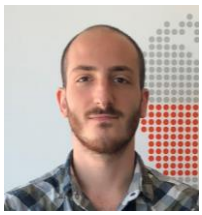
OBLIGATORIO PROGRAMACION 3

Universidad ORT Uruguay

Analista en Tecnologías de la Información - Grupo: N3B - Docente: Plinio Gañi

DOCUMENTACIÓN



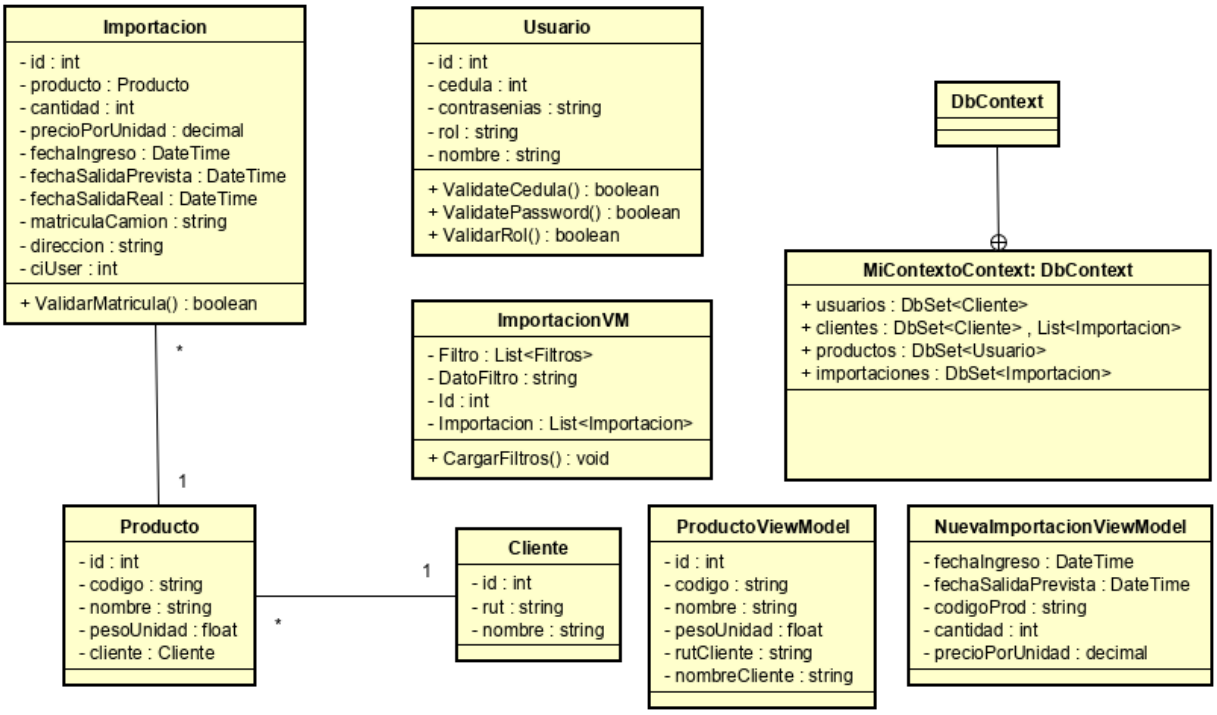
N° Estudiante: 237974 -Nombre: José Ignacio Rossi

N° Estudiante: 218588 - Nombre: Valentin Barreneche

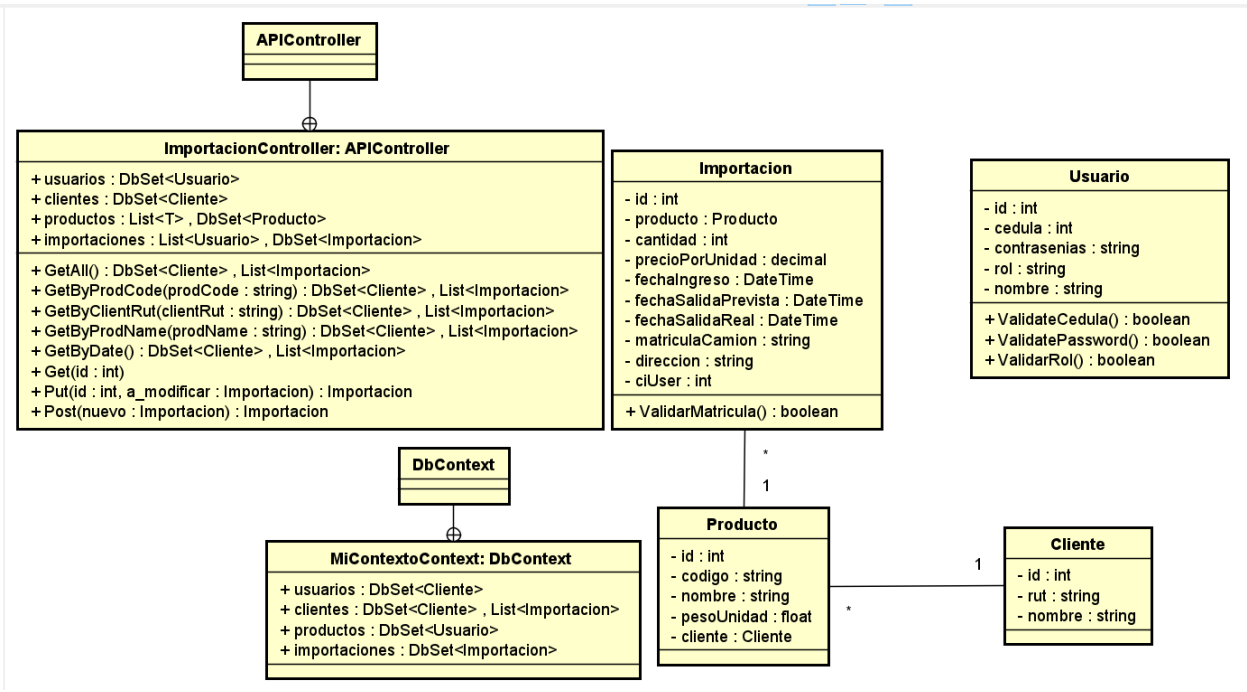
N° Estudiante: 215308 - Nombre: Jonathan Kaiser

