Assignment 2

FUNCTIONS DOCUMENTATION

It is very important for all of you to get familiar with these functions which you will be using throughout the assignment.

FILE TO BE INCLUDED:

<script src="SecurityManager.js"></script>

Must be included in your file before you use any of the function described below.

Use the following like **SecurityManager.FucntionName:**

ValidateAdmin (login, password):
Takes the login and password and true if the credentials are valid and false otherwise.

SaveUser (userObj, successFunction, errorFunc):
For provided user object will store user object. **successFunction** is function name which will be called in case object is added successfully. **errorFunc** is the function which will be called in case some error occurs.

DeleteUser (userID, successFunction, errorFunc):
For provided user id will delete user object. **successFunction** is function name which will be called in case object is deleted successfully. **errorFunc** is the function which will be called in case some error occurs.

GetUserById (userID)
For provided id will return the student object.

GetAllUsers ()
Returns an array of all student objects added so far.

SaveRole (roleObj, successFunction, errorFunc)
For provided role object will store role object. **successFunction** is function name which will be called in case object is added successfully. **errorFunc** is the function which will be called in case some error occurs.

DeleteRole (roleID, successFunction, errorFunc)
For provided role id will delete role object. **successFunction** is function name which will be called in case object is deleted successfully. **errorFunc** is the function which will be called in case some error occurs.

GetRoleById (roleID)
For provided id will return the role object.

GetAllRoles ()
Returns an array of all role objects added so far.

SavePermission (permissionObj, successFunction, errorFunc)
For provided permission object will store permission object. **successFunction** is function name which will be called in case object is added successfully. **errorFunc** is the function which will be called in case some error occurs.

DeletePermission (permissionID, successFunction, errorFunc)
For provided permission id will delete permission object. **successFunction** is function name which will be called in case object is deleted successfully. **errorFunc** is the function which will be called in case some error occurs.

GetPermissionById (permissionID)
For provided id will return the permission object.

GetAllPermissions ()
Returns an array of all permission objects added so far.

SaveRolePermission (rolePermissionObj, successFunction, errorFunc)
For provided rolePermission object will store rolePermission object. **successFunction** is function name which will be called in case object is added successfully. **errorFunc** is the function which will be called in case some error occurs.

DeleteRolePermission (rolePermissionID, successFunction, errorFunc)
For provided rolePermission id will delete rolePermission object. **successFunction** is function name which will be called in case object is deleted successfully. **errorFunc** is the function which will be called in case some error occurs.

GetRolePermissionById (rolePermissionID)
For provided id will return the rolePermission object.

GetAllRolePermissions ()
Returns an array of all rolePermission objects added so far.

SaveUserRole (userRoleObj, successFunction, errorFunc)
For provided userRole object will store userRole object. **successFunction** is function name which will be called in case object is added successfully. **errorFunc** is the function which will be called in case some error occurs.

DeleteUserRole (userRoleID, successFunction, errorFunc)
For provided userRole id will delete userRole object. **successFunction** is function name which will be called in case object is deleted successfully. **errorFunc** is the function which will be called in case some error occurs.

GetUserRoleById (userRoleID)
For provided id will return the userRole object.

GetAllUserRoles ()
Returns an array of all userRole objects added so far.

GetAllCountries ()
Return an array of all the country objects to be displayed.

GetCitiesByCountryId (cid)
For the provided coutryId return an array of all the matching city objects

