Cello How-To Guide

Data Level Security

Contents

1 Data Level Security

Entitlement management is a technique that grants, resolves, enforces, revokes and administers fine-grained access entitlements (also referred to as “authorizations,” “privileges,” “access rights,” “permissions” and/or “rules”). Its purpose is to execute IT access policies to structured/unstructured data, devices and services. Entitlement management can be delivered by different technologies, and is often different across platforms, applications, network components and devices.

CelloSaaS helps SaaS Application developers to build Security at all the levels and layers of the Application, thus Data level Security is one of the primary area, where configuration led system identifies who can view/add/edit/delete data at the data level.

How data scope works

When an end user invokes a SaaS application via a remote client like a Web browser and, upon authentication, requests the data from the Persistent Storage. The application processes the request and prepares the response. The data needed for the response is obtained from the database that's accessible via the DAL (Data Access Layer). The latter invokes the ES (Entitlement System) to authorize the request.

In the Datascope, user entitlement to the actual data is interpreted as an association between the user and a set of data filters. The data filter is a meta-definition of a real SQL filter, i.e., part of the "WHERE" clause. The data filter comprises the filter name and the ordered list of the filter parameters. To fill in filter parameters, the Datascope obtains the user profile and matches the user profile's attributes with the data filter parameters

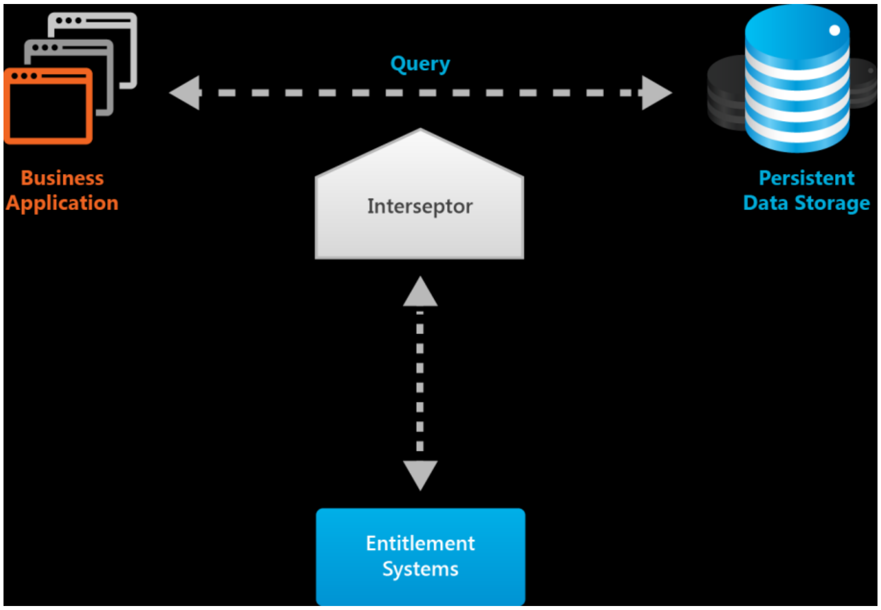


Figure -1

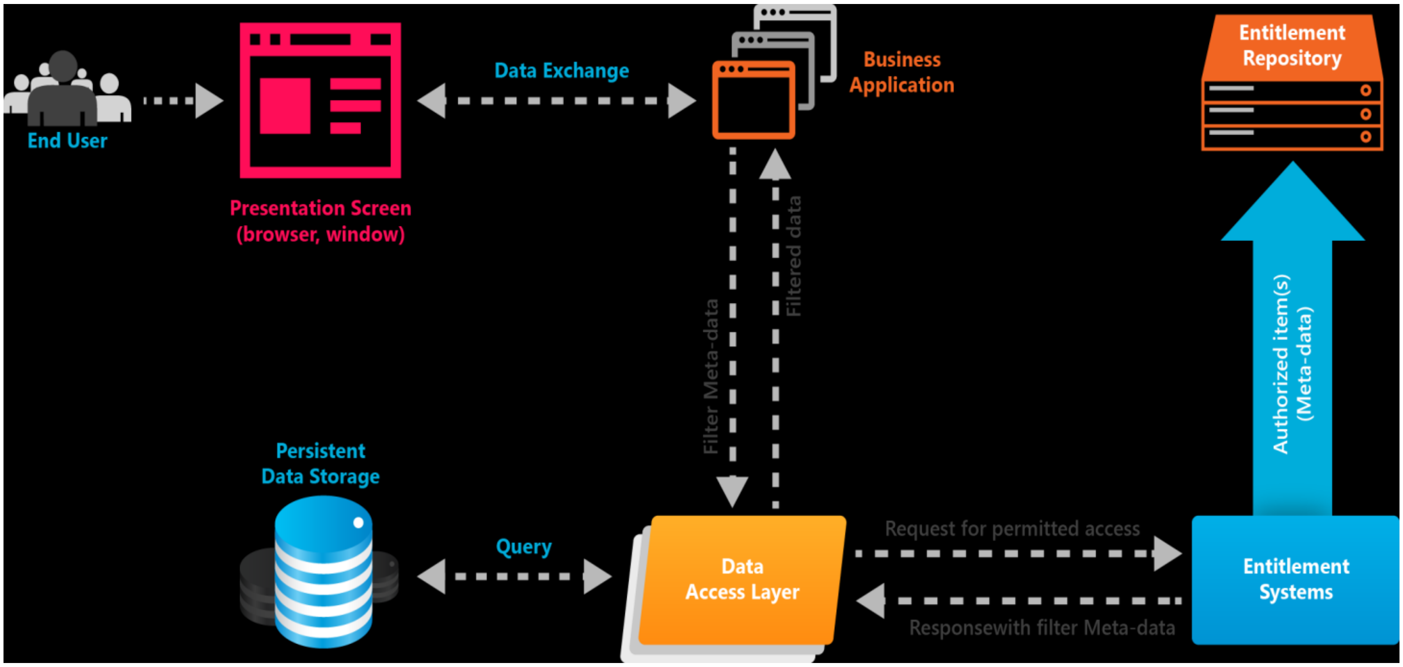


Figure -2

The Datascope returns a list of data filters the user is entitled to. At this moment, the DAL does not know which data filter to apply. The DAL iterates through the received data filter collection and, using the Entitlement API, enables matching filters defined in the DAL. For every enabled filter, the DAL sets filter parameters using values obtained by the ES from the user profile.

This document primarily focuses on different methods of entitlement to persistent data.

1.1 Data Scope

- Each tenant gets ability to configure their access control policy for their Roles and users at Data Level
- It helps to restrict the user access at data level
- Apply datascope against all the Application entities

