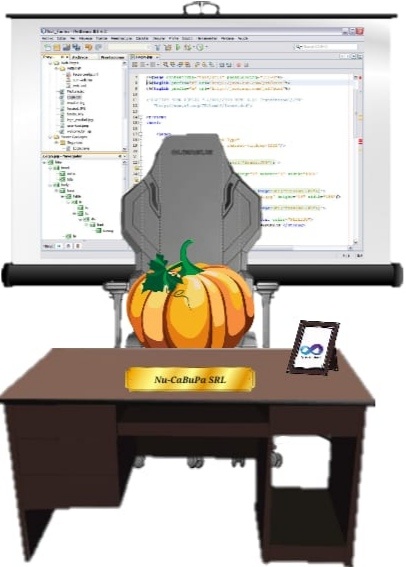
**Software de administración para la empresa**

**JVR Producciones**



Nu-CaBuPa. Software dedicado

Integrantes:

* Milagros Núñez
* Mateo Cabral
* ManuelBuslón
* Nahuel Pacheco

2020 Proyecto Integrador

Índice:

[Introducción: 3](#_Toc53172344)

[Fundamentación: 3](#_Toc53172345)

[Objetivos del proyecto: 4](#_Toc53172346)

[Nuestra empresa: 6](#_Toc53172347)

[FODA 8](#_Toc53172348)

[Organización de la empresa 9](#_Toc53172349)

[Recurso de la empresa: 10](#_Toc53172350)

[Estudio de costo y precio de venta: 11](#_Toc53172351)

[Análisis de la entrevista: 12](#_Toc53172352)

[Análisis de los requerimientos: 13](#_Toc53172353)

[Base de datos 15](#_Toc53172354)

[MER: 15](#_Toc53172355)

[Pasaje a tablas 16](#_Toc53172356)

[Consultas de creación MySql 17](#_Toc53172357)

[Creación de usuario 20](#_Toc53172358)

[Presentación del manual: 21](#_Toc53172359)

[Manual: 22](#_Toc53172360)

[Código: 27](#_Toc53172361)

[Modulos: 28](#_Toc53172362)

[Medidas de seguridad 67](#_Toc53172363)

[Configuración de red: 69](#_Toc53172364)

[Planos de la distribución: 69](#_Toc53172365)

[Conexión a la red: 69](#_Toc53172366)

[Configuración de router: 72](#_Toc53172367)

[Anexo 74](#_Toc53172368)

[S.R.L. 75](#_Toc53172369)

[Entrevista: 81](#_Toc53172370)

[Consultas de SQL 84](#_Toc53172371)

[Consultas de SQL 85](#_Toc53172372)

[Proyecto S.O 88](#_Toc53172373)

# Introducción:

El proyecto será desempeñado utilizando los conocimientos adquiridos en los años previos de estudio en el bachillerato de informática. También posee variedad de materias involucradas, las cuales son: Programación, Proyecto, Análisis y Desarrollo de Aplicaciones, Base de Datos, Formación Empresarial y Taller.

Nuestra empresa se definiría bajo el nombre “Nu-CaBuPa.SRL” y tendría su base en Uruguay, en el departamento de Salto. La misma se desarrollaría en el área tecnológica entorno a la producción de software dedicado desde fines administrativos hasta aplicaciones de entretenimiento.

El programa debe ser solo y únicamente para una empresa que necesite de este y pueda facilitarle tareas, las cuales serán indicadas por la empresa misma al igual que la organización del programa en caso de que el cliente desee especificarlo.

Nuestro software será diseñado para la empresa "JVR Producciones" y su propósito será enfocado en la administración y control de datos.

**Nombre:** Nu-CaBuPa.SRL



**Nombre social:**3M&N

**Dirección:**25 Av. Feliciano Viera (Esquina Uruguay y Brasil)

**Teléfono:**4733 5987 - +598 980 235 001

**E-mail:**[NuCaBuPa@gmail.com](mailto:NuCaBuPa@gmail.com)

**Socios:** Manuel Buslón, Nahuel Pacheco, Milagros Nuñez, Mateo Cabral

# Nuestra empresa:

Nu-CaBuPa.SRL, es una empresa de tecnología informática (Desarrollo de Software, reparación, mantenimiento y venta de Hardware) fundada en marzo del año 2020, que tiene como objetivo la satisfacción de las necesidades de servicios tecnológicos informáticos de los clientes.La empresa cuenta con varios factores que le permiten el alcance del objetivo, uno de ellos es la utilización de tecnologías avanzadas, junto a los conocimientos de los técnicos abarca los entornos tecnológicos de: Base de datos (MySQL) y los lenguajes de programación de Java y .Net.

La forma jurídica que está compuesta la empresa Nu-CaBuPa, es de una SRL (Sociedad de Responsabilidad Ilimitada) debido a que los socios cumplen con los siguientes puntos:

* Los socios no deberían tener embargos inscriptos.
* La Administración estará a cargo de los socios en la forma que ellos mismos acuerden, por ejemplo en forma conjunta o indistinta, etc.
* El giro puede ser el estándar y particularmente se incluye como principal, la actividad más importante que se va a realizar.
* Las Sociedades se entregan inscriptas en DGI y BPS, debiéndose establecer para BPS, cual es el socio que aporta y que va a tener actividad.
* Los aportes a BPS se pueden realizar por uno solo de los socios, que deberá ser el socio que declare actividad.
* Los socios serán responsables de las deudas sociales, hasta el máximo del capital social, respondiendo con sus propios bienes por las obligaciones de la sociedad.
* Los socios permanecerán en la sociedad y solo por cesión de sus cuotas sociales podrán dejar de participar en la misma. Las cesiones de cutas también se inscriben en el registro nacional de comercio y se publican en Diario Oficial y particular.

## Fundamentación:

A instancias de la necesidad por parte de las empresas de optimizar sus actividades han surgido proyectos de software de forma masiva. Cada uno adecuándose a las necesidades de un público específico o a un particular.

Este proyecto surge con los mismos fundamentos sobre los cuales emergen conjuntamente los demás, la posibilidad de volver más eficiente cualquier control e interacción entre los funcionarios y la gran cantidad de información a la que se enfrentan, en conjunto con las herramientas que estos deben utilizar continuamente.

Este software que será desarrollado, más que permitir una reducción en recursos y tiempo utilizado, lograrían tener una visión más amplia y objetiva de todas las actividades en las que se desarrolla la empresa. Se lo puede considerar directamente como una inversión realizada en el momento que se empieza el desarrollo, siguiendo esta visión, el software estará manteniendo una mejora continua en el tiempo sobre la utilización de recursos y gastos que gracias a este ya se volverían innecesarios.

En el sector de administración de recursos, lograría una automatización en la contabilidad de ganancias y un registro de las fuentes de ingreso en cada sector en la que se desarrolle. Permitiendo a dicho cliente un ahorro masivo en recursos tan valiosos como puede serlo el tiempo. Un ejemplo de este punto puede ser el acceso a variada cantidad de información almacenada ahora en servidores, pudiendo realizar consultas en un tiempo exponencialmente menor que el utilizado anteriormente. Una consecuencia directa de este ejemplo es la visibilidad de dicha información para el personal autorizado a acceder a ella, sin trabas externas a la empresa debido a las herramientas y la posibilidad de disminuir casi totalmente la perdida de información o destrucción de la misma por errores debido al método de almacenamiento empleado.

En conjunto al avance del tiempo y las tecnologías más actuales que se empleará a futuro el software contaría con la posibilidad de evolucionar y adaptarse al contexto, también contar con requisitos futuros en los que se deberá desempeñar. Dicha información almacenada y estructura en la codificación está regida por los paradigmas de la programación empleados en el mercado actual. Trayendo esto consigo una gran posibilidad de migración a diferentes software, remodelado de apartados en el programa y exportación de datos.

Como filosofía de este proyecto tenemos la utilización de una amplia gama de conocimientos, patrones de diseño y algoritmos estudiados y probados en entornos para el perfeccionamiento del programa en gran cantidad de casos que podrían ocurrir. Sumado a la correcta realización de pruebas y aplicación de diferentes herramientas en el desarrollo el software, contaría con el mejor desempeño y optimización adecuado a las necesidades del cliente. Todo esto permitiendo cambios futuros del mismo para obtener un mejor desempeño en las áreas que sean requeridas.

## Objetivos del proyecto:

Objetivos generales:

* Creación de un software administrativo, accesible para el usuario, que garantice una organización adecuada en la búsqueda y administración de datos.

Objetivos específicos:

* Diseñar una interfaz a medida y agradable para las necesidades del usuario.
* Crear un sistema de seguridad eficiente para la información ingresada.
* Conector adecuado entre la base de datos y el programa.
* Optimizar el tiempo de búsqueda de los datos solicitados.
* Software capacitado para la actualización de datos.
* Manejo adecuado de las publicidades emitidas por la radio “JVRProduciones”
* Software de bajos requerimientos.
* Utilización correcta y adecuada de los principios y bases de la ingeniería de software.

# 

# Misión:

La empresa Nu-CaBuPa es una empresa que se dedica a la creación de software, mantenimientos de equipos, servicios técnicos y ventas de insumos informáticos, que brinda e implementa nuevos métodos y nuevas tecnologías para ofrecer soluciones al desarrollo en el área informática de empresas y clientes, abarcando todo el territorio Uruguayo.

# Visión:

La empresa prende ser un referente internacional al momento que las personas y empresas busquen una solución informática. Para ello Nu-CaBuPa abordará las mejores soluciones y las formas de servicios que se ofrecen en la actualidad, brindando confianza y calidad al momento de elegirlos.

# Objetivos como empresa:

General:

Como empresa deseamos alcanzar un nivel de referencia internacional en cuanto a tecnología se refiere. Solucionar los problemas que el mundo tecnológico presente y crear nuevos estándares. Mantenernos actualizados a los últimos procedimientos informáticos e indirectamente iniciando nuevos avances debido a la necesidad del cambio. Siendo reconocida no por la velocidad, ni cantidad, si no calidad y el impacto.

Específicos:

-Mejorar la eficiencia de los programas y tecnologías utilizadas con cada nuevo proyecto.

-Fomentar un área de trabajo saludable y participativa entre todo el personal.

-Expandirse a otros mercados en el sector de software.

-Diseñar un modelo empresarial rentable para el futuro.

-Tener una estrategia de marketing en caso de ser necesario.

-Atraer más socios capacitados que ayuden a la empresa y traigan su propia visión con ellos.

# FODA

|  |  |  |
| --- | --- | --- |
|  | Interno | Externo |
| Ventaja: | Fortalezas:   * Conocimientos sobre desarrollo de aplicaciones y experiencia en el mismo. * Buena comunicación y organización interna. * Metas y objetivos claros. * Precios ajustados al producto y al trabajo requerido. * Clase y singularidad en los trabajos realizados. | Oportunidad:   * Facilidad de crecimiento como empresa debido al rubro. * Necesidad por parte de las empresas de aplicaciones y servicios. * Mercado objetivo claro. * Facilidad para distribución del producto. * Costo casi inexistente de producción de una vez finalizado el producto. |
| Desventajas: | Debilidades:   * No tener una fuerte relación previa con nuestros clientes. * Poca experiencia en el mercado. * Capital inicial inexistente. * Incapacidad de aceptar varios proyectos simultáneamente. | Amenazas:   * Gran competencia en el rubro. * Crisis mundial. * Económica en estado de ahorro y en estado de inversiones. * Trabajo y dependencia de terceros por el uso de sus productos. * Poco conocimiento del entorno sobre la factibilidad y posibilidad del software. |

# Organización de la empresa



Directiva de la empresa:

* Manuel Buslón
* Nahuel Pacheco
* Milagros Nuñez
* Mateo Cabral

Divisiones:

Programación:

* Manuel Buslón
* Nahuel Pacheco

Diseño:

* Nahuel Pacheco
* Milagros Nuñez

Documentación:

* *Carpeta*: Milagros Nuñez
* *Inglés:* Nahuel Pacheco, Mateo Cabral

# 

# Recurso de la empresa:

Recursos humanos:

Desarrolladores:

* Manuel Buslón
* Nahuel Pacheco
* Milagros Nuñez
* Mateo Cabral

Recursos materiales:

Fundamentales:

1. PC’s
2. Routers
3. Impresoras
4. Laptops

Extras:

1. Escritorios
2. Sillas
3. Monitores
4. Estantes
5. Repuestos de teclado y mouse

Recursos financieros:

Para el inicio de la empresa y la recaudación de los recursos materiales, los socios aportarán una suma de $USD2397,23, la cual se dividirá entre los cuatro, aportando cada uno el monto de $USD600,33.

Cada socio tomará la decisión de sacar un préstamo con la suma demandada para la recaudación.

# 

# Estudio de costo y precio de venta:

Presupuesto:

Por mes

|  |  |
| --- | --- |
| **Descripción** | **Importe** |
| Alquiler | $5.050 |
| UTE | $2.500 |
| ANTEL | $1.512 |
| OSE | $900 |
| Seguro | $4.000 |
| Salario | $42.496 |
| **Total** | $58.448 |
| **Total en dólares** | $USD 1377,27 |

Sueldo para desarrollador: $40.000

Sueldo hora/hombre = 40.000/30 = $1.333/8 = $166

Horas/mes = 64h

Sueldo nominal = 64x166 = 10.624

Aportes patronales = 10.624 x 0,075 + 10.624 x 0.05 + 10.624 x

0.00125 = 1.341

Sueldos = 10.624 X 4 = 42.496

Opciones de venta:

1- Software completo: $59.000 ($USD 1390,04)

2- Software completo + instalación: $60.000 ($USD 1413,60)

3- Software completo + instalación + manual: $60.500 ($USD 1425,38)

4- Software completo + instalación + servis por meses: $60.500 ($USD 1425,38)

5- Servis del Software: $700 ($USD 16,49)

## Análisis de la entrevista:

La entrevista a nuestro cliente JVR Producciones fue realizada por medio de la plataforma zoom el día 06 del 04 por motivos de una emergencia sanitaria. Esta fue planteada con anticipación, planeando con el equipo nuestra presentación y una metodología, la cual nos serviría para la obtención de datos y que a raíz de eso generaríamos más preguntas para la recolección de todos los requerimientos deseados por el cliente para poder realizar el software a medida.

Las primeras preguntas que fueron realizadas dieron un enfoque a la obtención de información sobre la empresa, para la realización del programa nuestro equipo tiene que estar en conocimientos sobre las actividades que realiza nuestro cliente y como estuvo administrando los datos hasta el momento, para poder identificar los métodos que este tenía. Por medio de la respuesta del cliente pudimos establecer que sus métodos de administración son bastante ineficientes, ya que estos demandan una buena organización por parte del mismo, además de que este método utilizado puede generar retrasos, errores y posibles inconvenientes.

Al momento de la obtención de la información el equipo comenzó a realizar preguntas relacionadas con el interés del cliente que se basaban en la gestión del programa de tv y una administración de control con relación a la radio, para poder comenzar a sugerir ciertas propuestas para la realización del programa dando sugerencias para su creación final.

A medida que se iba obteniendo información, se comenzó a consultar sobre los requisitos del programa, realizando preguntas especificas que estuvieran conectadas con la información que el cliente nos brindaba. Preguntando cómo es que se hacían ciertas actividades al momento de la administración, como es que se realizaban esas actividades y haciendo un énfasis en el interés del cliente, dando sugerencias y algunas explicaciones al momento que este pedía algo en específico.

Para concluir con la recolección de información necesaria, se comenzó a preguntar sobre el entendimiento que el cliente tenía sobre el área, además de preguntar sobre quienes estarían utilizando el programa cuando esté finalizado, para poder saber cómo crear el manual de usuario, además de conocer cuáles eran los equipos que este manejaba, ya que el programa será utilizado tanto en la radio como en la casa del cliente.

Al finalizar se concordaron ciertas especificaciones para el diseño que el cliente optó por propias.

## Análisis de los requerimientos:

|  |  |
| --- | --- |
| Definición de requisitos de usuario: | Especificación de requisitos del sistema: |
| El consumidor debe poder tener un control sobre los distintos usuarios y sus permisos. | El usuario poseería campos para seleccionar los distintos permisos que se tengan además de poder controlar los nombres y contraseñas (si posee los permisos necesarios). |

Usuarios:

Programa:

|  |  |
| --- | --- |
| El usuario podría administrar el precio de la cuota de mensual de cada programa. | Se podría elegir entra varias opciones de precios ya establecidos o agregar nuevos precios para otras categorías/programa. |
| El programa poseería fechas en las que se emiten y estas podrían ser controladas. | Se mostrarían en una lista las fechas ya establecidas y por medio de un botón se podría seleccionar en un calendario para agregar otra fecha.  Seleccionando una fecha de la lista se podría eliminar o modificar. |
| En el programa es posible que se muestren publicidades y se permitiría mantener el control sobre cuáles y hasta que fecha serían emitidas. | En una interfaz se podría seleccionar al programa que se desearía mostrar la publicidad y cuál de estas sería. |
| Se podría definir que un evento está relacionado con un programa. | Sería posible enlazarlo por medio de seleccionarlos de dos listas. |

Funcionarios:

|  |  |
| --- | --- |
| Se podría definir las funciones de cada funcionario. | A partir de seleccionar de un perfil del mismo dichas funciones (O agregarla si no se han creado con anterioridad). |
| El usuario puede establecer que funcionario y cumpliendo qué función tendría cada programa. | A base del funcionario y de la función se marca en que programa la ejercería. |

Publicidad:

|  |  |
| --- | --- |
| La publicidad tiene cuotas que se deben pagar y se registrarían las fechas. | Ingresando en el administrador de publicidades se pueden verificar las distintas cuotas a partir de una tabla, en la que se mostrar |
| Todas las publicidades pertenecen a alguna empresa y se debe poder tener registro de acual. | El administrador de empresa se mostraría las diferentes publicidades pertenecientes de dicha empresa en el orden en el cual se agregaron.  Se permitiría realizar una búsqueda por tema. |
| En la radio hay distintas tandas en las que pueden aparecer publicidades. | Permitiría administrar las tandas junto a las publicidades que aparecerían en la misma. Junto con un control de la fecha que empezarían a emitirse, junto a la fecha de finalización. |
| En los eventos se muestran publicidades y sería conveniente tener un registro de las mismas. | Mediante la unión a bese de la selección en listas (o buscando por temas) se relacionarían la publicidad con el evento. |

Videos:

|  |  |
| --- | --- |
| El usuario tendría el control de los videos y de las series, junto con la pertenencia de los mismos a las series. | Una vez agregado el video (el nombre y el tema) se podría seleccionar una serie existente a la cual pertenecería el mismo. |
| Sería posible que un evento tratara de un video de una serie y sería conveniente conocer los datos de ambos. | En una interfaz del control del evento habría una tabla con los videos relacionados permitiendo añadir, modificar y eliminar los mismos. |

