**KpiAdmin**

**Base de Datos**

El sistema está contemplado en una base de datos Master donde se encuentran los datos generales de todo el ecosistema.

La base de datos se llama **db\_sun.sql** Nuestra base de datos **MASTER**.

* Contiene los datos de los servidores y datos de conexión de las base de datos **DATAWAREHOUSE**.
* Preferiblemente mantener los datos de conexión en la base de datos. En la tabla clientes. No en un archivo dentro de la aplicación en Laravel. Ya que es más fácil de manipular, y realizar los backups.
* Contiene la información básica para que los usuarios puedan realizar el **login**, (email y password). Tabla **usuario\_clientes**

Luego existe la base de datos del usuario final.

En este primer caso la base de datos se llama **db\_kpi\_admin\_isuneon.**

* Llamaremos a esta  base de datos **DATAWAREHOUSE.**
* Base de datos en la que trabajara la aplicación.
* Pueden existir **N** base de datos.
* Puede estar en otro servidor.
* Todas las bases de datos **DATAWAREHOUSE** tendrán la misma estructura, módulos y roles para los usuarios (solo cambiara la data de las tablas de usuarios y las tablas con los prefijos **cxc\_**, **venta\_**, **cxp\_**, etc... ).
* Las bases de datos se crearan manualmente por el administrador de la base de datos.

**Flujo del sistema**

Paso 1

EL usuario ingresa email y password en la pantalla de login.

El sistema recibe el email y password y realiza el siguiente query para extraer los datos correspondientes al usuario que se está autenticando. Este Query está en **Store Procedure sp\_usuario\_clientes** .

SELECT a.id,a.name,a.email,a.password,a.remember\_token,a.co\_cli,a.nombre,a.apellido,a.descripcion,d.activo AS cliente\_activo,a.created\_at,a.updated\_at,a.deleted\_at, b.co\_lice,b.co\_tipo\_lice,b.co\_pro,b.dw\_server,b.dw\_dbname,b.dw\_user,b.dw\_pass,b.dw\_port,b.activa AS lice\_activa, c.fe\_inicio,c.fe\_fin

FROM usuario\_clientes AS a

INNER JOIN lice AS b ON a.co\_cli = b.co\_cli

INNER JOIN liced AS c ON b.co\_lice = c.co\_lice

AND b.co\_tipo\_lice = c.co\_tipo\_lice

AND b.co\_pro = c.co\_pro

INNER JOIN clientes AS d ON a.co\_cli = d.co\_cli

WHERE a.email = lcEmail AND a.password =lcPassword;

Con los datos arrojados por este Store Procedure obtendremos lo siguiente:

a.id  = id del usuario en la tabla usuario\_clientes.

a.name = name o seudónimo del usuario en la tabla usuario\_clientes.

a.email = email del usuario en la tabla usuario\_clientes.

a.password = password del usuario en la tabla usuario\_clientes.

a.remember\_token =

a.co\_cli = campo único e identificador del cliente.

a.nombre = nombre del usuario en la tabla usuario\_clientes.

a.apellido = apellido del usuario en la tabla usuario\_clientes.

a.descripcion = descripcion del usuario en la tabla usuario\_clientes.

d.activo AS cliente\_activo = indica si el cliente asociado al usuario está activo.

a.created\_at = fecha de creación del registro.

a.updated\_at = fecha de actualización del registro.

a.deleted\_at = fecha de eliminación del registro.

b.co\_lice = número de licencia, asociada a un cliente, campo único.

b.co\_tipo\_lice = tipo de licencia.

b.co\_pro = código de producto (en este caso es KPIADMIN).

b.dw\_server = dirección del servidor de la base de datos del cliente.

b.dw\_dbname = nombre de la base de datos del cliente.

b.dw\_user = usuario de la base de datos del cliente.

b.dw\_pass = password de la base de datos del cliente.

b.dw\_port = puerto de la base de datos del cliente.

b.activa AS lice\_activa = indica si la licencia del cliente está activa.

c.fe\_inicio = fecha de inicio de la licencia.

c.fe\_fin = fecha de caducidad de la licencia.

