30-10-2015

Diana Umanzor

INFOLOGIC

MANUAL TECNICO ADMINISTRACION

**Sistema de atención al contribuyente**

Contenido

[1. Breve descripción del Módulo de Administracion 2](#_Toc433959186)

[2. Diagrama Entidad – Relación 3](#_Toc433959187)

[3. Estructura de Componentes (Capa vista, controlador, servicio DAO y estructura de reportes) 4](#_Toc433959188)

[3.1 Controladores 7](#_Toc433959189)

[3.2 Descripción de controladores 8](#_Toc433959190)

[4. Descripción de POJOS de sesión 39](#_Toc433959191)

[4.1 Descripción de Clases POJO 41](#_Toc433959192)

[5. Listado de Repositorios. 49](#_Toc433959193)

[5.1 Descripción de Repositorios 51](#_Toc433959194)

[6. Listado de Componentes. 78](#_Toc433959195)

[Descripción de Componentes. 78](#_Toc433959196)

[7. Configuracion de proyectos (@Configuration y @Beans en general, esto es todo el paquete de sv.gob.mh.dgii.det.config) 79](#_Toc433959197)

[8. Documentos Relacionados a la Base de Datos 83](#_Toc433959198)

[8. Seguridad 103](#_Toc433959199)

[1. Anexos 105](#_Toc433959200)

# Breve descripción del Módulo de Administracion

Prefijo: /admin/

Modulo gestiona el funcionamiento de la administración (Mantenimiento) de los diferentes elementos que componen la parametrización de las Zonas (Áreas), alertas, configuración de los Tramites,los equipos, Escritorios, evaluación, Llamado, multimedia , prioridades, Reasignación, reproducción multimedia, Servicios, Tramites, Usuarios, también administra el formato que tendrán los tiquetes.

# Diagrama Entidad – Relación

A continuación se listan las diferentes tablas utilizadas en el Módulo de Administracion, junto con los nombres de los esquemas a donde pertenecen.

| Nombre del esquema | Nombres de tablas | Anexo |
| --- | --- | --- |
| Gestion.Cola | GC\_ESCRITORIO  GC\_TRAMITE  GC\_USUARIO  GC\_ALERTA  GC\_ZONA  GC\_CONF\_LLAMADO  GC\_CONF\_TIQUETE  GC\_CONF\_TRAMITE  GC\_EQUIPOS  GC\_MULTIMEDIA  GC\_PREGUNTAS  GC\_RESPUESTAS  GC\_PRIORIDAD  GC\_SERVICIOS  GC\_TIPO\_EQUIPO  GC\_TRAMITE  GC\_CONF\_TRAMITE | Anexo 1 |
| CATALOGOS | TB\_UNIDAD\_RECEP  TB\_LISTAS\_VALOR | Anexo 2 |

Tabla 2.1 tablas con sus esquemas respectivos

# Estructura de Componentes (Capa vista, controlador, servicio DAO y estructura de reportes)

**Vistas o páginas .jsp**

En la siguiente tabla se muestran todas las páginas .html usadas en el Módulode Administracion y el lugar en donde se pueden encontrar.

| Nombre de opción | Ubicación | Nombre |
| --- | --- | --- |
| Configuracion de Usuarios | src/main/resources/templates.admin | confiUsuarios.html  form.html  index.html  verUsuario.html |
| Mantenimiento de Alertas | src/main/resources/templates.alerta | form.html  index.html  show.html |
| Mantenimiento de Configuración de Llamado | src/main/resources/templates.cllamado | cllamados.html  newCllamado.html  verCllamado.html |
| Configuracion de Tiquete | src/main/resources/templates.conftiquete | conftiq.html |
| Mantenimiento de Configuración de Tramite | src/main/resources/templates.ctramite | ctramites.html  newCtramite.html  verCtramite.html |
| Mantenimiento de Equipos | src/main/resources/templates.equipo | equipos.html  newEquipo.html  verEquipo.html |
| Mantenimiento de Escritorios | src/main/resources/ templates.escritorio | editEscritorio.html |
|  |  | escritorios.html  newEscritorio.html  verEscritorio.html |
| Mantenimiento de Lista de Reproducción | src/main/resources/ templates.confrepro | Confrepro.html |
| Mantenimiento de Multimedia | src/main/resources/ templates.multimedia | editMultimedia.html  multimedia.html  newMultimedia­.html  verMultimedia.html |
| Mantenimiento de Configuración - Evaluación | src/main/resources/ templates.adminpreguntas | newPregunta.html  preguntas.html  verPregunta.html |
| Mantenimiento de Prioridades - Catálogo | src/main/resources/ templates.prioridad | newPrioridad.html  prioridades.html  verPrioridad.html |
| Mantenimiento de Secciones - Catálogo | src/main/resources/ templates.servicio | newServicio.html  servicios.html  verServicio.html |
| Mantenimiento de Tipos de Equipos - Catálogo | src/main/resources/ templates.tequipo | newTequipo.html  tequipos.html  verTequipo.html |
| Mantenimiento de Tramites - Catálogo | src/main/resources/ templates.tramite | newTramite.html  tramites.html  verTramite.html |
| Mantenimiento de Zonas - Catálogo | src/main/resources/ templates.zona | zonas.html  editZona.html  newZona.html  verZona.html |
| Configuración de Reasignación | src/main/resources/ templates.creasignacion | creasignaciones.html  newCreasignacion.html  verCreasignacion.html |

Tabla 3.1 nombres y ubicación de las páginas .jsp

## Controladores

En la siguiente tabla se muestran todas las clases controladores que se usan en el modulo de Administracion y la ubicación de los controladores que se encuentran en el paquete **sv.gob.mh.dgii.colas.controllers.admin** en donde se encuentran.

|  |  |
| --- | --- |
| Nombre de Opción | Nombre |
| Configuracion de Usuarios | AdminController.java |
| Mantenimiento de Alertas | AlertaController.java |
| Mantenimiento de Configuración de Llamado | ConfLlamadoController.java |
| Configuracion de Tiquete | ConfTiqueteController.java |
| Mantenimiento de Configuración de Tramite | ConfTramiteController.java |
| Mantenimiento de Equipos | EquipoController.java |
| Mantenimiento de Escritorios | EscritorioController.java |
| Mantenimiento de Lista de Reproducción | MultimedServicioController.java |
| Mantenimiento de Multimedia | MultimediaController.java |
| Mantenimiento de Configuración - Evaluación | PreguntaEvaluacionController.java |
| Mantenimiento de Prioridades - Catálogo | PrioridadController.java |
| Mantenimiento de Secciones - Catálogo | ServicioController.java |
| Mantenimiento de Tipos de Equipos - Catálogo | TipoEqueipoController.java |
| Mantenimiento de Tramites - Catálogo | TramiteController.java |
| Mantenimiento de Zonas - Catálogo | ZonaController.java |
| Configuración de Reasignación | ReasignacionController.JAVA |

Tabla 3.2 nombres y ubicación de los controladores

## Descripción de controladores

En las siguientes tablas se describen brevemente cada uno de los métodos de los diferentes controladores usados en el Módulo de Administracion.

| **PANTALLA CONFIGURACION DE USUARIOS** | |
| --- | --- |
| **Controlador:** | AdminController.java |
| **Descripción de Métodos de Negocio** | |
| **public UsuarioPojo usuariopojo();**  Método que crea una instancia a la clase UsuarioPojo e invoca los métodos prepareList() y getUnidad().  **@RequestMapping("")**  **public String iniciar\_usuarios(HttpServletRequest request);**  Método que carga la url redirect:/admin/  **@RequestMapping(value = "/", method = RequestMethod.GET)**  **public String index(HttpServletRequest request, @ModelAttribute UsuarioPojo usuariopojo, final ModelMap model);**  Método de tipo String que obtiene una lista de todos usuarios de la entidad GcUsuario con el repositorio GcAdminRepositoryImpl ubicado en el paquete sv.gob.mh.dgii.colas.repositories y el método getAllUser() que obtiene un listado de usuarios con sus tramites .  **@RequestMapping(value = "/agregar", method = RequestMethod.GET)**  **public String agregar(HttpServletRequest request, @ModelAttribute UsuarioPojo usuariopojo, final ModelMap model);**  Método de tipo String que se encarga de agregar un nuevo usuario creando una instancia a la clase UsuarioPojo, invocando los métodos preparelist() y getUnidad(); seteando los parametos cUsuario, dUsuario, Escritorio y cargando la url admin/form  **@RequestMapping(value = "/", method = RequestMethod.POST) public String guardar(HttpServletRequest request, @ModelAttribute UsuarioPojo usuariopojo, final ModelMap model);**  Método de tipo String que guarda el tramite asignado por el usuario inicializando la clase UsuarioPojo, seteando el parámetro AddTramite, utilizando el método save() del repositorio GcAdminRepositoryImpl ubicado en el paquete sv.gob.mh.dgii.colas.repositories y cargando la url redirect:/admin/  **@RequestMapping(value = "{user}/editar", method = RequestMethod.GET)**  **public String editar(HttpServletRequest request, @ModelAttribute UsuarioPojo usuariopojo, final ModelMap model,@PathVariable(value = "user") String user);**  Método de tipo String que llena el pojo a modificar con el usuario,escritorio y tramites y lo muestra en la vista cargando la url admin/form  **@RequestMapping(value = "{user}/show", method = RequestMethod.GET)**  **public String mostrar(HttpServletRequest request, @ModelAttribute UsuarioPojo usuariopojo, final ModelMap model,@PathVariable(value = "user") String user);**  Método de tipo String que obtiene los tramites por usuario llena el pojo con los parámetros seteados usuario, escritorio y los tramites lo muestra en la vista cargando la url admin/verUsuario.  **@RequestMapping(value = "/{user}/delete", method = RequestMethod.POST)**  **public String destroy(HttpServletRequest request,@ModelAttribute UsuarioPojo usuariopojo, final ModelMap model, @PathVariable(value = "user") String user);**  Método de tipo String que obtiene los tramites por usuario y elimina el usuario cargando la url redirect:/admin/  **@RequestMapping(value = "/getuserldap/{usuarioConsulta}", method = RequestMethod.GET)**  **public @ResponseBody String getContribuyenteNombre(HttpServletRequest request, final ModelMap model, @PathVariable(value = "usuarioConsulta") String usuario);**  Método de tipo String que obtiene nombre del contribuyente buscándolo por la llave primaria en el método findByPrimaryKey(usuario); por medio de personaLdap obtiene el primer nombre, segundo y apellido, obtiene la unidad receptora valida que si la unidad receptora esta nula obtiene la ubicación física.  **@RequestMapping(value = "/addTramite", method = RequestMethod.GET)**  **public String addTramite(HttpServletRequest request,@ModelAttribute UsuarioPojo usuariopojo, final ModelMap model, @RequestParam(value = "lista", required = false) String[] lista);**  Método de tipo String que obtiene un listado de tramites invocando el método addList() y la muestra en la vista cargando la url admin/form  **@RequestMapping(value = "/delTramite", method = RequestMethod.GET)**  **public String delTramite(HttpServletRequest request,@ModelAttribute UsuarioPojo usuariopojo, final ModelMap model, @RequestParam(value = "lista", required = false) String[] lista);**  Método de tipo String que elimina los tramites por el usuario invocando el método delList();  **public void addList(UsuarioPojo user, String[] lista);**  Método que obtiene un listado de tramites modificados de la entidad GcTramite invoca el iterador next();  **public void delList(UsuarioPojo user, String[] lista);**  Método que obtiene un listado de tramites modificados de la entidad GcTramite invoca el iterador next();  **public void removList(UsuarioPojo user, List<GcTramite> listado);**  Método que elimina el listado de tramites por el id de tramite invoca el método remove();  **public void prepareList(UsuarioPojo user, String unidad);**  Método que setea un listado de tramites y de escritorios de las entidades GcTramite y GcEscritorio  **@RequestMapping(value = "/getEscritorios", method = RequestMethod.GET)**  **public String getEscritorios(HttpServletRequest request, @ModelAttribute UsuarioPojo usuariopojo, final ModelMap model,@RequestParam(value = "cunidadRecepId", required = false) String unidad) ;**  Método de tipo String que obtiene un listado de escritorios vacios invocando el método getEscritoriosVacios que se encuentra en el repositorio escritorioRepository ubicado en el paquete sv.gob.mh.dgii.colas.repositories cargando la url admin/form  **public String getUnidad();**  Método de tipo String que obtiene la unidad receptora y la ubicación física. | |

Tabla 3.3 controlador **AdminController**

| **PANTALLA MANTENIMIENTO DE ALERTAS** | |
| --- | --- |
| **Controlador:** | AlertaController.java |
| **Descripción de Métodos de Negocio** | |
| **public GcAlerta gcAlertaList();**  Método que crea una nueva instancia de la clase **GcAlerta**  **@RequestMapping("")**  **public String iniciar(HttpServletRequest request);**  Método de tipo String que carga la url redirect:/alerta/  **@RequestMapping(value = "/", method = RequestMethod.GET)**  **public String index(HttpServletRequest request, @ModelAttribute GcAlerta gcalerta, final ModelMap model);**  Método de tipo String que obtiene todas las alertas activas y agrega el atributo listadoAlerta carga la url alerta/index  **@RequestMapping("/newAlerta")**  **public String newAlerta(HttpServletRequest request, @ModelAttribute GcAlerta gcalerta, final ModelMap model);**  Método String que prepara el formulario para agregar una nueva alerta invoca el método prepareList() ; cargando el formulario alerta/form  **@RequestMapping(value = "{alerta}/editAlerta", method = RequestMethod.GET)**  **public String editAlerta(HttpServletRequest request, @ModelAttribute GcAlerta gcalerta, final ModelMap model, @PathVariable(value = "alerta") Long alerta);**  Método de tipo String que prepara el formulario para editar la alerta, obtiene el correo de notificación carga la url alerta/form  **@RequestMapping(value = "/getzonas", method = RequestMethod.GET)**  **public String getZonasXunidad(HttpServletRequest request, @ModelAttribute GcAlerta gcalerta, final ModelMap model, @RequestParam(value = "cunidadRecepId", required = false) String unidad);**  Método de tipo String que obtiene las zonas por unidad cargando la url alerta/form  @RequestMapping(value = "{alerta}/delete", method = RequestMethod.POST)  **public String eliminar(HttpServletRequest request, @ModelAttribute GcAlerta gcalerta, final ModelMap model, @PathVariable(value = "alerta") Long alerta);**  Método de tipo String que elimina la alerta y guarda utilizando el método save();del repositorio gcAlertaRepository ubicado en el paquete sv.gob.mh.dgii.colas.repositories carga la url alerta/index  **@RequestMapping(value = "{alerta}/verAlerta", method = RequestMethod.GET)**  **public String verAlerta(HttpServletRequest request, @ModelAttribute GcAlerta gcalerta, final ModelMap model, @PathVariable(value = "alerta") Long alerta);**  Método que muestra las alertas cargando la url alerta/show  **@RequestMapping(value = "/", method = RequestMethod.POST)**  **public String guardar(HttpServletRequest request, @ModelAttribute GcAlerta gcalerta, final ModelMap model);**  Método de tipo String que setea los parámetros CUsuarioCrea, FModifica, BActiva, CUsuarioModi valida que se envíe la alerta al correo del usuario invoca el método save() con el repositorio gcAlertaRepository ubicado en el paquete sv.gob.mh.dgii.colas.repositories carga la url alerta/  **@RequestMapping(value = "/getusuario", method = RequestMethod.GET, headers = "Accept=application/json", produces = "application/json")**  **@ResponseBody**  **public TipoUsuarioPojo getUsuario(HttpServletRequest request);**  Método que inicializa el pojo TipoUsuarioPojo obtiene en el parámetro enviado el usuario y la unidad, asigna un tipo de técnico y obtiene los roles por el tipo de usuario.  **public void prepareList(AlertaPojo alertapojo);**  Método que setea un listado de zonas y obtiene todas las zonas activas.  **@InitBinder**  **public void initBinder(WebDataBinder binder);**  Inicializa la instancia binder para dar formato a los campos de fecha.  **public void enviarCorreos(GcAlerta alerta);**  Método tipo string que inserta registro de la plantilla en tb\_emial y envía la alerta al correo.  **public List<TbUnidadRecep> getUnidadesList();**  Método que devuelve una lista de unidades receptoras  **public String getUnidad();**  Método que obtiene la ubicación física. | |