Tabla de contenido

1. Diagrama de Clases



2. Diagrama de Servicios



3. Código Fuente

3.1. Dominio

3.1.1. Usuario

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Text.RegularExpressions;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace Dominio
{
    [Table("Usuarios")]
    public class Usuario
    {
        public int Id { get; set; }
        [Required]
        public int Cedula { get; set; }
        [Required]
        public string Nombre { get; set; }
        [Required]
        public string Contrasenia { get; set; }
        [Required]
        public string Rol { get; set; }


        public static bool ValidateCedula(string cedula)
        {
            var input = cedula;
            if (string.IsNullOrWhiteSpace(input))
            {
               return false;
            }
            var hasMiniMaxChars = new Regex(@".{7,9}");
            if (!hasMiniMaxChars.IsMatch(input))
            {
                return false;
            }
            else
            {
                return true;
            }
        }

        public static bool ValidatePassword(string password)
        {
            var input = password;
            if (string.IsNullOrWhiteSpace(input))
            {
                return false;
            }
```

```
            var hasNumber = new Regex(@"[0-9]+");
            var hasUpperChar = new Regex(@"[A-Z]+");
            var hasMiniMaxChars = new Regex(@".{6,15}");
            var hasLowerChar = new Regex(@"[a-z]+");

            if (!hasLowerChar.IsMatch(input))
            {
                return false;
            }
            else if (!hasUpperChar.IsMatch(input))
            {
                return false;
            }
            else if (!hasMiniMaxChars.IsMatch(input))
            {
                return false;
            }
            else if (!hasNumber.IsMatch(input))
            {
                return false;
            }
            else
            {
                return true;
            }
        }

        public static bool ValidarRol(string rolAsignado) {
            bool esValido = false;
            if (rolAsignado == "deposito" || rolAsignado == "admin")
            {
                esValido = true;
            }
            return esValido;
        }

    }
}
```

3.1.2. Cliente

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace Dominio
{
    [Table("Clientes")]
    public class Cliente
    {
        public int Id { get; set; }
        [Required]
        public string Rut { get; set; }
        [Required]
        public string Nombre { get; set; }

    }
}
```

3.1.3. Producto

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace Dominio
{
    [Table("Productos")]
    public class Producto
    {
        public int Id { get; set; }
        [Required]
        public string Codigo { get; set; }
        [Required]
        public string Nombre { get; set; }
        [Required]
        public float PesoUnidad { get; set; }
        public Cliente Cliente { get; set; }

    }
}
```

3.1.4. Importacion

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
using System.Text.RegularExpressions;

namespace Dominio
{
    [Table("Importaciones")]
    public class Importacion
    {
        public int Id { get; set; }
        [Required]
        public DateTime FechaIngreso { get; set; }
        [Required]
        public DateTime FechaSalidaPrevista { get; set; }
        public Producto Producto { get; set; }
        [Required]
        public int Cantidad { get; set; }
        [Required]
        public decimal PrecioPorUnidad { get; set; }
        public DateTime? FechaSalidaReal { get; set; }  //revisar ?
        public string MatriculaCamion { get; set; }
        public string Direccion { get; set; }
        public int? CiUser { get; set; }                //revisar ?


        public bool ValidarMatricula()
        {
            string matricula = MatriculaCamion;
            if (string.IsNullOrWhiteSpace(matricula))
            {
                return false;
            }
            bool matriculaValida = false;


            string firstThreeChars = matricula.Substring(0,3);
            string lastFourChars = matricula.Substring(3, 4);

            bool firstThreeCharsAreCAPS = IsAllUpper(firstThreeChars);
            bool lastFourCharsAreNumbers = IsDigitsOnly(lastFourChars);

            if (firstThreeCharsAreCAPS && lastFourCharsAreNumbers) {
                matriculaValida = true;
            }

            return matriculaValida;
        }


        private static bool IsDigitsOnly(string str)
        {
```

```
            foreach (char c in str)
            {
                if (c < '0' || c > '9')
                    return false;
            }

            return true;
        }

        private static bool IsAllUpper(string input)
        {
            for (int i = 0; i < input.Length; i++)
            {
                if (!Char.IsUpper(input[i]))
                    return false;
            }

            return true;
        }


    }
}
```

