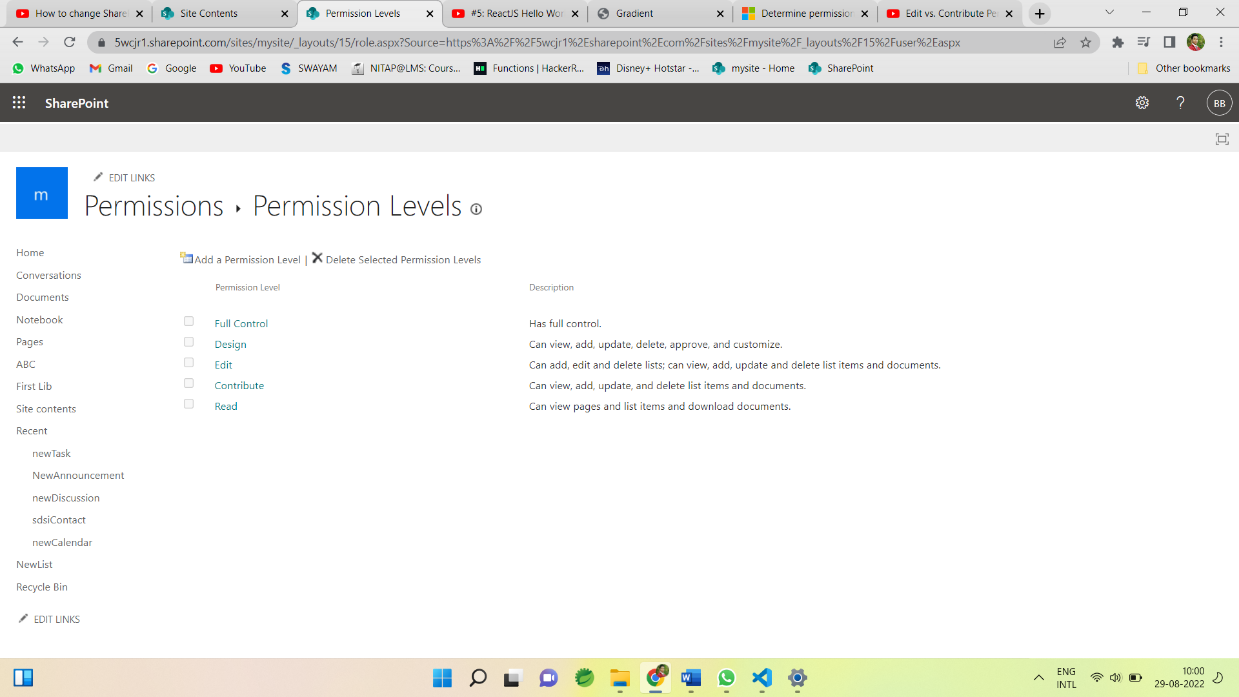
To grant permission click on Grant Permission tab and invite the person or group to give permission



Then this user is visible in the PERMISSIONS tab with specified permission level.

**Permissions levels :**

We can see the permissions levels on SharePoint through permission levels



Different permissions levels are:

* Full control: Grant full control to the users or groups.
* Design: Groups having permissions of design can view, add, update, delete, approve and customise.
* Edit: Groups having permissions of Edit can add, edit and delete lists; can view, add, update and delete list items and documents.
* Contribute: Groups having permissions of Contribute can view, add, update, and delete list items and documents, But can’t add, edit and delete lists;
* Read : Groups having permissions of Read can view pages and list items and download documents.