Tabla 3.4 controlador AlertaController

| **PANTALLA MANTENIMIENTO DE CONFIGURACIÓN DE LLAMADO** | |
| --- | --- |
| **Controlador:** | ConfLlamadoController.java |
| **Descripción de Métodos de Negocio** | |
| **public CllamadoPojo inicial();**  Método que crea una instancia a la clase CllamadoPojo.  **@RequestMapping(value="/", method = RequestMethod.GET)**  **public String indexConfLlamado(HttpServletRequest request, @ModelAttribute CllamadoPojo cllamadoPojo, final ModelMap model);**  Método que valida que el rol asignado sea igual al rol de administración de usuario en sesión también obtiene todas las configuraciones del repositorio gcConfLlamadoRepository ubicado en el paquete sv.gob.mh.dgii.colas.repositories muestra un listado de llamados cargando la url cllamado/cllamados  **@RequestMapping("/newCllamado")**  **public String newCllamado(HttpServletRequest request, @ModelAttribute CllamadoPojo cllamadoPojo, final ModelMap model);**  Método que crea un nuevo llamado obtiene una lista de unidades carga la url cllamado/newCllamado  **@RequestMapping(value="{cllamdo}/editCllamado", method = RequestMethod.GET)**  **public String editCllamado(HttpServletRequest request, @ModelAttribute CllamadoPojo cllamadoPojo, final ModelMap model, @PathVariable(value = "cllamdo") Long cllamdoId) ;**  Método que edita el llamado utilizando el repositorio gcConfLlamadoRepository para editarlo por el id del llamado llena el pojo con los parámetros setados obtenidos en el formulario carga la url carga la url cllamado/newCllamado  **RequestMapping(value="{cllamdo}/verCllamado", method = RequestMethod.GET)**  **public String verCllamado(HttpServletRequest request, @ModelAttribute CllamadoPojo cllamadoPojo, final ModelMap model, @PathVariable(value = "cllamdo") Long cllamdoId);**  Método de tipo String que muestra los parámetros seteados en el pojo cargando la url cllamado/verCllamado  **@RequestMapping(value = "{cllamaod}/delete", method = RequestMethod.POST)**  **public String eliminar(HttpServletRequest request, @ModelAttribute CllamadoPojo cllamadoPojo, final ModelMap model, @PathVariable(value = "cllamaod") Long cllamadoId);**  Método de tipo String que elimina por el id del llamado utilizando el repositorio gcConfLlamadoRepository ubicado en el paquete org.springframework.data.repository si se elimina el registro carga el mensaje "SE ELIMINO EL REGISTRO CORRECTAMENTE!" si no "ERROR: "+e.getMessage() carga la url cllamado/cllamados  **@RequestMapping(value = "/", method = RequestMethod.POST)**  **public String guardar(HttpServletRequest request, @ModelAttribute CllamadoPojo cllamadoPojo, final ModelMap model);**  Método de tipo String que guarda los parámetros seteados obtenidos del formularios utilizando los métodos findOne() para guardar por medio del ID del llamado y save() para guardar los parámetros enviados a la variable cLlamado estos métodos se encuentran en el repositorio gcConfLlamadoRepository carga la url cllamado/cllamados.  **public List<TbUnidadRecep> getUnidadesList();**  Método que obtiene una lista de unidades receptoras en servicios con el repositorio gcUnidadRecepRepository ubicado en el paquete sv.gob.mh.dgii.colas.repositories también obtiene un listado de combinaciones de unidades, captura el nombre de las unidades receptoras setea el listado de las unidades obtenidas.  **public String getUnidad();**  Método de tipo String que obtiene la unidad receptora y si viene vacio obtiene la ubicación física. | |

Tabla 3.5 controlador ConfLlamadoController

| **PANTALLA MANTENIMIENTO DE CONFIGURACIÓN DE TIQUETE** | |
| --- | --- |
| **Controlador:** | ConfTiqueteController.java |
| **Descripción de Métodos de Negocio** | |
| **public GcConfTiquete gcctramite();**  Método que crea una instancia a la clase GcConfTiquete  **@RequestMapping(value="/", method = RequestMethod.GET)**  **public String indexConfTiquete(HttpServletRequest request, @ModelAttribute GcConfTiquete gcConfTiquete, final ModelMap model);**  Método de tipo String que carga la url conftiquete/conftiq  **@SuppressWarnings("unchecked")**  **@RequestMapping(value="/getTq", method=RequestMethod.POST, headers="Accept=application/json", produces=MediaType.APPLICATION\_JSON\_VALUE,consumes=MediaType.APPLICATION\_JSON\_VALUE)**  **public @ResponseBody String getTiquete(@RequestBody Map<String, Integer> map);**  Método de tipo String que valida si la unidad receptora viene vacia entonces obtiene la ubicación física y captura n\_unidad\_tlv con el repositorio gcConfTiqueteRepository ubicado en el paquete sv.gob.mh.dgii.colas.repositories obtiene el nombre de la unidad receptora, obtiene una lista de configuraciones del tiquete y obtiene el máximo valor de la versión, también recibe una lista de plantillas para la configuración del tiquete.  **@RequestMapping(value="/saveTq",method=RequestMethod.POST, headers="Accept=application/json", produces=MediaType.APPLICATION\_JSON\_VALUE,consumes =MediaType.APPLICATION\_JSON\_VALUE)**  **public @ResponseBody String saveTiquete(@RequestBody Map<String, String> map);**  Método que realiza la funcion de guardar el tiquete , aplica un formato a la fecha, captura la ultima versión del tieuqt con el repositorio gcConfTiqueteRepository y el método maximaVersion() ubicado en el repositorio sv.gob.mh.dgii.colas.repositories , guarda el estado del riquete si es activo o inactivo  **@RequestMapping(value="/chgTq", method=RequestMethod.POST, headers="Accept=application/json", produces=MediaType.APPLICATION\_JSON\_VALUE,consumes = MediaType.APPLICATION\_JSON\_VALUE)**  **public @ResponseBody String chgTiquete(@RequestBody Map<String, String> map);**  Método de tipo String que asigna una plantilla por el numero de versión asignado por el usuario  **@RequestMapping(value="/reaPla", method=RequestMethod.POST, headers="Accept=application/json", produces=MediaType.APPLICATION\_JSON\_VALUE,consumes=MediaType.APPLICATION\_JSON\_VALUE)**  **public @ResponseBody String readPlantilla(@RequestBody Map<String, String> map);**  Método de tipo String que lee la plantilla invocando el método lineas\_plantilla() del repositorio gcConfTiqueteRepository ubicado en el paquete sv.gob.mh.dgii.colas.repositories  **@RequestMapping(value="/mirarTq",method=RequestMethod.POST,headers="Accept=application/json", produces=MediaType.APPLICATION\_JSON\_VALUE,consumes=MediaType.APPLICATION\_JSON\_VALUE)**  **public @ResponseBody String seeTiquete(@RequestBody Map<String, String> map);**  Método de tipo String que muestra el tiquete con los métodos visualizar() y getimg() del repositorio gcConfTiqueteRepository ubicado en el paquete sv.gob.mh.dgii.colas.repositories | |

Tabla 3.6 controlador ConfLlamadoController

| **PANTALLA DE** **MANTENIMIENTO DE CONFIGURACIÓN DE TRAMITE** | |
| --- | --- |
| **Controlador:** | ConfTramiteController.java |
| **Descripción de Métodos de Negocio** | |
| **public GcConfTramite gcctramite();**  Método que crea una instancia a la clase GcConfTramite  **@RequestMapping(value="/", method = RequestMethod.GET)**  **public String indexConfTramite(HttpServletRequest request, @ModelAttribute GcConfTramite gcConfTramite, final ModelMap model);**  Método que valida que el rol asignado sea igual al rol del usuario administrador si es asi captura una lista de configuraciones de tramite general si no captura una lista de tramites locales invocando en los tramites locales invoca el método getUnidad() carga la url ctramite/ctramites  **@RequestMapping("/newCtramite")**  **public String newCtramite(HttpServletRequest request,@ModelAttribute GcConfTramite gcConfTramite2, final ModelMap model);**  Método de tipo String que prepara el formulario para agrega uun nuevo tramite carga la url ctramite/newCtramite  **RequestMapping(value="{ctramite}/editCtramite", method = RequestMethod.GET)**  **public String editEquipo(HttpServletRequest request, @ModelAttribute GcConfTramite gcConfTramite, final ModelMap model, @PathVariable(value = "ctramite") Long ctramiteId);**  Método de tipo String que realiza la función de editar los tramites por medio de su ID , utiliza los métodos findOne() y getAllTramites() del repositorio gcTramiteRepository carga la url ctramite/newCtramite    **@RequestMapping(value = "/guardar", method = RequestMethod.POST)**  **public String guardarConfPOST(HttpServletRequest request, @ModelAttribute GcConfTramite gcConfTramite, final ModelMap model);**  Método de tipo String que guarda por el ID del tramite los parámetros seteados obtenidos en el formulario utiliza el método save() con el repositorio gcConfTramiteRepository ubicado en el paquete sv.gob.mh.dgii.colas.repositories y valida que los datos guardados no estén duplicados carga la url ctramite/ctramites  **@RequestMapping(value = "{ctramite}/delete", method = RequestMethod.POST)**  **public String eliminar(HttpServletRequest request, @ModelAttribute GcConfTramite gcConfTramite, final ModelMap model, @PathVariable(value = "ctramite") Long ctramiteId);**  Método de tipo String que elimina los tramites segun el ID utiliza el Método delete() por medio del repositorio gcConfTramiteRepository  **@RequestMapping(value="{ctramite}/verCtramite", method = RequestMethod.GET)**  **public String verEquipo(HttpServletRequest request, @ModelAttribute GcConfTramite gcConfTramite, final ModelMap model, @PathVariable(value = "ctramite") Long ctramiteId);**  Método de tipo String que mueustra los tramites por su id carga la url ctramite/verCtramite  **public List<TbUnidadRecep> getUnidadesList();**  Método de tipo String que obtiene una lista de unidades receptoras.  **public String getUnidad();**  Método de tipo String que obtiene la unidad receptora y si viene vacio obtiene la ubicación física. | |

Tabla 3.7 controlador ConfTramiteController

| **PANTALLA DE MANTENIMIENTO DE EQUIPOS** | |
| --- | --- |
| **Controlador:** | EquipoController.java |
| **Descripción de Métodos de Negocio** | |
| **public GcEquipos gcEquiposList();**  Método que crea una instancia a la clase GcEquipos  **@RequestMapping(value = "/", method = RequestMethod.GET)**  **public String indexEquipo(HttpServletRequest request, @ModelAttribute GcEquipos gcEquipos, final ModelMap model);**  Método de tipo String que obtiene un listado de equipo por unidad y activos con los métodos getEquiposActivos() y getEquiposByCS() del repositorio gcEquipoRepository ubicado en el paquete sv.gob.mh.dgii.colas.repositories carga la url equipo/equipos  **@RequestMapping("/newEquipo")**  **public String newEquipo(HttpServletRequest request, @ModelAttribute GcEquipos gcEquipos, final ModelMap model);**  Método de tipo String que obtiene un listado de todos los equipos con el método findAll() y el repositorio gcTipoEquipoRepository ubicado en el paquete sv.gob.mh.dgii.colas.repositories obtiene getUnidadesList() carga la url equipo/newEquipo  **@RequestMapping(value = "{equipo}/editEquipo", method = RequestMethod.GET)**  **public String editEquipo(HttpServletRequest request,@ModelAttribute GcEquipos gcEquipos, final ModelMap model,@PathVariable(value = "equipo")**  Método de tipo String que edita los equipos por medio del ID carga la url equipo/newEquipo  **@RequestMapping(value = "/guardar", method = RequestMethod.POST)**  **public String guardarEquipo(HttpServletRequest request, @ModelAttribute GcEquipos gcEquipos, final ModelMap model);**  Método de tipo String que realiza la función de preparar el formulario con los parámetros seteados obtenidos en el formulario que se van a guardar por medio de su ID carga la url equipo/equipos  **@RequestMapping(value = "/guardar", method = RequestMethod.GET)**  **public String guardarEquipoGET(HttpServletRequest request, @ModelAttribute GcEquipos gcEquipos, final ModelMap model);**  Método de tipo String que guarda por el id de equipo los parámetros seteados obtenidos en el formulario utilizando el método save(); con el repositorio gcEquipoRepository carga la url equipo/equipos  **@RequestMapping(value = "{equipo}/delete", method = RequestMethod.POST)**  **public String eliminar(HttpServletRequest request, @ModelAttribute GcEquipos gcEquipos, final ModelMap model, @PathVariable(value = "equipo") Long equipoId) ;**  Método de tipo String que elimina el registro por el id del equipo utiliza el método delete(); con el repositorio gcEquipoRepository carga la url equipo/equipos  **@RequestMapping(value = "{equipo}/verEquipo", method = RequestMethod.GET)**  **public String verEquipo(HttpServletRequest request, @ModelAttribute GcEquipos gcEquipos, final ModelMap model, @PathVariable(value = "equipo") Long equipoId);**  Método de tipo String que muestra el equipo por su id carga la url equipo/verEquipo  **public List<TbUnidadRecep> getUnidadesList();**  Método de tipo String que obtiene una lista de unidades  **public String getUnidad();**  Método de tipo String que obtiene la unidad receptora y si viene vacio obtiene la ubicación física. | |