# 

# Base de datos

## MER:

# 

|  |
| --- |
| Pasaje a tablas |
| Programa: { ID\_Programa , Fecha\_Finalizacion , Nombre\_Programa, Descripción}  FechaPrograma: { Fecha , Hora\_Inicio , Hora\_Fin, ID\_Programa}  ProgramaCuota: { ID\_Programa\_Cuota, ID\_Programa , Fecha\_Pago, Fecha\_Emision, Precio}  Publicidad: { ID\_Publicidad , Tema, ID\_Empresa}  Usuarios: {ID\_usuario, Nombre, Contrasena}  Acceso: {id\_acceso, Seccion, Tipo}  UsTieneAcceso:{id\_usuario, id\_acceso}  PMuestraPubli: {ID\_Publicidad, ID\_Programa , Fecha\_Finalizacion, Fecha\_Inicio}  PublicidadCuota: {ID\_Cuota, Fecha\_Pago, Fecha\_Emision, Precio, ID\_Publicidad}  Empresa:{ID\_Empresa , Nombre, Telefono, Mail}  Tanda: { Hora\_Inicio, Hora\_Fin}  AparecePubli: { ID\_Publicidad , Hora\_Inicio , Fecha\_Inicio, Fecha\_Finalizacion}  Evento: { ID\_Evento , Nombre, Descripcion}  EventoTrataDeV:{ID\_Evento, ID\_Video}  FechaEvento: { ID\_Evento, Fecha }  EventoMuestraPubli: { ID\_Evento , ID\_Publicidad }  EventoEsDeP: { ID\_Programa , ID\_Evento }  Funcionario: { ID\_Funcionario , Telefono, Nombre, Mail}  Funcion: { ID\_Funcion , Descripcion, Nombre}  TrabajaComo: { ID\_Funcionario, ID\_Funcion }  FunTrabaja: { ID\_Funcionario, ID\_Funcion, ID\_Programa , Fecha\_Inicio, Fecha\_Finalizacion}  Video: { ID\_Video , Contenido, Nombre, ID\_Serie, Fecha}  Serie: { ID\_Serie ,Fecha\_Finalizacion, Nombre} |

|  |
| --- |
| Consultas de creación MySql |
| CREATE TABLE `acceso` (  `id\_acceso` int(6) unsigned NOT NULL AUTO\_INCREMENT,  `seccion` varchar(8) NOT NULL,  `tipo` varchar(2) NOT NULL,  PRIMARY KEY (`id\_acceso`)  );  CREATE TABLE `aparecepubli` (  `ID\_Publicidad` int(6) unsigned NOT NULL,  `Hora\_Inicio` time NOT NULL,  `Fecha\_Inicio` date NOT NULL,  `Fecha\_Finalizacion` date DEFAULT NULL,  PRIMARY KEY (`ID\_Publicidad`,`Hora\_Inicio`,`Fecha\_Inicio`),  ForeignKEY (Hora\_Inicio) referencesTanda(Hora\_Inicio),  Foreign key (ID\_Publicidad) references Publicidad(ID\_Publicidad)  );  CREATE TABLE `empresa` (  `ID\_Empresa` int(6) unsigned NOT NULL AUTO\_INCREMENT,  `Nombre` varchar(64) NOT NULL,  `Telefono` varchar(16) DEFAULT NULL,  `Mail` varchar(64) DEFAULT NULL,  PRIMARY KEY (`ID\_Empresa`)  );  CREATE TABLE `publicidad` (  `ID\_Publicidad` int(6) unsigned NOT NULL AUTO\_INCREMENT,  `Tema` varchar(64) NOT NULL,  `ID\_Empresa` int(6) unsigned NOT NULL,  PRIMARY KEY (`ID\_Publicidad`),  Foreign KEY (ID\_Empresa) references Empresa(`ID\_Empresa`)  );  CREATE TABLE `evento` (  `ID\_Evento` int(6) unsigned NOT NULL AUTO\_INCREMENT,  `nombre` varchar(32) NOT NULL,  `descripcion` varchar(128) NOT NULL,  PRIMARY KEY (`ID\_Evento`)  );  CREATE TABLE `eventoesdep` (  `ID\_Programa` int(6) unsigned NOT NULL,  `ID\_Evento` int(6) unsigned NOT NULL,  PRIMARY KEY (`ID\_Programa`,`ID\_Evento`),  Foreign KEY (ID\_Evento) references Evento(`ID\_Evento`),  Foreign KEY (ID\_Programa) references Programa(`ID\_Programa `)  );  CREATE TABLE `eventotratadev` (  `ID\_Evento` int(6) unsigned NOT NULL,  `ID\_Video` int(6) unsigned NOT NULL,  PRIMARY KEY (`ID\_Evento`,`ID\_Video`),  Foreign KEY (ID\_Video)references Video(`ID\_Video`),  Foreign KEY (ID\_Evento) references Evento(ID\_Evento)  );  CREATE TABLE `eventomuestrapubli` (  `ID\_Evento` int(6) unsigned NOT NULL,  `ID\_Publicidad` int(6) unsigned NOT NULL,  PRIMARY KEY (`ID\_Evento`,`ID\_Publicidad`),  Foreign KEY (ID\_Publicidad) references Publicidad(`ID\_Publicidad`),  Foreign KEY (ID\_Evento) references Evento(ID\_Evento)  );  CREATE TABLE `fechaevento` (  `ID\_Evento` int(6) unsigned NOT NULL,  `fecha` datetime NOT NULL,  PRIMARY KEY (`ID\_Evento`,`fecha`),  Foreign key (ID\_Evento) references Evento(ID\_Evento)  );  CREATE TABLE `fechaprograma` (  `Fecha` date NOT NULL,  `Hora\_Inicio` time NOT NULL,  `Hora\_Fin` time NOT NULL,  `ID\_Programa` int(6) unsigned DEFAULT NULL,  PRIMARY KEY (`Fecha`,`Hora\_Inicio`),  Foreign KEY (ID\_Programa) references Programa(`ID\_Programa`)  );  CREATE TABLE `funcion` (  `ID\_Funcion` int(6) unsigned NOT NULL AUTO\_INCREMENT,  `descripcion` varchar(128) NOT NULL,  Nombrevarchar(16) not null  PRIMARY KEY (`ID\_Funcion`)  );  CREATE TABLE `funcionario` (  `ID\_Funcionario` int(6) unsigned NOT NULL AUTO\_INCREMENT,  `Telefono` varchar(16) DEFAULT NULL,  `nombre` varchar(48) NOT NULL,  `Mail` varchar(64) DEFAULT NULL,  PRIMARY KEY (`ID\_Funcionario`)  );  CREATE TABLE `trabajacomo` (  ID\_TrabajaComoint(6) unsigned not null auto\_increment,  `ID\_Funcionario` int(6) unsigned NOT NULL,  `ID\_Funcion` int(6) unsigned NOT NULL,  PRIMARY KEY (ID\_TrabajaComo),  Foreign KEY (ID\_Funcion) references Funcion(`ID\_Funcion`),  Foreign KEY (ID\_Funcionario) references Funcionario(ID\_Funcionario)  );  CREATE TABLE `programa` (  `ID\_Programa` int(6) unsigned NOT NULL AUTO\_INCREMENT,  `Nombre\_Programa` varchar(48) NOT NULL,  `Descripcion` varchar(128) DEFAULT NULL,  `Fecha\_Finalizacion` datetime DEFAULT NULL,  PRIMARY KEY (`ID\_Programa`)  );  CREATE TABLE `funtrabaja` (  `ID\_TrabajaComo` int(6) unsigned NOT NULL,  `ID\_Programa` int(6) unsigned NOT NULL,  `fecha\_inicio` date NOT NULL,  `fecha\_finalizacion` date DEFAULT NULL,  PRIMARY KEY (`ID\_TrabajaComo`,`ID\_Programa`,`fecha\_inicio`),  Foreign KEY (ID\_TrabajaComo) references Funcion(`ID\_Funcion`),  Foreign KEY (ID\_Programa)references Programa(`ID\_Programa`)  );  CREATE TABLE `pmuestrapubli` (  `ID\_Publicidad` int(6) unsigned NOT NULL,  `ID\_Programa` int(6) unsigned NOT NULL,  `Fecha\_Inicio` date NOT NULL,  `Fecha\_Finalizacion` date DEFAULT NULL,  PRIMARY KEY (`Fecha\_Inicio`,`ID\_Programa`,`ID\_Publicidad`),  Foreign KEY (ID\_Programa) references Programa(`ID\_Programa`),  Foreign KEY (ID\_Publicidad) references Publicidad(`ID\_Publicidad`)  );  CREATE TABLE `programacuota` (  `ID\_Programa\_Cuota` int(6) unsigned NOT NULL,  `ID\_Programa` int(6) unsigned NOT NULL,  `Fecha\_Pago` date DEFAULT NULL,  `Precio` decimal(5,2) unsigned NOT NULL,  `Fecha\_Emision` date NOT NULL,  PRIMARY KEY (`ID\_Programa\_Cuota`),  Foreign KEY (ID\_Programa) references Programa(`ID\_Programa`)  );  CREATE TABLE `publicidadcuota` (  `ID\_PublicidadCuota` int(6) unsigned NOT NULL AUTO\_INCREMENT,  `Fecha\_Pago` date DEFAULT NULL,  `Precio` int(6) unsigned NOT NULL,  `ID\_Publicidad` int(6) unsigned NOT NULL,  `Fecha\_Emision` date NOT NULL,  PRIMARY KEY (`ID\_PublicidadCuota`),  Foreign KEY (ID\_Publicidad) references Publicidad(`ID\_Publicidad`)  );  CREATE TABLE `serie` (  `ID\_Serie` int(6) unsigned NOT NULL AUTO\_INCREMENT,  `fecha\_finalizacion` date DEFAULT NULL,  `nombre` varchar(48) NOT NULL,  PRIMARY KEY (`ID\_Serie`)  );  CREATE TABLE `tanda` (  `Hora\_Inicio` time NOT NULL,  `Hora\_Fin` time NOT NULL,  PRIMARY KEY (`Hora\_Inicio`)  );  CREATE TABLE `ustieneacceso` (  `id\_usuario` int(6) unsigned NOT NULL,  `id\_acceso` int(6) unsigned NOT NULL,  PRIMARY KEY (`id\_usuario`,`id\_acceso`),  Foreign KEY (id\_acceso) acceso(`id\_acceso`)  );  CREATE TABLE `usuarios` (  `id\_usuario` int(6) unsigned NOT NULL AUTO\_INCREMENT,  `nombre` varchar(16) NOT NULL,  `contrasena` varbinary(256) NOT NULL,  PRIMARY KEY (`id\_usuario`)  );  CREATE TABLE `video` (  `ID\_Video` int(6) unsigned NOT NULL AUTO\_INCREMENT,  `contenido` varchar(128) NOT NULL,  `nombre` varchar(48) NOT NULL,  `ID\_Serie` int(6) unsigned DEFAULT NULL,  `fecha` date DEFAULT NULL,  PRIMARY KEY (`ID\_Video`),  Foreign KEY (ID\_Serie) Serie(`ID\_Serie`)  ); |

|  |
| --- |
| Creación de usuario |
| CREATE USER 'Admin'@'localhost' IDENTIFIED BY 'jvrp';  GRANT USAGE ON JVRPDATABASE.\* TO 'Admin'@'localhost';  GRANT ALL PRIVILEGES ON JVRPDATABASE.\*TO 'Admin'@'localhost';  FLUSH PRIVILEGES;  CREATE USER 'User'@'localhost' IDENTIFIED BY 'usuario';  GRANT USAGE ON JVRPDATABASE.\* TO 'User'@'localhost';  GRANT SELECT, INSERT, UPDATE, DELETEON JVRPDATABASE.\*TO 'User'@'localhost';  FLUSH PRIVILEGES;  CREATE USER 'Muestra'@'localhost' IDENTIFIED BY 'Default';  GRANT USAGE ON JVRPDATABASE.\*TO 'Muestra'@'localhost';  GRANT select ON JVRPDATABASE.\* TO 'Muestra'@'localhost';  FLUSH PRIVILEGES; |

# Presentación del manual:

* Justificación del manual bilingüe.

Un manual bilingüe tiene ventajas ya que normalmente se lo crea en el idioma que el producto fue creado y en un idioma universal (ingles) para que todos los usuarios estén al mismo nivel de compresión del funcionamiento de dicho producto.

El usuario que tenga la posibilidad de acceder a un manual bilingüe ayuda a la compresión del programa sin que este sepa el idioma que originalmente el manual fue redactado. Además de ayudar a la compresión de cada función que el programa brinda en su idioma, sin tener la necesidad de estar traduciendo. También el manual bilingüe ayuda a una mejor comunicación y comprensión de dos usuarios que no tengan el mismo idioma y estén interesados en el programa.

* Formas de crear un manual bilingüe accesible.

Para la creación del manual bilingüe hay que realizar ciertos pasos ayudándonos con algunas herramientas que nos permitirán una mejor accesibilidad para el usuario.

Primero hay que identificar a que tipos de usuarios irá dirigido el manual, en el caso de ser un manual bilingüe hay que tener en cuenta los dos idiomas en los cuales será redactado, en nuestro caso español e inglés.

Al momento de la identificación de los usuarios que tendrán conocimientos sobre el manual, hay que tener en cuenta la capacitación que estos tienen, normalmente es mejor crear el manual a un nivel de usuario estándar, ya que puede haber un usuario que no tenga una gran capacitación.

El formato y el estilo del manual debe ser apropiado, debe tener un formato sencillo pero coherente, el estilo de la escritura puede ser formal (si está dirigido para usuarios ya con conocimientos) o semi-formal. El seguimiento de la escritura debe ser fácil para que el usuario pueda seguir y encontrar con facilidad la información necesaria, además de implementar imágenes de guía. También se diferencia entre los títulos, subtítulos, advertencias, cuidados, etc.

El estilo y el formato del manual son elegidos por los miembros del equipo que se pondrán de acuerdo luego reconocer el tipo de usuario al cual este manual será dirigido. A raíz de eso se comenzará con la creación del mismo, preferentemente se sugiere ir creando el manual en un idioma para luego traducirlo.

Debido a que el destinatario del manual poseería conocimientos previos sobre el uso de dispositivos y herramientas en relación a la informática y la comunicación, nuestro manual omitirá explicaciones introductorias al uso de sistemas. Esto nos lleva a describir exclusivamente la organización, acceso, permisos y posibilidades implementadas dentro del software.

Primero se comienza con identificación de las partes que tendrá el mismo, especificando que irá primero antes de la inicialización del programa, que debe hacer el usuario para que este inicie correctamente, adjuntando imágenes de guía y el listado de pasos. Luego se deberá especificar que hará el usuario luego de haberlo hecho, en nuestro caso que es lo primero que ve el usuario al ingresar al programa, explicando que es lo que el usuario está viendo y que es lo que debe hacer. El manual irá especificando el uso de cada interfaz que el usuario está visualizando, con sus respectivos títulos, sus imágenes de guía, pasos a seguir, etc.

Luego de la realización de la especificación del programa y su funcionamiento, se deberá realizar un índice, para una mejor ayuda al momento de buscar alguna información concreta que el usuario desee, sin que este tenga que estar leyendo todo el manual para encontrarla.

Al tener finalizado el manual en un idioma se comienza con la traducción de este, teniendo en cuenta la estructura y organización, haciendo el mismo procedimiento que se realizó al comienzo.

Para poder finalizar la realización del manual bilingüe se exporta el mismo en los formatos pdf, docx y txt (sin imágenes en este último formato), además para una mejor distribución se lo imprime. En conjunto poseería un archivo del manual en formato de audio.

## Manual:

* 1. Primera ventana:



El primer formulario que se observa requiere el ingreso del usuario en el primer campo en blanco, la contraseña en el segundo.

Luego de esto y si los datos son válidos se podría abrir la ventana de configuración o entrar al sistema.

* 1. Ventana de configuración



Lo primero que se notará es que posee varios campos de texto para ingresar los datos de la conexión, con el correspondiente tipo de dato que se requiere. Posee la opción de restablecer para volver a los valores por defecto, o guardar los datos ingresados en el momento.

También posee en el panel una pestaña para administrar los usuarios (solo está disponible si el usuario tiene dicho permiso).

* 1. Pestaña de usuario



En la pestaña de usuario se puede observar una sección para la edición y otra para la creación de usuarios.

En la sección de creación hay un campo para el nombre y la contraseña para el usuario a crear, además de dos botones uno para crear y otro para limpiar.

En la sección de edición se observa una tabla con el id y el nombre de usuario, y al seleccionar alguno este se cargará en la sección de edición, pudiendo modificar el nombre y/o la contraseña. Siguiente a esto se puede aplicar los cambios a los usuarios o eliminar dicho usuario.

1. Pantalla de inicio





* 1. La región izquierda de la interfaz principal está dedicada a las tandas.

La división superior se puede visualizar las próximas tandas, enseñando la hora de comienzo y la hora de final. Mientras que la división inferior se visualizan los temas de las publicidades, junto a la fecha, la cual el usuario pude buscar una en especifico.



* 1. La región derecha de la interfaz, en la división superior está dedicada a los próximos eventos, mostrando el nombre y la fecha del mismo. En la división inferior hay un cuadro de notas para que el usuario pueda escribir algún recordatorio.
  2. En la región central encontramos dos pestañas: programas y multimedia.



Pestaña programas: Se visualizan los próximos programas, mostrando la hora de inicio, la hora final, el programa y el estado del mismo. El usuario podrá seleccionar una fecha en específico si desea ver qué programas se emitirán.

Seleccionando un programa, a su derecha se muestran los datos del mismo, los cuales son: la descripción, los funcionarios que trabajan en él y los temas de las publicidades.



Pestaña multimedia: Esta pestaña está dedicada para la búsqueda de vídeos y series.