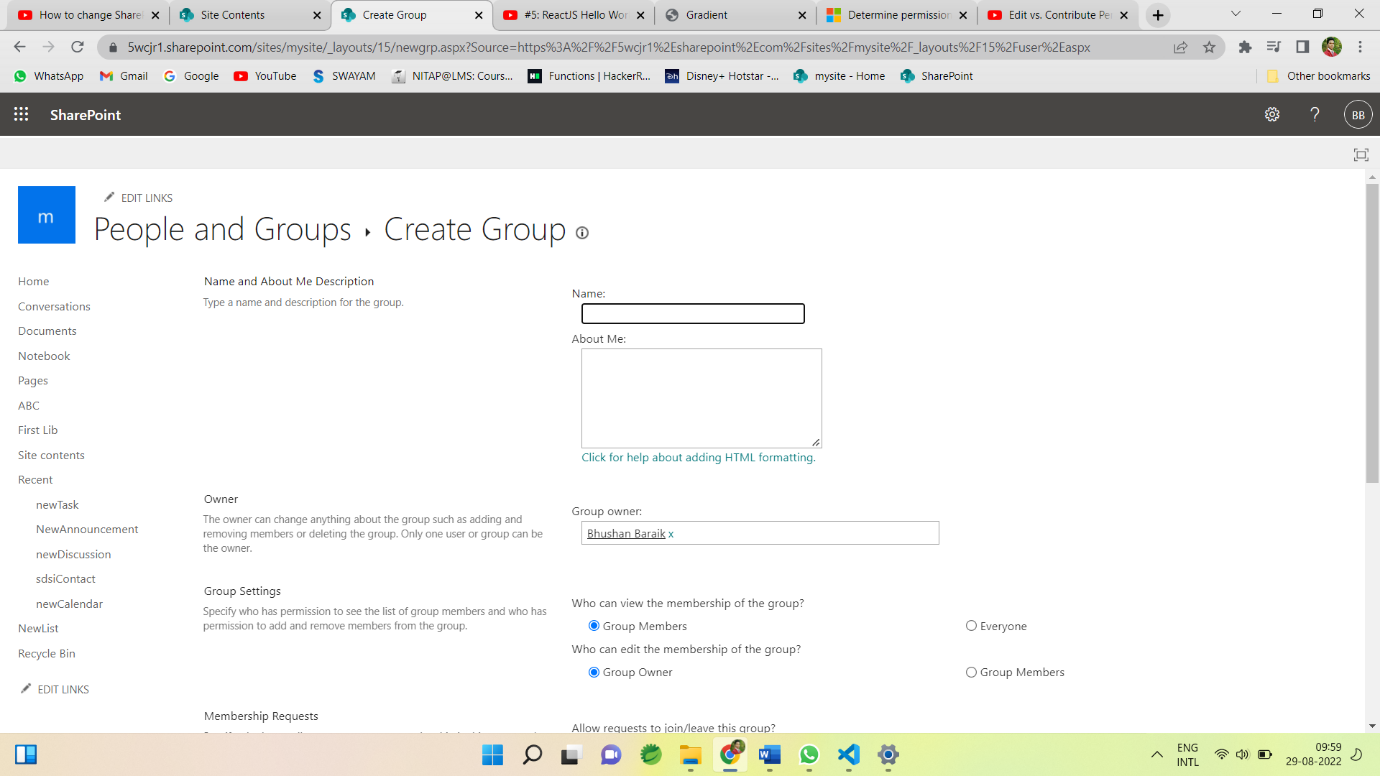
**Default Group with permissions :**

* Visitors 🡪Read
* Member 🡪Edit
* Owner 🡪 Full control
* Viewer 🡪view

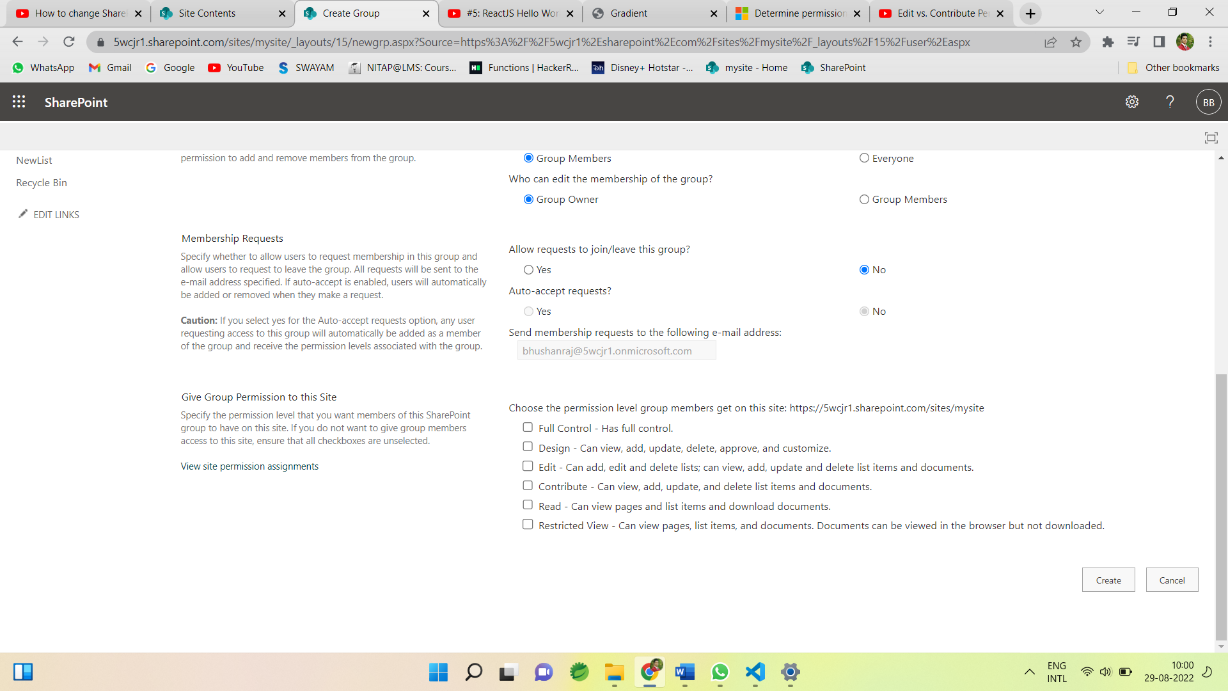
**Create Group:**

We can create group to specify the permission to that group.

