Manual técnico **Secop V 1.0**

Edgar Adrián Guzmán Vargas

**Prisma Web**

2019

**Tabla de contenido.**

[**Herramientas de Software Utilizadas.**](#_5rxarq1tyd22) **3**

[**Modelo Base de datos.**](#_4pdlsrie09cu) **4**

[**Estructura de archivos en el servidor.**](#_1vo65mjlnh3a) **11**

[**Definición de los Modelos.**](#_8rxjgzxlyyq0) **16**

[**Definición de trabajo Cron.**](#_a20yjgf6vuzz) **17**

[**Definición de eventos.**](#_2iaipqamq7jq) **17**

[**Referencias**](#_dx5o45rsq7yq) **19**

# 

# 

# 

# 

# 

# 

# Herramientas de Software Utilizadas.

* **Lenguajes de programación Servidor Web :** *PHP versión - 7.2*
  + **Librerías utilizadas**.
    - fideloper/proxy versión - 3.3.
    - laravel/tinker versión - 1.0.
    - maatwebsite/excel - 3.1
    - spatie/laravel-permission versión - 2.12.
* **Motor Base de datos :** MariaDB 10.2.12
* **Servidor Web:** Apache
* **Lenguaje de programación Cliente (Navegador Web):** *Javascript*
  + **Librerías utilizadas.**
    - Jquery.js versión - 3.3.1.
    - dropzone.js versión - 5.5.1.
    - datatables.js versión - 1.10.18.
    - dataTables.responsive versión - 2.2.3.
    - dataTables.buttons.min.js versión - 1.5.2.
    - datatables.pdfmake versión - 0.1.36
    - bootstrap-datepicker.js
    - select2.js versión - 4.0.6.
* **Lenguaje de marcado:** *HTML5* 
  + **Plantilla base.**
    - AdminLTE v2.4.2
* **Lenguaje de estilos :** *CSS3.*
  + Framework.
    - Bootstrap versión - 4.1.0.
* **Framework:** *Laravel 5.5.*

# Modelo Base de datos.

A continuación se listan en orden alfabético (A-Z) las tablas y su estructura existentes en la base de datos.

|  |  |  |
| --- | --- | --- |
| **control\_procesos** | | |
| **Nombre Campo** | **Tipo de dato** | **Observación** |
| **id** | int(10) primary key autoincrement |  |
| **numero\_proceso** | varchar(512) |  |
| **link\_proceso** | varchar(512) |  |
| **entidad** | varchar(512) |  |
| **objeto** | varchar(2048) |  |
| **dpto\_ciudad** | varchar(512) |  |
| **cuantia** | varchar(255) |  |
| **fecha\_apertura** | date |  |
| **id\_usuario\_asignado** | int(10) |  |
| **fecha\_cierre** | date |  |
| **tipo\_proceso** | varchar(255) |  |
| **estado\_proceso** | varchar(255) |  |
| **gestion\_comercial** | enum('Encontrado','No cumplimos','Pendiente Propuesta','Propuesta Presentada','Adjudicado','No Adjudicado') |  |
| **created\_at** | timestamp |  |
| **updated\_at** | timestamp |  |

**Descripción de la tabla:** Dentro de esta tabla se registrará la información de los anuncios.

|  |  |  |
| --- | --- | --- |
| **detalle\_empresas\_usuarios** | | |
| **Nombre Campo** | **Tipo de dato** | **Observación** |
| **id** | int(10) primary key autoincrement |  |
| **id\_usuario** | int(10) |  |
| **id\_empresa** | int(10) |  |
| **created\_at** | timestamp |  |
| **updated\_at** | timestamp |  |

**Descripción de la tabla:** Dentro de esta tabla se registrará la información de las apis usadas en la aplicación.

|  |  |  |
| --- | --- | --- |
| **empresas** | | |
| **Nombre Campo** | **Tipo de dato** | **Observación** |
| **id** | int(10) primary key autoincrement |  |
| **nombre\_empresa** | varchar(255) |  |
| **created\_at** | timestamp |  |
| **updated\_at** | timestamp |  |

**Descripción de la tabla:** Dentro de esta tabla se registrará la información de las bonificaciones registradas por cada registro o recarga de acuerdo a los referidos.

|  |  |  |
| --- | --- | --- |
| **migrations** | | |
| **Nombre Campo** | **Tipo de dato** | **Observación** |
| **id** | int(10) primary key autoincrement |  |
| **migration** | varchar(255) |  |
| **batch** | int(11) |  |