## Código:

Ingreso genérico de administrador:

Generador\_DB.vb

ImportsMySql.Data.MySqlClient

PublicClassGenerador\_DB

Dim dt AsNewDataTable

DimnTablaAsString = ""

PrivateSubGenerador\_DB\_Load(sender AsObject, e AsEventArgs) HandlesMyBase.Load

cbTablas.DataSource = ESQLSelect("show tables")

cbTablas.ValueMember = "Tables\_in\_jvrpdatabase"

nTabla = "acceso"

EndSub

PrivateSubbtnCTabla\_Click(sender AsObject, e AsEventArgs) HandlesbtnCTabla.Click

ModLog.Guardar(nTabla)

ForiAsInteger = dgvTabla.Columns.Count - 1 To 0 Step -1

dgvTabla.Columns.RemoveAt(i)

Next

dt = ESQLSelect("describe " + nTabla)

dgvTabla.DataSource = dt

ForiAsInteger = dgvTabla.Columns.Count - 1 To 1 Step -1

dgvTabla.Columns.RemoveAt(i)

Next

dgvTabla.Columns(0).ReadOnly = True

dgvTabla.Columns.Add("Datos", "Datos")

dgvTabla.Columns.Add("Mantener", "Mantener")

EndSub

PrivateSubbtnInsertar\_Click(sender AsObject, e AsEventArgs) HandlesbtnInsertar.Click

Dim Consulta AsString() = GeneradorDeString()

ModConector.ISQL(nTabla, Consulta(0), Consulta(1))

Limpiador()

EndSub

PublicSubLimpiador()

ForiAsInteger = 0 TodgvTabla.Rows.Count - 1

IfIsNothing(dgvTabla.Rows(i).Cells(2).Value) Then

dgvTabla.Rows(i).Cells(1).Value = ""

EndIf

Next

EndSub

PrivateFunctionGeneradorDeString() AsString()

DimTextoAsString() = {"", ""}

ForiAsInteger = 0 TodgvTabla.Rows.Count - 1

Ifdt.Rows(i).Item(3).ToString() = "no"OrNotString.IsNullOrWhiteSpace(dgvTabla.Rows(i).Cells(1).Value) Then

Texto(0) += dgvTabla.Rows(i).Cells(0).Value.ToString() + ","

Texto(1) += "'" + dgvTabla.Rows(i).Cells(1).Value.ToString() + "',"

EndIf

Next

Texto(0) = Texto(0).Remove(Texto(0).Length - 1)

Texto(1) = Texto(1).Remove(Texto(1).Length - 1)

ModLog.Guardar(Texto(0) + Texto(1))

Return Texto

EndFunction

PrivateSubLimpiar\_Click(sender AsObject, e AsEventArgs) HandlesLimpiar.Click

Limpiador()

EndSub

PrivateSubcbTablas\_SelectedValueChanged(sender AsObject, e AsEventArgs) HandlescbTablas.SelectedValueChanged

IfnTabla<>""Then

nTabla = cbTablas.SelectedValue

EndIf

EndSub

EndClass

# Modulos:

ModCodificador.vb:

ImportsSystem.Text

ImportsSystem.Security.Cryptography

'Se utiliza para encriptar y desencriptar datos de tipo string

'Usa una key y un vector de incializacion, que llamaremos key maestra

'Por mas información de esta ultima se puede buscar en:

'https://es.wikipedia.org/wiki/Vector\_de\_inicializaci%C3%B3n

ModuleModCodificador

PrivateKeyMaestraAsString = "rpaSPvIvVLlrcmtzPU9/c67Gkj7yL1S5"

Private Key AsString = "12345678"

PrivateIV() AsByte

PrivateEncryptionKey() AsByte

Privatebuffer() AsByte

Private des AsTripleDESCryptoServiceProvider

PublicSubEKey(ByValNKeyAsString)

IfNotString.IsNullOrEmpty(NKey) Then

Key = NKey

EndIf

EndSub

PublicSubActualizar()

IV = ASCIIEncoding.ASCII.GetBytes(Key) 'La clave debe ser de 8 caracteres

EncryptionKey = Convert.FromBase64String(KeyMaestra) 'No se puede alterar la cantidad de caracteres pero si la clave

des = NewTripleDESCryptoServiceProvider

des.Key = EncryptionKey

des.IV = IV

EndSub

PublicSubEKeyMaestra(ByValNKeyMaestraAsString)

IfNotString.IsNullOrEmpty(NKeyMaestra) Then

KeyMaestra = NKeyMaestra

EndIf

EndSub

PublicFunctionGKey() AsString

Return Key

EndFunction

PublicFunctionGKeyMaestra() AsString

ReturnKeyMaestra

EndFunction

PublicFunctionEncriptar(ByValTextoAsString) AsString

'convierte el texto a bytes utilizando el formato utf8

buffer = Encoding.UTF8.GetBytes(Texto)

'Crea un objeto de encriptacion, luego crea un valor hash con el buffer y utiliza todo su largo, comenzando en el byte 0

Return Convert.ToBase64String(des.CreateEncryptor().TransformFinalBlock(buffer, 0, buffer.Length()))

EndFunction

PublicFunctionDesencriptar(ByValTextoAsString) AsString

'convierte el texto encriptado a bytes

buffer = Convert.FromBase64String(Texto)

'desencripta el texto(ahora en array de bytes) y utiliza el formato utf8

Return Encoding.UTF8.GetString(des.CreateDecryptor().TransformFinalBlock(buffer, 0, buffer.Length()))

EndFunction

EndModule

ModConector.vb:

Imports MySql.Data.MySqlClient

Imports System.Data

Imports MySql.Data

Module ModConector

Private Debug As Boolean = True

Private conn As New MySqlConnection

Private connStr As String

Private Address, User, Database, Port, Pass As String

Private UsuarioID As Integer

Private Usuario, Password As String

Private objCmd As New MySqlCommand

Private ds As New DataSet

Private dt As New DataTable

#Region "GetDirection"

Public Function GDT(ByValNombre As String) As DataTable

Return ds.Tables(Nombre)

End Function

Public Function GUsuarioID() As Integer

Return UsuarioID

End Function

Public Sub BorrarUsuario()

Usuario = ""

Password = ""

UsuarioID = Nothing

End Sub

Public Function GDebug() As Boolean

Return Debug

End Function

Public Function GAddress() As String

Return Address

End Function

Public Function GUser() As String

Return User

End Function

Public Function GDatabase() As String

Return Database

End Function

Public Function GPort() As String

Return Port

End Function

Public Function GPass() As String

Return Pass

End Function

#End Region

#Region "Constructor"

Public Sub Crear(ByValRAddress As String, ByValRPort As String, ByValRDatabase As String, ByValRUser As String, ByValRPass As String)

If Not String.IsNullOrEmpty(Address) Then

Address = RAddress

Else

Address = "localhost"

End If

If Not String.IsNullOrEmpty(Address) Then

User = RUser

Else

User = "root"

End If

If Not String.IsNullOrEmpty(Address) Then

Database = RDatabase

Else

Database = "JVRPDB"

End If

If Not String.IsNullOrEmpty(Address) Then

Port = RPort

Else

Port = "3306"

End If

If Not String.IsNullOrEmpty(Address) Then

Pass = RPass

Else

Pass = "root"

End If

End Sub

Public Sub EAddress(ByValNAddress As String)

Address = NAddress

End Sub

Public Sub EPort(ByValNPort As String)

Port = NPort

End Sub

Public Sub EUser(ByValNUser As String)

User = NUser

End Sub

Public Sub EPass(ByValNPass As String)

Pass = NPass

End Sub

Public Sub EDatabase(ByValNDatabase As String)

Database = NDatabase

End Sub

#End Region

#Region "Conectar"

Public Sub Inicio()

Try

connStr = "DataSource=" + Address + "; Port=" + Port + "; Database=" + Database + "; Uid=" + User + "; Pwd=" + Pass + "; CharSet=utf8"

'connStr = "Server=" + Address + "; Database=" + Database + "; Uid=" + User + "; Pwd=" + Pass + "; CharSet=utf8mb4"

'connStr = "Server=" + Address + "; Database=" + Database + "; Uid=" + User + "; Pwd=root; CharSet=utf8mb4"

conn = New MySqlConnection(connStr)

ModLog.Guardar(connStr)

If Not conn.Ping Then

conn.Open()

End If

Catch ex As Exception

MessageBox.Show(ex.ToString())

End Try

End Sub

Public Sub desconectar()

conn.Close()

End Sub

Public Function RConexion() As MySqlConnection

Return conn

End Function

#End Region

#Region "Interpretar"

Public Function ESQL(ByValsql As String) As Boolean

Dim conT = New MySqlConnection(connStr)

conT.Open()

ModLog.Guardar(sql)

Try

objCmd = New MySqlCommand(sql, conn)

objCmd.Prepare()

objCmd.ExecuteNonQuery()

Catch ex As Exception

MessageBox.Show(ex.ToString)

Return False

End Try

conT.Close()

Return True

End Function

Public Function ESQLSelect(ByValsql As String) As DataTable

Dim dt As New DataTable

Dim sqladapter As MySqlDataAdapter

Dim conT = New MySqlConnection(connStr)

conT.OpenAsync()

ModLog.Guardar(sql)

Try

objCmd = New MySqlCommand(sql, conT)

objCmd.Prepare()

sqladapter = New MySqlDataAdapter(objCmd)

sqladapter.FillAsync(dt)

Catch ex As Exception

MessageBox.Show(ex.ToString)

Return Nothing

End Try

conT.CloseAsync()

Return dt

End Function

Public Function ESQLSelect(ByValobjCmd As MySqlCommand, ByValguardar As Boolean) As DataTable

dt = New DataTable

Dim sqladapter As MySqlDataAdapter

ModLog.Guardar(objCmd.CommandText)

Try

sqladapter = New MySqlDataAdapter(objCmd)

sqladapter.Fill(dt)

If guardar Then

If ds.Tables.Contains(dt.TableName) Then

ds.Tables.Remove(dt.TableName)

End If

ds.Tables.Add(dt)

End If

Catch ex As Exception

MessageBox.Show(ex.ToString)

Return Nothing

End Try

sqladapter.Dispose()

Return dt

End Function

#End Region

#Region "Comandos"

Public Sub BSQL(ByValnTabla As String, ByVal Condition As String)

ESQL("DELETE FROM " + nTabla + " WHERE " + Condition)

'ModLog.Guardar("DELETE FROM " + nTabla + " WHERE " + Condition)

End Sub

Public Sub ISQL(ByValnTabla As String, ByVal Column As String, ByVal Data As String)

ESQL("Insert into " + nTabla + " ( " + Column + " ) values (" + Data + " )")

End Sub

Public Function SSQL(ByValnColumn As String, ByValnTabla As String, ByVal Condition As String) As DataTable

If ds.Tables.Contains(nTabla) Then

dt = New DataTable

dt.Columns.Add(ds.Tables(nTabla).Columns(nColumn))

Return dt

Else

Return ESQLSelect("select " + nColumn + " FROM " + nTabla + " WHERE " + Condition)

End If

End Function

Public Function PSQL(ByValnColumn As String, ByValnTabla As String, ByVal Condition As String) As String

Return "select " + nColumn + " FROM " + nTabla + " WHERE " + Condition

End Function

Public Sub USQL(ByValnTabla As String, ByVal Orden As String, ByVal Condition As String)

ESQL("update " + nTabla + " set " + Orden + " WHERE " + Condition)

End Sub

#End Region

#Region "Usuarios"

Public Function AUsuarios(ByValactualizar As Boolean) As DataTable

If ds.Tables.Contains("usuarios") And Not actualizar Then

dt = New DataTable()

dt.Columns.Add(ds.Tables("usuarios").Columns("Nombre Usuarios"))

Return dt

Else

ReturnDevolverTabla(PSQL("id\_usuario as 'ID', nombre as 'Nombre Usuarios'", "usuarios", "True"))

End If

End Function

Public Function BUsuario(ByValnombre As String, ByValcontraseña As String) As Boolean

Try

objCmd = New MySqlCommand(PSQL("id\_usuario", "usuarios as User", "nombre = @nombre AND contrasena = AES\_ENCRYPT(@contrasena,sha2(@key,256))"), conn)

objCmd.Parameters.Add("@nombre", MySqlDbType.VarChar).Value = nombre

objCmd.Parameters.Add("@contrasena", MySqlDbType.VarChar).Value = contraseña

objCmd.Parameters.Add("@key", MySqlDbType.VarChar).Value = ModCodificador.GKey

objCmd.Prepare()

Dim dt As DataTable = ESQLSelect(objCmd, False)

If Not IsNothing(dt) Then

If dt.Rows.Count = 0 Then

MessageBox.Show("Contraseña o UsuarioIncorrecto")

Else

Usuario = nombre

Password = contraseña

UsuarioID = Integer.Parse(dt.Rows(0)("id\_usuario"))

Return True

End If

End If

Catch e As Exception

MessageBox.Show(e.ToString)

End Try

Return False

End Function

Public Sub IUsuario(ByValnombre As String, ByValcontraseña As String)

ISQL("usuarios", "nombre , contrasena", "'" + nombre + "', AES\_ENCRYPT('" + contraseña + "',sha2('" + ModCodificador.GKey + "',256))")

End Sub

#End Region

#Region "Main"

Public Function DevolverTabla(ByValDatos As String) As DataTable

Try

Dim dt As DataTable = ESQLSelect(Datos)

If Not IsNothing(dt) Then

If Not dt.Rows.Count = 0 Then

Return dt

End If

End If

Catch e As Exception

MessageBox.Show(e.ToString)

End Try

Return Nothing

End Function

#Region "Actualizar"

Public Function APrograma(fecha As Date) As DataTable

Return DevolverTabla(PSQL("p.id\_programa, time\_format(hora\_inicio, '%H:%i') as 'Inicio', time\_format(hora\_fin, '%H:%i') as 'Final', Nombre\_programa as 'Programa'", "fechaprograma f inner join programa p on f.id\_programa=p.id\_programa", "fecha = '" + Format(fecha, "yyyy-MM-dd") + "'"))

End Function

Public Function AFPrograma(idPrograma As Integer) As DataTable

Return DevolverTabla(PSQL("Nombre, Telefono", "programa p inner join funtrabaja f on f.id\_programa=p.id\_programa inner join funcionario ff on ff.id\_funcionario = f.id\_funcionario", "p.id\_programa = '" + idPrograma.ToString + "'"))

End Function

Public Function APublicidad(Fecha As Date, Hora As TimeSpan) As DataTable

Return DevolverTabla(PSQL("Tema", "publicidad p inner join aparecepubli a on p.id\_publicidad=a.id\_publicidad inner join tandat ont.Hora\_Inicio = a.hora\_inicio", "a.fecha\_inicio<= '" + Format(Fecha, "yyyy-MM-dd") + "' and a.fecha\_finalizacion>= '" + Format(Fecha, "yyyy-MM-dd").ToString + "' and t.hora\_inicio = '" + Hora.ToString + "'"))

End Function

Public Function APPublicidad(Fecha As Date, idPrograma As Integer) As DataTable

Return DevolverTabla(PSQL("Tema", "programa p inner join pmuestrapubli pp on p.id\_programa=pp.id\_programa inner join publicidadppp on pp.id\_publicidad = ppp.id\_publicidad", "pp.fecha\_inicio<= '" + Format(Fecha, "yyyy-MM-dd") + "' and pp.fecha\_finalizacion>= '" + Format(Fecha, "yyyy-MM-dd").ToString + "' and pp.id\_programa = '" + idPrograma.ToString + "'"))

End Function

Public Function AEventos() As DataTable

Return DevolverTabla(PSQL("e.id\_Evento, DATE\_FORMAT(Fecha,'%d/%m/%Y') as Fecha, Nombre", "evento e inner join fechaevento f on f.id\_evento=e.id\_evento", "f.fecha>= now()"))

End Function

Public Function ATandas() As DataTable

Return DevolverTabla(PSQL("time\_format(hora\_inicio, '%H:%i') as 'Inicio', time\_format(hora\_fin, '%H:%i') as 'Final'", "tanda", "(hora\_inicio<= curtime() and hora\_fin>= curtime()) or hora\_inicio>= curtime()"))

End Function

Public Function ADPrograma(idPrograma As Integer) As String

Dimdt As DataTable = DevolverTabla(PSQL("Descripcion", "programa", "id\_programa = '" + idPrograma.ToString + "'"))

Return dt.Rows(0)(0).ToString

End Function

#End Region

#End Region

End Module

ModInicializador.vb

' El modulo trabaja como union y acceso entre los formularios

ModuleModInicializador

Public Cancelar AsString = ""

PublicfrmPrinAsfrmPrincipal

PublicSubConfiguracion()

DimfrmConfigAsNewfrmConfiguracion

' Muestra el formulario de configuracion

frmConfig.ShowDialog()

EndSub

PublicSubGeneradores()

Dim Gen AsNewGenerador\_DB

' Muestra el formulario de configuracion

Gen.ShowDialog()

EndSub

PublicSubPrincipal()

frmPrin = NewfrmPrincipal

' Muestra el formulario de configuracion

frmPrin.Show()

EndSub

EndModule

ModLog.vb

Imports System.IO

ModuleModLog

':::Ruta donde crearemos nuestro archivo txt

Private ruta AsString = "..\User\"

':::Nombre del archivo

PrivatearchivoAsString = "Log.txt"

PublicSubGuardar(ByValAccionesAsString)

DimescribirAsNewStreamWriter(ruta&archivo, True)

escribir.WriteLine(Now.ToString + " " + My.Computer.Name + " ' " + Acciones + " ")

escribir.Close()

EndSub

EndModule

ModTablas.vb

ModuleModTablas

Publicdt\_programaAsDataTable

Publicdt\_dprogramaAsDataTable

Publicdt\_eventoAsDataTable

Publicdt\_tandasAsDataTable

Publicdt\_publiAsDataTable

Publicdt\_PpubliAsDataTable

Publicdt\_VideoAsDataTable

Publicdt\_SerieAsDataTable

Publicdt\_EmpresaAsDataTable

PublicFunctionCargarID(ByRefTablaAsDataTable, ByRefDgvAsDataGridView) AsInteger

If (NotIsNothing(Tabla)) Then

ReturnTabla.Rows(Dgv.SelectedRows(0).Index).Item(0).ToString

EndIf

Return -1

EndFunction

PublicFunctionCargarID(ByRefTablaAsDataTable, ByRefDgvAsDataGridView, ByValNumCol() AsByte) AsString()

If (NotIsNothing(Tabla)) Then

Dimres(NumCol.Length - 1) AsString

Dim j AsByte = 0

ForEachiAsByteInNumCol

res(j) = Tabla.Rows(Dgv.SelectedRows(0).Index).Item(i).ToString

j += 1

Next

Return res

EndIf

Dimerr(0) AsString

Returnerr

EndFunction

PublicSubClickCheck(ByRefDgvAsDataGridView)

Dgv.SelectedRows(0).Cells(Dgv.Columns.Count - 1).Value = NotDgv.SelectedRows(0).Cells(Dgv.Columns.Count - 1).Value

EndSub

PublicFunctionObtenerCheck(ByRefTablaAsDataTable, ByRefDgvAsDataGridView) AsString()

DimUltiColAsInteger = Dgv.Columns.Count - 1

DimIds(Dgv.Rows.Count - 1) AsString

DimValoresAsInteger = 0

ForiAsInteger = 0 ToDgv.Rows.Count - 1

If (Dgv.Rows(i).Cells(UltiCol).Value = True) Then

Ids(i) = Tabla.Rows(i).Item(0).ToString

Valores += 1

Else

Ids(i) = ""

EndIf

Next

DimID(Valores - 1) AsString

DimiterAsInteger = 0

ForEachiAsStringIn Ids

If (i<>"") Then

ID(iter) = i

iter += 1

EndIf

Next

Return ID

EndFunction

PublicSubActualizarTabla(ByRefTablaAsDataTable, ByRefDgvAsDataGridView)

IfNotIsNothing(Tabla) Then

DimTamanos(Dgv.Columns.Count() - 1) AsSingle

ForiAsInteger = 0 ToDgv.Columns.Count - 1

Tamanos(i) = Dgv.Columns(i).FillWeight()

Next

Dgv.Columns.Clear()

Dgv.DataSource = Tabla

ForiAsInteger = 0 ToDgv.Columns.Count - 1

Dgv.Columns(i).FillWeight() = Tamanos(i)

Dgv.Columns(i).SortMode = DataGridViewColumnSortMode.NotSortable

Next

Dgv.Refresh()

Else

If (Dgv.Rows.Count> 0) Then

Dgv.DataSource.Rows.Clear()

Dgv.Refresh()

EndIf

EndIf

Dgv.AllowUserToOrderColumns = False

EndSub

PublicSubActualizarTablaC(ByRefTablaAsDataTable, ByRefDgvAsDataGridView)

IfNotIsNothing(Tabla) Then

DimTamanos(Dgv.Columns.Count() - 1) AsSingle

ForiAsInteger = 0 ToDgv.Columns.Count - 1

Tamanos(i) = Dgv.Columns(i).FillWeight()

Next

DimColumnaAsDataGridViewColumn

Columna = Dgv.Columns(Tamanos.Length - 1)