3.2. MVC

3.2.1. **Models -** MiContexto Context

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Data.Entity;
using Dominio;

namespace MVC.Models
{
    public class MiContextoContext : DbContext
    {
        public DbSet<Producto> Productos { get; set; }
        public DbSet<Cliente> Clientes { get; set; }
        public DbSet<Importacion> importaciones { get; set; }
        public DbSet<Usuario> Usuarios { get; set; }

        public MiContextoContext() : base("Conexion")
        {

        }

    }
}
```

3.2.2. **Models -** Filtro

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace MVC.Models
{
    public class Filtro
    {
        public int Id { get; set; }
        public string Valor { get; set; }

        public Filtro() { }
    }
}
```

3.2.3. **Models -** Cargar Datos

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.IO;
using Dominio;
using System.Data.Entity;

namespace MVC.Models
{
    public class CargarDatos
    {
        static MiContextoContext db = new MiContextoContext();

        public static bool InsertarDatos(string rutaWeb)
        {
            bool exito = true;

            CargarUsuarios(rutaWeb);
            CargarClientes(rutaWeb);
            CargarProductos(rutaWeb);
            CargarImportaciones(rutaWeb);

            return exito;

        }

        public static void CargarUsuarios(string rutaWeb)
        {
            string rutaRelativa = rutaWeb + @"\MVC\Carga\";
            StreamReader sr = new StreamReader(rutaRelativa + "Usuarios.txt");

            string linea = sr.ReadLine();

            while (linea !=null)
            {
                string[] textoLinea = linea.Split(new char[] { '#' });
                //cargar datos en objeto
                Int32.TryParse(textoLinea[0], out int ci);

                Usuario unUsr = new Usuario()
                {
                    Cedula = ci,
                    Contrasenia = textoLinea[1],
                    Nombre = textoLinea[2],
                    Rol = textoLinea[3]
                };

                var query = db.Usuarios
                                        .Where(u => u.Cedula == unUsr.Cedula)
                                        .FirstOrDefault<Usuario>();

                if (query == null)
                {
                    //guardar en bd
                    db.Usuarios.Add(unUsr);
```

```
            db.SaveChanges();
        }
        linea = sr.ReadLine(); //mueve a la siguiente linea
    }
    sr.Close();
}

public static void CargarClientes(string rutaWeb)
{
    string rutaRelativa = rutaWeb + @"\MVC\Carga\";
    StreamReader sr = new StreamReader(rutaRelativa + "Clientes.txt");

    string linea = sr.ReadLine();

    while (linea != null)
    {
        string[] textoLinea = linea.Split(new char[] { '#' });

        Cliente unCli = new Cliente ()
        {
            Rut = textoLinea[0],
            Nombre = textoLinea[1],
        };

        var query = db.Clientes
                      .Where(c => c.Rut == unCli.Rut)
                      .FirstOrDefault<Cliente>();

        if (query == null)
        {
            //guardar en bd
            db.Clientes.Add(unCli);
            db.SaveChanges();
        }

        linea = sr.ReadLine(); //mueve a la siguiente linea
    }
    sr.Close();
}

public static void CargarProductos(string rutaWeb)
{
    string rutaRelativa = rutaWeb + @"\MVC\Carga\";
    StreamReader sr = new StreamReader(rutaRelativa + "Productos.txt");

    string linea = sr.ReadLine();

    while (linea != null)
    {
        string[] textoLinea = linea.Split(new char[] { '#' });
        //cargar datos en objeto
        string rut = textoLinea[3];

        var queryIdCliente = db.Clientes
                               .Where(c => c.Rut == rut)
                               .FirstOrDefault<Cliente>();