Tabla 3.8 controlador EquipoController

| **PANTALLA DE MANTENIMIENTO DE ESCRITORIOS** | |
| --- | --- |
| **Controlador:** | **EscritorioController.java** |
| **Descripción de Métodos de Negocio** | |
| **public EscritorioPojo escritotioPojo();**  Método que obtiene un listado de zonas si el rol asignado ROLE\_GC\_R1es igual al rol que tiene el usuario administrador si no obtiene las unidades retorna una lista de unidades y de zonas.  **@RequestMapping(value="/", method = RequestMethod.GET)**  **public String indexEscritorio(HttpServletRequest request, @ModelAttribute EscritorioPojo escritorioP, final ModelMap model);**  Método de tipo String que obtiene todos los escritorios si el rol asignado ROLE\_GC\_R1es igual al rol que tiene el usuario administrador con método allEscritorios()si no invoca el método getEscritoriosByUnidad () del repositorio gcEscritorioRepository ubicado en el paquete sv.gob.mh.dgii.colas.repositories; carga la url escritorio/escritorios  **@RequestMapping("/newEscritorio")**  **public String irAdministracionNewZona(HttpServletRequest request, @ModelAttribute EscritorioPojo escritorioP, final ModelMap model);**  Método de tipo String que setea un listado de zonas con el repositorio zonaRepository y el método findAll() también seta una lista de unidades invocando el método getUnidadesList() carga la url escritorio/newEscritorio  **@RequestMapping(value = "/", method = RequestMethod.POST)**  **public String index(HttpServletRequest request, @ModelAttribute EscritorioPojo escritorioP, final ModelMap model);**  Método de tipo String que obtiene la zona por ID y las unidades receptoras por ID,obtiene la información de los escritorios por el nombre inicializa la clase GcEscritorio y setea los parametros a obtener en el formulario luego utiliza el método save() con el repositorio gcEscritorioRepository ubicado en el paquete sv.gob.mh.dgii.colas.repositories para guardar los parametros seteados carga la url escritorio/escritorios.  **@RequestMapping(value = "/getzonas", method = RequestMethod.GET)**  **public String getZonasXunidad(HttpServletRequest request, @ModelAttribute EscritorioPojo escritorioPojo, final ModelMap model, @RequestParam(value = "cunidadRecepId", required = false) String unidad);**  Método de tipo String que obtiene un listado de zonas por unidad y setea el listado de zonas cargando la url escritorio/newEscritorio  **@RequestMapping(value="{escritorio}/editEscritorio", method = RequestMethod.GET)**  **public String editEscritorio(HttpServletRequest request, @ModelAttribute EscritorioPojo escritorioP, final ModelMap model, @PathVariable(value = "escritorio") Long escritorioId);**  Método de tipo String edita la información del escritorio de los parametros seteados carga la url escritorio/editEscritorio  **@RequestMapping(value = "{nEscritorioId}/delete", method = RequestMethod.POST)**  **public String eliminar(HttpServletRequest request, @ModelAttribute EscritorioPojo escritorioPojo, final ModelMap model, @PathVariable(value = "nEscritorioId") Long nEscritorioId);**  Método de tipo String que elimina la información del escritorio por el id si no es asignado por un usuario carga la url escritorio/escritorios  **@RequestMapping(value="{escritorio}/verEscritorio", method = RequestMethod.GET)**  **public String verEscritorio(HttpServletRequest request,ModelAttribute EscritorioPojo escritorioP, final ModelMap model, @PathVariable(value = "escritorio") Long escritorioId);**  Método de tipo String que muestra un listado de escritorios con la información de los parametros seteados carga la url escritorio/verEscritorio  **public List<TbUnidadRecep> getUnidadesList();**  Método de tipo String retorna una lista de unidades receptoras  **public String getUnidad();**  Método de tipo String que obtiene la unidad receptora y si viene vacio obtiene la ubicación física. | |

Tabla 3.9 controlador **EscritorioController**

| **PANTALLA DE MANTENIMIENTO DE LISTA DE REPRODUCCIÓN** | |
| --- | --- |
| **Controlador:** | MultimedServicioController**.java** |
| **Descripción de Métodos de Negocio** | |
| **@RequestMapping(value="/", method = RequestMethod.GET)**  **public String indexConfTiquete(HttpServletRequest request, @ModelAttribute GcConfTiquete gcConfTiquete, final ModelMap model);**  Método de tipo String que carga la url confrepro/confrepro  **@RequestMapping(value="/cargar", method = RequestMethod.POST)**  **public @ResponseBody String cargar(@RequestBody Map<String, Integer> map);**  Método de tipo String que obtiene las unidades en servicio con el Método getUnidadesEnServicio() del repositorio unidadRepository  **@RequestMapping(value="/leerLista", method = RequestMethod.POST)**  **public @ResponseBody String LeerLista(@RequestBody Map<String, String> map);**  Método de tipo String que obtiene una lista de unidades obtenida con el método getLista() del repositorio gcConfreproRepository del paquete sv.gob.mh.dgii.colas.repositories verifica que si la unidad obtenida esta vacia carga el método getListaVacia() del repositorio gcConfreproRepository  **public List<TbUnidadRecep> getUnidadesList();**  Método de tipo String retorna una lista de unidades receptoras  **public String getUnidad();**  Método de tipo String que obtiene la unidad receptora y si viene vacio obtiene la ubicación física. | |

Tabla 3.10 controlador MultimedServicioController

| **PANTALLA MANTENIMIENTO DE MULTIMEDIA** | |
| --- | --- |
| **Controlador:** | **MultimediaController.java** |
| **Descripción de Métodos de Negocio** | |
| **public MultimediaPojo serviciomulti();**  Método que invoca el método findA() del repositorio multimediaRepository ubicado en el paquete sv.gob.mh.dgii.colas.repositories; setea los paremetros de la clase GcMultimedia y retorna un listado de servicios multimedia.  **@RequestMapping(value="/", method = RequestMethod.GET)**  **public String indexServicio(HttpServletRequest request, @ModelAttribute MultimediaPojo servicioP, final ModelMap model)**  Método de tipo String que invoca el Método findA() del repositorio multimediaRepository setea los parametros obtenidos de GcMultimedia obtiene un listado de los parametros setados y carga la url multimedia/multimedia  **@RequestMapping(value = "/", method = RequestMethod.POST)**  **public String guardar(HttpServletRequest request, @ModelAttribute GcMultimedia gcMultimedia, final ModelMap model, @RequestParam("contenido") MultipartFile file)**  Método de tipo String que guarda los parametros seteados obtenidos en el formulario con el Método save() con el repositorio multimediaRepository cargando la url multimedia/multimedia  @RequestMapping("/newMultimedia")  **public String newServicio(HttpServletRequest request, @ModelAttribute GcMultimedia gcMultimedia, final ModelMap model);**  Método de tipo String que inicializa la clase GcMultimedia carga la url multimedia/newMultimedia  @RequestMapping(value="{servicio}/editMultimedia", method = RequestMethod.GET)  public String editMultimedia(HttpServletRequest request,@ModelAttribute GcMultimedia gcServicios\_2, final ModelMap model, @PathVariable(value = "servicio") Long servicioId);  Método de tipo String que edita por medio del ID los parametros seteados del pojo MultimediaPojo obtenidos del formulario y carga la url multimedia/editMultimedia  **@RequestMapping(value = "/saveMulti", method = RequestMethod.POST)**  **public @ResponseBody String guardar\_js(HttpServletRequest request,@RequestBody Map<String, String> map,final ModelMap model);**  Método de tipo String que le aplica un formato a la fecha y guarda los parametros obtenidos de la clase gcMultimedia  **@RequestMapping(value="{multimedia}/verMultimedia", method = RequestMethod.GET)**  **public String verMultimedia(HttpServletRequest request, @ModelAttribute GcMultimedia gcMultimedia, final ModelMap model, @PathVariable(value = "multimedia") Long multimediaId);**  Método de tipo String que busca por medio del id multimedia carga la url multimedia/verMultimedia  **@RequestMapping(value="{multimedia}/eliminar", method = RequestMethod.POST)**  **public String eliminar(HttpServletRequest request, ModelAttribute GcMultimedia gcMultimedia, final ModelMap model, @PathVariable(value = "multimedia") Long multimediaId);**  Método de tipo String que invoca el método borrarC() del repositorio multimediaRepository para eliminar el registro por el ID multimedia carga la url multimedia/multimedia  **@RequestMapping(value="/getBlob", method = RequestMethod.POST)**  **public @ResponseBody String getBlobo (HttpServletRequest request, @RequestBody Map<String, String> map);**  Método que captura el vid y lo almacena al disco duro para que se pueda visualizar | |

Tabla 3.11 controlador MultimediaController

| **PANTALLA MANTENIMIENTO DE CONFIGURACIÓN - EVALUACIÓN** | |
| --- | --- |
| **Controlador:** | **PreguntaEvaluacionController.java** |
| **Descripción de Métodos de Negocio** | |
| **@ModelAttribute("preguntapojo")**  **public PreguntaEvaluacionPojo preguntapojo();**  Método de tipo String que crea una nueva instancia de la clase pojo PreguntaEvaluacionPojo  **@RequestMapping(value = "/", method = RequestMethod.GET)**  **public String index(HttpServletRequest request, @ModelAttribute PreguntaEvaluacionPojo preguntaEvaluacionPojo, final ModelMap model);**  Método de tipo String que obtiene un listado de preguntas y ordena las respuestas por el id de pregunta.  Carga la url adminpreguntas/preguntas  **@RequestMapping("/newPregunta")**  **public String newPregunta(HttpServletRequest request, @ModelAttribute PreguntaEvaluacionPojo preguntaEvaluacionPojo, final ModelMap model);**  Método de tipo String que crea una instancia a la clase pojo preguntaEvaluacionPojo,invoca ael método getUnidadesList() y carga la url adminpreguntas/nzewPregunta  **@RequestMapping(value="{pregunta}/editPregunta", method = RequestMethod.GET)**  **public String editPregunta(HttpServletRequest request, @ModelAttribute PreguntaEvaluacionPojo preguntaEvaluacionPojo, final ModelMap model, PathVariable(value = "pregunta") Long preguntaId);**  Método de tipo String que edita la pregunta por su ID enviándole los parametros seteados obtenidos en el formulario cargando la url adminpreguntas/newPregunta  @RequestMapping(value="{pregunta}/verPregunta", method = RequestMethod.GET)  **public String verPregunta(HttpServletRequest request, @ModelAttribute PreguntaEvaluacionPojo preguntaEvaluacionPojo, final ModelMap model, @PathVariable(value = "pregunta") Long preguntaId);**  Método de tipo String que muestra por el id de pregunta los parametros seteados cargando la url adminpreguntas/verPregunta  **@RequestMapping(value="{pregunta}/delete", method = RequestMethod.POST)**  **public String delete(HttpServletRequest request, @ModelAttribute PreguntaEvaluacionPojo preguntaEvaluacionPojo, final ModelMap model,@PathVariable(value = "pregunta") Long preguntaId)**  Método de tipo String que elimina el registro seleccionado con el método delete() del repositorio gcRespuestasRepository ubicado en repositorio sv.gob.mh.dgii.colas.repositories carga la url adminpreguntas/preguntas  **@RequestMapping(value = "/", method = RequestMethod.POST)**  **public String guardar(HttpServletRequest request, @ModelAttribute PreguntaEvaluacionPojo preguntaEvaluacionPojo, final ModelMap model);**  Método de tipo String que guarda los cambios realizados carga la url adminpreguntas/preguntas  **public List<TbUnidadRecep> getUnidadesList();**  Método de tipo String retorna una lista de unidades receptoras  **public String getUnidad();**  Método de tipo String que obtiene la unidad receptora y si viene vacio obtiene la ubicación física. | |

Tabla 3.12 controlador PreguntaEvaluacionController

| **MANTENIMIENTO DE PRIORIDADES - CATÁLOGO** | |
| --- | --- |
| **Controlador:** | **PrioridadController.java** |
| **Descripción de Métodos de Negocio** | |
| **public GcPrioridad gcPrioridadList();**  Método de tipo String que crea una nueva instancia de la clase pojo GcPrioridad  **@RequestMapping(value="/", method = RequestMethod.GET)**  **public String indexPrioridad(HttpServletRequest request, @ModelAttribute GcPrioridad gcPrioridad, final ModelMap model);**  Método de tipo String que carga un listado de prioridades cargando la url prioridad/prioridades  **@RequestMapping(value="{prioridad}/editPrioridad", method = RequestMethod.GET)**  **public String editPrioridad(HttpServletRequest request,@ModelAttribute GcPrioridad gcPrioridad, final ModelMap model, @PathVariable(value = "prioridad") Long prioridadId) ;**  Método de tipo String que prepara el formulario para editarlo por medio del ID cargando la url prioridad/newPrioridad  **@RequestMapping(value = "/", method = RequestMethod.POST)**  **public String guardarPrioridad(HttpServletRequest request, @ModelAttribute GcPrioridad gcPrioridad, final ModelMap model);**  Método de tipo String que guarda los parámetros seteados obtenidos en el formulario con el método save() con el repositorio gcPrioridadRepository cargando la url prioridad/prioridades  **@RequestMapping(value = "{prioridad}/delete", method = RequestMethod.POST)**  **public String eliminar(HttpServletRequest request, @ModelAttribute GcPrioridad gcPrioridad, final ModelMap model, @PathVariable(value = "prioridad") Long prioridadId);**  Método de tipo String que elimina por el id de la prioridad seleccionada cargando la url prioridad/prioridades  **@RequestMapping(value="{prioridad}/verPrioridad", method = RequestMethod.GET)**  **public String verPrioridad(HttpServletRequest request, ModelAttribute GcPrioridad gcPrioridad, final ModelMap model, @PathVariable(value = "prioridad") Long prioridadId);**  Método de tipo String que muestra la prioridad seleccionada cargando la url prioridad/verPrioridad | |