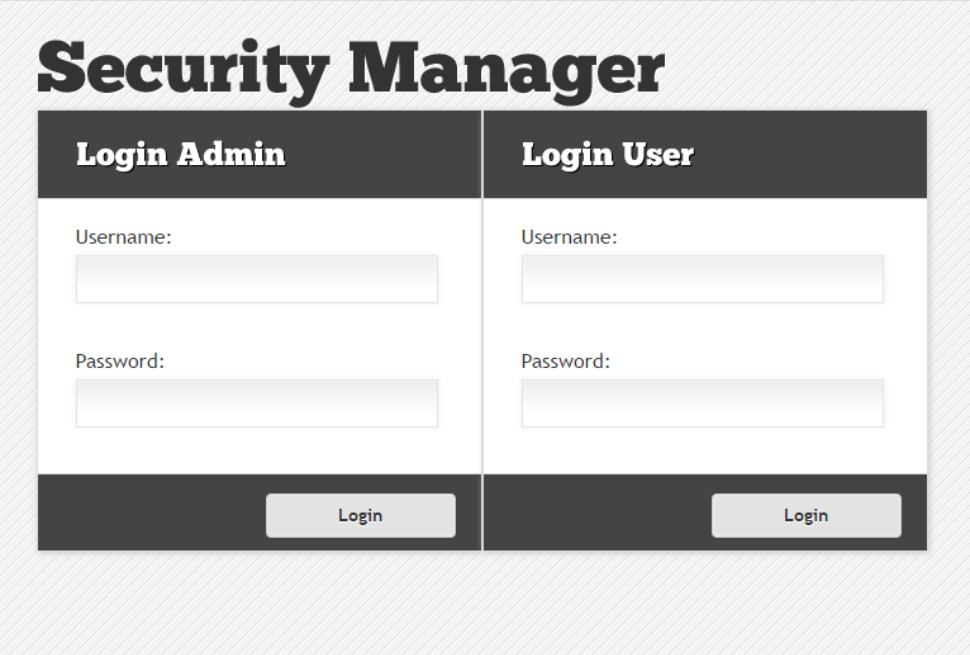
NOTE:

You are to implement all success and failure case functions to be passed to the functions with proper messages being displayed for every event. Not doing so may result n deduction of marks.

TASK1:

- Create login page.
- The page must have 2 login forms
    - One for admin login
    - One for user login.

Note: you are to perform all validations (like checking for empty fields, etc) for all fields of each form.

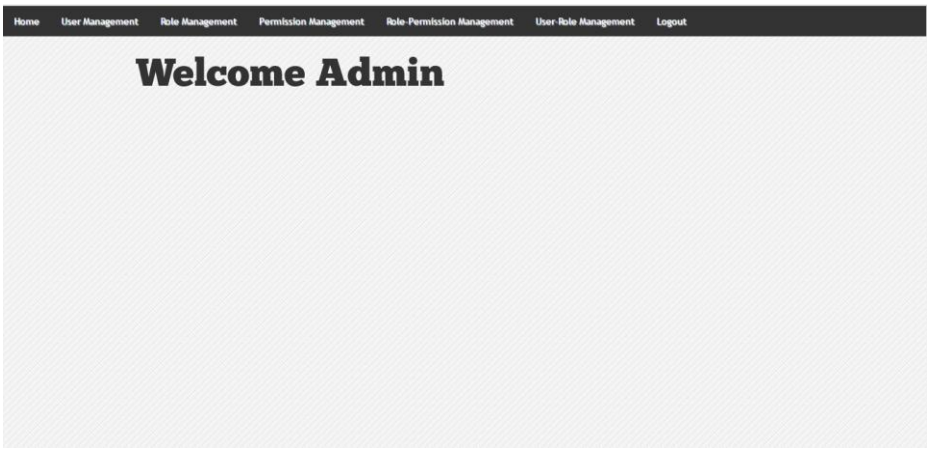


TASK2:

- Admin login must be handled using **ValidateAdmin**() described above.
- **In case you do not use the provided function to validate the admin login and/or use hard coded credentials you will be marked 0 in this task.**
- On successful login redirect to **AdminHome**.

TASK3:

- On successful login display **AdminHome** screen.