        Producto unProd = new Producto()
        {
```

```
                    Codigo = textoLinea[0],
                    Nombre = textoLinea[1],
                    PesoUnidad = float.Parse(textoLinea[2]),
                    Cliente = queryIdCliente
                };

                var query = db.Productos
                                    .Where(prod => prod.Codigo == unProd.Codigo)
                                    .FirstOrDefault<Producto>();

                if (query == null)
                {
                    //guardar en bd
                    db.Productos.Add(unProd);
                    db.SaveChanges();
                }

                linea = sr.ReadLine(); //mueve a la siguiente linea
            }
            sr.Close();
        }


        public static void CargarImportaciones(string rutaWeb)
        {
            string rutaRelativa = rutaWeb + @"\MVC\Carga\";
            StreamReader sr = new StreamReader(rutaRelativa + "Importacion.txt");

            string linea = sr.ReadLine();

            while (linea != null)
            {
                string[] textoLinea = linea.Split(new char[] { '#' });

                string codigo = textoLinea[5];
                //traigo el producto
                var queryProd = db.Productos
                                    .Where(prod => prod.Codigo == codigo)
                                    .FirstOrDefault<Producto>();
                Int32.TryParse(textoLinea[1], out int cant);

                Importacion unaImp = new Importacion()
                {
                    Cantidad = cant,
                    PrecioPorUnidad = decimal.Parse(textoLinea[2]),
                    FechaIngreso = DateTime.Parse(textoLinea[3]),
                    FechaSalidaPrevista = DateTime.Parse(textoLinea[4]),
                    Producto = queryProd
                };

                var queryImp = db.importaciones
                                    .Where(i => i.Cantidad == unaImp.Cantidad)
                                    .Where(i => i.PrecioPorUnidad ==
unaImp.PrecioPorUnidad)
                                    .Where(i => i.FechaIngreso == unaImp.FechaIngreso)
                                    .Where(i => i.FechaSalidaPrevista ==
unaImp.FechaSalidaPrevista)
                                    .Where(i => i.Producto.Codigo ==
unaImp.Producto.Codigo)
```

```
                                    .FirstOrDefault<Importacion>();

                if(queryImp == null)
                {
                    db.importaciones.Add(unaImp);
                    db.SaveChanges();
                }

                linea = sr.ReadLine(); //mueve a la siguiente linea
            }
            sr.Close();
        }
    }
}
```

3.2.4. **ViewModels** - ImportacionesVM

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using System.ComponentModel.DataAnnotations;
using Dominio;
using MVC.Models;

namespace MVC.ViewModels
{
    public class ImportacionesVM
    {
        [Display(Name = "Filtros")]
        public SelectList Filtros { get; set; }

        public string DatoFiltro { get; set; }

        public int Id { get; set; }

        public List<Importacion> Importaciones { get; set; }

        public ImportacionesVM(List<Importacion> import)
        {
            CargarFiltros();
            this.Importaciones = import;
        }

        public ImportacionesVM() { }

        public void CargarFiltros()
        {
            List<Filtro> lista = new List<Filtro>()
            {
                new Filtro(){  Id = 0, Valor= ""},
                new Filtro(){  Id = 1, Valor= "Código"},
                new Filtro(){  Id = 2, Valor= "Rut"},
                new Filtro(){  Id = 3, Valor= "Nombre producto"},
                new Filtro(){  Id = 4, Valor= "Pendientes de salida"},
            };

            this.Filtros = new SelectList(lista, "Id", "Valor");
        }
    }
}
```

3.2.5. **ViewModels** - NuevaImportacionViewModel

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace MVC.ViewModels
{
    public class NuevaImportacionViewModel
    {
        public int Id { get; set; }

        [Required]
        [DataType(DataType.Date)]
        [DisplayFormat(DataFormatString = "{0:dd/MM/yyyy}", ApplyFormatInEditMode = true)]
        public DateTime FechaIngreso { get; set; }

        [Required]
        [DataType(DataType.Date)]
        [DisplayFormat(DataFormatString = "{0:dd/MM/yyyy}", ApplyFormatInEditMode = true)]
        public DateTime FechaSalidaPrevista { get; set; }

        public string CodigoProd { get; set; }
        [Required]
        public int Cantidad { get; set; }
        [Required]
        public decimal PrecioPorUnidad { get; set; }

    }
}
```

3.2.6. **ViewModels** – ProductoViewModel

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace MVC.ViewModels
{
    public class ProductoViewModel
    {
        public int Id { get; set; }
        [Required]
        public string Codigo { get; set; }
        [Required]
        public string Nombre { get; set; }
        [Required]
        public float PesoUnidad { get; set; }
        public string RutCliente { get; set; }
        public string NombreCliente { get; set; }
    }
```

```
}
```

3.2.7. **Controllers -** Home Controller

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using Dominio;
using System.Data.Entity;
using MVC.Models;

namespace MVC.Controllers
{
    public class HomeController : Controller
    {
        static MiContextoContext db = new MiContextoContext();


        [HttpGet]
        public ActionResult Index()
        {
            if (Session["cedula"] != null)
            {
                return Redirect("/home/bienvenido");
            }
            else
            {
                return View();
            }
        }