1.2 Pre-Requisites

- Entity must inherit from ***CelloSaaS.Model.BaseEntity*** class.
- Each entity must be attributed with the privileges.
- Entity privileges should follow the naming conversion “{Action}_{EntityId}”.

Note: Action refers to CRUD operations and other actions.

Example

```
[EntityDescriptor(SchemaTableName = "Employees", PrimaryKeyName = "EmployeeID", IsExtensible=
false, SchemaTableConnectionStringName = "ApplicationConnectionString",
Privileges = "Add_EmployeeDetails,View_EmployeeDetails,Edit_EmployeeDetails,
Delete_EmployeeDetails")]
[EntityIdentifier(Name = "EmployeeDetails")]
public class EmployeeDetails : BaseEntity
{
        // Add required properties
}
```

Note: Assume your business application contains a Persistent model named Employee where it maintains all the Employee Information. The data must be secured and hence it cannot be exposed to all the users of the applications as such, so it requires a security entitlement to configure who get to see what.

In the above scenario, we could set the below data scope in order to secure the data.

Privilege	Description
My Team	The user can access only their team employee details.
My Department	The user can access only their department employee details.
All Data	The user can access all member details. This is more or less same as without mapping the datascope.
Nil	User cannot access any employee details.

1.2.1 How to add data scope

- Define the data scope in DataScope table and map the datascope with entity in ***EntityDataScope*** table.
- You need to define the ***dbo.datascope*** in the CelloSaaS Meta database and map it with the defined privilege.
- When you demand the privileges in the application code, CelloSaaS appends the datascope query with your actual query in the DAL and return the result with filtered condition (Refer Picture-2).
- DataScope query will be appended with your fetch and search query.

1.2.2 DataScope table Schema

Column	Data Type	Description
Name	Varchar(100)	Name of the data scope
SelectQuery	Varchar(255)	Filter query which is going to append with your fetch or search query
Condition	Varchar(255)	If required, you can give condition to your database query
AccessLevel	Small Int	Sets the priority of the datascope. Same privilege may be mapped with different datascopes in different roles. If a user has two roles, then that particular privilege will get two datascopes. That time CelloSaaS will consider the datascope which has high priority access level.