Tabla 3.13 controlador **PrioridadController**

| **MANTENIMIENTO DE PRIORIDADES - CATÁLOGO** | |
| --- | --- |
| **Controlador:** | **ServicioController.java** |
| **Descripción de Métodos de Negocio** | |
| **public ServicioPojo serviciopojo();**  Método de tipo String que inicializa la clase pojo ServicioPojo y retorna una lista de servicios.  **@RequestMapping(value="/", method = RequestMethod.GET)**  **public String indexServicio(HttpServletRequest request, @ModelAttribute ServicioPojo servicioP, final ModelMap model);**  Método de tipo String que setea una lista de servicios y carga la url servicio/servicios  **RequestMapping("/newServicio")**  **public String newServicio(HttpServletRequest request, @ModelAttribute GcServicios gcServicios, final ModelMap model);**  Método de tipo String que prepara el formulario para agregar un nuevo servicio cargando la url servicio/newServicio  **@RequestMapping(value="{servicio}/editServicio", method = RequestMethod.GET)**  **public String editZona(HttpServletRequest request, @ModelAttribute GcServicios gcServicios, final ModelMap model, @PathVariable(value = "servicio") Long servicioId);**  Método de tipo String que prepara el formulario para editar el servicio por su id cargando la url servicio/newServicio  **@RequestMapping(value = "/", method = RequestMethod.POST)**  **public String guardar(HttpServletRequest request, @ModelAttribute GcServicios gcServicios, final ModelMap model);**  Método de tipo String que se encarga de guardar los parametros seteados obtenidos del formulario con el método save() y el repositorio servicioRepository ubicado en el paquete GcServicioRepository cargando la url servicio/servicios  **@RequestMapping(value="{servicio}/verServicio", method = RequestMethod.GET)**  **public String verZona(HttpServletRequest request, @ModelAttribute GcServicios gcServicios, final ModelMap model, @PathVariable(value = "servicio") Long servicioId);**  Método de tipo String que muestra el servicio seleccionado cargando la url servicio/verServicio  **@RequestMapping(value = "{servicio}/delete", method = RequestMethod.POST)**  **public String eliminar(HttpServletRequest request, @ModelAttribute ZonaPojo zonap, final ModelMap model, @PathVariable(value = "servicio") Long servicioId);**  Método de tipo String que se encarga de eliminar el servicio por medio del id carga la url servicio/servicios | |

Tabla 3.14 controlador **ServicioController**

| **MANTENIMIENTO DE TIPOS DE EQUIPOS - CATÁLOGO** | |
| --- | --- |
| **Controlador:** | **TipoEqueipoController.java** |
| **Descripción de Métodos de Negocio** | |
| **public GcPrioridad gcPrioridadList();**  Método que crea una nueva instancia de la clase GcPrioridad  **@RequestMapping(value="/", method = RequestMethod.GET)**  **public String indexPrioridad(HttpServletRequest request, @ModelAttribute GcTipoEquipo gcTipoEquipo, final ModelMap model)**  Método de tipo String que muestra un listado de equipo cargando la url tequipo/tequipos  **@RequestMapping("/newTequipo")**  **public String newPrioridad(HttpServletRequest request, @ModelAttribute GcTipoEquipo gcTipoEquipo, final ModelMap model);**  Método de tipo String prepara el nuevo formulario para agregar nuevo equipo cargando la url tequipo/newTequipo  **@RequestMapping(value="{tequipo}/editTequipo", method = RequestMethod.GET)**  **public String editPrioridad(HttpServletRequest request, @ModelAttribute GcTipoEquipo gcTipoEquipo, final ModelMap model, @PathVariable(value = "tequipo") Long tequipoId);**  Método de tipo String que prepara el formulario para editarlo cargando la url tequipo/newTequipo  **@RequestMapping(value = "/", method = RequestMethod.POST)**  **public String guardarPrioridad(HttpServletRequest request, @ModelAttribute GcTipoEquipo gcTipoEquipo, final ModelMap model);**  Método de tipo String que guarda el equipo verificando que no exista carga la url tequipo/tequipos  **@RequestMapping(value="{tequipo}/verTequipo", method = RequestMethod.GET)**  **public String verPrioridad(HttpServletRequest request, @ModelAttribute GcTipoEquipo gcTipoEquipo, final ModelMap model, @PathVariable(value = "tequipo") Long tequipoId) ;**  Método de tipo String que muestra el equipo por su id cargando la url tequipo/verTequipo | |

Tabla 3.15 controlador **TipoEqueipoController**

| **MANTENIMIENTO DE TIPOS DE TRAMITES- CATÁLOGO** | |
| --- | --- |
| **Controlador:** | **TramiteController.java** |
| **Descripción de Métodos de Negocio** | |
| **public GcTramite gcTramiteList();**  Método que crea una nueva instancia de la clase GcTramite  **@RequestMapping(value="/", method = RequestMethod.GET)**  **public String indexTramite(HttpServletRequest request, @ModelAttribute GcTramite gcTramite, final ModelMap model);**  Método de tipo String que carga un listado de tramites cargando la url tramite/tramites  **@RequestMapping("/newTramite")**  **public String newTramite(HttpServletRequest request, @ModelAttribute GcTramite gcTramite, final ModelMap model);**  Método de tipo String que carga una lista de servicios y tramites cargando la url tramite/newTramite  **@RequestMapping(value="{tramite}/editTramite", method = RequestMethod.GET)**  **public String editTramite(HttpServletRequest request, @ModelAttribute GcTramite gcTramite, final ModelMap model, @PathVariable(value = "tramite") Long tramiteId);**  Método de tipo String que edita los parametros seteados del tramite por su id cargando la url tramite/newTramite  **@RequestMapping(value = "/", method = RequestMethod.POST)**  **public String guardar(HttpServletRequest request, @ModelAttribute GcTramite gcTramite, final ModelMap model);**  Método de tipo String que guarda los parametros seteado obtenidos en el formulario cargando la url tramite/tramites  **@RequestMapping(value = "{tramite}/delete", method = RequestMethod.POST)**  **public String eliminar(HttpServletRequest request, @ModelAttribute GcTramite gcTramite, final ModelMap model, @PathVariable(value = "tramite") Long tramiteId);**  Método de tipo String que obtiene los tramites relacionados por medio de su id carga la url tramite/tramites  **@RequestMapping(value="{tramite}/verTramite", method = RequestMethod.GET)**  **public String verTramite(HttpServletRequest request, @ModelAttribute GcTramite gcTramite, final ModelMap model,@PathVariable(value = "tramite") Long tramiteId);**  Método de tipo String que muestra el tramite por su id cargando la url tramite/verTramite | |

Tabla 3.16 controlador **TramiteController**

| **MANTENIMIENTO DE ZONAS - CATÁLOGO** | |
| --- | --- |
| **Controlador:** | **ZonaController.java** |
| **Descripción de Métodos de Negocio** | |
| **public GcTramite gcTramiteList();**  Método que crea una nueva instancia de la clase ZonaPojo seteando una lista de unidades obtenida del método getUnidadesList();  **@RequestMapping(value = "/", method = RequestMethod.GET)**  **public String indexZona(HttpServletRequest request, @ModelAttribute ZonaPojo zonapojo, final ModelMap model);**  Método de tipo String que recorre una lista de la entidad GcZona y obtiene todas las zonas cargando la url zona/zonas  **@RequestMapping("/newZona")**  **public String irAdministracionNewZona(HttpServletRequest request, @ModelAttribute ZonaPojo zonapojo, final ModelMap model);**  Método de tipo String que setea una lista de unidades obtenidas del pojo ZonaPojo y obtiene el nombre de la máxima zona con el método getMaxZona() del repositorio zonaRepository cargando la url zona/newZona  **public String getUnidad();**  Método de tipo String que obtiene la unidad receptora y si viene vacio obtiene la ubicación física.  **@RequestMapping(value = "/", method = RequestMethod.POST)**  **public String index(HttpServletRequest request, @ModelAttribute ZonaPojo zonapojo, final ModelMap model);**  Método de tipo String que setea los parametros de la entidad GcZona obtenidos en el formulario y los guarda verificando que los datos no se dupliquen cargando la url zona/zonas  **@RequestMapping(value = "{zona}/editZona", method = RequestMethod.GET)**  **public String editZona(HttpServletRequest request, @ModelAttribute ZonaPojo zonapojo, final ModelMap model, @PathVariable(value = "zona") Long zonaId);**  Método de tipo String que edita la zona por su id cargando la url zona/editZona  **@RequestMapping(value = "/guardar", method = RequestMethod.POST)**  **public String guardar(HttpServletRequest request, @ModelAttribute ZonaPojo zonapojo, final ModelMap model);**  Método de tipo String que guarda los parámetros seteados obtenidos en el formulario con el método save(); y el repositorio zonaRepository sv.gob.mh.dgii.colas.repositories cargando la url zona/zonas  **@RequestMapping(value = "{zona}/verZona", method = RequestMethod.GET)**  **public String mostrar(HttpServletRequest request, @ModelAttribute ZonaPojo zonaP, final ModelMap model, @PathVariable(value = "zona") Long zonaId);**  Método de tipo String que muestra los parametros seteados cargando la url zona/verZona  **@RequestMapping(value = "{zona}/delete", method = RequestMethod.POST)**  **public String eliminar(HttpServletRequest request, ModelAttribute ZonaPojo zonap, final ModelMap model, @PathVariable(value = "zona") Long zonaId);**  Método de tipo String que elimina los parametros seteados por su id cargando la url zona/zonas  **public List<TbUnidadRecep> getUnidadesList();**  Método de tipo String que retorna una lista de unidades receptoras.  **public String getUnidad();**  Método de tipo String que obtiene la unidad receptora y si viene vacio obtiene la ubicación física.  **public String getNextString(String letra);**  Método de tipo String que retorna un conjunto de letras de la A-Z con su posicion.  **@RequestMapping(value = "/getzona", method = RequestMethod.GET)**  **public String getZonasXunidad(HttpServletRequest request, @ModelAttribute ZonaPojo zonaPojo, final ModelMap model, @RequestParam(value = "cunidadRecepId", required = false) String unidad)**  Método de tipo String que obtiene las zonas por unidad cargando la url zona/newZona | |

Tabla 3.17 controlador **ZonaController**

| **CONFIGURACIÓN DE REASIGNACIÓN** | |
| --- | --- |
| **Controlador:** | **ReasignacionController.java** |
| **Descripción de Métodos de Negocio** | |
| **public ReasignacionPojo gcreasignacion();**  Método de tipo String que retorna una lista de tramites y de unidades del pojo ReasignacionPojo.  **@RequestMapping(value = "/", method = RequestMethod.GET)**  **public String indexConfReasignacion(HttpServletRequest request,@ModelAttribute ReasignacionPojo reasignacionPojo, final ModelMap model);**  Método de tipo String que obtiene una lista de configuraciones generales con el método listaConfiguracionesGeneral() del repositorio GcConfTramiteRepository ubicado en el paquete sv.gob.mh.dgii.colas.repositories  **@RequestMapping("/newCreasignacion")**  **public String newCreasignacion(HttpServletRequest request, @ModelAttribute ReasignacionPojo reasignacionPojo, final ModelMap model);**  Método de tipo String que prepara el formulario para agregar una nueva asignacion seteando una lista de tramites y configuraciones cargando la url creasignacion/newCreasignacion  **@RequestMapping(value = "{creasignacion}/editCreasignacion", method = RequestMethod.GET)**  **public String editreasignacion(HttpServletRequest request, @ModelAttribute ReasignacionPojo reasignacionPojo, final ModelMap model, @PathVariable(value = "creasignacion") Long creasignacionId);**  Método de tipo String que prepara el formulario con los parametros seteados para editarlo carga el formulario creasignacion/newCreasignacion  **@RequestMapping(value = "/", method = RequestMethod.POST)**  **public String guardarReasignacion(HttpServletRequest request, @ModelAttribute ReasignacionPojo reasignacionPojo, final ModelMap model);**  Método de tipo String que prepara el formulario que guarda los parametros seteados obtenidos en el formulario con el repositorio gcConfTramiteRepository y el método save ubicado en el paquete sv.gob.mh.dgii.colas.repositories carga la url creasignacion/creasignaciones  **@RequestMapping(value = "{creasignacion}/delete", method = RequestMethod.POST)**  **public String eliminar(HttpServletRequest request, @ModelAttribute ReasignacionPojo reasignacionPojo, final ModelMap model, @PathVariable(value = "creasignacion") Long creasignacionId);**  Método de tipo String que elimina los parametros seteados de la configuración del tramite con el repositorio gcConfTramiteRepository y el método save() ubicado en el paquete sv.gob.mh.dgii.colas.repositories  **@RequestMapping(value = "{creasignacion}/verCreasignacion", method = RequestMethod.GET)**  **public String verCreasignacionList(HttpServletRequest request, @ModelAttribute ReasignacionPojo reasignacionPojo, final ModelMap model, @PathVariable(value = "creasignacion") Long creasignacionId);**  Método de tipo String que retorna la url creasignacion/verCreasignacion  **public List<TbUnidadRecep> getUnidadesList();**  Método de tipo String que retorna una lista de unidades receptoras.  **public String getUnidad();**  Método de tipo String que obtiene la unidad receptora y si viene vacio obtiene la ubicación física. | |

Tabla 3.18 controlador **ReasignacionController**

# Descripción de POJOS de sesión

**Clases POJO**

En la siguiente tabla se listan las diferentes clases POJO que se usan en el Modulo de Administracion y los diferentes paquetes en donde se pueden encontrar.

| Nombre de Opción | Ubicación | | Nombre |
| --- | --- | --- | --- |
| Configuracion de Usuarios | | sv.gob.mh.dgii.colas.pojos.entities  sv.gob.mh.dgii.colas.pojos.security | UsuarioPojo.java  PersonW.java |
| Mantenimiento de Alertas | | sv.gob.mh.dgii.colas.pojos.entities | AlertaPojo.java  TipoUsuarioPojo |
| Mantenimiento de Configuración de Llamado | | sv.gob.mh.dgii.colas.pojos.entities | CllamadoPojo.java |
| Configuracion de Tiquete | | -- | -- |
| Mantenimiento de Configuración de Tramite | | -- | -- |
| Mantenimiento de Equipos | | -- | -- |
| Mantenimiento de Escritorios | | sv.gob.mh.dgii.colas.pojos.entities | EscritorioPojo.java |
| Mantenimiento de Lista de Reproducción | | -- | -- |
| Mantenimiento de Multimedia | | sv.gob.mh.dgii.colas.pojos.entities | MultimediaPojo.java |
| Mantenimiento de Configuración - Evaluación | | sv.gob.mh.dgii.colas.pojos.entities | PreguntaEvaluacionPojo.java |
| Mantenimiento de Prioridades - Catálogo | | -- | -- |
| Mantenimiento de Secciones - Catálogo | | sv.gob.mh.dgii.colas.pojos.entities | ServicioPojo.java  ZonaPojo.java |
| Mantenimiento de Tipos de Equipos - Catálogo | | -- | -- |
| Mantenimiento de Tramites - Catálogo | | -- | -- |
| Mantenimiento de Zonas - Catálogo | | sv.gob.mh.dgii.colas.pojos.entities | ZonaPojo.java |
| Configuración de Reasignación | | sv.gob.mh.dgii.colas.pojos.entities | ReasignacionPojo.java |