**Datos a validar**

1 - Cliente Activo

d.activo AS cliente\_activo

Si es 0 no hace login y muestra mensaje de **Cliente Inactivo**

2 - Licencia Inactiva

b.activa AS lice\_activa

Si es 0 no hace login y muestra mensaje de **Licencia Inactiva.**

3 - Licencia caducada

Compara la fecha actual con c.fe\_fin

Si la fecha actual y hora actual es mayor a c.fe\_fin

No hace login y muestra el mensaje **Su licencia a caducado.**

4 - Producto

Compara b.co\_pro con la palabra **KPIADMIN**

Si no es igual

No hace login y muestra el mensaje **Sus credenciales no corresponden al producto seleccionado**.

Nota: Las validaciones son en ese mismo orden de prioridades, siendo “1” la de mayor prioridad y “4” la de menor prioridad.

Paso 2

Con el resultado extraído del store procedure, se procede a conectarse a la base de datos **DATAWAREHOUSE,**  con los datos correspondientes.

b.dw\_server, b.dw\_dbname, b.dw\_user, b.dw\_pass, b.dw\_port.

Aquí es donde finalmente trabajara la aplicación.

La aplicación estará basada por módulos los cuales serán asignados a los usuarios, según asigne el usuario administrador del sistema.

Los cuales son:

BANCOS, COMPRA,CXC,CXP,DASH1,DASH2,DASH3,ORDCOM,PPTO,VENTAS,CUSTOM1, CUSTOM2,CUSTOM3

Estos módulos se encuentran en **db\_sun.tipo\_liced**, incluyendo el módulo de administración.

Para administrar los módulos. Usaremos el paquete de laravel “**zizaco/entrust**”

El cual se le asignara un permiso (modulo) por usuario.

Puntos a tener en cuenta, todas las bases de datos tendrán información predefinida tales como (esta información es cargada manualmente no es creada por el sistema):

* Módulos del sistema.
* Un usuario administrador, el cual creara otros usuarios y asignara permisos en la aplicación.
* Las consultas que se realicen a la base de datos incluyendo la de usuarios deben ser por medio de store procedures.

**Pruebas**

Para esta primera fase como solo probaremos el acceso de los distintos usuarios a sus distintos módulos, asignados por el/los usuarios administradores.

Y actualización de las tablas con el prefijo admin.

Se crearan usuarios con distintos accesos. Cada usuario podrá ver los módulos que el usuario administrador le asigno.

Debe ser multilenguaje. (Español, Ingles)

Las rutas del login y el template base son:

Login

'Auth\login'

Template base

'home'

**Tablas Iniciales para la fase 1**

SELECT \* FROM `admin\_config`;

SELECT \* FROM `admin\_horarios\_config`;

SELECT \* FROM `admin\_horarios\_dias`;

SELECT \* FROM `admin\_dia`;

SELECT \* FROM `admin\_horarios\_horas`;

SELECT \* FROM `admin\_notificacion`;

SELECT \* FROM `admin\_correos`;

SELECT \* FROM users;

SELECT \* FROM roles;

SELECT \* FROM role\_user;

SELECT a.name,a.descripcion,c.`dia`

FROM admin\_horarios\_config AS a INNER JOIN admin\_horarios\_dias AS b ON a.`id` = b.`id\_horarios`

INNER JOIN admin\_dia AS c ON b.`id\_dia` = c.`id`

WHERE a.`id` = 1 AND b.activo = 1;

SELECT a.`name`, a.`descripcion`,b.`hora\_sync`

FROM admin\_horarios\_config AS a INNER JOIN admin\_horarios\_horas AS b ON a.`id` = b.`id\_horarios`

WHERE a.`id` = 1 AND b.activo = 1;

SELECT a.`name`,a.`descripcion`, b.`to\_email`,b.`cc\_email`

FROM admin\_notificacion AS a INNER JOIN admin\_correos AS b

ON a.`id` = b.`id\_notifi`

WHERE a.`id` = '1' ;

**Privilegios de usuario Administrador**

admin\_horarios\_horas

Para el módulo de admin.