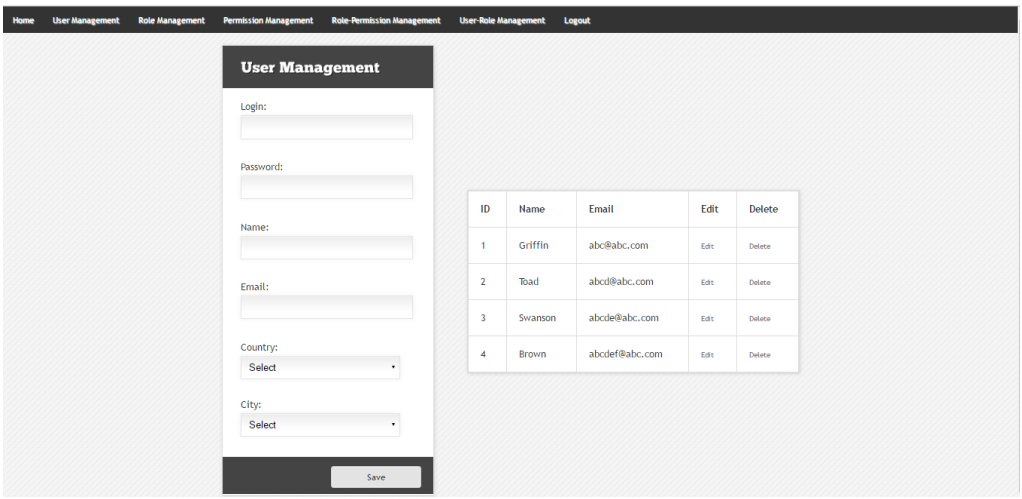
- This screen should have a welcome message.

- There should be links to all other admin screen on the top of screen.
- There must also be a **logout** button which is to take you back to the login screen.

TASK4:

User management screen:

- There must be a form to fill in to add a new User object.

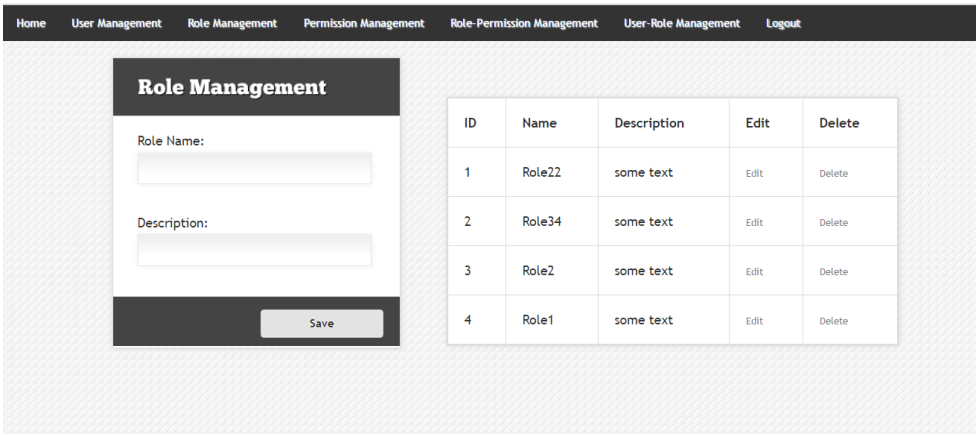


- You are to perform all the necessary validations including **checking for uniqueness of Login and Email.**
- In case of **clear** is pressed clear all the fields.
- The dropdown for country must be initialized using **GetAllCountries**().
- When a certain country is selected in the dropdown for country then get all the cities against that country id and show them as options in the city select dropdown. You are to use **GetCitiesByCountryId()** to accomplish this task.
- When save button is clicked than save the user object using **SaveUser**().
- The grid is to be initialized by all the added users using the **GetAllUsers**(). Every time a new user is **added/updated/deleted than the grid is to be refreshed**.
- For any record if delete is clicked you are to delete that object using **DeleteUser**(). You are also to **display a confirmation box before deleting** an object and only delete object **if user press OK or cancel otherwise.**
- For any record if update is clicked than load that object in the form's fields. When save is clicked than the user object must be updated using **SaveUser**(). **(Hint: when editing the object you are to omit the validation for checking uniqueness of email and login)**

TASK5:

Role management screen:

- There must be a form to enter a new role. You are to provide name and description of Role in separate fields.