To insert data scope details in database use the following script.

```
INSERT INTO [DataScope]([DataScope_ID],[DataScope_Name], [DataScope_SelectQuery],
[DataScope_Condition], [DataScope_AccessLevel],[DataScope_CreatedBy],
[DataScope_CreatedOn],[DataScope_Status])
VALUES('CD19DE8B-7ABE-4FA4-AF6B-EFBC47ED0727','My Department', 'select E.EmployeeId from Employees
as E left outer join EmployeeDepartment as ED on ED.EmployeeId = E.EmployeeID', 'ED.DepartmentId = (Select
EmpD.DepartmentId from EmployeeDepartment as EmpD inner join Employees as Emp on Emp.EmployeeID =
EmpD.EmployeeId  where Emp.UserId = ''@UserIdentity'' ) and E.TenantID = ''@TenantId''', 1,'3398F837-B988-
4708-999D-D3DFE11875B3', GETDATE(),1)
```

1.2.3 EntityDataScope table Schema

Column	Data Type	Description
DataScopeId	Varchar(100)	Data Scope identifier from DataScope table.
EntityId	Varchar(200)	Entity Identifier which has been from entity table/configuration
BridgeCondition	Varchar(255)	Conjunction property to append your fetch or search query with datascope query.

To insert entity datascope details in database use the following script.

```
INSERT INTO [EntityDataScope]([EntityDataScope_ID],
[EntityDataScope_DataScopeID],[EntityDataScope_EntityID],
[EntityDataScope_BridgeCondition],[EntityDataScope_CreatedBy],
[EntityDataScope_CreatedOn],[EntityDataScope_Status])
VALUES(NEWID(),'CD19DE8B-7ABE-4FA4-AF6B-EFBC47ED0727','EmployeeDetails',
          'EmployeeID In','3398F837-B988-4708-999D-D3DFE11875B3',GETDATE(),1)
```

1.2.4 How to set the DataScope and EntityDataScope in Service and DAL

- You can get the DataScope and EntityDataScope in Service using ***CelloSaaS.Services.BaseService.SetDataScopes*** methods.

- Pass datascope and EntityDataScope while create Data Search Request instance and pass it to DAL method.

```
    Namespace: CelloSaaS.Services
    Class: BaseService
/// <summary>
/// To set dataScope and entityBasedDataScope in out parameter
/// </summary>
/// <param name="entityIdentifier">Entity Identifier</param>
/// <param name="permissionName">Permission Name</param>
/// <param name="dataScope">Data Scope Details</param>
/// <param name="entityDataScope">Entity Based Data Scope Details</param>
void SetDataScopes(string entityIdentifier, string permissionName, out DataScope dataScope, out
EntityBasedDataScope entityDataScope)

/// <summary>
/// To set list dataScope and entityBasedDataScope in out parameter
/// </summary>
/// <param name="entityIdentifier">Entity Identifier</param>
/// <param name="permissionList">Permission List</param>
/// <param name="dicDataScope">Data Scope Details</param>
/// <param name="dicEntityDataScope">Entity Based Data Scope Details</param>
void SetDataScopes(string entityIdentifier, string[] permissionList, out Dictionary<string, DataScope>
dicDataScope, out Dictionary<string, EntityBasedDataScope> dicEntityDataScope)
```

1.2.5 Example for Data Scope Execution

```
public Dictionary<string, EmployeeDetails> GetEmployeeDetailssByTenantId(string tenantId)
{
DataScope dataScope = null;
EntityBasedDataScope entityBasedDataScope = null;
IEmployeeDetailsDAL employeeDetailsDAL = (IEmployeeDetailsDAL)
DALImplementationFactory.GetDALImplementation(typeof(IEmployeeDetailsDAL));

// Set datascope and entitydatascope using Base Service SetDataScopes method.
base.SetDataScopes(new EmployeeDetails().EntityIdentifier,
ManageEmployeeEmployeeDetailsConstants.ViewEmployeeDetails, out dataScope, out
entityBasedDataScope);

EmployeeDetailsSearchCondition employeeDetailsSearchCondition = new
EmployeeDetailsSearchCondition();
employeeDetailsSearchCondition.TenantID = tenantId;

// Pass datascope and entitydatascope in data search request
DataSearchRequest dataSearchRequest = new DataSearchRequest(employeeDetailsSearchCondition,
dataScope, entityBasedDataScope);

return employeeDetailsDAL.Search(dataSearchRequest);

}
```

Note: Get the DataScope and EntityDataScope from Data Search Request and pass it to FetchQuery instance creation.