        [HttpPost]
        public ActionResult Index(string cedula, string password)
        {
            if (cedula.Length != 0 && password.Length != 0)
            {
                Int32.TryParse(cedula, out int ciInt);
                Usuario usuarioIngresado = db.Usuarios.Where(u => u.Cedula ==
ciInt).FirstOrDefault<Usuario>();
                if (usuarioIngresado != null)
                {
                    if (usuarioIngresado.Contrasenia == password)
                    {
                        Session["cedula"] = usuarioIngresado.Cedula;
                        Session["nombre"] = usuarioIngresado.Nombre;
                        Session["rol"] = usuarioIngresado.Rol;
                        return Redirect("/Home/Bienvenido");
                    }
                    else
                    {
                        ViewBag.mensaje = "La password no es correcta.";
                    }
                }
                else
                {
                    ViewBag.mensaje = "El Usuario no existe.";
```

```
                }
            }
            else
            {
                ViewBag.mensaje = "Los campos cedula y password no pueden ser nulos.";
            }

            return View();
        }

        public ActionResult Bienvenido() {
            if (Session["cedula"] != null)
            {
                return View();
            }
            else
            {
                return Redirect("/home/index");
            }
        }


        public ActionResult Salir()
        {
            Session["rol"] = null;
            Session["cedula"] = null;
            return RedirectToAction("index");
        }

        public ActionResult Precarga()
        {
            string rutaWeb = HttpRuntime.AppDomainAppPath + @"..";
            bool cargaOk = CargarDatos.InsertarDatos(rutaWeb);
            if (cargaOk)
            {
                ViewBag.Mensaje ="Precarga realizada con exito.";
            }
            else {
                ViewBag.Mensaje ="Precarga realizada con exito.";
            }
            return View();
        }

        public ActionResult About()
        {
            ViewBag.Message = "Your application description page.";

            return View();
        }

        public ActionResult Contact()
        {
            ViewBag.Message = "Your contact page.";

            return View();
        }
    }
}
```

3.2.8. **Controllers** - Productos Controller

```
using Dominio;
using System.Data.Entity;
using System.Linq;
using System.Net;
using System.Web.Mvc;
using System.Collections.Generic;
using MVC.Models;
using MVC.ViewModels;

namespace MVC.Controllers
{
    public class ProductosController : Controller
    {
        private MiContextoContext db = new MiContextoContext();

        public ActionResult Index()
        {
            if (Session["cedula"] != null)
            {
                var listaProds1 = db.Productos
                            .Include(p => p.Cliente)
                            .ToList();

                List<ProductoViewModel> listaProdsViewModels = new
List<ProductoViewModel>();
                foreach (var unProd in listaProds1)
                {
                    ProductoViewModel unProdViewModel = new ProductoViewModel();
                    unProdViewModel.Id = unProd.Id;
                    unProdViewModel.Nombre = unProd.Nombre;
                    unProdViewModel.PesoUnidad = unProd.PesoUnidad;
                    unProdViewModel.Codigo = unProd.Codigo;
                    unProdViewModel.NombreCliente = unProd.Cliente.Nombre;
                    unProdViewModel.RutCliente = unProd.Cliente.Rut;

                    listaProdsViewModels.Add(unProdViewModel);
                }
                return View(listaProdsViewModels);
            }
            else
            {
                return Redirect("/home/Index");
            }
        }

        public ActionResult Details(int? id)
        {

            if (id == null)
            {
                return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
            }
            Producto producto = db.Productos.Find(id);
            if (producto == null)
            {
```

```
                return HttpNotFound();
            }
            return View(producto);
        }

        public ActionResult Create()
        {
            return View();
        }

        [HttpPost]
        [ValidateAntiForgeryToken]
        public ActionResult Create([Bind(Include = "Id,Codigo,Nombre,PesoUnidad")]
Producto producto)
        {
            if (ModelState.IsValid)
            {
                db.Productos.Add(producto);
                db.SaveChanges();
                return RedirectToAction("Index");
            }

            return View(producto);
        }

        public ActionResult Edit(int? id)
        {
            if (id == null)
            {
                return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
            }
            Producto producto = db.Productos.Find(id);
            if (producto == null)
            {
                return HttpNotFound();
            }
            return View(producto);
        }

        [HttpPost]
        [ValidateAntiForgeryToken]
        public ActionResult Edit([Bind(Include = "Id,Codigo,Nombre,PesoUnidad")] Producto
producto)
        {
            if (ModelState.IsValid)
            {
                db.Entry(producto).State = EntityState.Modified;
                db.SaveChanges();
                return RedirectToAction("Index");
            }
            return View(producto);
        }

        public ActionResult Delete(int? id)
        {
            if (id == null)
            {
                return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
            }
```

```
            Producto producto = db.Productos.Find(id);
            if (producto == null)
            {
                return HttpNotFound();
            }
            return View(producto);
        }