Tabla 4.1 nombre y ubicación de las clases pojo

## Descripción de Clases POJO

En la siguiente tabla se describen las diferentes clases POJO que se usan en el Módulo de Administracion.

|  |  |
| --- | --- |
| Pantalla: Configuracion de Usuarios | |
| Pojo | Descripción |
| UsuarioPojo.java | Las variables que se encuentran dentro de esta clase son las siguientes  private String cUsuario;  private String dUsuario;  private GcEscritorio escritorio;  private List<GcEscritorio> listadoEscritorio =  new AutoPopulatingList<GcEscritorio>(GcEscritorio.class);  private List<GcTramite> listradoTramite =  new AutoPopulatingList<GcTramite>(GcTramite.class);  private List<GcTramite> addTramite =  new AutoPopulatingList<GcTramite>(GcTramite.class);  private List<GcUsuario> usuarioList =  new AutoPopulatingList<GcUsuario>(GcUsuario.class); |
| PersonW.java | Las variables que se encuentran dentro de esta clase son las siguientes  private static final long serialVersionUID = -4454216593897L;  private Boolean authorities;  private String nit;  private String displayName;  private String cadmTrib;  private String rol; //uniqueMember  private String ssello;  private Long itecnico;  private String cargo;  private String tipoUsuario;  private String ubicacionFisica;  private String unidadRecep;  private String rawPassword;  private String detFtpHost;  private String detFtpUsr;  private String detFtpPass;  private String detFtpInitialDir;  private String nomInstitucion;  private String nitRepresentante;  private String dui;  private Boolean ctrlchgpwd;  private String email;  private String emailPi;  private String emailServ;  private String banderas;  private Object clave;  private String givenName;  private String middleName;  private String sn;  private String requerimiento;  private String tipoInstitucion;  private String preferredlanguage;  private String fh\_ingreso\_serv;  private String uidNumber;  private String unidad;  private String gidNumber;  private String username;  private String orclisenabled;  private String pasaporteUsr;  private String pasaporteRep;  private String ipUsuario; |

Tabla 4.2 nombre y ubicación de las clases pojo

| Pantalla: Mantenimiento de Alertas | |
| --- | --- |
| Pojo | Descripción |
| AlertaPojo.java | Las variables que se encuentran dentro de esta clase son las siguientes  private List<TbUnidadRecep> listadoUnidades  = new AutoPopulatingList<TbUnidadRecep>(TbUnidadRecep.class);  private List<GcZona> listadoZonas  = new AutoPopulatingList<GcZona>(GcZona.class); |
| TipoUsuarioPojo.java | Las variables que se encuentran dentro de esta clase son las siguientes  private String usuario;  private String tipo; |

Tabla 4.3 nombre y ubicación de las clases pojo

| Pantalla: Mantenimiento de Configuración de Llamado | |
| --- | --- |
| Pojo | Descripción |
| CllamadoPojo.java | Las variables que se encuentran dentro de esta clase son las siguientes  private static final long serialVersionUID = 1L;  private TbUnidadRecep cUnidadRecep;  private String unidadRecep;  private BigInteger nNumLlamadas;  private BigInteger nIntervaloLlamada;  private short bActiva;  private String sMensaje;  private String sDescripcion;  private Long nConfllamadoId; |

Tabla 4.4 nombre y ubicación de las clases pojo

| Pantalla: Mantenimiento de Escritorios | |
| --- | --- |
| Pojo | Descripción |
| EscritorioPojo.java | Las variables que se encuentran dentro de esta clase son las siguientes:  private Integer cEscritorio;  private String dEscritorio  private Long nEscritorioId;  private String cunidadRecepId;  private Long cZonaId;  private TbUnidadRecep cUnidadRecep;  private String dZona;  private String cUsuarioCrea;  private String cUsuarioModi;  private Date fiVigencia;  private BigInteger nNumEscritorio;  private GcZona nZonaId;  private String cIdentificador;  private short bActiva;  private List<TbUnidadRecep> unidadesList = new AutoPopulatingList<TbUnidadRecep>(TbUnidadRecep.class);  private List<GcZona> zonaList = new AutoPopulatingList<GcZona>(GcZona.class);  private List<GcEscritorio> escritorioList=new AutoPopulatingList<GcEscritorio>(GcEscritorio.class);  private String dunidadRecep;  private int nNumEscritorioI; |

Tabla 4.5 nombre y ubicación de las clases pojo

| Pantalla: Mantenimiento de Multimedia | |
| --- | --- |
| Pojo | Descripción |
| MultimediaPojo.java | Las variables que se encuentran dentro de esta clase son las siguientes:  public Long nMultimediaId;  public String sTipo;  private Serializable xArchivo;  public Long nDuracion;  public String fiMultimedia;  public String ffMultimedia;  public String dMultimedia;  public String cUsuarioCrea;  public String cUsuarioModi;  public String fiVigencia;  public String ffVigencia;  public String fModifica;  public short bActiva;  public String sRuta;  private String sMultimedia; |

Tabla 4.6 nombre y ubicación de las clases pojo

| Pantalla: Mantenimiento de Configuración - Evaluación | |
| --- | --- |
| Pojo | Descripción |
| PreguntaEvaluacionPojo.java | Las variables que se encuentran dentro de esta clase son las siguientes:    private String pregunta;  private String correcto;  private String respuestas;  private String respuest;  private String unidad;  private String descripcion;  private Long preguntaId; |

Tabla 4.7 nombre y ubicación de las clases pojo

| Pantalla: Mantenimiento de Configuración - Evaluación | |
| --- | --- |
| Pojo | Descripción |
| PreguntaEvaluacionPojo.java | Las variables que se encuentran dentro de esta clase son las siguientes:    private String pregunta;  private String correcto;  private String respuestas;  private String respuest;  private String unidad;  private String descripcion;  private Long preguntaId; |

Tabla 4.8 nombre y ubicación de las clases pojo

| Mantenimiento de Secciones - Catálogo | |
| --- | --- |
| Pojo | Descripción |
| PreguntaEvaluacionPojo.java | Las variables que se encuentran dentro de esta clase son las siguientes:    private String pregunta;  private String correcto;  private String respuestas;  private String respuest;  private String unidad;  private String descripcion;  private Long preguntaId; |

Tabla 4.9 nombre y ubicación de las clases pojo

# Listado de Repositorios.

|  |  |  |
| --- | --- | --- |
| Nombre de la opcion | Ubicación | Nombre |
| Configuracion de Usuarios | sv.gob.mh.dgii.colas.repositories | GcAdminRepositoryImpl.java  GcEscritorioRepository.java  GcTramiteRepository.java  GcUsuarioRepository.java  PersonWRepository.java |
| Mantenimiento de Alertas | sv.gob.mh.dgii.colas.repositories | DeNotifEmailRepository.java  GcAlertaRepository.java  GcUnidadRecepRepository.java  GcZonaRepository.java |
| Mantenimiento de Configuración de Llamado | sv.gob.mh.dgii.colas.repositories | GcConfLlamadoRepository.java  GcUnidadRecepRepository.java |
| Configuracion de Tiquete | sv.gob.mh.dgii.colas.repositories | GcConfTiqueteRepository.java |
| Mantenimiento de Configuración de Tramite | sv.gob.mh.dgii.colas.repositories | GcTramiteRepository.java  GcUnidadRecepRepository.java |
| Mantenimiento de Equipos | sv.gob.mh.dgii.colas.repositories | GcEquipoRepository.java  GcTipoEquipoRepository.java  GcUnidadRecepRepository.java |
| Mantenimiento de Escritorios | sv.gob.mh.dgii.colas.repositories | GcEscritorioRepository.java  GcUnidadRecepRepository.java  GcUsuarioRepository.java  GcZonaRepository.java |
| Mantenimiento de Lista de Reproducción | sv.gob.mh.dgii.colas.repositories | GcMultiServicioRepository.java  GcUnidadRecepRepository.java |
| Mantenimiento de Multimedia | sv.gob.mh.dgii.colas.repositories | GcMultimediaRepository.java  GcMultimediaRepositoryImpl.java |
| Mantenimiento de Configuración - Evaluación | sv.gob.mh.dgii.colas.repositories | GcPreguntasRepository.java  GcRespuestasRepository.java  GcUnidadRecepRepository.java |
| Mantenimiento de Prioridades - Catálogo | sv.gob.mh.dgii.colas.repositories | GcPrioridadRepository.java |
| Mantenimiento de Secciones - Catálogo | sv.gob.mh.dgii.colas.repositories | GcServicioRepository.java |
| Mantenimiento de Tipos de Equipos - Catálogo | sv.gob.mh.dgii.colas.repositories | GcTipoEquipoRepository.java |
| Mantenimiento de Tramites - Catálogo | sv.gob.mh.dgii.colas.repositories | GcServicioRepository.java  GcTramiteRepository.java |
| Mantenimiento de Zonas - Catálogo | sv.gob.mh.dgii.colas.repositories | GcEscritorioRepository.java  GcUnidadRecepRepository.java  GcZonaRepository.java |
| Configuración de Reasignación | sv.gob.mh.dgii.colas.repositories | GcConfTramiteRepository.java  GcTramiteRepository.java  GcUnidadRecepRepository.java |

Tabla 5.1 Listado de repositorios

## Descripción de Repositorios

En la siguiente tabla se describen cada una de los repositorios usados en el módulo Declaración Sugerida de Renta (Declaración en Línea)

|  |  |
| --- | --- |
| OPCION: Configuracion de Usuarios | |
| Repositorio | **Descripción de métodos** |
| PersonWRepository.java | **public PersonW findByPrimaryKey(String name);**  Método que busca a la persona por su id |
| GcTramiteRepository.java | **public List<GcTramite> getTramitesByUser(String user);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM GcTramite t INNER JOIN t.gcUsuarioList u WHERE u.cUsuario = ?1 ") |
| GcAdminRepository.java | **public List<GcUsuario> getAllUser();**  Método de tipo String que obtiene todos los usuarios de la entidad GcUsuario |
| GcEscritorioRepository.java | **public List<GcEscritorio> getEscritoriosVacios(String unidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT e FROM GcEscritorio e where e.nEscritorioId not in (SELECT u.nEscritorioId FROM GcUsuario u) and c\_unidad\_recep= ?1 and e.ffVigencia is null") |

Tabla 5.2 Nombre y descripción de repositorios

| OPCION: Mantenimiento de Alertas | |
| --- | --- |
| Repositorio | Descripción de métodos |
| GcAlertaRepository.jva | **public Iterable<GcAlerta> findAllActivas(Date fecha);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT alerta FROM GcAlerta alerta WHERE alerta.bActiva = 1 AND ?1 BETWEEN alerta.fiVigencia AND alerta.ffVigencia ORDER BY alerta.nAlertaId ASC") |
| GcUnidadRecepRepository.java | **public List<TbUnidadRecep> getUnidadesEnServicio();**  Método que ejecuta la siguiente sentencia:  **@Query("SELECT t FROM TbUnidadRecep t WHERE t.mservicio IN('M','C') AND c\_unidad\_Recep NOT IN(SELECT d.id.clistaDet FROM TbListasValorDet d where c\_modulo='GC')")**  **public String getNombreUnidadCombinacion(String unidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query(value="select D\_LISTA from TB\_LISTAS\_VALOR where c\_lista =(select c\_lista from TB\_LISTAS\_VALOR\_DET where c\_modulo='GC' and c\_lista\_det=?1 and rownum = 1)",nativeQuery=true)  **public String getNombreUnidadCombinacion(String unidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query(value="select D\_LISTA from TB\_LISTAS\_VALOR where c\_lista =(select c\_lista from TB\_LISTAS\_VALOR\_DET where c\_modulo='GC' and c\_lista\_det=?1 and rownum = 1)",nativeQuery=true) |
| GcZonaRepository.java | **public List<GcZona> getZonasByUnidad(String cunidadRecepId);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM GcZona t where C\_UNIDAD\_RECEP = ?1 and t.ffVigencia is null ")  **public List<GcZona> getAllZonasActivas();**  Método que ejecuta la siguiente sentencia:  **@Query("SELECT t FROM GcZona t WHERE t.bActiva = 1 and t.ffVigencia is null")** |
| DeNotifEmailRepository.java | **public Long getLast();**  Método que ejecuta la siguiente sentencia:  @Query("Select max(o.iindice) from DeNotifEmail o ") |
| CrudRepository.java | **public <S extends T> S save(S s);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository |

Tabla 5.3 Nombre y descripción de repositorios

| OPCION: Mantenimiento de Configuración de Llamado | |
| --- | --- |
| Repositorio | Descripción de métodos |
| GcUnidadRecepRepository.java | **public T findOne(ID id);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository |
| GcUnidadRecepRepository.java | **public List<TbUnidadRecep> getUnidadesEnServicio();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM TbUnidadRecep t WHERE t.mservicio IN('M','C') AND c\_unidad\_Recep NOT IN(SELECT d.id.clistaDet FROM TbListasValorDet d where c\_modulo='GC')")  **public List<TbUnidadRecep> getUnidadesCombinaciones();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM TbUnidadRecep t WHERE t.mservicio IN('M','C') AND c\_unidad\_Recep IN(SELECT MIN(d.id.clistaDet) FROM TbListasValorDet d where c\_modulo='GC' group by c\_lista)")  **public String getNombreUnidadCombinacion(String unidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query(value="select D\_LISTA from TB\_LISTAS\_VALOR where c\_lista =(select c\_lista from TB\_LISTAS\_VALOR\_DET where c\_modulo='GC' and c\_lista\_det=?1 and rownum = 1)",nativeQuery=true) |
| GcConfLlamadoRepository.java | **public List<GcConfLlamado> getAllConf();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT g FROM GcConfLlamado g WHERE g.bActiva=1 ")  **public List<GcConfLlamado> getConfsByCS(String cunidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT g FROM GcConfLlamado g WHERE g.bActiva=1 and c\_unidad\_recep = ?1") |
| CrudRepository.java | **public void delete(T t);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository  **public <S extends T> S save(S s);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository  **public T findOne(ID id);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository |
| GcConfLlamadoRepository.java | **public int getCounByCS(String cunidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT count(g) FROM GcConfLlamado g WHERE g.bActiva=1 and c\_unidad\_recep = ?1") |