Example

```
protected override Dictionary<string, EmployeeDetails> DoSearch(DataSearchRequest
dataSearchRequest)
{
        FetchQuery fetchQuery = new FetchQuery(getConnectionStringName(),
        dataSearchRequest.DataScope, dataSearchRequest.EntityBasedDataScope);

        //Do DAL Logic.
}
```

Note: You can use the ***BaseService.ValidateAccess()*** method to check the datascope while update and delete operation in service.

```
Namespace: CelloSaaS.Services
Class: BaseService
/// <summary>
/// To validate access for given entity and entityAction
/// </summary>
/// <param name="entityAction">Entity Action</param>
/// <param name="entity">Enity Details</param>
/// <returns>returns True/False</returns>
bool ValidateAccess(EntityAction entityAction, BaseEntity entity)
```

Example for Update:

```
    public void UpdateEmployeeDetails(EmployeeDetails employeeDetails)
    {
        EntityAction entityAction = new
        EntityAction(ManageEmployeeEmployeeDetailsConstants.EditEmployeeDetails);
        entityAction.Operation = EntityOperation.Update;
        if (!base.ValidateAccess(entityAction, employeeDetails))
        {
                throw new UnauthorizedAccessException("Permission to update the given employee
                is denied.");
        }

        // Do you update logic
    }
```

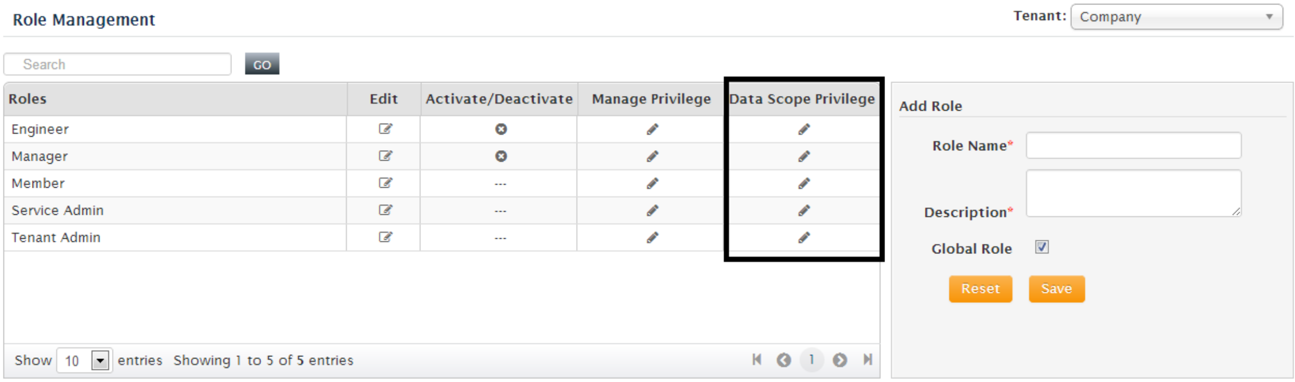
Example for Delete:

```
    public void DeleteEmployeeDetails(string employeeID)
    {
        EmployeeDetails employeeDetails = this.GetEmployeeDetailsByEmployeeID(employeeID);
         EntityAction entityAction = new
        EntityAction(ManageEmployeeEmployeeDetailsConstants.DeleteEmployeeDetails);
        entityAction.Operation = EntityOperation.Delete;
        if (!base.ValidateAccess(entityAction, employeeDetails))
        {
                throw new UnauthorizedAccessException("Permission to delete the given employee is
                denied.");
        }

        // Do your delete logic.
     }
```

1.2.6 How to set the datascope for entity privilege

- Navigate to **Access Control -> Manage Roles**. This page will list all the available roles for logged-in user tenant. For more details refer **Role Management** here.
- Click Data Scope Privilege icon to manage data scope for a particular role.