        [HttpPost, ActionName("Delete")]
        [ValidateAntiForgeryToken]
        public ActionResult DeleteConfirmed(int id)
        {
            Producto producto = db.Productos.Find(id);
            db.Productos.Remove(producto);
            db.SaveChanges();
            return RedirectToAction("Index");
        }

        protected override void Dispose(bool disposing)
        {
            if (disposing)
            {
                db.Dispose();
            }
            base.Dispose(disposing);
        }
    }
}
```

3.2.9. **Controllers** - Importaciones Controller

```
using System;
using System.Collections.Generic;
using System.Configuration;
using System.Linq;
using System.Net.Http;
using System.Threading.Tasks;
using System.Web;
using System.Web.Mvc;
using Dominio;
using Newtonsoft.Json;
using System.Text;
using MVC.ViewModels;
using MVC.Models;

namespace MVC.Controllers
{
    public class ImportacionesController : Controller
    {
        private string url = ConfigurationManager.AppSettings["urlWebAPI"];

        [HttpGet]
        public ActionResult Index()
        {

            if (Session["cedula"] != null)
            {

                ImportacionesVM vm = new ImportacionesVM();

                List<Importacion> lista = FiltroApiImportaciones(url + "/getall");
                vm = new ImportacionesVM(lista);
                return View(vm);
            }
            else
            {
                return Redirect("/home/Index");
            }

        }

        [HttpPost]
        public ActionResult Index(ImportacionesVM vm)
        {

            if (Session["cedula"] != null)
            {
                List<Importacion> importaciones = new List<Importacion>();

                if (vm.Id == 0) //trae todo
                    importaciones = FiltroApiImportaciones(url + "/getall");
                else if (vm.Id == 1) // filtra por codigo
                    importaciones = FiltroApiImportaciones(url + "/getByProdCode/" +
vm.DatoFiltro);
                else if (vm.Id == 2) //filtra por rut
                    importaciones = FiltroApiImportaciones(url + "/getByClientRut/" +
vm.DatoFiltro);
```

```
                else if (vm.Id == 3) //filtra por nombre producto
                    importaciones = FiltroApiImportaciones(url + "/GetByProdName/" +
vm.DatoFiltro);
                else if (vm.Id == 4)//filtra los que deberían estar fuera de depósito
                    importaciones = FiltroApiImportaciones(url + "/GetByDate");

                vm.CargarFiltros();
                vm.Importaciones = importaciones;
                if (importaciones == null || importaciones.Count() == 0)
                    ViewBag.Error = "No se obtuvieron datos.";
                return View(vm);
            }
            else
            {
                return Redirect("/home/Index");
            }
        }

        public ActionResult Create()
        {
            if (Session["cedula"] != null)
            {
                return View();
            }
            else
            {
                return Redirect("/home/Index");
            }
        }

        [HttpPost]
        public ActionResult Create(NuevaImportacionViewModel nuevo)
        {
            if (Session["cedula"] != null)
            {
                Importacion nuevaImportacion = new Importacion();
                bool existe = false;

                using (MiContextoContext db = new MiContextoContext())
                {
                    var listaProds = db.Productos
                                .Join(db.Clientes, prod => prod.Cliente.Id, c => c.Id,
(prod, c) => new { prod, c })
                                .Where(e => e.prod.Codigo ==
nuevo.CodigoProd).FirstOrDefault();
                    if (listaProds != null)
                    {
                        nuevaImportacion.Producto = listaProds.prod;
                        existe = true;
                    }
                }
                if (existe)
                {
                    nuevaImportacion.Cantidad = nuevo.Cantidad;
                    nuevaImportacion.FechaIngreso = nuevo.FechaIngreso;
                    nuevaImportacion.FechaSalidaPrevista = nuevo.FechaSalidaPrevista;
                    nuevaImportacion.PrecioPorUnidad = nuevo.PrecioPorUnidad;
                    try
                    {
```

```
                    Uri uri = new Uri(url + "/Post/");

                    HttpClient cliente = new HttpClient();

                    Task<HttpResponseMessage> tarea = cliente.PostAsJsonAsync(uri,
nuevaImportacion);
                    tarea.Wait();

                    if (!tarea.Result.IsSuccessStatusCode)
                    {
                        ViewBag.Error = tarea.Result.StatusCode;
                        return View(nuevo);
                    }
                    else
                    {
                        return RedirectToAction("Index");
                    }
                }
                catch
                {
                    return View(nuevo);
                }
            }
            else
            {
                ViewBag.Mensaje = "No existe producto con tal codigo";
                return View(nuevo);
            }
        }
        else
        {
            return Redirect("/home/Index");
        }
    }

    [HttpGet]
    public ActionResult DarSalida(int id)
    {
        if (Session["rol"].ToString() == "deposito")
        {
            Importacion impor = null;

            Uri uri = new Uri(url + "/Get/" + id);

            HttpClient cliente = new HttpClient();