Columna.ReadOnly = False

Dgv.Columns.Clear()

Dgv.DataSource = Tabla

If (Tabla.Columns.Count> 1) Then

Dgv.Columns.RemoveAt(0)

EndIf

If (Tabla.Columns.Count = Tamanos.Length) Then

Dgv.Columns.Add(Columna)

EndIf

ForiAsInteger = 0 ToDgv.Columns.Count - 1

Dgv.Columns(i).FillWeight() = Tamanos(i)

Dgv.Columns(i).SortMode = DataGridViewColumnSortMode.NotSortable

Dgv.Columns(i).ReadOnly = If(Tabla.Columns.Count = Tamanos.Length, False, True)

Next

Dgv.Refresh()

Else

If (Dgv.Rows.Count> 0) Then

Dgv.DataSource.Rows.Clear()

Dgv.Refresh()

EndIf

EndIf

EndSub

PublicSubPrepararUpdate(ByValtablaAsString, ByValdatos() AsString, ByVal id AsString)

Dim res AsString = ""

Dim dt AsDataTable = ESQLSelect("describe " + tabla)

DimiAsInteger = 1

ForEachdatoIndatos

res += String.Format(If(dato = "null", "{0} = {1},", "{0} = '{1}',"), dt.Rows(i).Item(0).ToString, dato)

i += 1

Next

res = res.Remove(res.Length - 1)

USQL(tabla, res, String.Format("{0} = '{1}'", dt.Rows(0).Item(0).ToString, id))

EndSub

PublicSubPrepararInsert(ByValtablaAsString, ByValdatos() AsString)

Dim col AsString = ""

DimvaluAsString = ""

Dim dt AsDataTable = ESQLSelect("describe " + tabla)

DimiAsInteger = 1

ForEachdatoIndatos

col += String.Format("{0},", dt.Rows(i).Item(0).ToString)

valu += String.Format(If(dato = "null", "{0},", "'{0}',"), dato)

i += 1

Next

col = col.Remove(col.Length - 1)

valu = valu.Remove(valu.Length - 1)

ISQL(tabla, col, valu)

EndSub

PublicSubPrepararUpdate(ByValtablaAsString, ByValColumna() AsString, ByValdatos() AsString, ByValCondiciones() AsString, ByVal id() AsString)

Dim res AsString = ""

DimiAsInteger = 0

ForEachdatoIndatos

res += String.Format(If(dato = "null", "{0} = {1},", "{0} = '{1}',"), Columna(i), dato)

i += 1

Next

i = 0

res = res.Remove(res.Length - 1)

DimcondicionAsString = ""

ForEach con InCondiciones

condicion += String.Format(If(id(i) = "null", "{0} = {1} and", "{0} = '{1}' and"), con, id(i))

i += 1

Next

condicion = condicion.Remove(condicion.Length - 4)

USQL(tabla, res, condicion)

EndSub

PublicSubPrepararDelete(ByValtablaAsString, ByValdatos() AsString, ByVal ids() AsString)

Dim res AsString = ""

DimiAsInteger = 0

ForEachdatoIndatos

res += String.Format(If(ids(i) = "null", "{0} = {1} and", "{0} = '{1}' and"), datos(i), ids(i))

i += 1

Next

res = res.Remove(res.Length - 3)

BSQL(tabla, res)

EndSub

PublicSubPrepararDelete(ByValtablaAsString, ByValdatosAsString, ByVal ids() AsString)

Dim res AsString = datos + " in ("

ForEach id In ids

res += String.Format(If(id = "null", "{0} ,", "'{0}' ,"), id)

Next

res = res.Remove(res.Length - 1)

res += ")"

BSQL(tabla, res)

EndSub

PublicFunctionBuscarDatos(ByValtablaAsString, ByValColumnas() AsString, ByVal campo AsString, ByVal id AsString) AsString()

Dim res AsString = ""

Dimresultado(Columnas.Length - 1) AsString

ForEachcolumnaInColumnas

res += String.Format("{0}, ", columna)

Next

res = res.Remove(res.Length - 2)

DimdtNAsDataTable = DevolverTabla(PSQL(res, tabla, String.Format("{0} = '{1}'", campo, id)))

If (NotIsNothing(dtN)) Then

For j AsInteger = 0 TodtN.Columns.Count - 1

resultado(j) = dtN.Rows(0).Item(j).ToString

Next

EndIf

Returnresultado

EndFunction

PublicSubLlenarCombo(ByRef con AsComboBox, ByVal dt AsDataTable, ByVal col AsString)

con.Items.Clear()

con.Items.Add("No está relacionado")

If (NotIsNothing(dt)) Then

If (dt.Rows.Count> 0) Then

For j AsInteger = 0 Todt.Rows.Count - 1

con.Items.Add(dt.Rows(j).Item(col).ToString)

Next

EndIf

EndIf

EndSub

PublicFunctionCompararValores(ByVal s1() AsString, ByVal s2() AsString) AsBoolean

For j AsInteger = 0 To s2.Length - 1

If s1(j) <> s2(j) Then

ReturnFalse

EndIf

Next

ReturnTrue

EndFunction

PublicFunctionVaciarNull(ByVal s1() AsString) AsString()

DimsN(s1.Length - 1) AsString

For j AsInteger = 0 To s1.Length - 1

sN(j) = s1(j)

If s1(j) = "null"Then

sN(j) = ""

EndIf

Next

ReturnsN

EndFunction

EndModule

ModUser.vb

Imports System

Imports System.IO

ImportsSystem.Collections

ModuleModUser

':::Ruta donde crearemos nuestro archivo txt

Private ruta AsString = "..\User\"

':::Nombre del archivo

PrivatearchivoAsString = "User.txt"

PrivateBarchivoAsString = "Origin.txt"

PrivatekarchivoAsString = "Key.txt"

'Establece la key y una verificacion

PublicSubInicio()

IfNotDirectory.Exists(ruta) Then

Directory.CreateDirectory(ruta)

EndIf

IfFile.Exists(ruta&karchivo) Then

Dim leer AsNewStreamReader(ruta&karchivo)

ModCodificador.EKey(leer.ReadLine())

ModCodificador.EKeyMaestra(leer.ReadLine())

leer.Close()

ModCodificador.Actualizar()

LeeDatos()

Verify()

Else

IfFile.Exists(ruta&archivo) Then

File.Delete(ruta&archivo)

EndIf

IfFile.Exists(ruta&Barchivo) Then

File.Delete(ruta&Barchivo)

EndIf

DimescribirAsNewStreamWriter(ruta&karchivo, False)

Dim r AsNew Random

escribir.WriteLine(RandomString(r, 8))

escribir.WriteLine(RandomString(r, 32))

escribir.Close()

Inicio()

EndIf

EndSub

FunctionRandomString(r As Random, ByValcntAsByte)

Dim s AsString = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789"

Dim sb AsNewText.StringBuilder

ForiAsInteger = 1 Tocnt

DimidxAsInteger = r.Next(0, s.Length)

sb.Append(s.Substring(idx, 1))

Next

Returnsb.ToString()

EndFunction

PublicSubVerify()

IfNotFile.Exists(ruta&archivo) Then

IfNotFile.Exists(ruta&Barchivo) Then

File.Create(ruta&Barchivo).Close()

Else

File.Copy(ruta &Barchivo, ruta & archivo)

EndIf

EndIf

EndSub

PublicSubBorrar()

IfFile.Exists(ruta&archivo) Then

File.Delete(ruta&archivo)

EndIf

EndSub

PublicSubLeeDatos()

IfFile.Exists(ruta&archivo) Then

Dim leer AsNewStreamReader(ruta&archivo)

DimDatos(4) AsString

Dim n AsByte = 0

Whileleer.Peek<> -1

'Leemos cada linea del archivo TXT

DimlineaAsString = leer.ReadLine()

'Agregramos los datos

Datos(n) = ModCodificador.Desencriptar(linea)

n += 1

EndWhile

ModConector.Crear(Datos(0), Datos(1), Datos(2), Datos(3), Datos(4))

leer.Close()

EndIf

EndSub

PublicSubGuardar(ByVal Desarrollo AsBoolean)

DimarchivosAsString

If Desarrollo Then

archivos = Barchivo

Else

archivos = archivo

EndIf

Dim escribir AsNewStreamWriter(ruta & Archivos, False)

escribir.WriteLine(ModCodificador.Encriptar(ModConector.GAddress()))

escribir.WriteLine(ModCodificador.Encriptar(ModConector.GPort()))

escribir.WriteLine(ModCodificador.Encriptar(ModConector.GDatabase()))

escribir.WriteLine(ModCodificador.Encriptar(ModConector.GUser()))

escribir.WriteLine(ModCodificador.Encriptar(ModConector.GPass()))

escribir.Close()

EndSub

EndModule

Formularios:

frmCliente.vb

' TODO: Indicador de editar (?)

PublicClassfrmCliente

DimempresaIDAsInteger

Dim editando AsBoolean = False' Controla si se esta en modo de edicion o no

DimtmpDatos(2) AsString

Dim cambio AsBoolean = False' Controla si han habido cambios desde el ultimo modo de edicion

Dimdt\_PublicidadAsNewDataTable

PublicSubNew(ByValDatosI() AsString)

InitializeComponent()

'Los siguientes datos se obtienen de la tabla en el elemento padre

empresaID = DatosI(0)

txtNombre.Text = DatosI(1)

txtTelefono.Text = DatosI(2)

txtMail.Text = DatosI(3)

' ...Esto tambien, supongo.....

''ModLog.Guardar(PSQL("id\_video, fecha as Fecha, nombre as Nombre", "video", String.Format("id\_serie = '{0}'", DatosI(0))))

EndSub

PrivateSubfrmSerie\_Load(sender AsObject, e AsEventArgs) HandlesMyBase.Load

ActualizarTabla()

If (empresaID = -1) Then

Alternar()

EndIf

EndSub

PrivateSubbtnSEditar\_Click(sender AsObject, e AsEventArgs) HandlesbtnSEditar.Click

'' editando = True -> Se guardaran los cambios

'' editando = False -> Se le permitira al usuario escribir en los campos

IfempresaID = -1 Then

Dimdatos() AsString = {txtNombre.Text, txtTelefono.Text, txtMail.Text}

PrepararInsert("Empresa", datos)

vaciar()

ElseIf editando Then

If cambio Then

Dimdatos() AsString = {txtNombre.Text, txtTelefono.Text, txtMail.Text}

IfNotCompararValores(VaciarNull(datos), tmpDatos) Then

PrepararUpdate("Empresa", datos, empresaID)

EndIf

AlternarCambioHandlers()

EndIf

Else

tmpDatos(0) = txtNombre.Text

tmpDatos(1) = txtTelefono.Text

tmpDatos(2) = txtMail.Text

EndIf

Alternar()

EndSub

Subvaciar()

txtNombre.Text = ""

txtTelefono.Text = ""

txtMail.Text = ""

EndSub

PrivateSubbtnSSalir\_Click(sender AsObject, e AsEventArgs) HandlesbtnSSalir.Click

IfNoteditandoOrempresaID<> -1 Then

Close()

Else

IfcambioThen

txtNombre.Text = tmpDatos(0)

txtTelefono.Text = tmpDatos(1)

txtMail.Text = tmpDatos(2)

AlternarCambioHandlers()

EndIf

Alternar()

EndIf

EndSub

PrivateSubSerie\_FormClosing(sender AsObject, e AsFormClosingEventArgs) HandlesMe.FormClosing

IfcambioAndempresaID<> -1 Then

Dimdatos() AsString = {txtNombre.Text, txtTelefono.Text, txtMail.Text}

IfNotCompararValores(VaciarNull(datos), tmpDatos) Then

Dim g AsNewfrmGuardarEdicion("Empresa", datos, empresaID)

g.ShowDialog()

IfModInicializador.Cancelar.Contains("Empresa") Then

e.Cancel = True

ModInicializador.Cancelar = ModInicializador.Cancelar.Replace("Empresa", "")

Else

AlternarCambioHandlers()

EndIf

EndIf

EndIf

EndSub

'' Alternarbotones

PrivateSubAlternar()

IfempresaID = -1 Then

btnSEditar.Text = "Ingresar"

btnBorrar.Visible = False

btnSSalir.Text = "Salir"

ActiveForm.Text = "Ingresar Cliente"

ElseIfeditandoThen

btnSEditar.Text = "Editar"

btnSSalir.Text = "Salir"

ActiveForm.Text = "Ver Cliente"

Else

btnSSalir.Text = "Cancelar"

btnSEditar.Text = "Guardar"

ActiveForm.Text = "Editar Cliente"

EndIf

editando = Not editando

txtNombre.ReadOnly = Not editando

txtTelefono.ReadOnly = Not editando

txtMail.ReadOnly = Not editando

EndSub

'' Checkean si hay cambios hechos

'' Si cambio = True, no se llamaran

PrivateSubtxtNombre\_ModifiedChanged(sender AsObject, e AsEventArgs) HandlestxtNombre.ModifiedChanged

IftxtNombre.ModifiedThen

AlternarCambioHandlers()

EndIf

EndSub

' AquiyacIA mi variable, recordatorio de lo que una vez fue Y AUN ES

' cambio: I LIVED BITCH

'' Alterna el estado de cambios y activa/desactiva la deteccion de cambios

PrivateSubAlternarCambioHandlers()

'' cambio = True -> Se aniadiran los handlers, se los necesita

'' cambio = False -> Se ha detectado un cambio, y los handlers se removeran hasta la siguiente llamada del metodo

IfcambioThen

AddHandlertxtNombre.ModifiedChanged, AddressOftxtNombre\_ModifiedChanged

AddHandlertxtMail.ModifiedChanged, AddressOftxtMail\_ModifiedChanged

AddHandlertxtTelefono.ModifiedChanged, AddressOftxtTelefono\_ModifiedChanged

Else

RemoveHandlertxtNombre.ModifiedChanged, AddressOftxtNombre\_ModifiedChanged

RemoveHandlertxtMail.ModifiedChanged, AddressOftxtMail\_ModifiedChanged

RemoveHandlertxtTelefono.ModifiedChanged, AddressOftxtTelefono\_ModifiedChanged

EndIf

cambio = Notcambio

EndSub

PrivateSubdgvVSM\_CellDoubleClick(sender AsObject, e AsDataGridViewCellEventArgs) HandlesdgvVSM.CellDoubleClick

'Dim i As Integer = CargarID(dt\_Video, dgvVSM)

'If (i<> -1) Then

'Dim formVideo As New frmVideo(i)

'AddHandlerformVideo.FormClosed, AddressOfFormVideo\_FormClosed

'formVideo.ShowDialog()

'EndIf

EndSub

'' Actualiza la tabla mostrando los videos asociados cuando el formulario de mostrar video se ha cerrado

PrivateSubFormVideo\_FormClosed(sender AsObject, e AsFormClosedEventArgs)

ActualizarTabla()

EndSub

PrivateSubActualizarTabla()

dt\_Publicidad = DevolverTabla(PSQL("ID\_Publicidad, Tema", "publicidad", String.Format("ID\_Empresa = '{0}'", empresaID)))

ActualizarTablaC(dt\_Publicidad, dgvVSM)

EndSub

PrivateSubbtnBorrar\_Click(senderAsObject, e AsEventArgs) HandlesbtnBorrar.Click

DimformDeleteAsNewfrmConfirmarBorrado("Empresa", {empresaID}, True)

formDelete.ShowDialog(Me)

EndSub

PrivateSubtxtMail\_ModifiedChanged(sender AsObject, e AsEventArgs) HandlestxtMail.ModifiedChanged

IftxtMail.ModifiedThen

AlternarCambioHandlers()

EndIf

EndSub

PrivateSubtxtTelefono\_ModifiedChanged(sender AsObject, e AsEventArgs) HandlestxtTelefono.ModifiedChanged

IftxtTelefono.ModifiedThen

AlternarCambioHandlers()

EndIf

EndSub

EndClass

frmConfirmarBorrado.vb

PublicClassfrmConfirmarBorrado

DimtablaAsString

Dimid() AsString

Dim c AsBoolean = False

PublicSubNew(ByVal t AsString, ByValidentificador() AsString, ByValCerrarAsBoolean)

InitializeComponent()

tabla = t

id = identificador

c = Cerrar

EndSub

PrivateSubbtnCancelar\_Click(sender AsObject, e AsEventArgs) HandlesbtnCancelar.Click

Close()

EndSub

PrivateSubbtnBorrar\_Click(sender AsObject, e AsEventArgs) HandlesbtnBorrar.Click

SelectCasetabla

Case"Serie"

PrepararDelete("video", "id\_serie", id)

PrepararDelete("Serie", "id\_serie", id)

Case"Video"

PrepararDelete("Video", "id\_video", id)

Case"Empresa"

PrepararDelete("Empresa", "id\_empresa", id)

EndSelect

If (c) Then

Owner.Close()

EndIf

Close()

EndSub

PrivateSubBtnBorrarTodo\_Click(sender AsObject, e AsEventArgs) HandlesbtnBorrarSerie.Click

PrepararUpdate("video", {"id\_serie"}, {"null"}, {"id\_serie"}, id)

PrepararDelete("Serie", "id\_serie", id)

If (NotIsNothing(dt\_Video)) Then

frmPrin.btnlimpiarv.PerformClick()

EndIf

If (c) Then

Owner.Close()

EndIf

Close()

EndSub

PrivateSubfrmConfirmarBorrado\_Load(sender AsObject, e AsEventArgs) HandlesMyBase.Load

If (tabla = "Serie") Then

btnBorrarSerie.Visible = True

EndIf

EndSub

EndClass

frmGuardarEdicion.vb

ImportsSystem.ComponentModel

PublicClassfrmGuardarEdicion

DimtablaAsString

Dimdatos() AsString

Dim id AsString

PublicSubNew(ByVal t AsString, ByVal d() AsString, ByValidentificadorAsString)

InitializeComponent()

tabla = t

datos = d

id = identificador

Beep()

EndSub

PrivateSubbtnCancelar\_Click(sender AsObject, e AsEventArgs) HandlesbtnCancelar.Click

ModInicializador.Cancelar += tabla

Close()

EndSub

PrivateSubbtnNoGuardar\_Click(sender AsObject, e AsEventArgs) HandlesbtnNoGuardar.Click

Close()

EndSub

PrivateSubbtnGuardar\_Click(sender AsObject, e AsEventArgs) HandlesbtnGuardar.Click

PrepararUpdate(tabla, datos, id)

Close()

EndSub

EndClass

frmPrincipal.vb

Imports System.ComponentModel

Imports System.Drawing.Drawing2D

Imports System.IO

Public Class frmPrincipal

Private DescripcionP As String

PrivatePNombre As String = "Datos del programa"

Private TBuscada As String = ""

Private TBusca As New DataTable

Public dt\_Buscada As New DataTable

#Region "Carga y descarga"

Private Sub frmPrincipal\_FormClosing(ByVal sender As Object, ByVal e As System.Windows.Forms.FormClosingEventArgs) Handles Me.FormClosing

ModConector.desconectar()

End

End Sub

Private Sub frmPrincipal\_Closing(sender As Object, e As CancelEventArgs) Handles Me.Closing

GuardarNotas()

End Sub

Public Sub EPropiedades()