Role Management

Tenant: Company

Search GO

Roles	Edit	Activate/Deactivate	Manage Privilege	Data Scope Privilege
Engineer	✎	⊗	✎	✎
Manager	✎	⊗	✎	✎
Member	✎	---	✎	✎
Service Admin	✎	---	✎	✎
Tenant Admin	✎	---	✎	✎

Show 10 entries Showing 1 to 5 of 5 entries

Add Role

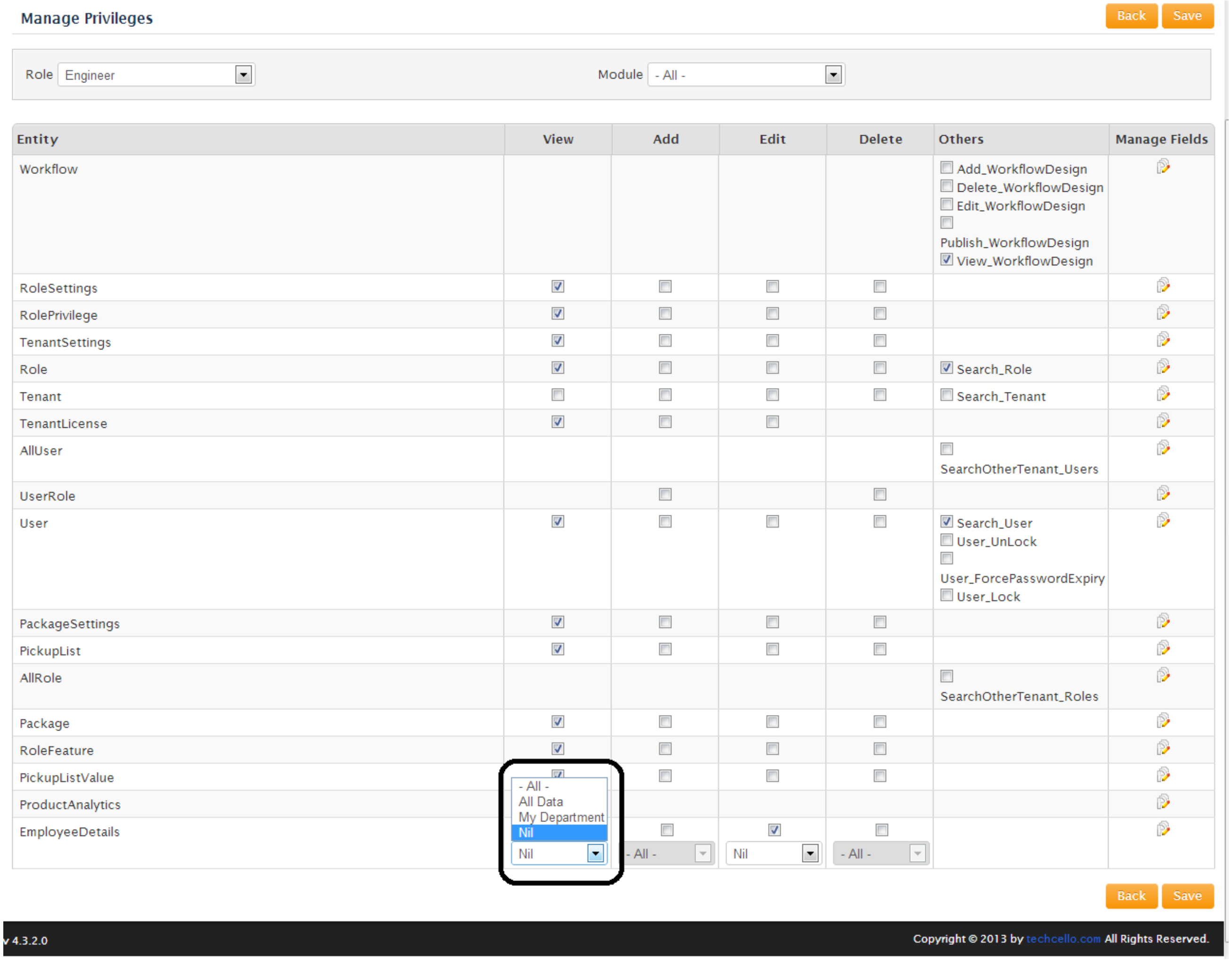
Role Name*

Description*

Global Role ☑

Reset Save

- Manage Data Scope page will display all the entity and its privileges with the DataScope details.
- You can select the required datascope against the entity privilege.