Tabla 5.4 Nombre y descripción de repositorios

| OPCION: Configuracion de Tiquete | |
| --- | --- |
| Repositorio | Descripción de métodos |
| GcConfTiqueteRepository.java | **public TbListasValor n\_unidad\_tlv(String cUnidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query("Select a from TbListasValor a where a.id.clista=(select b.id.clista from TbListasValorDet b where b.id.clistaDet= ?1 and b.id.cmodulo='GC')")  **public TbUnidadRecep n\_unidad\_tur(String cUnidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query("Select c from TbUnidadRecep c where cunidadRecep=?1")  **public List<GcConfTiquete> listaConfiguraciones(String cUnidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT c FROM GcConfTiquete c where C\_UNIDAD\_RECEP = ?1 order by nVersion desc")  **public List<String> lista\_plantilla();**  Método que ejecuta la siguiente sentencia:  @Query(value="select n\_plantilla\_tiq||';'||s\_nombre from gc\_plantilla\_tiq",nativeQuery = true)  **public GcConfTiquete maxVer(String nv,String unidad);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT c FROM GcConfTiquete c where N\_VERSION = ?1 and C\_UNIDAD\_RECEP=?2 order by nVersion desc")  **public String lineas\_tiquete(String cUnidadRecep,String maxver);**  Método que ejecuta la siguiente sentencia:  **@Query("select trim(replace(replace(regexp\_replace(regexp\_replace(X\_CONTENIDO, '</[^>]\*>','%%'),'<[^>]\*>','&&'),'&& ',''),'%% ','')) from GcConfTiquete a where C\_UNIDAD\_RECEP=?1 and N\_VERSION=?2")**  **public int maximaVersion(String unidad);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT nvl(max(nVersion),0) FROM GcConfTiquete c where C\_UNIDAD\_RECEP=?1")  **public GcConfTiquete currTq(String id);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT c FROM GcConfTiquete c where N\_CONF\_TIQ\_ID = ?1 order by nVersion desc")  **public void noActivo(String cUnidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query("update GcConfTiquete set B\_ACTIVA=0 where C\_UNIDAD\_RECEP=?1")  **public void setActivo(String cUnidadRecep, String ver);**  Método que ejecuta la siguiente sentencia:  @Query("update GcConfTiquete set B\_ACTIVA=1 where C\_UNIDAD\_RECEP=?1 and N\_VERSION=?2" )  **public List<GcConfTiquete> listaConfiguraciones(String cUnidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT c FROM GcConfTiquete c where C\_UNIDAD\_RECEP = ?1 order by nVersion desc") |
| CrudRepository.java | **public <S extends T> S save(S s);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository |

Tabla 5.5 Nombre y descripción de repositorios

| OPCION: Mantenimiento de Configuración de Tramite | |
| --- | --- |
| Repositorio | Descripción de métodos |
| GcUnidadRecepRepository.java | **public String getNombreUnidadCombinacion(String unidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query(value="select D\_LISTA from TB\_LISTAS\_VALOR where c\_lista =(select c\_lista from TB\_LISTAS\_VALOR\_DET where c\_modulo='GC' and c\_lista\_det=?1 and rownum = 1)",nativeQuery=true)  **public List<TbUnidadRecep> getUnidadesCombinaciones();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM TbUnidadRecep t WHERE t.mservicio IN('M','C') AND c\_unidad\_Recep IN(SELECT MIN(d.id.clistaDet) FROM TbListasValorDet d where c\_modulo='GC' group by c\_lista)")  **public String getNombreUnidadCombinacion(String unidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query(value="select D\_LISTA from TB\_LISTAS\_VALOR where c\_lista =(select c\_lista from TB\_LISTAS\_VALOR\_DET where c\_modulo='GC' and c\_lista\_det=?1 and rownum = 1)",nativeQuery=true) |
| GcTramiteRepository.java | **public List<GcTramite> getAllTramites();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM GcTramite t where t.bActiva=1") |

Tabla 5.6 Nombre y descripción de repositorios

| OPCION: Mantenimiento de Equipos | |
| --- | --- |
| Repositorio | Descripción de métodos |
| GcEquipoRepository.java | **public List<GcEquipos> getEquiposActivos();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM GcEquipos t where t.bActiva = 1 ")  **public List<GcEquipos> getEquiposByCS(String unidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM GcEquipos t where c\_unidad\_recep = ?1 and t.bActiva = 1 ") |
| CrudRepository.java | **public T findOne(ID id);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository  **public <S extends T> S save(S s);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository  **public void delete(ID id);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository  **public Iterable<T> findAll();**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository |
| GcUnidadRecepRepository.java | **public List<TbUnidadRecep> getUnidadesEnServicio();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM TbUnidadRecep t WHERE t.mservicio IN('M','C') AND c\_unidad\_Recep NOT IN(SELECT d.id.clistaDet FROM TbListasValorDet d where c\_modulo='GC')")  **public List<TbUnidadRecep> getUnidadesCombinaciones();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM TbUnidadRecep t WHERE t.mservicio IN('M','C') AND c\_unidad\_Recep IN(SELECT MIN(d.id.clistaDet) FROM TbListasValorDet d where c\_modulo='GC' group by c\_lista)")  **public String getNombreUnidadCombinacion(String unidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query(value="select D\_LISTA from TB\_LISTAS\_VALOR where c\_lista =(select c\_lista from TB\_LISTAS\_VALOR\_DET where c\_modulo='GC' and c\_lista\_det=?1 and rownum = 1)",nativeQuery=true)  **public List<TbUnidadRecep> getUnidadesEnServicio(String cunidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM TbUnidadRecep t where t.mservicio IN('C','M') and C\_UNIDAD\_RECEP = SIIT.PKG\_COLAS\_UTILS.CENTRO\_SERVICIO(nvl(?1,C\_UNIDAD\_RECEP))") |

Tabla 5.6 Nombre y descripción de repositorios

| OPCION: Mantenimiento de Escritorios | |
| --- | --- |
| Repositorio | Descripción de métodos |
| GcZonaRepository.java | **public List<GcZona> getAllZonas();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM GcZona t WHERE t.ffVigencia is null ")  **public List<GcZona> getZonasByUnidad(String cunidadRecepId);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM GcZona t where C\_UNIDAD\_RECEP = ?1 and t.ffVigencia is null ") |
| GcUnidadRecepRepository.java | **public T findOne(ID id);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository  **public List<TbUnidadRecep> getUnidadesEnServicio();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM TbUnidadRecep t WHERE t.mservicio IN('M','C') AND c\_unidad\_Recep NOT IN(SELECT d.id.clistaDet FROM TbListasValorDet d where c\_modulo='GC')")  **public List<TbUnidadRecep> getUnidadesCombinaciones();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM TbUnidadRecep t WHERE t.mservicio IN('M','C') AND c\_unidad\_Recep IN(SELECT MIN(d.id.clistaDet) FROM TbListasValorDet d where c\_modulo='GC' group by c\_lista)")  **public String getNombreUnidadCombinacion(String unidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query(value="select D\_LISTA from TB\_LISTAS\_VALOR where c\_lista =(select c\_lista from TB\_LISTAS\_VALOR\_DET where c\_modulo='GC' and c\_lista\_det=?1 and rownum = 1)",nativeQuery=true) |
| GcEscritorioRepository.java | **public List<GcEscritorio> allEscritorios();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT e FROM GcEscritorio e where e.ffVigencia is null")  **public List<GcEscritorio> getEscritoriosByUnidad(String Unidad);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT e FROM GcEscritorio e where C\_UNIDAD\_RECEP = ?1 and e.ffVigencia is null")  **public GcEscritorio getEscritorioByName(String cIdentificador,Long EscritorioId, String cUnidadRecep, Long nZonaId);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT e FROM GcEscritorio e where c\_identificador = ?1 and n\_num\_escritorio = ?2 and c\_unidad\_recep = ?3 and n\_zona\_id= ?4 and e.ffVigencia is null ")  **public GcEscritorio getEscritorioByNameEdit(String cIdentificador,BigInteger nNumEscritorio, TbUnidadRecep cUnidadRecep, GcZona nZonaId,Long nEscritorioId);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT e FROM GcEscritorio e where e.cIdentificador = ?1 and e.nNumEscritorio = ?2 and e.cUnidadRecep = ?3 and e.nZonaId = ?4 and e.ffVigencia is null and e.nEscritorioId != ?5") |
| CrudRepository.java | **public <S extends T> S save(S s);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository  **public T findOne(ID id);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository |
| GcUsuarioRepository.java | **public int getUsersEscritorio(Long escritorio);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT count(u) FROM GcUsuario u where n\_escritorio\_id= ?1") |

Tabla 5.7 Nombre y descripción de repositorios

| OPCION: Mantenimiento de Lista de Reproducción | |
| --- | --- |
| Repositorio | Descripción de métodos |
| GcMultiServicioRepository.java | **public List<String> getLista(String cunidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query(value="select str from( " +  "select n\_multimedia\_id||';'||d\_multimedia||';'||n\_orden str,n\_orden orden " +  "from gc\_multimedia a inner join gc\_multimed\_servicio b using (n\_multimedia\_id) " +  "where b\_activa=1 and " +  "sysdate between a.FI\_MULTIMEDIA and a.FF\_MULTIMEDIA " +  "and (c\_unidad\_recep=?1) " +  "union " +  "select n\_multimedia\_id||';'||d\_multimedia||';'||10000 str,10000 orden " +  "from gc\_multimedia a " +  "where b\_activa=1 and " +  "sysdate between a.FI\_MULTIMEDIA and a.FF\_MULTIMEDIA " +  "and n\_multimedia\_id not in (select n\_multimedia\_id from gc\_multimed\_servicio where c\_unidad\_recep=?1) " +  ") order by orden",nativeQuery = true)    **public List<String> getListaVacia();**  Método que ejecuta la siguiente sentencia:  @Query(value="select n\_multimedia\_id||';'||d\_multimedia||';'||10000 from gc\_multimedia a " +  "where b\_activa=1 and " +  "sysdate between a.FI\_MULTIMEDIA and a.FF\_MULTIMEDIA " +  "order by n\_multimedia\_id",nativeQuery = true)  **public void borrar(String cUnidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query(value="delete from GC\_MULTIMED\_SERVICIO where C\_UNIDAD\_RECEP=?1",nativeQuery = true)  **public void insertar(String cUnidadRecep,int nmulti,int norden);**  Método que ejecuta la siguiente sentencia:  @Query(value="insert into GC\_MULTIMED\_SERVICIO(C\_UNIDAD\_RECEP,N\_MULTIMEDIA\_ID,N\_ORDEN) values(?1,?2,?3)",nativeQuery = true) |
| GcUnidadRecepRepository.java | **public List<TbUnidadRecep> getUnidadesEnServicio();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM TbUnidadRecep t WHERE t.mservicio IN('M','C') AND c\_unidad\_Recep NOT IN(SELECT d.id.clistaDet FROM TbListasValorDet d where c\_modulo='GC')")  **public List<TbUnidadRecep> getUnidadesCombinaciones();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM TbUnidadRecep t WHERE t.mservicio IN('M','C') AND c\_unidad\_Recep IN(SELECT MIN(d.id.clistaDet) FROM TbListasValorDet d where c\_modulo='GC' group by c\_lista)")  **public String getNombreUnidadCombinacion(String unidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query(value="select D\_LISTA from TB\_LISTAS\_VALOR where c\_lista =(select c\_lista from TB\_LISTAS\_VALOR\_DET where c\_modulo='GC' and c\_lista\_det=?1 and rownum = 1)",nativeQuery=true)  **public List<TbUnidadRecep> getUnidadesEnServicio(String cunidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM TbUnidadRecep t where t.mservicio IN('C','M') and C\_UNIDAD\_RECEP = SIIT.PKG\_COLAS\_UTILS.CENTRO\_SERVICIO(nvl(?1,C\_UNIDAD\_RECEP))")  **public String getNombreUnidadCombinacion(String unidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query(value="select D\_LISTA from TB\_LISTAS\_VALOR where c\_lista =(select c\_lista from TB\_LISTAS\_VALOR\_DET where c\_modulo='GC' and c\_lista\_det=?1 and rownum = 1)",nativeQuery=true) |

Tabla 5.8 Nombre y descripción de repositorios

| OPCION: Mantenimiento de Multimedia | |
| --- | --- |
| Repositorio | Descripción de métodos |
| GcMultimediaRepository.java | **public List<String> findA();**  Método que ejecuta la siguiente sentencia:  @Query(value="SELECT n\_Multimedia\_Id||';'||s\_Tipo||';'||" +  " n\_Duracion||';'||TO\_CHAR(fi\_Multimedia,'DD/MM/YYYY')||';'||TO\_CHAR(ff\_Multimedia,'DD/MM/YYYY')||';'||" +  " d\_Multimedia||';'||c\_Usuario\_Crea||';'||c\_Usuario\_Modi||';'||" +  " to\_char(fi\_Vigencia,'DD/MM/YYYY')||';'||to\_char(f\_Modifica,'DD/MM/YYYY')||';'||" +  " b\_Activa||';'||s\_Ruta FROM gc\_multimedia order by n\_Multimedia\_Id ", nativeQuery = true)  **public void actualizar(long i,long ndur,String cumod,String dmult,Date fmodi,int bacti,String stipo,Date ffmul,Date fimul,String sruta);**  Método que ejecuta la siguiente sentencia:  @Query(value="UPDATE GC\_MULTIMEDIA " +  " SET N\_DURACION = ?2 ," +  " C\_USUARIO\_MODI = ?3 ," +  " D\_MULTIMEDIA = ?4 ," +  " F\_MODIFICA = ?5 ," +  " B\_ACTIVA = ?6 ," +  " S\_TIPO = ?7 , " +  " FF\_MULTIMEDIA = ?8 , " +  " FI\_MULTIMEDIA = ?9 , " +  " S\_RUTA = ?10 " +  " WHERE N\_MULTIMEDIA\_ID = ?1 ",nativeQuery = true )  **public String findO(Long i);**  Método que ejecuta la siguiente sentencia:  @Query(value="SELECT n\_Multimedia\_Id||';'||s\_Tipo||';'||" +  " n\_Duracion||';'||TO\_CHAR(fi\_Multimedia,'DD/MM/YYYY')||';'||TO\_CHAR(ff\_Multimedia,'DD/MM/YYYY')||';'||" +  " d\_Multimedia||';'||c\_Usuario\_Crea||';'||c\_Usuario\_Modi||';'||" +  " to\_char(fi\_Vigencia,'DD/MM/YYYY')||';'||to\_char(f\_Modifica,'DD/MM/YYYY')||';'||" +  " b\_Activa||';'||s\_Ruta||';'||s\_multimedia||';'||TO\_CHAR(fi\_vigencia,'DD/MM/YYYY') FROM gc\_multimedia where n\_Multimedia\_Id=?1 ", nativeQuery = true)  **public void borrarC(long i);**  Método que ejecuta la siguiente sentencia:  @Query(value="delete from gc\_multimedia where n\_multimedia\_id=?1",nativeQuery = true ) |

