**Prac3:**

In this practical set, you will add the following functionalities to the website you have developed in

Prac2:

Add pages for our university’s security staff to manage parking permits.

Create two roles for the website: one role for the general users and another role for the security staff.

You should continue to work on the **ParkingPermit** project from Prac2, and accomplish the tasks described in the sections A and B below. Note that you should keep an original copy of the project you submitted for Prac2, in case your submission for Prac2 cannot run properly at your tutor’s computer during marking.

**A. Creating and managing roles**

First, you should create two roles for this website: **users** and **staff**. Then, you should complete the following tasks.

**A.1 Registration of users**

When a person registers at this website, he/she will be assigned the role of **users**. You should add source code to the code-behind of **Register.aspx** to achieve this.

**A.2 Registration of security staff**

In the normal running of the website, security staff members should not register through the Register.aspx page. Instead, their accounts should be created in advance. You should at least create one account with the **staff** role for your website, and this account should have the username

[‘permits@westernsydney.edu.a](mailto:permits@westernsydney.edu.au)u’ and the password ‘Pa$$word1’.

To achieve this, you can register [‘permits@westernsydney.edu.a](mailto:permits@westernsydney.edu.au)u’ through Register.aspx first, and then assign it the role of **staff** by editing the **AspNetUserRoles** table in the Identity's database.

Note that security staff members should share the login and logout links with **users**.

**A.3 Authorization of users and security staff**

Create authorization rules such that only logged-in **users** can access the aspx pages inside the

“users” folder of your project.

Create a new folder called “staff” in your project, and create authorization rules such that only logged-in **staff** can access the aspx pages inside the “staff” folder.

**A.4 Dynamic Links**

The links contained in the navigation section should be dynamic. The detailed requirements are as follows:

The links for anonymous surfers should include the following: Home, About, Contact, Registration, and Login. Note that the Login link here is used by both **users** and **staff**.

The links for logged-in **users** should include the following: Home, About, Contact, Personal Details, My Purchases, Make Purchase and Logout.

The links for logged-in **staff** should include the following: Home, About, Contact, Manage

Permits, Plot Charts, and Logout.

**A.5 Dynamic Default.aspx page**

The Home link mentioned above leads to this page.

For anonymous surfers, this page should display a general introduction to the three types of parking permits our university offers.

For logged-in **users**, this page should display an introduction to the links that the users can use.

For logged-in **staff**, this page should display an introduction to the links that a security staff member can use.

**B. Pages for security staff**

The aspx pages required in this section should be:

based on the master page “Site.Master”.

placed inside the “staff” folder of your project.

**B.1 Managing Parking Permits — ManagePermits.aspx**

Recall that the website sells three types of parking permits. Each permit has its own description and different rates for different durations. This page should allow a security staff member to:

List, delete and insert parking permits.

Edit all fields of the **permits** table in the database except the primary key '**type**'.

Specifically, you should accomplish the following:

After logging in, the staff member should see a list of all permits displayed by a **ListView**

control. For each permit, the following fields should be displayed: type, description, quarter\_price, halfyear\_price, and year\_price.

The editing/deleting/inserting interfaces of the ListView should be enabled.

Use AJAX controls (i.e., UpdatePanel and ScriptManager) to wrap around the ListView control such that only this ListView is posted back during editing/deleting/inserting, while

other parts of the page not. That is, put the entire ListView control inside the

<ContentTemplate> of the UpdatePanel control.

For the editing and inserting interfaces, use the **FilteredTextBoxExtender** control from the

AJAX Control Toolkit to implement validations for the following fields:

a. **type** (only for inserting interface): should only consist of upper or lower case English

letters.

b. **description**: should only consist of upper or lower case English letters, hyphen,

comma, full stop, and space.

c. **quarter\_price**, **halfyear\_price**, and **year\_price**: should be a decimal number (i.e., the decimal point is allowed. You can use the ValidChars property to implement

this).

Note that, in your implementation, the FilteredTextBoxExtender will allow “.” to appear multiple times in a decimal number. We accept this kind of limitation in marking, so you will not lose marks because of this. The advantage of FilteredTextBoxExtender is that it doesn’t allow users to type invalid letters at all, whereas the standard validators from ASP.NET allow users to type them. If you want a full implementation of the validation for the above fields, you can combine FilteredTextBoxExtender with those standard validators such as RequiredFieldValidator and RegularExpressionValidator. However, this is not required in this practical set.

**B.2 Plotting Chart on Purchases — PlotChart.aspx**

This page will allow a security staff member to plot an ASP.NET chart on the number of purchases against different types of permits. To do this, your SQL query should count the number of purchases for each type of permit.

The upper part of the page should display the following two DropDownLists, which allow the staff member to select the type and dimension he/she wants for the chart respectively:

The chart type: Column, Pie, or Line

The chart dimension: 2D or 3D

When the page is loaded initially, “Column” should be already selected for the chart type, and “2D” for the chart dimension. And the chart with the selected type and dimension should be plotted under the two DropDownLists. After the staff member changes any selection in the two DropDownLists, the chart should be replotted based on the new selections. (To enable this, the “AutoPostBack” property of the two DropDownLists should be set to “true”.)

Moreover, for any chart plotted, appropriate titles, pie labels or axis labels should be used.