Manage Privileges

Back Save

Role Engineer Module - All -

Entity	View	Add	Edit	Delete	Others	Manage Fields
Workflow					□ Add_WorkflowDesign □ Delete_WorkflowDesign □ Edit_WorkflowDesign □ Publish_WorkflowDesign ☑ View_WorkflowDesign	✎
RoleSettings	☑	□	□	□		✎
RolePrivilege	☑	□	□	□		✎
TenantSettings	☑	□	□	□		✎
Role	☑	□	□	□	☑ Search_Role	✎
Tenant	□	□	□	□	□ Search_Tenant	✎
TenantLicense	☑	□	□			✎
AllUser					□ SearchOtherTenant_Users	✎
UserRole		□		□		✎
User	☑	□	□	□	☑ Search_User □ User_UnLock □ User_ForcePasswordExpiry □ User_Lock	✎
PackageSettings	☑	□	□	□		✎
PickupList	☑	□	□	□		✎
AllRole					□ SearchOtherTenant_Roles	✎
Package	☑	□	□	□		✎
RoleFeature	☑	□	□	□		✎
PickupListValue	☑	□	□	□		✎
ProductAnalytics						✎
EmployeeDetails	- All - All Data My Department Nil Nil	□ - All -	☑ Nil	□ - All -		✎