**Descripción de la tabla:** Aquí se registran la campañas que tenderán los cupones y sus respecti vas restricciones, usuario, tipo de canje y valor.

|  |  |  |
| --- | --- | --- |
| **models\_has\_permissions** | | |
| **Nombre Campo** | **Tipo de dato** | **Observación** |
| **permission\_id** | int(10) |  |
| **model\_type** | varchar(255) |  |
| **model\_id** | bigint(20) |  |

**Descripción de la tabla:** En esta tabla se registran lo s cupones y sus estados para cada campaña.

|  |  |  |
| --- | --- | --- |
| **model\_has\_roles** | | |
| **Nombre Campo** | **Tipo de dato** | **Observación** |
| **role\_id** | int(10) |  |
| **model\_type** | varchar(255) |  |
| **model\_id** | bigint(20) |  |

**Descripción de la tabla:** Aquí se registran los clic hechos por los usuarios a los anuncios, se registra tipo, fechas de clic y el valor descontado de la recarga

|  |  |  |
| --- | --- | --- |
| **observacion\_procesos** | | |
| **Nombre Campo** | **Tipo de dato** | **Observación** |
| **id** | int(10) primary key autoincrement |  |
| **observacion** | varchar(1024) |  |
| **id\_usuario\_observacion** | int(10) |  |
| **id\_control\_proceso** | int(10) |  |
| **tipo\_observacion** | enum('manual', 'auto') |  |
| **created\_at** | timestamp |  |
| **updated\_at** | timestamp |  |

**Descripción de la tabla:** En esta tabla se registran los horarios de cada usuario y el estado del día de la se mana.

|  |  |  |
| --- | --- | --- |
| **password\_resets** | | |
| **Nombre Campo** | **Tipo de dato** | **Observación** |
| **email** | varchar(255) |  |
| **token** | varchar(255) |  |
| **created\_at** | timestamp |  |

**Descripción de la tabla:**Aquí se registra los detalles de las recargas hechas por los usuarios.

|  |  |  |
| --- | --- | --- |
| **permissions** | | |
| **Nombre Campo** | **Tipo de dato** | **Observación** |
| **id** | int(10) primary key autoincrement |  |
| **name** | varchar(255) |  |
| **guard\_name** | varchar(255) |  |
| **created\_at** | timestamp |  |
| **updated\_at** | timestamp |  |

**Descripción de la tabla:** Aquí se registran los usuarios que tengan un referido.

|  |  |  |
| --- | --- | --- |
| **roles** | | |
| **Nombre Campo** | **Tipo de dato** | **Observación** |
| **id** | int(10) primary key autoincrement |  |
| **name** | varchar(255) |  |
| **guard\_name** | varchar(255) |  |
| **created\_at** | timestamp |  |
| **updated\_at** | timestamp |  |