            Task<HttpResponseMessage> tarea = cliente.GetAsync(uri);
            tarea.Wait();

            if (tarea.Result.IsSuccessStatusCode)
            {
                Task<string> tarea2 = tarea.Result.Content.ReadAsStringAsync();
                tarea2.Wait();

                string json = tarea2.Result;
                impor = JsonConvert.DeserializeObject<Importacion>(json);
            }
            else
            {
```

```
                ViewBag.Error = tarea.Result.StatusCode;
            }
            return View(impor);
        }
        else
        {
            return Redirect("/home/Index");
        }
    }



    [HttpPost]
    public ActionResult DarSalida(Importacion importacion)
    {
        if (Session["rol"].ToString() == "deposito")
        {
            bool matriculaValida = importacion.ValidarMatricula();
            if (matriculaValida && importacion.Direccion != null)
            {
                try
                {
                    Uri uri = new Uri(url + "/Put/" + importacion.Id);

                    HttpClient cliente = new HttpClient();

                    importacion.FechaSalidaReal = DateTime.Now;
                    Int32.TryParse(Session["cedula"].ToString(), out int s);
                    importacion.CiUser = s;

                    Task<HttpResponseMessage> tarea = cliente.PutAsJsonAsync(uri,
importacion);
                    tarea.Wait();

                    if (!tarea.Result.IsSuccessStatusCode)
                    {
                        Task<string> tarea2 =
tarea.Result.Content.ReadAsStringAsync();
                        tarea2.Wait();

                        return View(importacion);
                    }
                    else
                    {
                        return RedirectToAction("Index");
                    }
                }
                catch
                {
                    return View(importacion);
                }
            }
            else
            {
                ViewBag.Mensaje = "La matricula no es valida.";
                return View(importacion);
            }
        }
        else
```

```
            {
                return Redirect("/home/Index");
            }
        }

        //METODO QUE SOLICITA A LA API LOS DATOS FILTRADOS
        public List<Importacion> FiltroApiImportaciones(string url)
        {
            List<Importacion> importaciones = new List<Importacion>();

            Uri uri = new Uri(url);

            HttpClient cliente = new HttpClient();

            Task<HttpResponseMessage> tarea = cliente.GetAsync(uri);
            tarea.Wait();

            if (tarea.Result.IsSuccessStatusCode)
            {
                Task<string> tarea2 = tarea.Result.Content.ReadAsStringAsync();
                tarea2.Wait();

                string json = tarea2.Result;
                importaciones = JsonConvert.DeserializeObject<List<Importacion>>(json);
            }
            else
            {
                ViewBag.Error = tarea.Result.StatusCode;
            }
            return importaciones;
        }