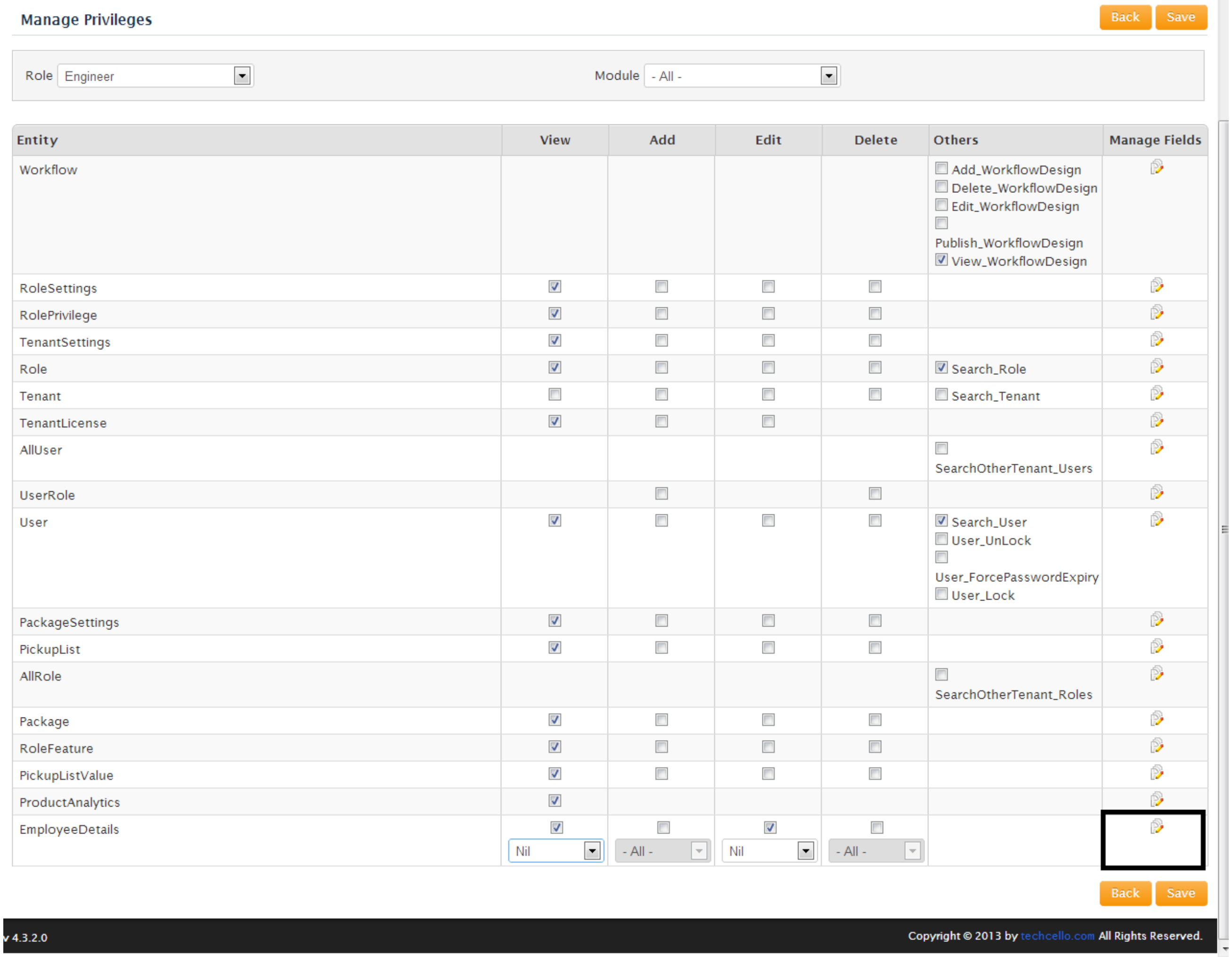
Back Save

v 4.3.2.0

Copyright © 2013 by techcello.com All Rights Reserved.

1.3 *Field Level Security*

- You can control the access at a field level in CelloSaaS.
- Field privileges can be set by using Mange Role Privilege screen which is provided by CelloSaaS.
- Each entity will has set of fields. These fields have Edit and View privileges.
- **Select Access Control-> Manage Roles** then select the **Manage Data Scope** icon. You will get the registered entity details with privileges. Select **Manage Fields** icon to set the field privileges.