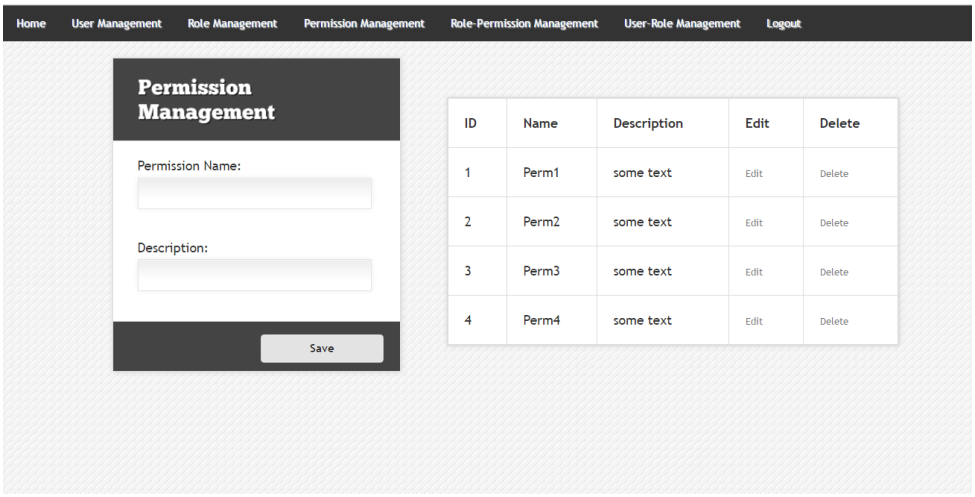


- **The grid will work same as in case of the Task4**. However now you are to display Role records which you will get using **GetAllRoles**().
- Use **SaveRole**() to save a permission object. Also perform all validations necessary like **not leaving a field empty or if Role name already exist.**
- Moreover delete (**also show confirmation alert**) and edit for all records must also work similar to that in Task4. However use **DeleteRole**() to delete role and **SaveRole**() for updating role.
- **The grid must also be refreshed every time a record is added/updated/deleted.**

TASK6:

Permission management screen:

- There must be a form to enter a new Permission. You are to provide name and description of Permission in separate fields.

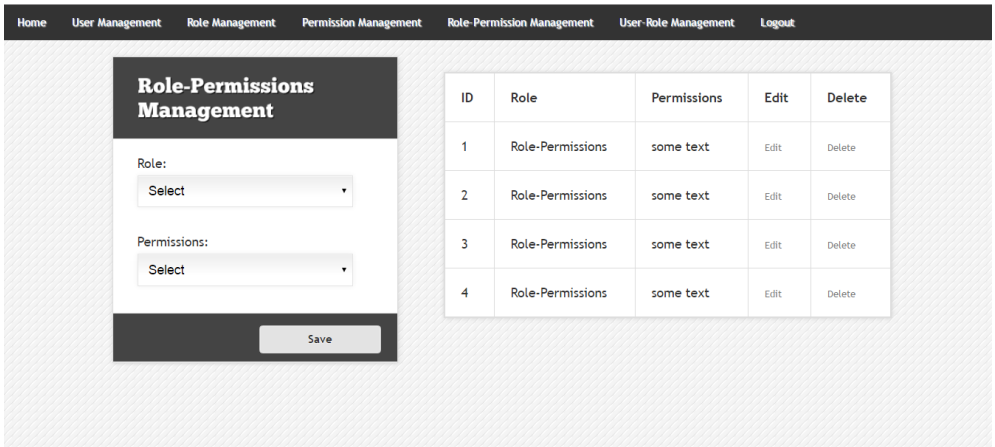


- **The grid will work same as in case of the Task4.** However now you are to display Role records which you will get using **GetAllPermissions**().
- Use **SavePermission**() to save a permission object. Also perform all validations necessary like **not leaving a field empty or if Permission name already exist.**
- Moreover delete (**also show confirmation alert**) and edit for all records must also work similar to that in Task4. However use **DeletePermission** () to delete Permission and **SavePermission** () for updating Permission.
- **The grid must also be refreshed every time a record is added/updated/deleted.**

TASK7:

RolePermission management screen:

- There must be a form to enter a new RolePermission object. There must be two dropdowns.
  - One to select Role which is to be initialized using **GetAllRoles**().
  - One to select Permission which is to be initialized using **GetAllPermissions**().