Se crearan los siguientes sub-modulos.

**Usuarios**

**Parámetros administrativos**

**Notificaciones**

**Configurar sincronizaciones**

**Usuarios:**

Crea, actualiza o elimina usuarios. (Recuerda que estos usuarios deben estar también en la base de datos db\_sun para que puedan hacer login)

Se le asignan los módulos a los que podrá acceder.

Los datos a visualizar en el listado de usuarios son:

Email

\*Nombre

\*Apellido

Descripción

Estado (activo o no)

Módulos asignados.

\*Creado por: (créate\_user)

\*Actualizado por: (Update\_user)

\*Creado el : Created\_at

\* Actualizado el : updated\_at

Nota los Campos con \* serán mostrados solo en la edición individual de los usuarios, no en el listado.

Los datos a visualizar al crear un usuario o editar:

Email

Nombre

Apellido

Descripción

Tipo de usuario

Estado (activo o no) podrá activarlo o desactivar al usuario

Módulos asignados.

Módulos a asignar.

Creado por: (créate\_user)

Actualizado por: (Update\_user)

Creado el : Created\_at

Actualizado el : updated\_at

**Notificaciones**

Datos a mostrar y/o actualizar.

Aquí solo se mostraran las notificaciones que se encuentren activas.

admin\_notificacion.descripcion

admin\_notificacion.sender\_user

admin\_notificacion.sender\_pass

admin\_notificacion.asunto

admin\_notificacion.cuerpo

Datos a mostrar/actualizar

admin\_correos.to\_email

admin\_correos.cc\_email

admin\_correos.bcc\_email

Las tablas admin\_notificacion y admin\_correos están relacionadas por el id. Donde cada registro de notificación debe tener su registro en admin\_correos donde enlace sus to\_email,cc\_email,bbc\_email.

**Parámetros administrativos**

Datos a mostrar/Actualizar

De la tabla **admin\_config**

`defa\_cod\_proveedor` - Codigo proveedor

`defa\_cod\_tip\_banco`, Cod. Tipo Banco

`defa\_costo` Costo por defecto

`defa\_precio`, Precio por defecto

`defa\_pmaxgan` Porcentaje Max. Ganancia

`defa\_p\_est\_cos`, Porcentaje Estructura de Costos

`defa\_language`, Idioma por defecto

`defa\_moneda` Moneda por defecto

`alert\_mont\_odp`, Orden de pago mayor a

`alert\_mont\_odc`, Orden de compra mayor a

`alert\_mont\_com`, Compras mayor a

`alert\_mont\_fact` Facturas mayor a

`alert\_mont\_cotz` Cotizaciones mayor a

**Configurar sincronizaciones**

Los datos a mostrar serán:

Por cada registro de admin\_horarios\_config que este activo se mostrara:

**admin\_horarios\_config.descripcion** (este campo es solo lectura)

* **descripcion**

Los días de la semana están en la tabla “admin\_dia”

Las tablas admin\_horarios\_config y admin\_horarios\_horas

Están relacionadas de 1 a N. Por cada registro en la admin\_horarios\_config puede existir N registro en admin\_horarios\_horas.

Mostrando

**admin\_horarios\_horas.hora\_sync -> Hora (la hora es editable)**

Correspondiente a cada registro.

Estas horas serán asignadas por cada día de la semana que se encuentre

Activo en la tabla “admin\_horarios\_dias”

Los días podrán ser visualizados y activarse o desactivarse.

Las tablas que se encuentran relacionadas para los horarios son

`admin\_dia`;

`admin\_horarios\_config`;

`admin\_horarios\_horas`;

admin\_horarios\_dias`;

La intención de este módulo es asociar por cada registro en “admin\_horarios\_config” los días de la semana que el usuario desee en los horarios que se encuentren en la tabla “admin\_horarios\_horas”.