Tabla 5.9 Nombre y descripción de repositorios

| OPCION: Mantenimiento de Configuración - Evaluación | |
| --- | --- |
| Repositorio | Descripción de métodos |
| GcPreguntasRepository.java | **public List<GcPreguntas> preguntaYrespuestas(String cUnidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT c FROM GcPreguntas c where C\_UNIDAD\_RECEP = ?1 ") |
| CrudRepository.java | **public T findOne(ID id);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository  **public void delete(ID id);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository |
| GcRespuestasRepository.java | **public List<GcRespuestas> respuestasByPregutna(Long nPreguntaId);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT c FROM GcRespuestas c where N\_PREGUNTA\_ID = ?1 ")  **public List<GcRespuestas> respuestasByPregutna(Long nPreguntaId);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT c FROM GcRespuestas c where N\_PREGUNTA\_ID = ?1 ") |
| GcUnidadRecepRepository.java | **public List<TbUnidadRecep> getUnidadesEnServicio();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM TbUnidadRecep t WHERE t.mservicio IN('M','C') AND c\_unidad\_Recep NOT IN(SELECT d.id.clistaDet FROM TbListasValorDet d where c\_modulo='GC')")  **public List<TbUnidadRecep> getUnidadesCombinaciones();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM TbUnidadRecep t WHERE t.mservicio IN('M','C') AND c\_unidad\_Recep IN(SELECT MIN(d.id.clistaDet) FROM TbListasValorDet d where c\_modulo='GC' group by c\_lista)")  **public String getNombreUnidadCombinacion(String unidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query(value="select D\_LISTA from TB\_LISTAS\_VALOR where c\_lista =(select c\_lista from TB\_LISTAS\_VALOR\_DET where c\_modulo='GC' and c\_lista\_det=?1 and rownum = 1)",nativeQuery=true) |

Tabla 5.10 Nombre y descripción de repositorios

| OPCION: Mantenimiento de Prioridades - Catálogo | |
| --- | --- |
| Repositorio | Descripción de métodos |
| GcPrioridadRepository.java | **public Long getPrioridadDuplicada(String nombre,Long id);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT count(p) FROM GcPrioridad p WHERE p.bActiva in(1,0) AND REPLACE(upper(p.sNombre),' ','') = REPLACE(upper(?1) ,' ','') and p.nPrioridadId != ?2") |
| CrudRepository.java | **public Iterable<T> findAll();**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository  **public T findOne(ID id);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository  **public <S extends T> S save(S s);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository |

Tabla 5.11 Nombre y descripción de repositorios

| OPCION: Mantenimiento de Secciones - Catálogo | |
| --- | --- |
| Repositorio | Descripción de métodos |
| GcServicioRepository.java | **public List<GcServicios> getAllSecciones();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT s FROM GcServicios s WHERE s.bActiva in(0,1) ")  **public Integer getServicioDuplicado(String servicio,Long servicioId);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT count(s) FROM GcServicios s where REPLACE(upper(s.sNombre),' ','') = REPLACE(upper(?1) ,' ','') and s.nServiciosId != nvl(?2,0) and s.bActiva in(0,1) ")  **public BigInteger getServicioMaxOrden();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT max(s.nOrden)+1 FROM GcServicios s WHERE s.bActiva in(0,1) ")  **public Integer getServicioRelacionado(Long servisioId);**  Método que ejecuta la siguiente sentencia:  @Query(value="SELECT count(\*) FROM gc\_servicios s where EXISTS (select \* from gc\_tramite tra where tra.n\_servicios\_id= s.n\_servicios\_id) and s.n\_servicios\_id=?1 s.b\_Activa in(0,1)", nativeQuery = true) |
| CrudRepository.java | **public T findOne(ID id);**Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository  **public <S extends T> S save(S s);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository |

Tabla 5.12 Nombre y descripción de repositorios

| OPCION: Mantenimiento de Tipos de Equipos - Catálogo | |
| --- | --- |
| Repositorio | Descripción de métodos |
| GcTipoEquipoRepository.java | **public String getTipoEquipoByName(String sNombre);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t.sNombre FROM GcTipoEquipo t where t.sNombre = ?1 ") |
| CrudRepository.java | **public Iterable<T> findAll();**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository  **public T findOne(ID id);**Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository  **public <S extends T> S save(S s);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository  **public void delete(T t);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository |

Tabla 5.13 Nombre y descripción de repositorios

| OPCION: Mantenimiento de Tramites - Catálogo | |
| --- | --- |
| Repositorio | Descripción de métodos |
| GcTramiteRepository.java | **public int getTramiteDuplicado(String tramite,Long tramiteId);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT count(s) FROM GcTramite s where REPLACE(upper(s.sNombre),' ','') = REPLACE(upper(?1) ,' ','') and s.nTramiteId != nvl(?2,0)")  **public int getTramiteRelacionados(Long tramiteId);**  Método que ejecuta la siguiente sentencia:  @Query(value="select count(\*) from gc\_tramite tra where( (EXISTS (select \* from gc\_conf\_tramite ct where ct.n\_tramite\_id=tra.n\_tramite\_id)) or (exists (select \* from gc\_usr\_tra ut where ut.n\_tramite\_id=tra.n\_tramite\_id))) and tra.n\_tramite\_id=?1 ", nativeQuery = true) |
| CrudRepository.java | **public Iterable<T> findAll();**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository  **public T findOne(ID id);**Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository  **public <S extends T> S save(S s);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository  **public void delete(T t);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository |

Tabla 5.14 Nombre y descripción de repositorios

| OPCION: Mantenimiento de Zonas - Catálogo | |
| --- | --- |
| Repositorio | Descripción de métodos |
| CrudRepository.java | **public T findOne(ID id);**Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository  **public <S extends T> S save(S s);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository |
| GcZonaRepository.java | **public List<GcZona> getAllZonas();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM GcZona t WHERE t.ffVigencia is null ")  **public List<GcZona> getZonasByUnidad(String cunidadRecepId);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM GcZona t where C\_UNIDAD\_RECEP = ?1 and t.ffVigencia is null ")  **public String getMaxZona(String unidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query(value = "SELECT max(s\_nombre) FROM gc\_zona where C\_UNIDAD\_RECEP = ?1 and ff\_Vigencia is null", nativeQuery = true)  **public String getZonasByUnidadName(String cunidadRecepId,String sNombre);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t.sNombre FROM GcZona t where C\_UNIDAD\_RECEP = ?1 and REPLACE(upper(t.sNombre),' ','') =REPLACE(upper(?2),' ','') and t.ffVigencia is null")  **public GcZona getZonasByUnidadNameId(String cunidadRecepId,String sNombre, Long nZonaId);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM GcZona t where C\_UNIDAD\_RECEP = ?1 and t.sNombre = ?2 and t.nZonaId != ?3 and rownum=1 and t.ffVigencia is null ") |
| GcUnidadRecepRepository.java | **public TbUnidadRecep getUnidadEnServicioE(String cunidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM TbUnidadRecep t WHERE t.mservicio IN('C','M') and t.cunidadRecep = ?1 ")  **public List<TbUnidadRecep> getUnidadesCombinaciones();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM TbUnidadRecep t WHERE t.mservicio IN('M','C') AND c\_unidad\_Recep IN(SELECT MIN(d.id.clistaDet) FROM TbListasValorDet d where c\_modulo='GC' group by c\_lista)")  **public String getNombreUnidadCombinacion(String unidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query(value="select D\_LISTA from TB\_LISTAS\_VALOR where c\_lista =(select c\_lista from TB\_LISTAS\_VALOR\_DET where c\_modulo='GC' and c\_lista\_det=?1 and rownum = 1)",nativeQuery=true)  **public List<TbUnidadRecep> getUnidadesEnServicio(String cunidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM TbUnidadRecep t where t.mservicio IN('C','M') and C\_UNIDAD\_RECEP = SIIT.PKG\_COLAS\_UTILS.CENTRO\_SERVICIO(nvl(?1,C\_UNIDAD\_RECEP))") |
| GcEscritorioRepository.java | **public int getEscritoriosByZona(Long zona);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT count(e) FROM GcEscritorio e where n\_zona\_id = ?1 and e.ffVigencia is null") |

Tabla 5.15 Nombre y descripción de repositorios

| OPCION: Configuración de Reasignación | |
| --- | --- |
| Repositorio | Descripción de métodos |
| GcConfTramiteRepository.java | **public List<GcConfTramite> listaConfiguracionesGeneral();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT c FROM GcConfTramite c where c.nComportamiento is not null")  **public List<GcConfTramite> listaConfiguracionesLocal(String cUnidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT c FROM GcConfTramite c where C\_UNIDAD\_RECEP = ?1 and c.nComportamiento is not null")  **public GcConfTramite existe(String cUnidadRecep, Long tramiteId);**  Método que ejecuta la siguiente sentencia:  @Query("SELECT c FROM GcConfTramite c where C\_UNIDAD\_RECEP = ?1 and N\_TRAMITE\_ID = ?2 ")  **public List<TbUnidadRecep> getUnidadesEnServicio();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM TbUnidadRecep t WHERE t.mservicio IN('M','C') AND c\_unidad\_Recep NOT IN(SELECT d.id.clistaDet FROM TbListasValorDet d where c\_modulo='GC')")  **public List<TbUnidadRecep> getUnidadesCombinaciones();**  Método que ejecuta la siguiente sentencia:  @Query("SELECT t FROM TbUnidadRecep t WHERE t.mservicio IN('M','C') AND c\_unidad\_Recep IN(SELECT MIN(d.id.clistaDet) FROM TbListasValorDet d where c\_modulo='GC' group by c\_lista)")  **public String getNombreUnidadCombinacion(String unidadRecep);**  Método que ejecuta la siguiente sentencia:  @Query(value="select D\_LISTA from TB\_LISTAS\_VALOR where c\_lista =(select c\_lista from TB\_LISTAS\_VALOR\_DET where c\_modulo='GC' and c\_lista\_det=?1 and rownum = 1)",nativeQuery=true) |
| CrudRepository.java | **public T findOne(ID id);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository  **public <S extends T> S save(S s);**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository  **public Iterable<T> findAll();**  Método generado de la interfaz CrudRepository ubicada en el repositorio org.springframework.data.repository |

Tabla 5.16 Nombre y descripción de repositorios

# Listado de Componentes.

|  |  |
| --- | --- |
| Nombre del Componente | Ubicación |
| AppInfoInterceptor.java | sv.gob.mh.dgii.det.components |

Tabla 6.1 Listado y Ubicación de componentes

## Descripción de Componentes.

| Nombre del Componente | Descripcion |
| --- | --- |
| AppInfoInterceptor.java | Clase que extiende de HandlerInterceptorAdapter la cual sobre-escribe los siguientes Métodos:  **public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler);**  Esta aplicación siempre devuelve cierto.  **public void postHandle(HttpServletRequest request, HttpServletResponse response, Object handler, ModelAndView modelAndView);**  Esta aplicación está vacía. |

Tabla 6.2 Nombre y descripción de componentes

# Configuracion de proyectos (@Configuration y @Beans en general, esto es todo el paquete de sv.gob.mh.dgii.det.config)

| Nombre del archivo de configuración | Ubicación  sv.gob.mh.dgii.colas.config | Descripción |
| --- | --- | --- |
| HibernateConfig | **public LocalSessionFactoryBean alertsSessionFactory();**  Método que establece el origen de datos para ser utilizados por la SessionFactory, Especifica el paquete "sv.gob.mh.dgii.model" para buscar la autodetección de sus clases de entidad en la ruta de clases, establece las propiedades de hibernate, establecer la ubicación de un único archivo de configuración de Hibernate XML, por ejemplo, como recurso de ruta de clases "ruta de clases: hibernate.cfg.xml".  **public HibernateTransactionManager transactionManager();**  Método que ajusta la instancia que debe gestionar las transacciones y obtiene el objeto del método alertsSessionFactory().  **public HibernateExceptionTranslator exceptionTranslation();**  Método que crea una nueva instancia de **HibernateExceptionTranslator.**  **final Properties hibernateProperties();**  Método que establece las propiedades de la clase Properties   |  | | --- | |  | | | |
| RepositoryConfig | **@EnableJpaRepositories(basePackages = { "sv.gob.mh.dgii" },**  **includeFilters = @ComponentScan.Filter(pattern = ".\*.repositories.\*", type = FilterType.REGEX))**  Esta anotación me dice que paquete inyectara como repositorios.  **@ComponentScan(basePackages = "sv.gob.mh.dgii", useDefaultFilters = false, includeFilters = @Filter(pattern = ".\*.components.\*", type = FilterType.REGEX));**  Configura directivas de escaneo de componentes y me dice que paquete inyectara como repositorios. | | |
| SecurityConfig | **protected void configure(HttpSecurity http);**  Método que contiene información sobre cómo autenticar a los usuarios, Asegura que cualquier petición a nuestra aplicación requiere que el usuario sea autenticado,permite que los usuarios se autentican con formulario basado entrada,permite que los usuarios se autentican con autenticación básica HTTP.  **public DefaultLdapAuthoritiesPopulator ldapAuthoritiesPopulator();**  Constructor de escenarios de búsqueda de grupo y  suministra los contextos utilizados para buscar roles de usuario.  **public DgiiFilterInvocationSecurityMetadataSource dgiiSecurityMetadataSource();**  Método que invoca el paquete "sv.gob.mh.dgii.colas.security.PropertyFileSecurityBuilder".  **public AuthenticationManager authenticationManagerBean();**  Método de anulación authenticationManagerBean en WebSecurityConfigurerAdapter para exponer el AuthenticationManager construido usando configure(AuthenticationManagerBuilder)  **public RoleVoter roleVoter();**  Método que especifica un prefijo de rol al usuario.  **public AuthenticatedVoter authenticatedVoter();**  Método para la autenticación de usuario dependiendo si es anónima o si desea que se recuerde.  **public FilterSecurityInterceptor dgiiFilterSecurityInterceptor();**  Método que realiza el  manejo de la seguridad de los recursos HTTP a través de un filtro de aplicación.  **BaseLdapPathContextSource contextSource();**  Interfaz para ser implementado por ContextSources que son capaces de proporcionar la ruta LDAP base.  **public void configureAuthentification(AuthenticationManagerBuilder auth)**  Método que obtiene la configuración requerirá que cualquier URL que se solicita será necesario un usuario con el rol de "ROLE\_USER". | | |
| SecurityWebApplicationInitializer | Clase que extiende de AbstractSecurityWebApplicationInitializer la cual sobre-escribe los siguientes Métodos:  **public class SecurityWebApplicationInitializer extends AbstractSecurityWebApplicationInitializer.** | | |
| WebConfig | Clase que extiende de WebMvcConfigurerAdapter la cual sobre-escribe los siguientes Métodos:  **public String appName();**  Método que obtiene el nombre completo de la app.  **public void addResourceHandlers(ResourceHandlerRegistry registry) ;**  Agregar controladores para servir recursos estáticos como imágenes, js y css, archivos desde ubicaciones específicas bajo raíz de la aplicación web, la ruta de clase, y otros.    **public LocaleResolver localeResolver();**  Interfaz de estrategias de resolución de localización basadas en web que permite tanto la resolución de la configuración regional a través de la solicitud y la modificación de la configuración regional a través de la solicitud y la respuesta.  **public ViewResolver viewResolver() ;**  Método para ver si el estado no cambia durante el funcionamiento de la aplicación.  **public CommonsMultipartResolver multipartResolver():**  Método encargado de setear por defecto la configuración utf-8 y de ¡configurar los tamaños máximos de upload y el uso de la memoria máxima. | | |
| WebSocketConfig | **public void configureMessageBroker(MessageBrokerRegistry config):**  Método encargado de configurar enableSimpleBroker y setApplicationDestinationPrefixes.  **public void registerStompEndpoints(StompEndpointRegistry registry):**  Método encargado de configurar el sock en addEndpoint withSockJS. | | |