Me.Width = Screen.PrimaryScreen.WorkingArea.Width \* 0.85

Me.Height = Screen.PrimaryScreen.WorkingArea.Height \* 0.8

dtp.BackColor = Color.FromArgb(64, 64, 64)

dtp.ForeColor = Color.White

dtpTanda.BackColor = Color.FromArgb(64, 64, 64)

dtpTanda.ForeColor = Color.White

End Sub

Public Sub ONBW()

BWProgramas.RunWorkerAsync()

BWTandas.RunWorkerAsync()

BWEventos.RunWorkerAsync()

End Sub

Private Sub frmPrincipal\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

EPropiedades()

ONBW()

LeerNotas()

End Sub

#End Region

Private Sub BWNumberOne\_DoWork(sender As Object, e As DoWorkEventArgs) Handles BWProgramas.DoWork

dt\_programa = ModConector.APrograma(dtp.Value.Date)

End Sub

Private Sub ActualizarProgramas()

dgvPrograma.Columns.Clear()

If Not IsNothing(dt\_programa) Then

dgvPrograma.DataSource = dt\_programa

dgvPrograma.Columns().RemoveAt(0)

dgvPrograma.Columns.Add("PEstado", "Estado")

dgvProgramaColor()

For i As Integer = 0 To dgvPrograma.Columns.Count - 1

dgvPrograma.Columns(i).SortMode = DataGridViewColumnSortMode.NotSortable

Next

Else

dt\_programa = New DataTable

dgvPrograma.DataSource = dt\_programa

dgvPrograma.Columns.Add("PInicio", "Inicio")

dgvPrograma.Columns.Add("PFinal", "Final")

dgvPrograma.Columns.Add("PPrograma", "Programa")

dgvPrograma.Columns.Add("PEstado", "Estado")

End If

End Sub

Private Sub ActualizarEvento()

ActualizarTabla(dt\_evento, dgvEventos)

dgvEventos.Columns().RemoveAt(0)

dgvEventos.ClearSelection()

End Sub

Public Sub dgvProgramaColor()

Dim fin As TimeSpan

Dim inicio As TimeSpan

Dim colores As Color = dgvPrograma.DefaultCellStyle.ForeColor

Dim colorNuevo As Color

Dim estado As String

Dim Activo As Integer = -1

If (Now.Date>= dtp.Value.Date) And dgvPrograma.Rows.Count> 0 Then

For i As Integer = 0 To dgvPrograma.Rows.Count - 1

If Not (TimeSpan.TryParse(dgvPrograma.Rows(i).Cells(0).Value().ToString, inicio) And TimeSpan.TryParse(dgvPrograma.Rows(i).Cells(1).Value().ToString, fin)) Then

Exit Sub

End If

If (inicio<Now.TimeOfDay Or Now.Date>dtp.Value.Date) Then

If (Now.TimeOfDay< fin And Now.Date = dtp.Value.Date) Then

colorNuevo = Color.FromArgb(245, 94, 94)

estado = "Activo"

dgvPrograma.Rows(i).Selected = True

Activo = i

Else

estado = "Finalizado"

colorNuevo = Color.FromArgb(154, 94, 245)

End If

dgvPrograma.Rows(i).Cells(dgvPrograma.Rows(i).Cells.Count - 1).Style.ForeColor = colorNuevo

Else

If Activo = -1 Then

Activo = i

dgvPrograma.Rows(i).Selected = True

End If

estado = "Activo"

End If

dgvPrograma.Rows(i).Cells(dgvPrograma.Rows(i).Cells.Count - 1).Value = estado

Next

Else

For i As Integer = 0 To dgvPrograma.Rows.Count - 1

dgvPrograma.Rows(i).Cells(dgvPrograma.Rows(i).Cells.Count - 1).Value = "Proximo"

Next

End If

End Sub

Private Sub BWNumberOne\_RunWorkerCompleted(sender As Object, e As RunWorkerCompletedEventArgs) Handles BWProgramas.RunWorkerCompleted

ActualizarProgramas()

Funcionarios()

End Sub

Private Sub dtp\_ValueChanged(sender As Object, e As EventArgs)

If Not BWProgramas.IsBusy Then

BWProgramas.RunWorkerAsync()

End If

End Sub

Public Sub GuardarNotas()

Dim ruta As String = "..\User\"

Dim archivo As String = "Notas.txt"

Dim escribir As New StreamWriter(ruta&archivo, False)

escribir.Write(TBNotas.Text)

escribir.Close()

End Sub

Public Sub LeerNotas()

Dim ruta As String = "..\User\"

Dim archivo As String = "Notas.txt"

If File.Exists(ruta&archivo) Then

TBNotas.Text = ""

Dim leer As New StreamReader(ruta&archivo)

TBNotas.Text = leer.ReadToEnd()

leer.Close()

End If

End Sub

Private Sub BWDPRogramas\_DoWork(sender As Object, e As DoWorkEventArgs) Handles BWDPRogramas.DoWork

If (dgvPrograma.Rows.Count> 0) And dgvPrograma.SelectedRows.Count> 0 Then

Dim idRow As Integer = dgvPrograma.SelectedRows(0).Index

If (idRow>= 0) And Not IsNothing(dt\_programa) Then

Dim id As Integer = CInt(dt\_programa.Rows(idRow)(0).ToString)

dt\_dprograma = ModConector.AFPrograma(id)

DescripcionP = ModConector.ADPrograma(id)

dt\_Ppubli = ModConector.APPublicidad(dtp.Value.Date, id)

PNombre = "Datos del programa : " + dt\_programa.Rows(idRow)(3).ToString

End If

Else

PNombre = "Datos del programa"

dt\_Ppubli = Nothing

dt\_dprograma = Nothing

End If

End Sub

Private Sub BWDPRogramas\_RunWorkerCompleted(sender As Object, e As RunWorkerCompletedEventArgs) Handles BWDPRogramas.RunWorkerCompleted

ActualizarTabla(dt\_dprograma, dgvFuncionarios)

ActualizarTabla(dt\_publi, dgvPPublicidades)

GBFuncionarios.Text = PNombre

TBDescripcion.Text = DescripcionP

dgvFuncionarios.ClearSelection()

dgvPPublicidades.ClearSelection()

dgvPrograma.ClearSelection()

End Sub

Public Sub Funcionarios()

If Not (BWDPRogramas.IsBusy) Then

If Not (BWProgramas.IsBusy) Then

BWDPRogramas.RunWorkerAsync()

End If

End If

End Sub

Private Sub dgvPrograma\_Click(sender As Object, e As EventArgs) Handles dgvPrograma.Click

Funcionarios()

End Sub

Private Sub BWEventos\_DoWork(sender As Object, e As DoWorkEventArgs) Handles BWEventos.DoWork

dt\_evento = ModConector.AEventos()

End Sub

Private Sub BWEventos\_RunWorkerCompleted(sender As Object, e As RunWorkerCompletedEventArgs) Handles BWEventos.RunWorkerCompleted

ActualizarEvento()

End Sub

Private Sub BWPublicidades\_DoWork(sender As Object, e As DoWorkEventArgs) Handles BWPublicidades.DoWork

If dgvTandas.Rows.Count> 0 And dgvTandas.SelectedRows.Count> 0 Then

Dim Hora As TimeSpan = TimeSpan.Parse(dgvTandas.SelectedRows(0).Cells(0).Value)

If Not IsNothing(dt\_tandas) Then

dt\_publi = ModConector.APublicidad(dtpTanda.Value.Date, Hora)

End If

Else

dt\_publi = Nothing

End If

End Sub

Private Sub BWPublicidades\_RunWorkerCompleted(sender As Object, e As RunWorkerCompletedEventArgs) Handles BWPublicidades.RunWorkerCompleted

ActualizarTabla(dt\_publi, dgvPublicidades)

End Sub

Private Sub BWTandas\_DoWork(sender As Object, e As DoWorkEventArgs) Handles BWTandas.DoWork

dt\_tandas = ModConector.ATandas()

End Sub

Public Sub Tandas()

ActualizarTabla(dt\_tandas, dgvTandas)

End Sub

Private Sub BWTandas\_RunWorkerCompleted(sender As Object, e As RunWorkerCompletedEventArgs) Handles BWTandas.RunWorkerCompleted

Tandas()

BWPublicidades.RunWorkerAsync()

End Sub

Private Sub dtpTanda\_ValueChanged(sender As Object, e As EventArgs)

If Not BWTandas.IsBusy Then

BWPublicidades.RunWorkerAsync()

End If

End Sub

Private Sub dgvTandas\_Click(sender As Object, e As EventArgs) Handles dgvTandas.Click

If Not BWPublicidades.IsBusy Then

BWPublicidades.RunWorkerAsync()

End If

End Sub

Private Sub dtp\_ValueChanged\_1(sender As Object, e As EventArgs) Handles dtp.ValueChanged

If Not BWProgramas.IsBusy Then

BWProgramas.RunWorkerAsync()

End If

End Sub

Private Sub dtpTanda\_ValueChanged\_1(sender As Object, e As EventArgs) Handles dtpTanda.ValueChanged

If Not BWTandas.IsBusy Then

BWPublicidades.RunWorkerAsync()

End If

End Sub

Private Sub BWBuscador\_DoWork(sender As Object, e As DoWorkEventArgs) Handles BWBuscador.DoWork

TBusca = DevolverTabla(e.Argument)

ModLog.Guardar(e.Argument)

End Sub

Private Sub btnbuscarv\_Click(sender As Object, e As EventArgs) Handles btnbuscarv.Click

TBuscada = "Video"

Dimcondicion As String = "true" ' FIXME: Al poner limit 50 no sirve buscar solo por fecha. Asi que lo he quitado por ahora.

If (NotString.IsNullOrWhiteSpace(txtVnombre.Text)) Then

condicion = String.Format("v.nombrelike '%{0}%'", txtVnombre.Text)

End If

If (cbFecha.Checked) Then

condicion += String.Format(" and fecha = '{0}'", Format(dtpfechavideo.Value, "yyyy-MM-dd").ToString)

End If

If (Not String.IsNullOrWhiteSpace(txtVcontenido.Text)) Then

condicion += String.Format(" and v.contenido like '%{0}%'", txtVcontenido.Text)

End If

If Not (BWBuscador.IsBusy) Then

BWBuscador.RunWorkerAsync(PSQL("id\_video, fecha as Fecha, v.nombre as Nombre, (selects.nombrefrom serie s wheres.id\_serie=v.id\_serie) as Serie", "video v", condicion))

End If

End Sub

Private Sub BWBuscador\_RunWorkerCompleted(sender As Object, e As RunWorkerCompletedEventArgs) Handles BWBuscador.RunWorkerCompleted

Select Case TBuscada

Case "Video"

dt\_Video = TBusca

ActualizarTablaC(dt\_Video, dgvVB)

Case "Serie"

dt\_Serie = TBusca

ActualizarTablaC(dt\_Serie, dgvBS)

Case "Empresa"

dt\_Empresa = TBusca

ActualizarTablaC(dt\_Empresa, dgvClientes)

End Select

TBusca = Nothing

TBuscada = ""

End Sub

Private Sub DgvVB\_CellDoubleClick(sender As Object, e As DataGridViewCellEventArgs) Handles dgvVB.CellDoubleClick

Dim i As Integer = CargarID(dt\_Video, dgvVB)

If (i<> -1) Then

Dim formVideo As New frmVideo(i)

AddHandlerformVideo.FormClosed, AddressOfFormVideo\_FormClosed

formVideo.ShowDialog()

End If

End Sub

Private Sub btnLimpiarBS\_Click(sender As Object, e As EventArgs) Handles btnLimpiarBS.Click

txtBSnombre.Clear()

ctpSerie.Value = Now.Date

cbS.Checked = False

ActualizarTabla(Nothing, dgvBS)

End Sub

Private Sub frmPrincipal\_Resize(sender As Object, e As EventArgs) Handles Me.Resize

Refresh()

End Sub

Private Sub btnlimpiarv\_Click(sender As Object, e As EventArgs) Handles btnlimpiarv.Click

txtVcontenido.Clear()

txtVnombre.Clear()

dtpfechavideo.Value = Now.Date

cbFecha.Checked = False

ActualizarTabla(Nothing, dgvVB)

End Sub

Private Sub btnBuscarBS\_Click(sender As Object, e As EventArgs) Handles btnBuscarBS.Click

TBuscada = "Serie"

Dimcondicion As String = "true" ' FIXME: Al poner limit 50 no sirve buscar solo por fecha. Asi que lo he quitado por ahora.

If (NotString.IsNullOrWhiteSpace(txtBSnombre.Text)) Then

condicion = "nombre like '%" + txtBSnombre.Text + "%'"

End If

If (cbS.Checked) Then

condicion += String.Format(" and fecha\_finalizacion = '{0}'", Format(ctpSerie.Value, "yyyy-MM-dd").ToString)

End If

IfNot (BWBuscador.IsBusy) Then

BWBuscador.RunWorkerAsync(PSQL("id\_serie, fecha\_finalizacion as Fecha, nombre as Nombre", "serie", condicion))

End If

End Sub

Private Sub dgvBS\_CellDoubleClick(sender As Object, e As DataGridViewCellEventArgs) Handles dgvBS.CellDoubleClick

Dim i() As String = CargarID(dt\_Serie, dgvBS, {0, 1, 2})

If (i.Length<> 1) Then

Dim formSerie As New frmSerie(i)

AddHandlerformSerie.FormClosed, AddressOfFormSerie\_FormClosed

formSerie.ShowDialog()

Else

TBNotas.Text += "a"

End If

End Sub

Private Sub FormVideo\_FormClosed(sender As Object, e As FormClosedEventArgs)

btnbuscarv.PerformClick()

End Sub

Private Sub FormSerie\_FormClosed(sender As Object, e As FormClosedEventArgs)

btnBuscarBS.PerformClick()

End Sub

Private Sub dgvVB\_CellClick(sender As Object, e As DataGridViewCellEventArgs) Handles dgvVB.CellClick

ClickCheck(dgvVB)

End Sub

Private Sub btnBorrar\_Click(sender As Object, e As EventArgs) Handles btnBorrar.Click

If (dt\_Video.Rows.Count> 0) Then

Dim Id() As String = ObtenerCheck(dt\_Video, dgvVB)

Dim formDelete As New frmConfirmarBorrado("Video", Id, False)

formDelete.ShowDialog(Me)

btnbuscarv.PerformClick()

End If

End Sub

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles btnBorrarS.Click

If (dt\_Serie.Rows.Count> 0) Then

Dim Id() As String = ObtenerCheck(dt\_Serie, dgvBS)

Dim formDelete As New frmConfirmarBorrado("Serie", Id, False)

formDelete.ShowDialog(Me)

btnBuscarBS.PerformClick()

End If

End Sub

Private Sub dgvBS\_CellClick(sender As Object, e As DataGridViewCellEventArgs) Handles dgvBS.CellClick

ClickCheck(dgvBS)

End Sub

Private Sub btnIngresarV\_Click(sender As Object, e As EventArgs) Handles btnIngresarV.Click

Dim formVideo As New frmVideo(-1)

formVideo.ShowDialog()

End Sub

Private Sub Button2\_Click(sender As Object, e As EventArgs) Handles btnsIngresar.Click

Dim formSerie As New frmSerie({-1, Now.Date.ToString, ""})

formSerie.ShowDialog()

End Sub

Private Sub btncBuscar\_Click(sender As Object, e As EventArgs) Handles btncBuscar.Click

TBuscada = "Empresa"

Dimcondicion As String = "true" ' FIXME: Al poner limit 50 no sirve buscar solo por fecha. Asi que lo he quitado por ahora.

If (NotString.IsNullOrWhiteSpace(txtCNombre.Text)) Then

condicion = "Nombre like '%" + txtCNombre.Text + "%'"

End If

If (Not String.IsNullOrWhiteSpace(txtCTel.Text)) Then

condicion += " and Telefono = '" + txtCTel.Text + "'"

End If

If (Not String.IsNullOrWhiteSpace(txtCMail.Text)) Then

condicion += " and Mail = '" + txtCMail.Text + "'"

End If

If Not (BWBuscador.IsBusy) Then

BWBuscador.RunWorkerAsync(PSQL("id\_Empresa, Nombre, Telefono, Mail", "empresa", condicion))

End If

End Sub

Private Sub btncBorrar\_Click(sender As Object, e As EventArgs) Handles btncBorrar.Click

If (dt\_Empresa.Rows.Count> 0) Then

Dim Id() As String = ObtenerCheck(dt\_Empresa, dgvClientes)

Dim formDelete As New frmConfirmarBorrado("Empresa", Id, False)

formDelete.ShowDialog(Me)

btncBuscar.PerformClick()

End If

End Sub

Private Sub dgvClientes\_CellClick(sender As Object, e As DataGridViewCellEventArgs) Handles dgvClientes.CellClick

ClickCheck(dgvClientes)

End Sub

Private Sub FormCliente\_FormClosed(sender As Object, e As FormClosedEventArgs)

btncBuscar.PerformClick()

End Sub

Private Sub dgvClientes\_CellDoubleClick(sender As Object, e As DataGridViewCellEventArgs) Handles dgvClientes.CellDoubleClick

Dim i() As String = CargarID(dt\_Empresa, dgvClientes, {0, 1, 2, 3})

If (i.Length<> 1) Then

Dim formCliente As New frmCliente(i)

AddHandlerformCliente.FormClosed, AddressOfFormCliente\_FormClosed

formCliente.ShowDialog()

Else

TBNotas.Text += "a"

End If

End Sub

Private Sub btncIngresar\_Click(sender As Object, e As EventArgs) Handles btncIngresar.Click

Dim formCliente As New frmCliente({-1, "", "", ""})

formCliente.ShowDialog()

End Sub

Private Sub btncLimpiar\_Click(sender As Object, e As EventArgs) Handles btncLimpiar.Click

txtCNombre.Clear()

txtCMail.Clear()

txtCTel.Clear()

ActualizarTabla(Nothing, dgvClientes)

End Sub

End Class

frmConfiguracion.vb

PublicClassfrmConfiguracion

PrivateUserIDAsInteger = Nothing

PrivateUsuarioDatosAsDataTable

Public dt AsDataTable

PrivateSubbtnRestablecer\_Click(ByVal sender AsSystem.Object, ByVal e AsSystem.EventArgs) HandlesbtnRestablecer.Click

ModUser.Borrar()

ModUser.Inicio()

EndSub

PrivateSubbtnGuardar\_Click(ByVal sender AsSystem.Object, ByVal e AsSystem.EventArgs) HandlesbtnGuardar.Click

If (txtIp.Text<>"") Then

ModConector.EAddress(txtIp.Text)

EndIf

If (txtPuerto.Text<>"") Then

ModConector.EPort(txtPuerto.Text)

EndIf

If (txtUsuario.Text<>"") Then

ModConector.EUser(txtUsuario.Text)

EndIf

ModConector.EPass(txtContraseña.Text)

If (txtBD.Text<>"") Then

ModConector.EDatabase(txtBD.Text)

EndIf

ModUser.Guardar(False)

ModUser.LeeDatos()

EndSub

PrivateSubDebugCrear\_Click(ByVal sender AsSystem.Object, ByVal e AsSystem.EventArgs) HandlesDebugCrear.Click

If (txtIp.Text<>"") Then

ModConector.EAddress(txtIp.Text)

EndIf

If (txtPuerto.Text<>"") Then

ModConector.EPort(txtPuerto.Text)

