Universidad Nacional de La Matanza Materia: Programación Web 3 2C 2021 – Recup. 1er Parcial

Apellido y Nombre: _____ DNI: _____

Teoría

1) Verdadero o falso ASP.NET MVC. Justifique en los casos que sea falso:

a) Todas las clases que representan al Model, deben terminar con la palabra "Model" por convención.
b) El modo de enrutamiento de la aplicación se define en el archivo appsettings.json.
c) Un middleware es configurado en el método "**Configure**" de la clase "**Startup".** Solo puede existir un middleware por aplicación Web.
d) El controlador ejecuta lógica pero el Modelo es el responsable de elegir la vista que será procesada y devuelta a cliente.

2) Dado un controller con una única acción para el sitio www.misitio.com acción:

```
public class ProductosController : Controller
{
        public ActionResult MostrarTodosProductos()
        {
                return View(ProductosServicio.ObtenerTodosProductos());
        }
}
```

Y considerando la configuración de tabla de ruteo default de MVC mencionada en clase, indique cuáles de las siguientes urls son válidas/correctas (no arrojarían error y encuentran una acción correspondiente en el controlador) . **Justifique muy bien su respuesta**:

a) http://www.misitio.com/ProductosController/MostrarTodosProductos
b) http://www.misitio.com/ProductosController/MostrarTodosActionProductos
c) http://www.misitio.com/Productos/MostrarTodos
d) http://www.misitio.com/Productos/MostrarTodosProductosAction
e) http://www.misitio.com/Productos/MostrarTodosProductos/ObtenerTodosProductos
f) Ninguna es valida

3) Verdadero o falso ASP.NET - MVC. Justifique en los casos que sea falso:

a) Respecto a las variables de sesión en ASP.NET CORE, existe un mecanismo automático para informar al código de la aplicación (servidor) que el navegador se ha cerrado.
b) El tag helper <partial name="_Vista" /> es la forma tradicional de renderizar una vista parcial.
c) En ASP.NET CORE el uso de variables de sesión está activado por defecto; solo es necesario utilizar el objeto o clase: "HttpContext.Session".
d) ViewBag es similar a ViewData, pero se diferencian en su "tiempo de vida", así ViewData permite mantener variables y su contenido después de una redirección, pero ViewBag no.

4) Se sabe que la siguiente Vista NO es un layout y tiene errores. Asuma que existe una clase "**Cliente**" con una propiedad pública "**RazonSocial**" con espacio de nombres: "**PrimerParcialEjercicios.Models**" y que el controller y la acción son correctos.

```
@Model PrimerParcialEjercicios.Models.Cliente


@{
    Layout = "Bienvenidxs al primer parcial de Web 3"; }

@RenderBody()

{

    <h2> Bienvenido @model.RazonSocial a la evaluación</h2> }
```

Indique y explique a que se debe que la vista no funcione correctamente y corrija los errores.

5) Indique V o F. Justifique si es falso

a) En una aplicación ASP.NET MVC el runtime (Framework o Core) debe estar instalado en el server y se recomienda que también este instalado en el cliente para evitar problemas de compatibilidad de versiones.
b) Un Assembly (.dll / .exe) es la unidad mínima de ejecución cuyo código MSIL es creado por el CLR.
c) .Net Core tiene similitudes a .Net Framework y fue escrito sobre la base de .Net Framework.
d) En .Net para castear un string a un int se puede usar la forma de conversión: **int** i **= (int) mystring**;

Practica:

Realizar una aplicación "Registro de Ventas", en Asp.Net MVC que incluya las siguientes funcionalidades y restricciones:

1) Al ejecutar la aplicación, esta debe iniciar por defecto en un controlador llamado "**Presentacion**" en una acción "**Bienvenido**". La vista correspondiente deberá incluir un mensaje de bienvenida al sitio: "Bienvenido al sitio de evaluación"

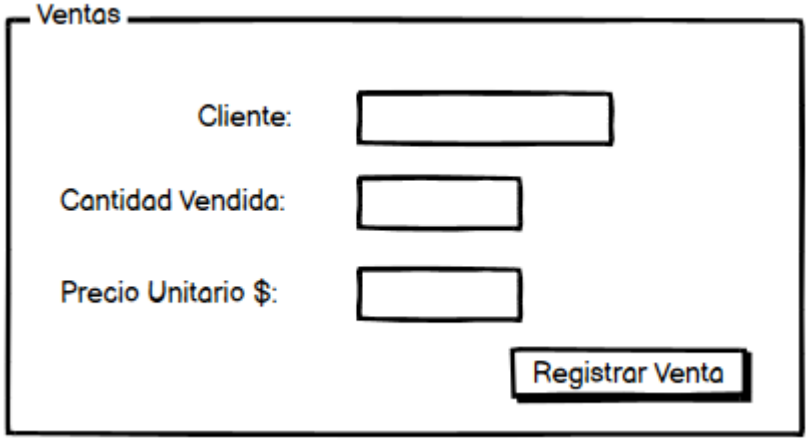
2) **2.1)** Agregar un layout SIN cabecera Fija, pero que contenga un pie o footer Fijo: con la información del alumno:

Nombre y Apellido
DNI

2-2) Se deberán incluir dos links en la vista Bienvenido:
a) **Registrar Venta**
b) **Resultados**

Se pretende que la aplicación tenga un controlador "Ventas", que tenga la lógica necesaria para poder mostrar alguna de las dos vistas según la elección del usuario: a): **Registrar Venta** o b) **Resultados**

3) La vista Registrar Venta, tendrá el siguiente formato:



Al hacer clic en **Registrar** se deberá programar una acción que permita realizar el cálculo y registro de la venta correspondiente con su lógica de negocio respectiva. En términos generales, el cálculo del registro de venta es el siguiente: **Total venta = Cantidad Vendida * Precio unitario.**

4) Se deberá agregar las siguientes validaciones al registro anterior:

- Cliente: Hasta 50 caracteres. Campo requerido
- Cantidad Vendida: Numérico, Campo Obligatorio. La cantidad vendida debe ser mayor a 1 y menor a 300.
- Precio unitario: Numérico, campo obligatorio. El precio unitario debe ser mayor o igual a 10 y menor a 1000.

NOTA: Agregar el código necesario a fin de que NO se registre una venta que no cumple con las validaciones respectivas.

5) Inmediatamente después de realizar el cálculo y registro correspondiente en el punto 3), se deberá agregar el resultado de ese cálculo y la información relacionada a una lista de resultados a los efectos de mantenerlos y no perderlos entre los distintos cálculos / registros (requests) que se realicen. Los elementos de esta lista se mostrarán en una vista de Resultados de la siguiente forma:

Resultados:

#IdVenta	Cliente	Cantidad Vendida	Precio Unitario ($)	Total Venta
1	Metegoles S.A	250	20	$ 5000
2	Medialunas SRL	100	10	$ 1000
3	Tecleando S.A	20	200	$ 4000
…	….	….		….