Provide the name to the groups 🡪



And specify the permission level for group 🡪



Security – User Group and Permissions is the most important part of SharePoint as SharePoint is a content sharing and management tool so its important to give the correct permissions to the different groups and users for the security of data/content.

**-:Day 2nd :-**

**Claim Based Authentications**

There are a few Authentication methods and we can use any of them.

1. Windows authentication
   1. NTLM
   2. Kerberos
   3. Digest
   4. Basic
2. Forms-based authentication (FBA)
3. Security Assertion Markup Language (SAML) token-based authentication Windows Authentication:

In Windows authentication, SharePoint takes advantage of the default authentication provider (ADDS) to authenticate the logged-in user. In this type of authentication, SharePoint site takes the same credentials, which was used by the user to log in to his machine.

* 1. NTLM
  2. Kerberos
  3. Digest
  4. Basic

**Windows claims Authentication process in SharePoint**

1. Assume, on clients computer meres no any clams-Based security token, user requests SharePoint resource (Site, list, page, etc....).
2. SharePoint asks the user for the credential.
3. The user provides the credentials.
4. SharePoint takes the credentials and goes to the identity provider to verify the credentials
5. The identity provider will verify the credentials and will send back the Windows Security Token
6. SharePoint asks Domain Controller for the list of Security Groups that the user belongs to.
7. Domain Controller sends the list of Security Groups t? SharePoint
8. Using this information (Windows Security Token and Group Membership of the user account), Security Token Service in SharePoint server creates claims-based Security Token.
9. Security Token Service (STS) stores claims in Distributed Cache Service (DCS) in ShareP0int farm.
10. IIS server on the SharePoint server sends authorization code to the client computer.
11. The end-user accesses the page if the user is authorized, through analysis of the claims in the security token and configured permission.
12. After authorization is successful, the SharePoint server sends the content of the page.

**Note:** Authorization code is used for authentication of any subsequent requests.

**Forms-Based Authentication (FBA)**

Form-based authentication is a process of checking the user's claim based identity with the help of ASP.Net membership and eole provider; You can use Forms-based authentication if the user credentials are stored in one of the below authentication providers

1. ADDS
2. SQL Server or equivalent database
3. Lightweight Directory Access Protocol (LDAP) Datastore

**Forms-based claims Authentication process in SharePoint**

1. Assume, on client's computer there's no any claims-based security token, User requests SharePoint resource (Site, list, page, etc....).
2. SharePoint sends Form-Based login page to the user and asks the user to enter credential.
3. The user provides the credentials.
4. SharePoint sends the credentials to Membership Provider.
5. Membership Provider verifies the credentials
6. SharePoint queries Role Provider for the credentials (user)
7. Role Provider sends the list of Roles that are associated with the user.
8. Based on Information from Membership and Role Provider, Security Token Service in SharePoint server creates claims-based Security Token.
9. Security Token Service stores claims in Distributed Cache Service (DCS) in SharePoint farm. Claims in Security Token include user identity and the roles of the user account.
10. IIS server on the SharePoint server sends Federated Authentication or Fed Cookie to the client machine. The cookie contains encrypted key or index to the security token (which is created in SharePoint server in Distributed Cache Service - DCS).
11. The end-user accesses the page, If the user is authorized, through analysis of the claims in the security token and configured permission.
12. After authorization is successful, the SharePoint server sends the content of the page. Fed Auth Cookie is used for authentication of any subsequent requests.

**Security Assertion Markup Language (SAML) token-based authentication**

1. Assume, on client's computer there's no any claims-based security token, User requests SharePoint resource (Site, list, page, etc....) and SAML Authentication process starts with this request.
2. SharePoint server redirects the user (client computer) to the Identity Federation server (ADFS here) server to obtain SAML-based login page
3. The user enters credentials in the login page and sends back to Identity Federation server (ADFS here) server and requests SAML security token
4. Identity Federation server (ADFS here) sends credentials to Identity Provider (ADDS here)
5. Identity Provider (ADDS here) confirms credentials and send a reply to Identity Federation server (ADFS here)
6. Identity Federation server (ADFS here) then creates a SAML token for the user, signs it and send it to the user.
7. The client computer sends a fresh request to SharePoint resource (Site, list, page, etc....). This time SAML security token is also included in the request that it got from the Identity Federation server (ADFS here).
8. Security Token Service in SharePoint server creates claims-based Security Token from SAML security token received from the Identity Federation server (ADFS here).
9. Security Token Service stores claims in Distributed Cache Service in SharePoint farm
10. IIS server on the SharePoint server sends Federated Authentication or ned Auth Cookie to the client machine. The cookie contains encrypted key or index to the security token (which is created in SharePoint server in Distributed Cache Service).
11. The end-user accesses the page if the user is authorized, through analysis of the claims in the security token and configured permission.
12. After authorization is successful, the SharePoint server sends the content of the page. Fed Auth Cookie is used for authentication of any subsequent requests.

**-:Day 3rd :-**