EndIf

If (txtUsuario.Text<>"") Then

ModConector.EUser(txtUsuario.Text)

EndIf

ModConector.EPass(txtContraseña.Text)

If (txtBD.Text<>"") Then

ModConector.EDatabase(txtBD.Text)

EndIf

ModUser.Guardar(True)

ModUser.LeeDatos()

Me.Dispose()

EndSub

PrivateSubDebugCrearUsuario\_Click(ByVal sender AsSystem.Object, ByVal e AsSystem.EventArgs)

ModConector.IUsuario(txtUsuario.Text, txtContraseña.Text)

EndSub

PrivateSubfrmConfiguracion\_FormClosing(ByVal sender AsObject, ByVal e AsSystem.Windows.Forms.FormClosingEventArgs) HandlesMe.FormClosing

ModUser.Inicio()

ModConector.Inicio()

EndSub

PrivateSubfrmConfiguracion\_Load(ByVal sender AsSystem.Object, ByVal e AsSystem.EventArgs) HandlesMyBase.Load

IfModConector.GDebugThen

DebugCrear.Visible = True

btnGenerador.Visible = True

btnGenerador.Enabled = True

DebugCrear.Enabled = True

EndIf

Control.CheckForIllegalCrossThreadCalls = False

EndSub

PrivateSub TabPage2\_Enter(ByVal sender AsObject, ByVal e AsSystem.EventArgs) HandlesTPusuarios.Enter

'ActualizarUsuarios(False)

BW.RunWorkerAsync(False)

LimpiarEditar()

limpiar()

EndSub

#Region"Limpiadores"

PrivateSublimpiar()

txtNombre.Text = ""

txtContrasena.Text = ""

dgvNombreUsuario.ClearSelection()

EndSub

PrivateSubLimpiarUsuario\_Click(ByVal sender AsSystem.Object, ByVal e AsSystem.EventArgs) HandlesLimpiarUsuario.Click

limpiar()

EndSub

PrivateSubLimpiarEditar()

txtENombre.Text = ""

txtEContrasena.Text = ""

UserID = Nothing

EndSub

PrivateSubUAplicar\_Click(ByVal sender AsSystem.Object, ByVal e AsSystem.EventArgs) HandlesUAplicar.Click

IfNotIsNothing(UserID) Then

USQL("usuarios", "nombre ='" + txtENombre.Text + "',contrasena = AES\_ENCRYPT('" + txtEContrasena.Text() + "',sha2('" + ModCodificador.GKey + "',256))", "id\_usuario ='" + UserID.ToString() + "'")

'ActualizarUsuarios(False)

BW.RunWorkerAsync(False)

IfUserID = ModConector.GUsuarioIDThen

ModConector.BorrarUsuario()

Me.Dispose()

EndIf

EndIf

LimpiarEditar()

EndSub

#EndRegion

#Region"Modificar Usuarios"

PrivateSubUBorrar\_Click(ByVal sender AsSystem.Object, ByVal e AsSystem.EventArgs) HandlesUBorrar.Click

IfNotIsNothing(UserID) Then

BSQL("usuarios", "id\_usuario ='" + UserID.ToString + "'")

'ActualizarUsuarios(False)

BW.RunWorkerAsync(False)

IfUserID = ModConector.GUsuarioIDThen

ModConector.BorrarUsuario()

Me.Dispose()

EndIf

EndIf

LimpiarEditar()

EndSub

PrivateSubActualizarUsuarios()

IfNotIsNothing(dt) Then

Me.dgvNombreUsuario.DataSource = dt

Else

MessageBox.Show("No se cargo correctamente.")

EndIf

EndSub

PrivateSubCrearUsuario\_Click(ByVal sender AsSystem.Object, ByVal e AsSystem.EventArgs) HandlesCrearUsuario.Click

ModConector.IUsuario(txtNombre.Text, txtContrasena.Text)

'ActualizarUsuarios(True)

BW.RunWorkerAsync(True)

limpiar()

EndSub

#EndRegion

PrivateSubdgvNombreUsuario\_CellMouseClick(ByVal sender AsObject, ByVal e AsSystem.Windows.Forms.DataGridViewCellMouseEventArgs) HandlesdgvNombreUsuario.CellMouseClick

Try

LimpiarEditar()

txtENombre.Text = dgvNombreUsuario.CurrentRow.Cells("Nombre Usuarios").Value.ToString

UserID = dgvNombreUsuario.CurrentRow.Cells("ID").Value

Catch es As Exception

EndTry

EndSub

PrivateSubBW\_DoWork(ByVal sender AsObject, ByVal e AsSystem.ComponentModel.DoWorkEventArgs) HandlesBW.DoWork

dt = ModConector.AUsuarios(e.Argument)

EndSub

PrivateSubBW\_RunWorkerCompleted(ByVal sender AsObject, ByVal e AsSystem.ComponentModel.RunWorkerCompletedEventArgs) HandlesBW.RunWorkerCompleted

ActualizarUsuarios()

EndSub

PrivateSubbtnGenerador\_Click(senderAsObject, e AsEventArgs) HandlesbtnGenerador.Click

ModInicializador.Generadores()

EndSub

PrivateSubbtnAccesos\_Click(sender AsObject, e AsEventArgs) HandlesbtnAccesos.Click

EndSub

EndClass

frmLUsuario.vb

PublicClassfrmLUsuario

PrivateSubbtnOpciones\_Click(ByVal sender AsSystem.Object, ByVal e AsSystem.EventArgs) HandlesbtnOpciones.Click

IfBUsuario(txtUsuario.Text, txtPass.Text) OrModConector.GDebugThen

ModInicializador.Configuracion()

EndIf

EndSub

PrivateSubbtnSalir\_Click(ByVal sender AsSystem.Object, ByVal e AsSystem.EventArgs)

End

ModConector.desconectar()

EndSub

PrivateSubLimpiar()

txtUsuario.Text = ""

txtPass.Text = ""

EndSub

PrivateSubbtnEntrar\_Click(ByVal sender AsSystem.Object, ByVal e AsSystem.EventArgs) HandlesbtnEntrar.Click

IfModConector.BUsuario(txtUsuario.Text, txtPass.Text) Then

ModInicializador.Principal()

Me.Dispose()

EndIf

EndSub

PrivateSubfrmLUsuario\_FormClosing(ByVal sender AsObject, ByVal e AsSystem.Windows.Forms.FormClosingEventArgs) HandlesMe.FormClosing

ModConector.desconectar()

EndSub

PrivateSubfrmLUsuario\_Load(ByVal sender AsSystem.Object, ByVal e AsSystem.EventArgs) HandlesMyBase.Load

Try

ModUser.Inicio()

ModConector.Inicio()

Catch m As Exception

If (ModConector.GDebug) Then

MessageBox.Show(e.ToString)

EndIf

EndTry

EndSub

PrivateSublblPass\_Click(ByVal sender AsSystem.Object, ByVal e AsSystem.EventArgs) HandleslblPass.Click

EndSub

EndClass

frmVideo.vb

PublicClassfrmVideo

DimvideoIDAsInteger

PubliceditandoAsBoolean = False

Dimdatos() AsString

DimdatosI() AsString

DimdtVAsDataTable

Dimposition() AsString

Dim pos As UInt16 = 0

DimserieIDAsString = ""

PublicSubNew(ByVal vid AsInteger)

InitializeComponent()

videoID = vid

EndSub

PublicSubBuscar()

Dimcolumnas() AsString = {"contenido", "Nombre", "id\_serie", "DATE\_FORMAT(Fecha,'%Y-%m-%d') as Fecha"}

datosI = BuscarDatos("video", columnas, "id\_video", videoID)

Rellenar()

EndSub

PrivateSubRellenar()

txtNombre.Text = datosI(1)

IfdatosI(3) <>""Then

dtpFecha.Value = CDate(datosI(3))

Else

txtTapar.Visible = True

EndIf

txtContenido.Text = datosI(0)

If (datosI(2) = "") Then

serieID = 0

Else

serieID = datosI(2)

If (datosI(2) = 0) Then

datosI(2) = ""

EndIf

EndIf

CargarCombo()

EndSub

SubCargarCombo()

dtV = DevolverTabla(PSQL("id\_serie, nombre", "Serie", "True"))

LlenarCombo(cbSerie, dtV, "nombre")

If (NotIsNothing(dtV)) Then

ExtraerDatos()

EndIf

cbSerie.SelectedIndex = pos

EndSub

PublicSubExtraerDatos()

ReDimposition(dtV.Rows.Count - 1)

For j AsInteger = 0 TodtV.Rows.Count - 1

position(j) = dtV.Rows(j).Item(0).ToString

Next

ForiAsInteger = 0 Toposition.Length - 1

IfserieID = position(i) Then

pos = i + 1

ExitFor

EndIf

Next

EndSub

PrivateSubbtnEditar\_Click(sender AsObject, e AsEventArgs) HandlesbtnEditar.Click

IfvideoID = -1 Then

ActualizarDatos()

PrepararInsert("video", datos)

vaciar()

ElseIf editando Then

ActualizarDatos()

IfNotCompararValores(VaciarNull(datos), datosI) Then

PrepararUpdate("video", datos, videoID)

datosI = VaciarNull(datos)

' ModInicializador.frmPrin.btnbuscarv.PerformClick()

EndIf

Alternar()

Else

Alternar()

EndIf

EndSub

SubActivar()

txtNombre.ReadOnly = editando

txtContenido.ReadOnly = editando

editando = Not editando

cbSerie.Enabled = editando

dtpFecha.Enabled = editando

chbTieneFecha.Visible = editando

EndSub

PrivateSubAlternar()

Activar()

btnEditar.Text = If(editando, "Guardar", "Editar")

btnSalir.Text = If(editando, "Cancelar", "Salir")

chbTieneFecha.Checked = datosI(3) <>""

IfeditandoThen

txtTapar.Visible = False

Else

IfdatosI(3) = ""Then

txtTapar.Visible = True

chbTieneFecha.Visible = False

EndIf

EndIf

EndSub

PrivateSubbtnSalir\_Click(sender AsObject, e AsEventArgs) HandlesbtnSalir.Click

IfNoteditandoOrvideoID = -1 Then

Close()

Else

Rellenar()

Alternar()

EndIf

EndSub

PrivateSubfrmVideo\_Load(sender AsObject, e AsEventArgs) HandlesMyBase.Load

IfvideoID<> -1 Then

Buscar()

btnSalir.Select()

Else

btnEditar.Text = "Insertar"

btnBorrar.Visible = False

Activar()

EndIf

dtpFecha.BackColor = Color.FromArgb(64, 64, 64)

dtpFecha.ForeColor = Color.White

btnSalir.Select()

EndSub

Subvaciar()

txtNombre.Text = ""

txtContenido.Text = ""

cbSerie.SelectedIndex = -1

dtpFecha.Value = Now.Date

chbTieneFecha.Checked = False

EndSub

PrivateSubfrmVideo\_FormClosing(sender AsObject, e AsFormClosingEventArgs) HandlesMe.FormClosing

IfeditandoAndvideoID<> -1 Then

ActualizarDatos()

IfNotCompararValores(VaciarNull(datos), datosI) Then

Dim g AsNewfrmGuardarEdicion("Video", datos, videoID)

g.ShowDialog()

IfModInicializador.Cancelar.Contains("Video") Then

e.Cancel = True

ModInicializador.Cancelar = ModInicializador.Cancelar.Replace("Video", "")

' ModInicializador.frmPrin.btnbuscarv.PerformClick()

EndIf

EndIf

EndIf

EndSub

PrivateSubActualizarDatos()

Dim con AsString = txtContenido.Text

DimnomAsString = txtNombre.Text

DimdatAsString = If(chbTieneFecha.Checked, Format(dtpFecha.Value, "yyyy-MM-dd"), "null")

ModLog.Guardar(cbSerie.SelectedIndex)

Dim ser AsString = If(cbSerie.SelectedIndex<= 0, "null", position(cbSerie.SelectedIndex - 1))

datos = {con, nom, ser, dat}

EndSub

PrivateSubbtnBorrar\_Click(sender AsObject, e AsEventArgs) HandlesbtnBorrar.Click

DimformDeleteAsNewfrmConfirmarBorrado("Video", {videoID}, True)

formDelete.ShowDialog(Me)

EndSub

EndClass

frmSerie.vb

' TODO: Indicador de editar (?)

PublicClassfrmSerie

DimserieIDAsInteger

Dim editando AsBoolean = False' Controla si se esta en modo de edicion o no

DimtmpDatos(1) AsString

Dim cambio AsBoolean = False' Controla si han habido cambios desde el ultimo modo de edicion

Dimdt\_VideoAsNewDataTable

PublicSubNew(ByValDatosI() AsString)

InitializeComponent()

'Los siguientes datos se obtienen de la tabla en el elemento padre

serieID = DatosI(0)

txtNombre.Text = DatosI(2)

IfDatosI(1) <>""Then

RemoveHandlerdtpFecha.ValueChanged, AddressOfdtpFecha\_ValueChanged

dtpFecha.Value = CDate(DatosI(1))

AddHandlerdtpFecha.ValueChanged, AddressOfdtpFecha\_ValueChanged

' Fix temporal... no me maten

Else

txtTapar.Visible = True

EndIf

RemoveHandlerchbIncluir.CheckedChanged, AddressOfchbIncluir\_CheckedChanged

chbIncluir.Checked = DatosI(1) <>""

AddHandlerchbIncluir.CheckedChanged, AddressOfchbIncluir\_CheckedChanged

' ...Esto tambien, supongo.....

''ModLog.Guardar(PSQL("id\_video, fecha as Fecha, nombre as Nombre", "video", String.Format("id\_serie = '{0}'", DatosI(0))))

EndSub

PrivateSubfrmSerie\_Load(sender AsObject, e AsEventArgs) HandlesMyBase.Load

dtpFecha.BackColor = Color.FromArgb(64, 64, 64)

dtpFecha.ForeColor = Color.White

ActualizarTabla()

If (serieID = -1) Then

Alternar()

EndIf

EndSub

PrivateSubbtnSEditar\_Click(sender AsObject, e AsEventArgs) HandlesbtnSEditar.Click

'' editando = True -> Se guardaran los cambios

'' editando = False -> Se le permitira al usuario escribir en los campos

IfserieID = -1 Then

Dimdatos() AsString = {If(chbIncluir.Checked, Format(dtpFecha.Value, "yyyy-MM-dd"), "null"), txtNombre.Text}

PrepararInsert("Serie", datos)

vaciar()

ElseIf editando Then

If cambio Then

Dimdatos() AsString = {If(chbIncluir.Checked, Format(dtpFecha.Value, "yyyy-MM-dd"), "null"), txtNombre.Text}

IfNotCompararValores(VaciarNull(datos), tmpDatos) Then

PrepararUpdate("Serie", datos, serieID)

EndIf

AlternarCambioHandlers()

EndIf

Else

tmpDatos(0) = If(chbIncluir.Checked, Format(dtpFecha.Value, "yyyy-MM-dd"), "")

tmpDatos(1) = txtNombre.Text

EndIf

Alternar()

EndSub

Subvaciar()

txtNombre.Text = ""

dtpFecha.Value = Now.Date

chbIncluir.Checked = True

EndSub

PrivateSubbtnSSalir\_Click(sender AsObject, e AsEventArgs) HandlesbtnSSalir.Click

IfNoteditandoOrserieID<> -1 Then

Close()

Else

IfcambioThen

IftmpDatos(0) <>""Then

dtpFecha.Value = CDate(tmpDatos(0))

Else

dtpFecha.Value = Now.Date

chbIncluir.Checked = False

EndIf

txtNombre.Text = tmpDatos(1)

AlternarCambioHandlers()

EndIf

Alternar()

EndIf

EndSub

PrivateSubSerie\_FormClosing(sender AsObject, e AsFormClosingEventArgs) HandlesMe.FormClosing

IfcambioAndserieID<> -1 Then

Dimdatos() AsString = {If(chbIncluir.Checked, Format(dtpFecha.Value, "yyyy-MM-dd"), "null"), txtNombre.Text}

IfNotCompararValores(VaciarNull(datos), tmpDatos) Then

Dim g AsNewfrmGuardarEdicion("Serie", datos, serieID)

g.ShowDialog()

IfModInicializador.Cancelar.Contains("Serie") Then

e.Cancel = True

ModInicializador.Cancelar = ModInicializador.Cancelar.Replace("Serie", "")

Else

AlternarCambioHandlers()

EndIf

EndIf

EndIf

EndSub

'' Alternarbotones

PrivateSubAlternar()

IfserieID = -1 Then

btnSEditar.Text = "Ingresar"

btnBorrar.Visible = False

btnSSalir.Text = "Salir"

ActiveForm.Text = "Ingresar Serie"

txtTapar.Visible = False

ElseIfeditandoThen

btnSEditar.Text = "Editar"

btnSSalir.Text = "Salir"

ActiveForm.Text = "Ver Serie"

txtTapar.Visible = NotchbIncluir.Checked

Else

btnSSalir.Text = "Cancelar"

btnSEditar.Text = "Guardar"

ActiveForm.Text = "Editar Serie"

RemoveHandlerchbIncluir.CheckedChanged, AddressOfchbIncluir\_CheckedChanged

chbIncluir.Checked = tmpDatos(0) <>""

AddHandlerchbIncluir.CheckedChanged, AddressOfchbIncluir\_CheckedChanged

' ...Sufro con poner esas dos lineas

txtTapar.Visible = False

EndIf

editando = Not editando

txtNombre.ReadOnly = Not editando

dtpFecha.Enabled = editando

chbIncluir.Enabled = editando

EndSub

'' Checkean si hay cambios hechos

'' Si cambio = True, no se llamaran

PrivateSubtxtNombre\_ModifiedChanged(sender AsObject, e AsEventArgs) HandlestxtNombre.ModifiedChanged

IftxtNombre.ModifiedThen

AlternarCambioHandlers()

EndIf

EndSub

PrivateSubdtpFecha\_ValueChanged(sender AsObject, e AsEventArgs) HandlesdtpFecha.ValueChanged

AlternarCambioHandlers()

EndSub

PrivateSubchbIncluir\_CheckedChanged(sender AsObject, e AsEventArgs) HandleschbIncluir.CheckedChanged

AlternarCambioHandlers()

EndSub

' AquiyacIA mi variable, recordatorio de lo que una vez fue Y AUN ES

' cambio: I LIVED BITCH

'' Alterna el estado de cambios y activa/desactiva la deteccion de cambios

PrivateSubAlternarCambioHandlers()

'' cambio = True -> Se aniadiran los handlers, se los necesita

'' cambio = False -> Se ha detectado un cambio, y los handlers se removeran hasta la siguiente llamada del metodo

IfcambioThen

AddHandlerdtpFecha.ValueChanged, AddressOfdtpFecha\_ValueChanged

AddHandlertxtNombre.ModifiedChanged, AddressOftxtNombre\_ModifiedChanged

AddHandlerchbIncluir.CheckedChanged, AddressOfchbIncluir\_CheckedChanged

Else

RemoveHandlerdtpFecha.ValueChanged, AddressOfdtpFecha\_ValueChanged

RemoveHandlertxtNombre.ModifiedChanged, AddressOftxtNombre\_ModifiedChanged

RemoveHandlerchbIncluir.CheckedChanged, AddressOfchbIncluir\_CheckedChanged

EndIf

cambio = Notcambio

EndSub

PrivateSubdgvVSM\_CellDoubleClick(sender AsObject, e AsDataGridViewCellEventArgs) HandlesdgvVSM.CellDoubleClick