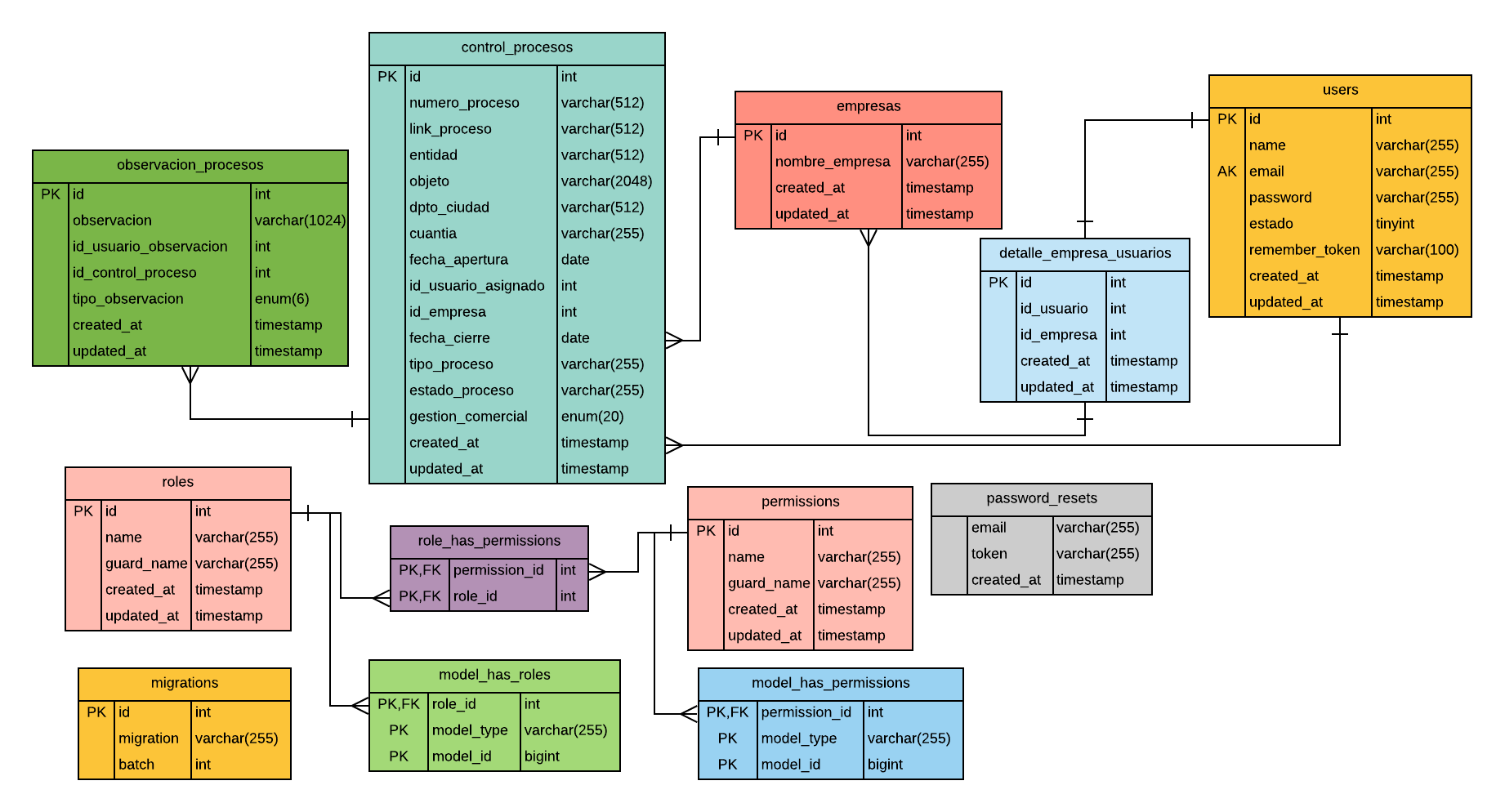
**Descripción de la tabla:** Tabla generada automáticamente por el framework Laravel para registrar las mi graciones.

|  |  |  |
| --- | --- | --- |
| **roles\_has\_permissions** | | |
| **Nombre Campo** | **Tipo de dato** | **Observación** |
| **permission\_id** | int(10) |  |
| **role\_id** | int(10) |  |

**Descripción de la tabla:** Tabla creada por el paquete ***spatie/laravel-permission versión - 2.12*** y se registran los permisos para cada modelo (usuarios).

|  |  |  |
| --- | --- | --- |
| **users** | | |
| **Nombre Campo** | **Tipo de dato** | **Observación** |
| **id** | int(10) primary key autoincrement |  |
| **name** | bigint(20) |  |
| **email** | varchar(191) |  |
| **password** | varchar(255) |  |
| **estado** | tinyint(1) |  |
| **remember\_token** | varchar(100) |  |
| **created\_at** | timestamp |  |
| **updated\_at** | timestamp |  |

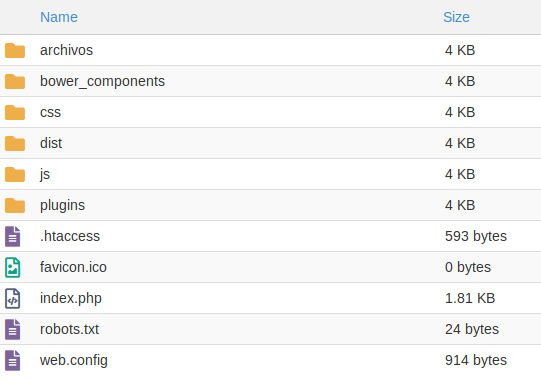
**Descripción de la tabla:** Tabla creada por el paquete ***spatie/laravel-permission versión - 2.12*** y se registran los roles para cada modelo (usuarios)



# Estructura de archivos en el servidor.

A continuación se describe la estructura de archivos bajo la cual se encuentra en funcionamiento la aplicación.