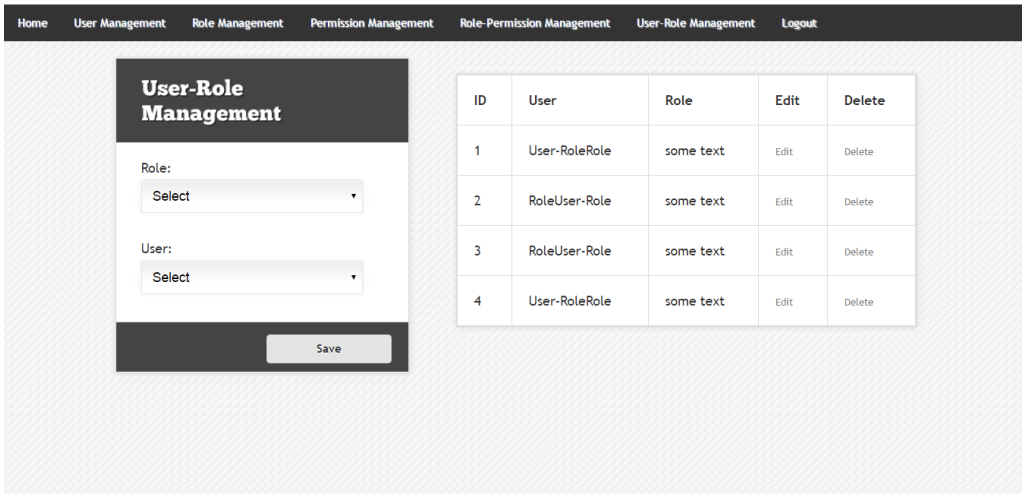


- Next when both options are selected you are to save record using **SaveRolePermission** ().
- The grid will work same as in case of the Task4. However now you are to display Role records which you will get using **GetAllRolePermissions**().
- Moreover delete (**also show confirmation alert**) and edit for all records must also work similar to that in Task4. However use **DeleteRolePermission** () to delete RolePermission and **SaveRolePermission** () for updating RolePermission.
- **The grid must also be refreshed every time a record is added/updated/deleted.**

TASK8:

UserRole management screen:

- There must be a form to enter a new UserRole object. There must be two dropdowns.
    - One to select Role which is to be initialized using **GetAllRoles**().
    - One to select User (**User Login which is unique**) which is to be initialized using **GetAllUsers**().



- Next when both options are selected you are to save record using **SaveUserRole** ().
- The grid will work same as in case of the Task4. However now you are to display Role records which you will get using **GetAllUserRoles** ().
- Moreover delete (**also show confirmation alert**) and edit for all records must also work similar to that in Task4. However use **DeleteUserRole** () to delete UserRole and **SaveUserRole** () for updating UserRole.
- **The grid must also be refreshed every time a record is added/updated/deleted.**

TASK9:

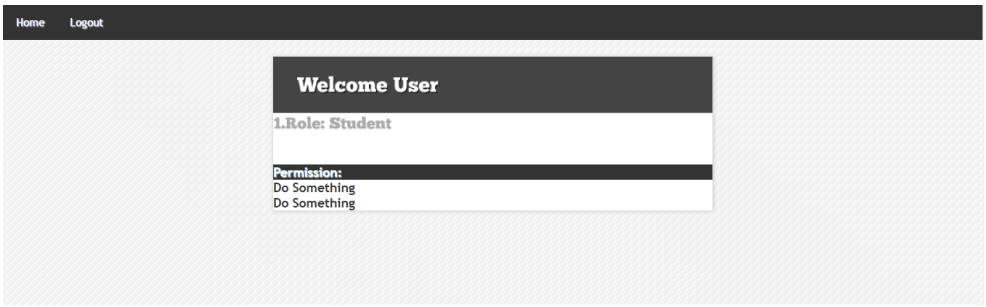
- User log in has to be handled using **GetAllUsers**(). (see screen in task1)
- **The idea here is to get all user records and then write your own logic to check if a user with entered name and password exists.**

- You are to perform all validation as well.
- **If user exist redirect to User Home Screen.**

TASK10:

User Home Screen:

- There must be a welcome message on home screen.



- There must be log out button which will redirect to the main login screen.
- Next you are to display all the Roles and Permissions for that Role for the specified User. **(Hint: you have to work out some mapping of the objects created above. In the first step you are to get all UserRole objects with same user name as the login of current user. Next for that role in UserRole object get all RolePermission objects with same role name and so on.)**

TASK BONUS:

If you implement all the forms with same formatting in the pictures in this document you will get 10% bonus marks in this or any of your previous assignment of yours wanting.