DimiAsInteger = CargarID(dt\_Video, dgvVSM)

If (i<> -1) Then

DimformVideoAsNewfrmVideo(i)

AddHandlerformVideo.FormClosed, AddressOfFormVideo\_FormClosed

formVideo.ShowDialog()

EndIf

EndSub

'' Actualiza la tabla mostrando los videos asociados cuando el formulario de mostrar video se ha cerrado

PrivateSubFormVideo\_FormClosed(sender AsObject, e AsFormClosedEventArgs)

ActualizarTabla()

EndSub

PrivateSubActualizarTabla()

dt\_Video = DevolverTabla(PSQL("id\_video, fecha as Fecha, nombre as Nombre", "video", String.Format("id\_serie = '{0}'", serieID)))

ActualizarTablaC(dt\_Video, dgvVSM)

EndSub

PrivateSubbtnBorrar\_Click(senderAsObject, e AsEventArgs) HandlesbtnBorrar.Click

DimformDeleteAsNewfrmConfirmarBorrado("Serie", {serieID}, True)

formDelete.ShowDialog(Me)

EndSub

EndClass

Controles personalizados

ColorDateTimePicker.vb

ImportsSystem.Windows.Forms.VisualStyles

PublicClassColorDateTimePicker

InheritsDateTimePicker

PublicSubNew()

MyBase.New()

SetStyle(ControlStyles.UserPaint, True)

ForeColor = Color.Black

BackColor = Color.White

DropDownAlign = LeftRightAlignment.Right

EndSub

PublicOverridesPropertyForeColorAs Color

PublicOverridesPropertyBackColorAs Color

ProtectedOverridesSubOnPaint(e AsPaintEventArgs)

Dim g As Graphics = Me.CreateGraphics()

DimdropDownRectangleAs Rectangle = NewRectangle(ClientRectangle.Width - 17, 0, 17, ClientRectangle.Height)

DimbkgBrushAs Brush = NewSolidBrush(BackColor)

DimvisualStateAsComboBoxState = ComboBoxState.Normal

g.FillRectangle(bkgBrush, 0, 0, ClientRectangle.Width, ClientRectangle.Height)

DimtextBrushAs Brush = NewSolidBrush(ForeColor)

g.DrawString(Text, Font, textBrush, 0, 2)

ComboBoxRenderer.DrawDropDownButton(g, dropDownRectangle, visualState)

g.DrawRectangle(Pens.Gray, 0, 0, ClientRectangle.Width - 1, ClientRectangle.Height - 1)

g.Dispose()

bkgBrush.Dispose()

textBrush.Dispose()

EndSub

EndClass

# Medidas de seguridad

Medidas para la ejecución del software:

El programa será ejecutado por el usuario administrado del sistema operativo.

Medidas de seguridad al momento de la utilización del Software:

Existen dos tipos de usuarios, “administrador” y “funcionarios”, que tienen implementadas funciones específicas.

Al momento de la inicialización del programa el consumidor debe ingresar su usuario y contraseña correspondiente.



## 

*Usuario administrador:*

El usuario administrador, es el que tiene todos los permisos asignados en el programa (Edición, selección, actualización, eliminar), es el único usuario que puede crear, eliminar y editar más usuarios, y es el que asigna los permisos a los mismos. Además de tener permitido todos los accesos a las áreas del programa. Este usuario será utilizado por el dueño de la empresa.

*Usuario Funcionario:*

Los usuarios funcionaros son creados por el usuario administrador, el cual le asigna los permisos correspondientes a su función y la utilización que le dará al programa. Además de que estos usuarios pueden estar restringidos a ciertas áreas del programa. Esos usuarios serán utilizados por funcionarios.

Sistema de Logueo:

Las credenciales de usuario están guardadas en la base de datos, en una tabla llamada Usuarios.

Los nombres de usuario están guardados en texto simple. Las contraseñas, por otro lado, están encriptados usando AES con una key convertida por SHA-256 al momento de ser usadas.

Las keys están almacenadas en un archivo llamado Key.txt dentro de la carpeta bin/User/ (Sujeto a cambios). Encontrándose allí una key común de 8 caracteres y una key maestra de 32 caracteres. Ambas son un string de letras y números aleatorios cada una. Si dicho archivo no existe, el programa generara un nuevo conjunto de keys y creara el archivo.

Al momento de loguearse en el programa, la contraseña ingresada es convertida con el método ya mencionado y comparado con la base de datos.

Tanto como para entrar al programa o cambiar la configuración de la base de datos, el usuario deberá ingresar credenciales validas.

Datos de Conexión a la Base de Datos:

Los datos requeridos para que el programa se conecte a la base de datos (usuario, contraseña, IP, puerto, nombre) se guardan en el archivo User.txt dentro de la carpeta bin/User/ (sujeto a cambios).

Los datos dentro de dicho archivo se encuentran encriptados en Base64 por encima de TripleDES, usando ambas keys del archivo Key.txt.

El administrador puede cambiar los datos de conexión en la ventana de Configuración. Los cambios serán guardados en el archivo mencionado anteriormente.

Además del archivo User.txt, se encontrara un archivo Origin.txt dentro de la misma carpeta, conteniendo los datos por defecto. Si el administrador no ha hecho ningún cambio, ambos archivos serán idénticos; si el archivo User.txt no existe, uno nuevo será creado con los datos de Origin.txt.

# Configuración de red:

## Planos de la distribución:



## Conexión a la red:



Sala de programas:



**Router:**

Rutas estáticas:

IP: 192.168.4.0 - 255.255.255.0 10.0.2.3

**PC’S:**

PCPrograma:

Dirección IP: 192.168.4.2

Máscara: 255.255.255.0

Gateway por defecto: 192.168.4.1

PCPrograma:

Dirección IP: 192.168.4.3

Máscara: 255.255.255.0

Gateway por defecto: 192.168.4.1

Sala de administración:



**Router:**

Rutas estáticas:

IP: 192.168.3.0 - 255.255.255.0 10.0.2.1

**PC’S:**

PCAdministración:

Dirección IP: 192.168.3.2

Máscara: 255.255.255.0

Gateway por defecto: 192.168.3.1

Sala de radio:



**Router:**

Rutas estáticas:

IP: 192.168.1.0 - 255.255.255.0 10.0.2.2

**PC’S:**

PCRadio:

Dirección IP: 192.168.1.2

Máscara: 255.255.255.0

Gateway por defecto: 192.168.1.1

PCRadio1:

Dirección IP: 192.168.1.3

Máscara: 255.255.255.0

Gateway por defecto: 192.168.1.1

PCRadio2:

Dirección IP: 192.168.1.4

Máscara: 255.255.255.0

Gateway por defecto: 192.168.1.1

PCRadio5:

Dirección IP: 192.168.1.5

Máscara: 255.255.255.0

Gateway por defecto: 192.168.1.1

## Configuración de router:

***Router Radio:***

Router#configure terminal

Router(config)#hostname Router Radio

Router(config)#enable secret redes

Router(config)#interface fastEthemet 0/0

Router(config-if)#ip address 192.168.1.0 - 255.255.255.0

Router(config-if)#no shutdown

Router(config)#exit

Router(config)#interface serial 0/1

Router(config-if)#ip address 10.0.2.2 255.255.255.252

Router(config-if)#no shutdown

Router(config-if)#exit

Router(config)#line console 0

Router(config-line)#password redes

Router(config-line)#login

Router(config-line)#line vty 0 4

Router(config-line)#password redes

Router(config-line)#login

Router(config-line)#exit

Router(config)#exit

***Router Programa:***

Router#configuracion terminal

Router(config)#hostname Router Admnistración

Router(config)#enable secret redes

Router(config)#interface fastEthemet 0/0

Router(config-if)#ip address 192.168.4.0 - 255.255.255.0

Router(config-if)#no shutdown

Router(config)#exit

Router(config)#interface serial 0/1

Router(config-if)#ip address 10.0.2.3 255.255.255.252

Router(config-if)#no shutdown

Router(config-if)#exit

Router(config)#line console 0

Router(config-line)#password redes

Router(config-line)#login

Router(config-line)#line vty 0 4

Router(config-line)#password redes

Router(config-line)#login

Router(config-line)#exit

Router(config)#exit

***Router Administración:***

Router#configuracion terminal

Router(config)#hostname Router Admnistración

Router(config)#enable secret redes

Router(config)#interface fastEthemet 0/0

Router(config-if)#ip address 192.168.3.0 - 255.255.255.0

Router(config-if)#no shutdown

Router(config)#exit

Router(config)#interface serial 0/1

Router(config-if)#ip address 10.0.2.1 255.255.255.252

Router(config-if)#no shutdown

Router(config-if)#exit

Router(config)#line console 0

Router(config-line)#password redes

Router(config-line)#login

Router(config-line)#line vty 0 4

Router(config-line)#password redes

Router(config-line)#login

Router(config-line)#exit

Router(config)#exit

# 

# Anexo

## S.R.L.

Contraro Sociedad de Responsabilidad Limitada

En la ciudad de Salto, en fecha de 6/4/2020 reunidos los señores Nahuel Pacheco con el número de cedula 5.432.959-1, Milagros Núñez con el número de cedula 5.257.879-8, Mateo Cabral con el número de cedula 5.538.392-2 y Manuel Buslón con la cedula número 5.328.350-8 resuelven celebrar el siguiente contrato de Sociedad de Responsabilidad Limitada que se regirá por las cláusulas que se indican a continuación:

PRIMERO. (Denominación) La naturaleza de la sociedad será de responsabilidad limitada, para la que adoptan la denominación de Nu-CaBuPa.SRL. Se regirá por la Ley 16.060, el decreto 155/010 de 24 de mayo de 2010, y demás normas acordadas. Los socios

SEGUNDO. (Domicilio) La sociedad tendrá su domicilio en el departamento de Salto, pudiendo establecer sucursales, filiales, agencias o representaciones en todo el país o en el extranjero.

TERCERO. (Plazo) El plazo de duración de la sociedad será de 8 años a contar de hoy, no obstante, cada año, a contar desde hoy, cualesquiera de los socios podrá hacer uso del derecho de renuncia, debiendo en tal caso hacer saber su decisión con una anticipación no menor de tres meses al vencimiento del período respectivo, a los demás socios por telegrama colacionado.

CUARTO. (Objeto) La sociedad tendrá por objeto la realización de la siguiente actividad: A) Desarrollo de Software. B) Gestión, administración, soporte de los productos desarrollados por nuestra empresa; C) Ayuda técnica con el uso del sistema informático D) realización de toda clase de actos y/u operaciones civiles, comerciales, industriales, ya sea por cuenta propia, de terceros o de ambos a la vez, pudiendo a tales efectos comprar, vender, hipotecar, caucionar, administrar, explotar, arrendar y dar en arrendamiento, cualquier clase de bienes y derechos, sean muebles, inmuebles, corporales o incorporales, pudiendo a tal efecto participar en toda clase de sociedades y empresas existentes o que se constituyan en el futuro. La sociedad podrá realizar si lo estima conveniente para el mejor desarrollo de la empresa o para un integral aprovechamiento de su capital, cualquier acto, negocio, o contrato de cualquier naturaleza y de cualquier ramo de actividad comercial. Como medios adecuados para el cumplimiento de sus objetivos, la sociedad podrá realizar todos los actos jurídicos referidos para el desarrollo de sus negocios, tanto sean de disposición, afectación y administración, otorgando y efectuando cualquier clase de operaciones ya fueran de enajenación, adquisición, afectación y gravámenes sobre toda clase de bienes y derechos, sin limitación alguna, pudiendo al efecto participar en toda clase de sociedades y empresas existentes o que se constituyan en el futuro; E) Implementación, diseño, compra, venta de equipos informáticos y sistemas de redes de ordenadores; F) Desarrollo de contenido audio visual con fines de entrenamiento; G) Traducción y exposición de cursos y conferencias en relación al ámbito tecnológico-psicológico-filosófico-matemático; H) Instalación y obtención de software relacionado a herramientas de edición de texto.

QUINTO. (Capital) El capital de la sociedad se fija en la suma de $288.640 (pesos uruguayos que quedan divididos en 40 cuotas)

Cuotas de $7.216 cada una, correspondiéndole a los socios Sres.

La sociedad estará conformada por: Manuel Buslón, Mateo Cabral, Milagros Nuñez, Nahuel Pacheco. Cada uno aportando 10 cuotas, equivalentes a $72.160, lo cual ya fue pagado.

SEXTO. (Aportes) Por su obligación de aportar, cada uno de los socios manifiesta que aportó a la sociedad antes de este acto y en efectivo, la suma correspondiente a sus cuotas, con lo que queda integrado en su totalidad el capital societario previsto, razón por la cual se otorgan recíprocas cartas de pago por sí y en representación de la sociedad.

SEPTIMO. (Calidad de los socios) La responsabilidad de los socios quedará limitada a la cantidad o cantidades estipuladas como aporte de capital en el contrato social. ————————————–

OCTAVO. (Administración) La administración de la sociedad y el uso de la firma social, con las más amplias facultades y con el ejercicio de la representación de la empresa estarán a cargo de cualquiera de las socias, actuando indistintamente, sin perjuicio de la facultad de otorgar mandatos a terceros. A título enunciativo se establece que el administrador en nombre y representación de la sociedad podrá: a) realizar toda clase de actos de disposición, administración y afectación quedando facultado para enajenar y/o gravar toda clase de bienes; b) celebrar toda clase de contratos; c) ejecutar todo género de actos de administración; d) otorgar todo tipo de mandatos; e) representar a la sociedad ante cualquier autoridad judicial o administrativa; f)suscribir todo tipo de documentos civiles y/o comerciales; g) registrar todo tipo de Marcas y Patentes. El administrador tendrá todas las facultades necesarias para el gobierno, administración y disposición de los bienes de la sociedad, así como también la representación judicial, extrajudicial y administrativa de la misma, y podrá actuar por sí o hacerse representar por apoderado con facultades suficientes. El domicilio del administrador será el indicado como suyo en la comparecencia de este contrato. El administrador deberá suscribir la documentación de la siguiente forma: por S.R.L. seguido de su firma habitual. ——————————————–

NOVENO. (Inventario-balance) Anualmente se efectuará un inventario-balance al cierre del ejercicio económico el que deberá estar concluido dentro de los ciento veinte (120) días siguientes a la finalización del ejercicio. Los socios tendrán las más amplias facultades para controlar la confección de los balances y formular por escrito las observaciones que les merezcan. Los socios acuerdan que el ejercicio económico finalice cuando lo establezca la asamblea de socios. ————

DECIMO. (Fondo de reserva.) De las ganancias líquidas de cada ejercicio se deducirá un diez por ciento (10%) para formar un fondo de reserva, hasta que éste alcance un cien por ciento (100%) del capital social. ———————————————————————————–

DECIMO PRIMERO. (Distribución de ganancias y pérdidas) El resto de las ganancias serán repartidas y las pérdidas soportadas, entre los socios en proporción de sus aportes. Las pérdidas de cada ejercicio serán compensadas con las utilidades del o de los ejercicios subsiguientes.—————————————————————

DECIMO SEGUNDO. En caso de ausencia, fallecimiento o incapacidad de cualquiera de los socios, la sociedad continuará entre los demás socios.——– ——————————————————————

DECIMO TERCERO. La sociedad se disolverá cuando así lo decida la Asamblea Extraordinaria convocada a tales efectos y será ésta quien decidirá la o las personas que realizarán la liquidación de los bienes sociales fijando sus atribuciones y remuneraciones. Una vez liquidados los bienes sociales del activo y abonadas todas las obligaciones de la sociedad, el remanente será adjudicado a los socios en proporción a sus respectivos capitales. La sociedad puede disolverse por las siguientes causales: A); Por decisión unánime de los socios; B) Por expiración del plazo; C) Por pérdidas que reduzcan el patrimonio social a una cifra inferior a la cuarta parte del capital social integrado; D) Por reducción a uno del número de socios según lo dispuesto en el art. 156 de la ley 16.060. ————————————————————

DECIMO CUARTO. (Cesión de cuotas sociales). Las cuotas sociales podrán ser cedidas por cualquiera de los socios conforme a las disposiciones legales. Cuando un socio desee ceder su cuota a un tercero ajeno a la sociedad, se convocará a una Asamblea Extraordinaria, diez (10) días después de que el socio interesado comunique por telegrama colacionado la intención de ceder su cuota. En tal caso la cesión deberá ser aceptada por unanimidad. Si no se notificara la oposición se presumirá el consentimiento. Si se formulara alguna oposición se seguirá el procedimiento previsto por el art. 232 de la ley 16.060. La sociedad tendrá prioridad frente a los socios para la adquisición de las cuotas. Si hubiera varios socios interesados en la adquisición, las cuotas se distribuirán a prorrata y si no fuera posible, se atribuirán por sorteo.———————————————————————————–

DECIMO QUINTO. Los socios podrán por unanimidad, fijarse remuneraciones en calidad de sueldo y otros conceptos, así como autorizar retiros a cuenta de utilidades con débito a sus respectivas cuentas. —————————————————————-

DECIMO SEXTO. Los socios se obligan muy especialmente a aportar su trabajo a los negocios de la sociedad, pudiendo no obstante realizar operaciones mercantiles por su cuenta o como integrante de otras empresas, siempre que su realización no perjudique a la sociedad, ni menoscabe la actividad que le corresponda desarrollar en la misma. —-

DECIMO SEPTIMO. Anualmente dentro de los ciento veinte (120) días de cerrado el ejercicio económico se celebrará una Asamblea Ordinaria que tendrá por objeto aprobar el balance, examinar y juzgar la gestión de la administración, así como la cuenta de ganancias y pérdidas, establecer el porcentaje destinado al capital de reserva cuando éste fuera mayor al fijado, y resolver cualquier otro punto que se encuentre en el orden del día. Los miembros de la Asamblea serán citados por telegrama colacionado, con una anticipación de por lo menos diez (10) días a la fecha de realización de la misma. La Asamblea Extraordinaria podrá ser convocada en cualquier momento, por mayoría de socios, debiendo comunicar por escrito a los restantes la realización de la misma con una antelación no menor a diez (10) días. El quórum necesario para sesionar y tomar resoluciones en Asambleas Ordinarias o Extraordinarias, salvo disposición en contrario, estará constituido por la mayoría de los socios que representen la mayoría del capital social, correspondiéndoles a tales efectos un voto por cuota social. ——————————————

DECIMO OCTAVO. Queda prohibido a la sociedad constituirse fiadora de los socios y de terceras personas. ———————————–

DECIMO NOVENO. La asamblea social, por decisión de la mayoría de socios que represente las tres cuartas partes del capital social podrá tomar las siguientes decisiones: a) transformar a la SRL en sociedad anónima; b) fusionarla con otra; c) modificar el objeto social; d) revocar y nombrar administradores; e) prorrogar el plazo de la sociedad. Los socios disidentes o ausentes tendrán derecho a receso. ———————————————————————————-

VIGESIMO. Los socios podrán renunciar a la sociedad en los siguientes casos: a) en caso establecido en la cláusula tercera de este contrato; b) si el ejercicio social diere una pérdida que excediere el 30% del capital social; c) si no se conformaren con la resolución que tomen los restantes socios respecto a lo previsto en la cláusula décimo novena de la presente. En tales casos la participación del socio saliente se determinará conforme al balance practicado del día de la manifestación de voluntad, y la cuota que correspondiere le será abonada por la sociedad hasta en cuatro mensualidades consecutivas, iguales a contar de la fecha del balance, las que devengarán el máximo interés legal, calculado sobre los saldos deudores pagadero conjuntamente con el capital. ———————————————————————————

VIGESIMO PRIMERO. Cualquier diferencia o controversia entre los socios, sea por aplicación o interpretación de este contrato, será resuelto inapelablemente por árbitros, designados uno por cada una de las partes discrepantes y un tercero designado por los árbitros ya designados. La designación se hará dentro del plazo de 30 días a contar del telegrama colacionado que el discrepante deberá enviar a la administración de la sociedad, manifestando su disconformidad. —–

VIGESIMO SEGUNDO. En todo lo que no estuviere previsto en este contrato se aplicarán las disposiciones de la ley 16.060 sección IV, “De las sociedades de responsabilidad limitadas” y sus concordantes.———

**ENTREVISTA**

# Entrevista:

Luego de la presentación de nuestro grupo, contando quiénes somos y lo que podemos ofrecer al cliente se dio la entrevista de requerimientos.