Manage Privileges

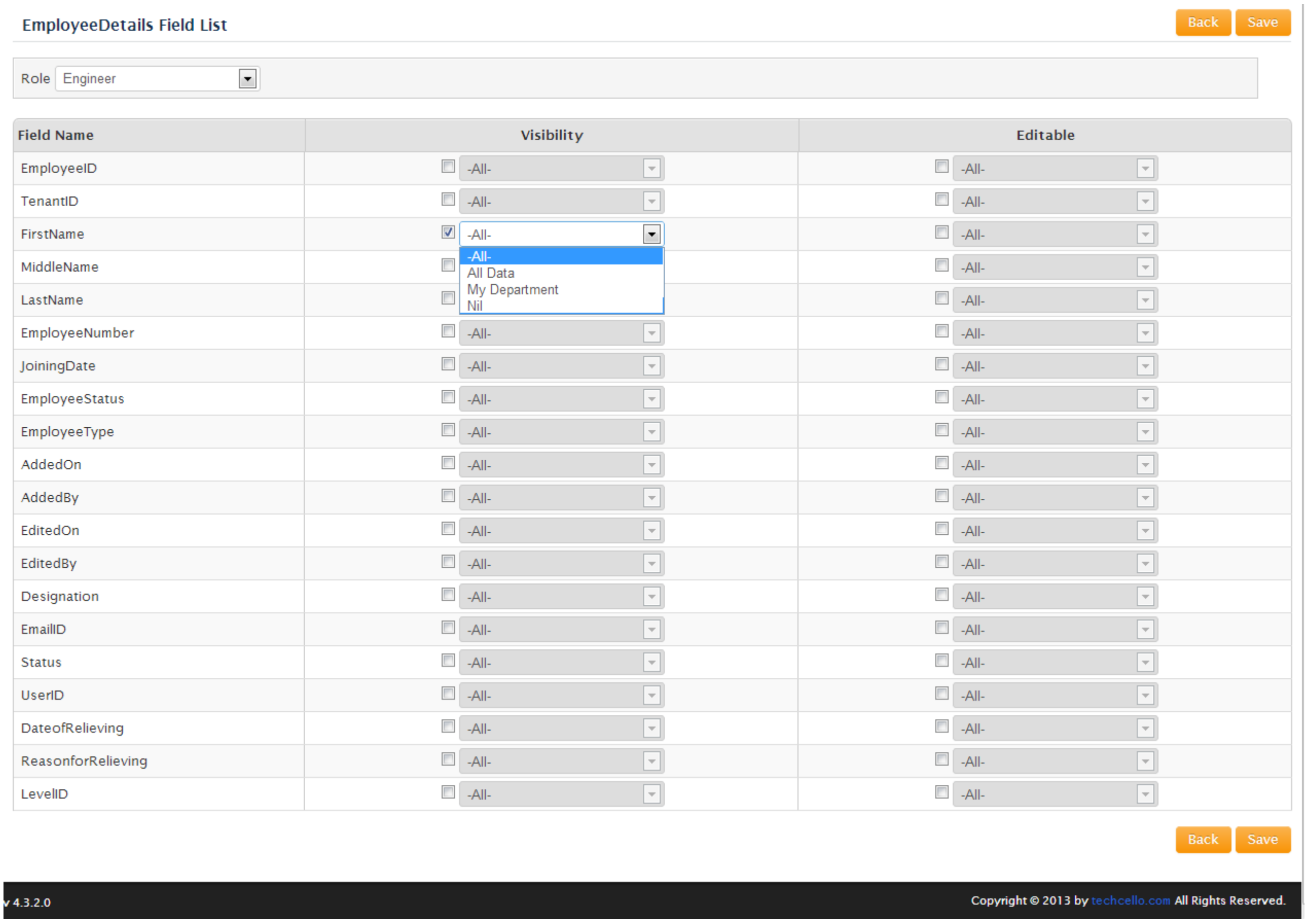
Role: Engineer   Module: - All -

Entity	View	Add	Edit	Delete	Others	Manage Fields
Workflow					□ Add_WorkflowDesign □ Delete_WorkflowDesign □ Edit_WorkflowDesign □ Publish_WorkflowDesign ☑ View_WorkflowDesign	✎
RoleSettings	☑	□	□	□		✎
RolePrivilege	☑	□	□	□		✎
TenantSettings	☑	□	□	□		✎
Role	☑	□	□	□	☑ Search_Role	✎
Tenant	□	□	□	□	□ Search_Tenant	✎
TenantLicense	☑	□	□			✎
AllUser					□ SearchOtherTenant_Users	✎
UserRole		□		□		✎
User	☑	□	□	□	☑ Search_User □ User_UnLock □ User_ForcePasswordExpiry □ User_Lock	✎
PackageSettings	☑	□	□	□		✎
PickupList	☑	□	□	□		✎
AllRole					□ SearchOtherTenant_Roles	✎
Package	☑	□	□	□		✎
RoleFeature	☑	□	□	□		✎
PickupListValue	☑	□	□	□		✎
ProductAnalytics	☑					✎
EmployeeDetails	☑ Nil	□ - All -	☑ Nil	□ - All -		✎

v 4.3.2.0

Copyright © 2013 by techcello.com All Rights Reserved.

- Manage Field page will list the field details of selected entity. All fields will have visibility and editable privilege. You can select the visibility and editable privilege with datascope.



EmployeeDetails Field List

Role: Engineer

Field Name	Visibility	Editable
EmployeeID	☐ -All-	☐ -All-
TenantID	☐ -All-	☐ -All-
FirstName	☑ -All-	☐ -All-
MiddleName	☐	☐ -All-
LastName	☐	☐ -All-
EmployeeNumber	☐ -All-	☐ -All-
JoiningDate	☐ -All-	☐ -All-
EmployeeStatus	☐ -All-	☐ -All-
EmployeeType	☐ -All-	☐ -All-
AddedOn	☐ -All-	☐ -All-
AddedBy	☐ -All-	☐ -All-
EditedOn	☐ -All-	☐ -All-
EditedBy	☐ -All-	☐ -All-
Designation	☐ -All-	☐ -All-
EmailID	☐ -All-	☐ -All-
Status	☐ -All-	☐ -All-
UserID	☐ -All-	☐ -All-
DateofRelieving	☐ -All-	☐ -All-
ReasonforRelieving	☐ -All-	☐ -All-
LevelID	☐ -All-	☐ -All-

v 4.3.2.0

Copyright © 2013 by techcello.com All Rights Reserved.

- Datascope which has been mapped with privileges will also be applied in field level access.
- It is not required to enter the Field Level Privileges. If the privilege does not exist, cellosaas will put the entry while assigning the privileges for a user.
- Cello follows a naming convention for privileges name.
  - For Visibility: ViewField_FieldIdentifier
  - Example: ViewField_FirstName
  - For Editable: EditField_FieldIdentifier
  - Example: EditField_FirstName
- If you do not set the privileges to a field, Cello will apply parent entity privileges.
- Dataview table and cello Grid will take care of field level access by using FieldIdentifier which have been given.

Limitations

- CelloSaaS Data scope is a technique which is developer centric and requires developers involvement in order to change the Data scope policies

- Data entitlement can only be set at the Database level and not at the physical resources.

2 Contact Information

Any problem using this guide (or) using Cello Framework. Please feel free to contact us, we will be happy to assist you in getting started with Cello.

Email: support@techcello.com

Phone: +1(609)503-7163

Skype: techcello