La aplicación se encuentra almacenada bajo la url [**prismaweb.biz/secop**](http://prismaweb.biz/secop) y ubicada en la carpeta **core** como carpeta de producción de igual manera se cuenta con una carpeta para pruebas y se puede acceder a través de la url [**prismaweb.biz/test**](http://prismaweb.biz/test) y almacenada en la carpeta **test**, las url’s anteriormente mencionadas cuentan con sus archivos almacenados dentro de la carpeta ***public\_html*** del servidor**,** dentro de estas carpetas solo están almacenados los archivos correspondientes a la carpeta public del framework **Laravel 5.5**, es decir que solo encontremos allí archivos como imágenes o scripts de tipo .js (javascript), archivos de estilos y el archivo **index.php** encargado de iniciar el framework, a continuación se anexa una imagen para facilitar el identificación de los archivos



Por otra parte se encuentran los archivos que contiene la lógica y estructura del framework, de igual manera se cuentan con dos carpetas una para producción (***tutramite***) y pruebas (***tutramitetest***), dichas carpetas contiene la estructura correspondiente a la versión del framework ***Laravel 5.5***, y se encuentran almacenadas en la raíz (***home/turdorco***) de el servidor, a continuación se listan y se describe su contenido brevemente, de igual manera se anexa una imagen para ayudar a identificar las carpetas mencionadas.



* **app**: carpeta que contiene los controladores modelos y es donde se desarrolla la lógica de la aplicación, a continuación se listan las carpetas más relevantes del proyecto dentro de app.



* + **Console:** aquí se almacenarán los trabajos de tipo cron de la aplicación.
  + **Events:** aquí se almacenarán los eventos de la aplicación.
  + **Http/Controllers :** Contiene todos los archivos con las clases de los controladores que sirven para interactuar con los modelos, las vistas y manejar la lógica de la aplicación.
* **boostrap:** Carpeta con contenido de configuración de y no se debe modificar manualmente.
* **config:** aquí se encuentran todos los archivos de configuración de la aplicación: base datos, cache, correos, sesiones o cualquier otra configuración general de la aplicación.
* **databases:** En esta carpeta se incluye todo lo relacionado con la definición de la base de datos de nuestro proyecto.
* **resources:** esta carpeta contiene a su vez tres carpetas: *assets*, *views* y *lang*: 
  + **resources/views** – Este directorio contiene las vistas de nuestra aplicación. En general serán plantillas de HTML que usan los controladores para mostrar la información. Hay que tener en cuenta que en esta carpeta no se almacenan los Javascript, CSS o imágenes, ese tipo de archivos se tienen que guardar en la carpeta public.
  + **resources/lang –** En esta carpeta se guardan archivos PHP que contiene arrays con los textos de nuestro sitio web en diferentes lenguajes, solo será necesario utilizarla en caso que se desee que la aplicación se pueda traducir.
  + **resources/assets –** Se utiliza para almacenar los fuentes de los assets tipo *less* o *sass* que se tendrían que compilar para generar las hojas de estilo públicas. No es necesario usar esta carpeta ya que podemos escribir directamente las las hojas de estilo dentro de la carpeta *public*.
* **routes:** esta carpeta contiene los archivos con las rutas de acceso permitidas para acceder a los recursos de la aplicación a través del protocolo HTTP.
* **storage:** en esta carpeta Laravel almacena toda la información interna necesarios para la ejecución de la web, como son los archivos de sesión, la caché, la compilación de las vistas, meta información y los logs de error del sistema. Normalmente tampoco tendremos que tocar nada dentro de esta carpeta, únicamente se suele acceder a ella para consultar los logs, en algunos casos como en Linux es necesario habilitar los permisos de escritura.
* **tests** : Esta carpeta se utiliza para los ficheros con las pruebas automatizadas. Laravel incluye un sistema que facilita todo el proceso de pruebas con PHPUnit.
* **vendor** : En esta carpeta se alojan todas las librerías y dependencias que conforman el *framework* de Laravel. Esta carpeta tampoco la tendremos que modificar, ya que todo el código que contiene son librerías que se instalan y actualizan mediante la herramienta Composer.
* **Archivo .env:**  en este archivo se almacena los datos de configuración de la aplicación.
* **Archivo composer.json:**  este fichero es el utilizado por Composer para realizar la instalación de Laravel. En una instalación inicial únicamente se especificará la instalación de un paquete, el propio *framework* de Laravel, pero podemos especificar la instalación de otras librerías o paquetes externos que añaden funcionalidad a ***Laravel***.