    }
}
```

3.3. ImportacionesWebApi2

3.3.1. **Controllers -** Importaciones Controller

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using System.Web;
using System.Web.Mvc;
using System.Data.Entity;
using System.Configuration;

namespace ImportacionWebAPI2.Controllers
{
    [System.Web.Http.RoutePrefix("api/Importacion")]

    public class ImportacionController : ApiController
    {
        public IHttpActionResult GetAll()
        {
            List<Importacion> lstImportacion = new List<Importacion>();

            try
            {
                using (MiContextoContext db = new MiContextoContext())
                {

                    lstImportacion = db.importaciones
                                       .Include(i => i.Producto)
                                       .Include(i => i.Producto.Cliente)
                                       .ToList();
                }
            }
            catch
            {
                return InternalServerError();
            }

            return Ok(lstImportacion);
        }

        [System.Web.Http.HttpGet()]
        [System.Web.Http.Route("getByProdCode/{prodCode}")]
        public IHttpActionResult GetByProdCode(string prodCode)
        {
            List<Importacion> lstImportacion = new List<Importacion>();
            try
            {
                using (MiContextoContext db = new MiContextoContext())
                {
                    lstImportacion = db.importaciones
                                       .Include(i => i.Producto)
                                       .Include(i => i.Producto.Cliente)
                                       .Where(i => i.Producto.Codigo == prodCode)
                                       .ToList();
```

```
        }
    }
    catch
    {
        return InternalServerError();
    }
    return Ok(lstImportacion);
}

[System.Web.Http.HttpGet()]
[System.Web.Http.Route("getByClientRut/{clientRut}")]
public IHttpActionResult GetByClientRut(string clientRut)
{
    List<Importacion> lstImportacion = new List<Importacion>();
    try
    {
        using (MiContextoContext db = new MiContextoContext())
        {
            lstImportacion = db.importaciones
                            .Include(i => i.Producto)
                            .Include(i => i.Producto.Cliente)
                            .Where(i => i.Producto.Cliente.Rut == clientRut)
                            .ToList();
        }
    }
    catch
    {
        return InternalServerError();
    }
    return Ok(lstImportacion);
}


[System.Web.Http.HttpGet()]
[System.Web.Http.Route("getByProdName/{prodName}")]
public IHttpActionResult GetByProdName(string prodName)
{
    List<Importacion> lstImportacion = new List<Importacion>();
    try
    {
        using (MiContextoContext db = new MiContextoContext())
        {
            lstImportacion = db.importaciones
                            .Include(i => i.Producto)
                            .Include(i => i.Producto.Cliente)
                            .Where(i => i.Producto.Nombre.Contains(prodName))
                            .ToList();
        }
    }
    catch
    {
        return InternalServerError();
    }
    return Ok(lstImportacion);
}

[System.Web.Http.HttpGet()]
[System.Web.Http.Route("getByDate/")]
public IHttpActionResult GetByDate()
```

```
        {
            List<Importacion> lstImportacion = new List<Importacion>();

            try
            {
                using (MiContextoContext db = new MiContextoContext())
                {
                    lstImportacion = db.importaciones
                                    .Include(i => i.Producto)
                                    .Include(i => i.Producto.Cliente)
                                    .Where(i => i.FechaSalidaPrevista < DateTime.Today &&
i.FechaSalidaReal == null)
                                    .ToList();
                }
            }
            catch
            {
                return InternalServerError();
            }
            return Ok(lstImportacion);
        }

        public IHttpActionResult Get(int id)
        {
            Importacion impor = null;
            try
            {
                using (MiContextoContext db = new MiContextoContext())
                {
                    impor = db.importaciones.Include(i => i.Producto)
                                    .Include(i => i.Producto.Cliente)
                                    .Where(i => i.Id == id)
                                    .FirstOrDefault();
                    if (impor == null) return NotFound();
                }
            }
            catch
            {
                return InternalServerError();
            }

            return Ok(impor);
        }

        public IHttpActionResult Put(int id, Importacion a_modificar)
        {
            if (id != a_modificar.Id) return BadRequest();

            try
            {
                using (MiContextoContext db = new MiContextoContext())
                {
                    int cantidad = db.importaciones.Count(p => p.Id == id);
                    if (cantidad == 0) return NotFound();

                    db.Entry(a_modificar).State = EntityState.Modified;
                    db.SaveChanges();

                    return Ok(a_modificar);
```

```
                }
            }
            catch (Exception ex)
            {
                return InternalServerError(ex);
            }
        }

        public IHttpActionResult Post(Importacion nuevo)
        {
            if (!ModelState.IsValid)
            {
                return BadRequest(ModelState);
            }

            try
            {
                using (MiContextoContext db = new MiContextoContext())
                {
                    db.importaciones.Add(nuevo);
                    db.SaveChanges();
                }
            }
            catch
            {
                return InternalServerError();
            }
            return Created("api/productos/" + nuevo.Id, nuevo);
        }
    }
}
```

3.3.2. **Models -** MiContexto Context

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Data.Entity;

namespace ImportacionWebAPI2
{
    public class MiContextoContext : DbContext
    {
        public DbSet<Producto> Productos { get; set; }
        public DbSet<Cliente> Clientes { get; set; }
        public DbSet<Importacion> importaciones { get; set; }
        public DbSet<Usuario> Usuarios { get; set; }

        public MiContextoContext() : base("ConexionWebAPI")
        {

        }
    }
}
```

3.3.3. **Models -** Cliente

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ImportacionWebAPI2
{
    [Table("Clientes")]
    public class Cliente
    {
        public int Id { get; set; }
        [Required]
        public string Rut { get; set; }
        [Required]
        public string Nombre { get; set; }

    }
}
```

3.3.4. **Models -** Usuario

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Text.RegularExpressions;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ImportacionWebAPI2
{
    [Table("Usuarios")]
    public class Usuario
    {
        public int Id { get; set; }
        [Required]
        public int Cedula { get; set; }
        [Required]
        public string Nombre { get; set; }
        [Required]
        public string Contrasenia { get; set; }
        [Required]
        public string Rol { get; set; }
    }
}
```

3.3.5. **Models -** Producto

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ImportacionWebAPI2
{
    [Table("Productos")]
    public class Producto
    {
        public int Id { get; set; }
        [Required]
        public string Codigo { get; set; }
        [Required]
        public string Nombre { get; set; }
        [Required]
        public float PesoUnidad { get; set; }
        public Cliente Cliente { get; set; }

    }
}
```

3.3.6. **Models -** Importacion

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace ImportacionWebAPI2
{
    [Table("Importaciones")]
    public class Importacion
    {
        public int Id { get; set; }
        [Required]
        public DateTime FechaIngreso { get; set; }
        [Required]
        public DateTime FechaSalidaPrevista { get; set; }
        public Producto Producto { get; set; }
        [Required]
        public int Cantidad { get; set; }
        [Required]
        public decimal PrecioPorUnidad { get; set; }
        public DateTime? FechaSalidaReal { get; set; }
        public string MatriculaCamion { get; set; }
        public string Direccion { get; set; }
        public int? CiUser { get; set; }

    }
}
```