Grupo: ¿Puede contarnos qué tipo de actividades realiza su empresa?

Cliente: Principalmente la gestión y el control de la radio Bemba

Grupo: ¿Y cómo se estuvo administrando los datos de la radio hasta ahora?

Cliente: Tengo una libreta en el que anoto los pagos de cada programa y más o menos los horarios son sobre lo que se recuerda y lo que queda hablado con el dueño de cada programa. Las publicidades dependen de la tanda en que se contrataron y queda guardado lo que se debe decir o mostrar en la computadora

Grupo: ¿Y de esos programas le interesaría almacenar alguna información?

Cliente: Solo más o menos saber que conductores hay y los pagos de cada mes por parte del dueño del programa. También estaría el operador pero es casi siempre el mismo conductor el que hace esa función y actúa como el dueño.

Grupo: ¿Los precios para los programas están definidos antes o se acuerdan con el que alquilaría el espacio?

Cliente: Se fijan en el momento teniendo en cuenta lo que duraría, repeticiones por semana y de que trata

Grupo: Entonces ¿un programa puede tener varios días distintos cada semana?

Cliente: Sí, y eso puede cambiar

Grupo: ¿Y el tema de publicidades cómo es?

Cliente: En el momento de la tanda entre cada programa se comentan o muestran las publicidades que contratan las empresas, cada una con las repeticiones que pagó. De las publicidades que aparecen en cada programa no interesaría, ya que el dueño del programa se encarga de eso.

Grupo: De eso ¿solo le interesaría saber lo de los pagos, la tanda en que aparecen y las repeticiones entonces?

Cliente: Sí, me gustaría que quede guardado algo sobre publicidades que han sido contratadas y los pagos para luego calcular. ¿Se podrá?

Grupo: Sí, se podría a base de la información del pago de los programas y las publicidades hacer que el software calcule las ganancias de los meses o días que se quiera saber. ¿En la empresa quienes usarían el software?

Cliente: Principalmente yo y otro que esté ahí ayudando a registrar las cosas, el operador de la radio puede ser. También quisiera poder usarlo desde casa u otro lugar.

Grupo: Bueno, podemos hacer que se administre de forma online. ¿Tiene un servidor propio?

Cliente: Sí, tengo uno en el que está la página web de la radio y en el que se retransmite la misma

Grupo: ¿Qué equipos tiene? ¿Son equipos de uso propio o exclusivos para la administración de la radio?

Cliente: Mis equipos de administración son de alta eficiencia, en la radio las computadoras son exclusivos para los programas, mientras que la computadora que tengo en mi casa está más para el trabajo.

Grupo: Está bien, en una futura entrevista acordamos como implementamos eso y qué características tiene el servidor si se puede usar ese. ¿Hay algo más que quisiera mantener en los registros?

Cliente: Cada tanto hay eventos que se realizan con algún tema por la empresa, lo importante sería saber la fecha y de que trata más o menos. Después está lo de Mil voces y otras series que produciríamos, de eso es solo tener que video de la serie se va a emitir y de que trata.

Grupo: ¿Esos eventos tienen relación con algo de la radio o las series?

Cliente: Puede tratar de eso sí, estar en conjunto con algún programa sobre el que se realice o por algún video o tema de la serie para promocionarlo. En el evento se mostraría publicidades también.

Grupo: ¿Algo más que quiera poder administrar?

Cliente: Por ahora serían solo esas cosas, si pienso algo más les envío un mensaje.

Grupo: ¿Le gustaría mantener control de las publicidades que aparecerían en algún programa de la radio?

Eso dentro del software

Cliente: Sí, me gustaría, claro si es posible.

# Consultas de SQL

# Consultas de SQL

#Se utilizan los “describe” directamente en el software para obtener nombre y tipo de las columnas de las tablas, recibiéndolas y aplicando funciones y métodos. Es necesario para poder generalizar funciones y que sean independientes, ahorrando el incluir todo.

describe Empresa

describe video

describe Serie

#Los delete tienen su propia sección en la cual permite editar o borrar los datos del software, para eso existe un método que recibe el nombre de la tabla, la columna(s) a comparar un array de los datos a borrar. Esta crea la sentencia sql y utiliza un in ya que usualmente recibe varios datos como se ve abajo.

DELETE FROM video WHERE id\_serie in ('9', ‘6’)

DELETE FROM Video WHERE id\_video in ('6', ‘1’, ‘9’ )

DELETE FROM Serie WHERE id\_serie in ('8' , ‘5’ )

DELETE FROM Empresa WHERE id\_empresa in ('2' )

#Esta sentencia se utiliza para verificar si existe el usuario ingresado en el login, utiliza el @nombre remplazando eso por el nombre ingresado y el @contraseña para lo mismo, a esto se utiliza un sha2 en conjunto a una key almacenada en los archivos del software, sin la cual el login no funcionará, a esto a su vez se aplica un aesencrypt.

select id\_usuario FROM usuarios as User WHERE nombre = @nombre AND contrasena = AES\_ENCRYPT(@contrasena,sha2(@key,256))

#Se selecciona la hora de inicio y final de las próximas tandas, esto se mostrará en el menú principal y permite seleccionar una para ver sus publicidades

selecttime\_format(hora\_inicio, '%H:%i') as 'Inicio', time\_format(hora\_fin, '%H:%i') as 'Final' FROM tanda WHERE (hora\_inicio<= curtime() and hora\_fin>= curtime()) orhora\_inicio>= curtime()

#Se selecciona el id, la hora de inicio, de fin y el nombre de los programas del día actual, aunque el dato cambia dependiendo de lo seleccionado en un datatimepicker. Esto se modificará, se oculta el id y se lo carga en una tabla.

select p.id\_programa, time\_format(hora\_inicio, '%H:%i') as 'Inicio', time\_format(hora\_fin, '%H:%i') as 'Final', Nombre\_programa as 'Programa' FROM fechaprograma f inner join programa p on f.id\_programa=p.id\_programa WHERE fecha = '2020-09-26'

#Se selecciona la fecha de los próximos eventos para mostrarlos en una lista

select e.id\_Evento, DATE\_FORMAT(Fecha,'%d/%m/%Y') as Fecha, Nombre FROM evento e inner join fechaevento f on f.id\_evento=e.id\_evento WHERE f.fecha>= now()

#Se cargan los temas de una publicidad de la tanda seleccionada en el día seleccionado, por defecto es el día actual.

select Tema FROM publicidad p innerjoinaparecepubli a onp.id\_publicidad=a.id\_publicidadinnerjoin tanda t ont.Hora\_Inicio = a.hora\_inicio WHERE a.fecha\_inicio<= '2020-09-26' and a.fecha\_finalizacion>= '2020-09-26' and t.hora\_inicio = '15:00:00'

#Muestra el nombre y teléfonos de los funcionarios que trabajan en un programa seleccionado

select Nombre, Telefono FROM programa p innerjoinfuntrabaja f onf.id\_programa=p.id\_programainnerjoin funcionario ffonff.id\_funcionario = f.id\_funcionario WHERE p.id\_programa = '3'

#Se utiliza para mostrar la descripción del programa seleccionado

select Descripcion FROM programa WHERE id\_programa = '3'

#Similar a las publicidades cargadas para las tandas en esta sección se cargan las publicidades del programa seleccionado en el día que se desee

select Tema FROM programa p inner join pmuestrapubli pp on p.id\_programa=pp.id\_programa inner join publicidadppp on pp.id\_publicidad = ppp.id\_publicidad WHERE pp.fecha\_inicio<= '2020-08-10' and pp.fecha\_finalizacion>= '2020-08-10' and pp.id\_programa = '1'

#Esta es necesaria para mostrar los datos de los videos, utilizamos en where true debido a que las consultas están generalizadas y requieren tres parámetros para el método, se puede simplemente eliminar, pero preferimos mostrar tal como quedan los logs del software

#Realizamos un select para la búsqueda de videos en su interfaz correspondiente, la subconsulta es útil porque ayuda a que aunque el video no tenga serie este se mostrará en la interfaz, como es una relación de 1 a n no hay posibilidad de que devuelva más de uno.

selectid\_video, fecha as Fecha, v.nombre as Nombre, (selects.nombrefrom serie s wheres.id\_serie=v.id\_serie) as Serie FROM video v WHERE true

#Se selecciona el contenido, nombre y serie de los videos para cuando selecciona alguno y lo quiere ver o editar, estos datos seleccionados se arreglan para mostrar una interfaz con ellos, el id\_serie se usa para seleccionas la serie en un combo box de las series a las que se puede asociar.

select contenido, Nombre, id\_serie, DATE\_FORMAT(Fecha,'%Y-%m-%d') as Fecha FROM video WHERE id\_video = '6'

#Se seleccionan las series para mostrar en la sección de asignación en la edición o agregación de un video, el id se oculta y guarda en otra parte.

select id\_serie, nombre FROM Serie WHERE True

#Se hace un update para actualizar los datos del video a los elegidos por el usuario, el null es para quitarlo de una serie o su fecha de emisión.

update video set contenido = 'sid',nombre = 'no',ID\_Serie = null,fecha = null WHERE ID\_Video = '6'

#Se actualizan los datos de una serie desde su formulario de edición y al guardar se actualizan con lo ingresado en los campos

update Serie set fecha\_finalizacion = null,nombre = 'holid' WHERE ID\_Serie = '6'

#Se seleccionan las series con sus nombres y fechas para el menú de búsqueda

selectid\_serie, fecha\_finalizacion as Fecha, nombre as Nombre FROM serie WHERE true

#Se muestran los temas de las publicidades asociadas a una empresa en su menú de datos

selectID\_Publicidad, Tema FROM publicidad WHERE ID\_Empresa = '1'

#Se actualizan los datos de la empresa en su interfaz de edición

update Empresa set Nombre = 'Mil',Telefono = '092521203',Mail = 'dd@gmail.com' WHERE ID\_Empresa = '1'

#Se ingresa un video con los datos que el usuario desea

Insertinto video ( contenido,nombre,fecha ) values ('None','Voces de ...','2020-02-02' )

#Se ingresa una serie con los datos que el usuario desea

Insertinto serie ( ID\_Serie,nombre ) values ('10','Voces' )

#Se ingresa un video con los datos que el usuario desea pero desde una interfaz de administración, dicha interfaz solo se puede acceder con permisos de administrador y permite ingresar datos en todas las tablas de forma más accesible que en consola

Insertinto video ( ID\_Video,nombre,ID\_Serie ) values ('15','None','10' )

#Se ingresa una empresa con los datos que el usuario desea

Insertinto empresa ( Nombre ) values ('NuCaBuPA' )

### Proyecto S.O

1. Crear un script que permita crear un árbol de directorios (con subdirectorios y archivos). En primer lugar se debe preguntar cuál será el directorio “base” del árbol, que deberá crearse dentro del directorio personal del usuario utilizado.

Posteriormente se deben mostrar opciones dando la posibilidad de crear un archivo o subdirectorio, y en ambos casos preguntar dentro de qué rama del árbol se lo desea crear. A medida que se van agregando archivos y subdirectorios, deberá desplegarse el árbol correspondiente para poder verificar que se hizo en forma correcta.

Implementarse **al menos 2 funciones** de modo de no repetir código innecesariamente: una para crear directorio y otra para crear archivo.

2. Utilizando el árbol creado anteriormente, crear otro script que permita renombrar una lista de archivos, cambiando alguno de sus caracteres. En primer lugar se deberá mostrar el árbol de directorios junto a archivos, preguntar en qué directorio se desea hacer la modificación, luego el carácter a ser cambiado y por cuál. Antes de proceder al cambio, mostrar qué se renombrará y pedir confirmación, en cuyo caso se procederá a realizarlo.

#### Script 1:

#!/bin/bash

Crear () {

    printf "%s\n" "Desea crear un archivo o subdirectorio?" "1)Archivo" "2)Subdirectorio" "Otro)Salir"

    read OP

    Eleccion=""

    if [ $OP = "1" ]; then

        Eleccion="archivo"

    elif [ $OP = "2" ]; then

        Eleccion="subdirectorio"

    else

        exit

    fi

    clear

    RecD $TD

}

Nombre(){

    printf "%s\n" "Indique un nombre para el $Eleccion"

    read Nombre

    if [ $OP = "1" ]; then

        touch "$DirTemp/$Nombre"

    elif [ $OP = "2" ]; then

        mkdir "$DirTemp/$Nombre"

    else

        Error "Nombre"

    fi

    clear

    printf "%s\n" "$Eleccion creado exitosamente en $DirTemp"

    tree $TD

    Crear

}

#Recorre el directorio

RecD () {

    printf "%s\n" "Donde desea crearlo?"

    tree $TD

    read nombre

    if [ nombre = "" ]; then

        exit

    fi

    Direcciones=($(find $TD -type d -name "$nombre"))

    Result=${#Direcciones[@]}

    if [ $Result -eq 1 ] && [ "${Direcciones[0]}" != "" ]; then

        DirTemp=${Direcciones[0]}

        printf "%s\n" "Encontrado: $DirTemp"

        Nombre

    elif [ $Result -gt 1 ]; then

        printf "%s\n" "A cual directorio se refiere?"

        Num=0

        iter=`expr $Result - 1`

        for i in `seq 0 $iter`; do

            printf "$Num)%s\n" "${Direcciones[$i]}"

            Num=`expr $Num + 1`

        done

        read Num

        eleccion=${Direcciones[$Num]}

        echo $eleccion

        if [ -d "$eleccion" ]; then

            DirTemp=$eleccion

            Nombre

        else

            Error "RecD"

        fi

    else

        Error "RecD"

    fi

}

Error(){

        printf "%s\n" "No se encontró" "Ingrese cualquier caracter para continuar o 0 para salir"

        read Salir

        if [ $Salir -eq 0 ]; then

            exit

        else

            $1

        fi

}

TD=""

DirTemp=""

OP=""

#Se pregunta cual es la dirección en la que se creará el arbol

printf "%s\n" "Indique la dirección en la que se creará el arbol"

read TD

#Verificamos que no sea camino absoluto y de que exista

TD=$(echo $TD | sed 's/^\///')

TD="$HOME/"$TD

if [ -d "$TD" ] ; then

    printf "%s\n\n" "Directorio encontrado"

else

    printf "%s\n\n" "Directorio creado"

    mkdir $TD

fi

Crear

#### Script 2:

Letra(){

    printf "%s\n" "Indique el caracter a ser cambiado"

    read Letra

    Verificacion=$(echo $Letra | egrep "^[a-Z0-9]$")

    if [ Verificacion = "" ]; then

        Error "Letra"

    fi

    Letra2

}

Letra2(){

    printf "%s\n" "Indique el caracter por el cual reemplazar"

    read Letra2

    Verificacion=$(echo $Letra2 | egrep "^[a-Z0-9]$")

    if [ Verificacion = "" ]; then

        Error "Letra2"

    fi

    Cambiar

}

Cambiar(){

    clear

    printf "%s" $(find $DirTemp -maxdepth 1 -type f | sed "s/^.\*\///" | sed "s/.\*/'&'/")

    printf "\n%s\n\n" "Pasará a ser:"

    printf "%s" $(find $DirTemp -maxdepth 1 -type f | sed "s/^.\*\///" | sed "s/$Letra/$Letra2/g" | sed "s/.\*/'&'/")

    printf "\n%s\n\n" "Para aceptar inserte 'y', para salir inserte cualquier otro"

    read Salir

        if [ "$Salir" = "y" ]; then

            Renombrar

        else

            exit

        fi

}

Renombrar(){

    Archivos=$(find "$DirTemp" -maxdepth 1 -type f | sed "s/.\*/'&'/")

    for i in $Archivos

    do

        OldName=$(echo "$i" | sed "s/'//g")

        NewName="$DirTemp/$(echo "$i" | sed "s/^.\*\///" | sed "s/$Letra/$Letra2/g" | sed "s/'//")"

        if [ $OldName != $NewName ]; then

            mv $OldName $NewName

        fi

    done

    tree "$TD"

    printf "\n%s\n\n" "Si quiere seguir modificando inserte 'y', para salir inserte cualquier otro"

    read Salir

        if [ "$Salir" = "y" ]; then

            Letra

        else

            exit

        fi

}

RecD () {

    printf "%s\n" "Donde desea modificar?"

    tree "$TD"

    read nombre

    if [ nombre = "" ]; then

        Direcciones=($(find "$TD" -type d))

    else

        Direcciones=($(find "$TD" -type d -name "$nombre"))

    fi

    Result=${#Direcciones[@]}

    if [ $Result -eq 1 ] && [ "${Direcciones[0]}" != "" ]; then

        clear

        DirTemp=${Direcciones[0]}

        printf "%s\n" "Encontrado: $DirTemp"

        Letra

    elif [ $Result -gt 1 ]; then

        clear

        printf "%s\n\n" "A cual directorio se refiere?"

        Num=0

        iter=`expr $Result - 1`

        for i in `seq 0 $iter`; do

            printf "$Num)%s\n" "${Direcciones[$i]}"

            Num=`expr $Num + 1`

        done

        read Nume

        eleccion=${Direcciones[$Nume]}

        echo $eleccion

        if [ -d "$eleccion" ]; then

            DirTemp=$eleccion

            Letra

        else

            Error "RecD"

        fi

    else

        Error "RecD"

    fi

}

Error(){

        printf "%s\n" "No se encontró" "Ingrese cualquier caracter para continuar o 0 para salir"

        read Salir

        if [ $Salir -eq 0 ]; then

            exit

        else

            $1

        fi

}

Inicio(){

TD=""

DirTemp=""

OP=""

#Se pregunta cual es la dirección en la que se creará el arbol

printf "%s\n" "Indique la dirección en la que se modificará el arbol"

read TD

#Verificamos que no sea camino absoluto y de que exista

TD=$(echo $TD | sed 's/^\///')

TD="/home/$USER/$TD"

if [ -d "$TD" ] ; then

    printf "%s\n\n" "Directorio encontrado"

else

    Error "Inicio"

fi

RecD

}

Inicio