# Definición de los Modelos.

A continuación se listan las clases definidas en el sistema para interactuar con la base de datos, y manejo de servicios como el acceso a la api de coinmarket.

* **ControlProceso**: Clase asociada a la entidad control\_procesos.
* **DetalleEmpresaUsuario:** Clase asociada a la entidad\_detalle\_empresa\_usuario
* **Empresa**: Clase asociada a la entidad empresas
* **ObservacionProceso**: Clase asociada a la entidad observacion\_procesos.
* **User**: Clase asociada a la entidad users.

# Definición de trabajo Cron.

La aplicación cuenta con un trabajo de tipo cron el cual se encuentra registrado en el framework bajo el nombre ***proceso:cierre\_proximo,*** y está almacenado en la carpeta ***app/console/commands*** el cual se ejecuta a media noche, en el servidor se encuentra registrado para su respectiva ejecución.

***php-cli /home/priswebco/secop/artisan schedule:run 1>> /dev/null 2>&1***

# Definición de eventos.

La aplicación cuenta con 5 eventos para notificar a los usuarios.

* **Event:** Evento el cual notifica al usuario

Todos los eventos envían correos electrónicos las vistas de estos correo están almacenadas en la carpeta ***resourde/views/emails.***

Especificación de evento **Event.**

Este evento maneja diferentes notificaciones por lo cual tiene una estructura necesaria para su uso, a continuación se describe su estructura y se brinda un ejemplo de uso.

Para la invocación del evento ***Event*,** primero se debe usar el paquete ***use App\Events\Event,*** luego se puede acceder al método estático **dispatch** en el cual debemos enviar los parámetros registrados en la clase ***Event.php,*** dicha clase se encuentra ubicada en ***app/Events,***  allí encontrará 4 propiedades de la clase **$user**, **$datos**, **$tipo**, los cuales se describen a continuación.

* **$user**:Debe ser un objeto de tipo User.
* **$datos**: Este objeto puede ser de tipo array o el que se requiera para el mensaje específico registrado en la variable **tipo.**
* **$tipo:** Este valor debe ser de tipo string y debe estar asociado a una de las opciones registradas en el archivo ***NotificationMail*** ubicado en la carpeta ***app/Mail*** *dentro de la función* ***build.***

ejemplo.

***Event::dispatch($suario[0],[$ad[0],$uc[0]],"ProximoCierre");***

***Aquí observamos que se envían los 3 campos requeridos (separados por , )***

# Referencias

PHP: Hypertext Preprocessor. (2019, 4 abril). Recuperado 30 abril, 2019, de <https://www.php.net/>

MariaDB 10.2.12 Release Notes. (s.f.). Recuperado 30 abril, 2019, de <https://mariadb.com/kb/en/library/mariadb-10212-release-notes/>

fideloper/proxy - Packagist. (s.f.). Recuperado 30 abril, 2019, de <https://packagist.org/packages/fideloper/proxy>

payu webcheckout - Payu Recuperado 30 abril, 2019, de

<http://developers.payulatam.com/es/web_checkout/>

laravel/tinker. (s.f.). Recuperado 30 abril, 2019, de <https://github.com/laravel/tinker>

spatie/laravel-permission. (s.f.). Recuperado 30 abril, 2019, de <https://github.com/spatie/laravel-permission>

JavaScript. (2019, 23 abril). Recuperado 30 abril, 2019, de <https://developer.mozilla.org/en-US/docs/Web/JavaScript>

jQuery. (s.f.). Recuperado 30 abril, 2019, de <https://jquery.com/>

Dropzone.js. (s.f.). Recuperado 30 abril, 2019, de <https://www.dropzonejs.com/>

Manual. (s.f.-b). Recuperado 30 abril, 2019, de <https://datatables.net/manual/>

Getting Started | Select2 - The jQuery replacement for select boxes. (s.f.-b). Recuperado 30 abril, 2019, de <https://select2.org/>

Free Bootstrap Admin Template | AdminLTE.IO. (s.f.-b). Recuperado 30 abril, 2019, de <https://adminlte.io/>

Download. (s.f.-b). Recuperado 30 abril, 2019, de <https://getbootstrap.com/docs/4.1/getting-started/download/>

Installation - Laravel - The PHP Framework For Web Artisans. (s.f.). Recuperado 30 abril, 2019, de <https://laravel.com/docs/5.5/>

Places Library | Maps JavaScript API | Google Developers. (2019, 16 abril). Recuperado 30 abril, 2019, de <https://developers.google.com/maps/documentation/javascript/places>