TABLA 7.1 Configuraciones del Módulo de Colas

# Documentos Relacionados a la Base de Datos

**Tablas**

En las siguientes páginas se describen cada uno de los campos de las tablas de la base de datos. Cuando la **descripción no tiene nada**, quiere decir que dicha tabla no **tiene comentarios en la base de datos**.

| GC\_ESCRITORIO  Nombre de secuencia: SEQ\_GC\_ESCRITORIO | | | |
| --- | --- | --- | --- |
| Nombre del atributo | Tipo de dato | Null | Descripción |
| N\_ESCRITORIO\_ID | NUMBER | No | Llave primaria de la tabla |
| N\_ZONA\_ID | NUMBER | No | Referencia a la zona a que pertenece el escritorio |
| C\_UNIDAD\_RECEP | VARCHAR2(5 BYTE) | No | Código del centro de servicio |
| B\_ACTIVA | NUMBER(1,0) | No | Bandera que indica si el registro esta activo o no |
| D\_ESCRITORIO | VARCHAR2(256 BYTE) | Yes | Descripción del registro |
| C\_USUARIO\_CREA | VARCHAR2(100 BYTE) | No | Código del usuarioque crea el registro |
| C\_USUARIO\_MODI | VARCHAR2(100 BYTE) | No | Código del usuarioque modifica el registro |
| FI\_VIGENCIA | DATE | No | Fecha en que el registro es creado |
| FF\_VIGENCIA | DATE | Yes | Fecha en que el registro deja de tener vigencia |
| F\_MODIFICA | DATE | Yes | Fecha en que el registro fue modificado |
| C\_IDENTIFICADOR | VARCHAR2(1 BYTE) | Yes | Letra que identifica a un escritorio al momento del llamado |
| N\_NUM\_ESCRITORIO | NUMBER | No | Número con el cual se identifica al escritorio al momento del llamado |
| B\_FILA\_ESP | NUMBER(1,0) | No | Si el escritorio será Fila Especial |

**Tabla 8.1** GC\_ESCRITORIO

|  |  |  |  |
| --- | --- | --- | --- |
| GC\_TRAMITE  Nombre de secuencia: SEQ\_GC\_TRAMITE | | | |
| Nombre del atributo | **Tipo de dato** | **Null** | **Descripción** |
| N\_TRAMITE\_ID | NUMBER | No | Llave primaria de la tabla |
| N\_SERVICIOS\_ID | NUMBER | No | Referencia al servicio a que pertenece el trámite |
| N\_PESO | NUMBER | No | Cuantificación de la prioridad que debe tener el tramite luego de aplicarle todos los criterios |
| B\_NIT\_REQUERIDO | NUMBER(1,0) | No | Bandera que indica si es obligatorio proporionar el nit |
| B\_ESCALAMIENTO | NUMBER(1,0) | No | Bandera que indica si el tramite será escalado al supervisor |
| D\_TRAMITE | VARCHAR2(256 BYTE) | Yes | Descripción del registro |
| C\_USUARIO\_CREA | VARCHAR2(100 BYTE) | No | Código del usuarioque crea el registro |
| C\_USUARIO\_MODI | VARCHAR2(100 BYTE) | No | Código del usuarioque modifica el registro |
| FI\_VIGENCIA | DATE | No | Fecha en que el registro es creado |
| FF\_VIGENCIA | DATE | Yes | Fecha en que el registro deja de tener vigencia |
| F\_MODIFICA | DATE | Yes | Fecha en que el registro fue modificado |
| B\_ACTIVA | NUMBER(1,0) | No | Bandera que indica si el registro esta activo o no |
| S\_NOMBRE | VARCHAR2(256 BYTE) | No | Nombre del tramite |
| N\_ORDEN | NUMBER | Yes | Orden en que se presentaran los tramites en pantalla |

**Tabla 8.2** GC\_TRAMITE

|  |  |  |  |
| --- | --- | --- | --- |
| GC\_USUARIO  Nombre de secuencia: -- | | | |
| Nombre del atributo | **Tipo de dato** | **Null** | **Descripción** |
| N\_ESCRITORIO\_ID | NUMBER(38,0) | No | Escritorio asociado al usuario |
| C\_USUARIO | VARCHAR2(256 BYTE) | No | Código del usuario |

**Tabla 8.3** GC\_USUARIO

|  |  |  |  |
| --- | --- | --- | --- |
| GC\_ALERTA  Nombre de secuencia: SEQ\_GC\_ALERTA | | | |
| Nombre del atributo | **Tipo de dato** | **Null** | **Descripción** |
| N\_ALERTA\_ID | NUMBER(38,0) | No | Llave primaria de la tabla |
| N\_SERVICIOS\_ID | NUMBER | No | Referencia al servicio al que esta asociado la alerta |
| C\_UNIDAD\_RECEP | VARCHAR2(5 BYTE) | Yes | Código del centro de servicio |
| S\_MENSAJE | VARCHAR2(1024 BYTE) | No | Mensaje que desplegará o enviará la alerta |
| S\_TIPO | VARCHAR2(15 BYTE) | No | Tipo de alerta |
| FH\_NOTIFICACION | DATE | Yes | Fecha y hora en que se ejecutará la alerta programada |
| B\_CORREO | NUMBER(1,0) | No | Bandera que indica si la Alerta enviará correo o no |
| B\_PANTALLA | NUMBER(1,0) | No | Bandera que indica si se mostrará o no en la pantalla |
| C\_USUARIO\_NOTIFICA | VARCHAR2(1024 BYTE) | Yes | Listado de Usuarios a notificar |
| S\_CORREO\_NOTIFICA | VARCHAR2(1024 BYTE) | Yes | Listado de correos a los que la alerta notificará |
| C\_USUARIO\_CREA | VARCHAR2(100 BYTE) | No | Código del usuarioque crea el registro |
| C\_USUARIO\_MODI | VARCHAR2(100 BYTE) | No | Código del usuarioque modifica el registro |
| FI\_VIGENCIA | DATE | No | Fecha en que el registro es creado |
| FF\_VIGENCIA | DATE | Yes | Fecha en que el registro deja de tener vigencia |
| F\_MODIFICA | DATE | Yes | Fecha en que el registro fue modificado |
| B\_ACTIVA | NUMBER(1,0) | No | Bandera que indica si el registro esta activo o no |

**Tabla 8.4** GC\_ALERTA

|  |  |  |  |
| --- | --- | --- | --- |
| GC\_ZONA  Nombre de secuencia: SEQ\_GC\_ZONA | | | |
| Nombre del atributo | **Tipo de dato** | **Null** | **Descripción** |
| N\_ZONA\_ID | NUMBER | No | Llave primaria de la tabla |
| C\_UNIDAD\_RECEP | VARCHAR2(5 BYTE) | No | Código del centro de servicio |
| N\_SILLAS\_ESPERA | NUMBER | Yes | Número de sillas en la zona de atención |
| B\_ACTIVA | NUMBER(1,0) | No | Bandera que indica si el registro esta activo o no |
| D\_ZONA | VARCHAR2(256 BYTE) | Yes | Descripción del registro |
| C\_USUARIO\_CREA | VARCHAR2(100 BYTE) | No | Código del usuarioque crea el registro |
| C\_USUARIO\_MODI | VARCHAR2(100 BYTE) | No | Código del usuarioque modifica el registro |
| FI\_VIGENCIA | DATE | No | Fecha en que el registro es creado |
| FF\_VIGENCIA | DATE | Yes | Fecha en que el registro deja de tener vigencia |
| F\_MODIFICA | DATE | Yes | Fecha en que el registro fue modificado |
| S\_NOMBRE | VARCHAR2(256 BYTE) | Yes | Nombre de la zona |

**Tabla 8.5** GC\_ZONA

|  |  |  |  |
| --- | --- | --- | --- |
| GC\_CONF\_LLAMADO  Nombre de secuencia: SEQ\_GC\_CONF\_LLAMADO | | | |
| Nombre del atributo | **Tipo de dato** | **Null** | **Descripción** |
| C\_UNIDAD\_RECEP | VARCHAR2(5 BYTE) | No | Código del centro de servicio |
| N\_NUM\_LLAMADAS | NUMBER | No | Número de llamadas que se harán al convocar a un nuevo contribuyente |
| N\_INTERVALO\_LLAMADA | NUMBER | No | Segundos que transcurirán entre una llamada y la siguiente |
| B\_ACTIVA | NUMBER(1,0) | No | Bandera que indica si el registro esta activo o no |
| S\_MENSAJE | VARCHAR2(1024 BYTE) | No | Mensaje que desplegará en pantalla al hacer el llamado |
| C\_USUARIO\_CREA | VARCHAR2(100 BYTE) | No | Código del usuarioque crea el registro |
| C\_USUARIO\_MODI | VARCHAR2(100 BYTE) | No | Código del usuarioque modifica el registro |
| FI\_VIGENCIA | DATE | No | Fecha en que el registro es creado |
| FF\_VIGENCIA | DATE | Yes | Fecha en que el registro deja de tener vigencia |
| F\_MODIFICA | DATE | Yes | Fecha en que el registro fue modificado |
| S\_DESCRIPCION | VARCHAR2(256 BYTE) | Yes | Descripcion del registro |
| N\_CONFLLAMADO\_ID | NUMBER | No | Llave primaria de la tabla |

**Tabla 8.6** GC\_CONF\_LLAMADO

|  |  |  |  |
| --- | --- | --- | --- |
| GC\_CONF\_TIQUETE  Nombre de secuencia: SEQ\_GC\_CONF\_TIQUETE | | | |
| Nombre del atributo | **Tipo de dato** | **Null** | **Descripción** |
| C\_UNIDAD\_RECEP | VARCHAR2(5 BYTE) | No | Código del centro de servicio |
| B\_ACTIVA | NUMBER(1,0) | No | Bandera que indica si el registro esta activo o no |
| X\_CONTENIDO | CLOB | No | Formato del tiquete que se entregaran en los centros de servicio |
| C\_USUARIO\_CREA | VARCHAR2(100 BYTE) | No | Código del usuarioque crea el registro |
| C\_USUARIO\_MODI | VARCHAR2(100 BYTE) | No | Código del usuarioque modifica el registro |
| FF\_VIGENCIA | DATE | Yes | Fecha en que el registro deja de tener vigencia |
| FI\_VIGENCIA | DATE | No | Fecha en que el registro es creado |
| F\_MODIFICA | DATE | Yes | Fecha en que el registro fue modificado |
| N\_CONF\_TIQ\_ID | NUMBER | No | Llave primaria de la tabla |
| N\_VERSION | NUMBER | Yes | Versión del tiquete en relación alcentro de servicio |
| X\_IMAGEN | BLOB | Yes | Imagen que va en el tiquete. |

**Tabla 8.7** GC\_CONF\_TIQUETE

|  |  |  |  |
| --- | --- | --- | --- |
| GC\_CONF\_TRAMITE  Nombre de secuencia: SEQ\_GC\_CONF\_TRAMITE | | | |
| Nombre del atributo | **Tipo de dato** | **Null** | **Descripción** |
| N\_CONF\_TRA\_ID | NUMBER | No | Llave primaria de la tabla |
| C\_UNIDAD\_RECEP | VARCHAR2(5 BYTE) | No | Código del centro de servicio |
| N\_TRAMITE\_ID | NUMBER | No | Referencia al tramite a que se refiere la configuración |
| N\_ATENCION\_PROM | NUMBER | No | Tiempo promedio de atención |
| N\_TIEMPO\_ESPERA | NUMBER | Yes | Tiempo, en minutos, de espera máximo que debe tardarse en la cola |
| N\_TIEMPO\_HOLGURA | NUMBER | Yes | Tiempo, en minutos, que debe esperarse antes de incluirlo en la cola |
| C\_USUARIO\_CREA | VARCHAR2(100 BYTE) | No | Código del usuarioque crea el registro |
| C\_USUARIO\_MODI | VARCHAR2(100 BYTE) | No | Código del usuarioque modifica el registro |
| FI\_VIGENCIA | DATE | No | Fecha en que el registro es creado |
| FF\_VIGENCIA | DATE | Yes | Fecha en que el registro deja de tener vigencia |
| F\_MODIFICA | DATE | Yes | Fecha en que el registro fue modificado |
| N\_COMPORTAMIENTO | NUMBER(6,0) | Yes | Comportamiento: 0-al final, 1-al inicio, 2- en medio, 3 por peso |
| N\_PESO | NUMBER(6,0) | Yes | peso de la reasignacion |
| N\_PROM\_ESPERA | NUMBER | Yes | Promedio real (en segundos) de espera en cola |
| N\_PROM\_ATENCION | NUMBER | Yes | Promedio real (en segundos) de atención en escritorio |

**Tabla 8.8** GC\_CONF\_TRAMITE

|  |  |  |  |
| --- | --- | --- | --- |
| GC\_CONF\_TRAMITE  Nombre de secuencia: SEQ\_GC\_EQUIPOS | | | |
| Nombre del atributo | **Tipo de dato** | **Null** | **Descripción** |
| N\_CONF\_TRA\_ID | NUMBER | No | Llave primaria de la tabla |
| C\_UNIDAD\_RECEP | VARCHAR2(5 BYTE) | No | Código del centro de servicio |
| N\_TRAMITE\_ID | NUMBER | No | Referencia al tramite a que se refiere la configuración |
| N\_ATENCION\_PROM | NUMBER | No | Tiempo promedio de atención |
| N\_TIEMPO\_ESPERA | NUMBER | Yes | Tiempo, en minutos, de espera máximo que debe tardarse en la cola |
| N\_TIEMPO\_HOLGURA | NUMBER | Yes | Tiempo, en minutos, que debe esperarse antes de incluirlo en la cola |
| C\_USUARIO\_CREA | VARCHAR2(100 BYTE) | No | Código del usuarioque crea el registro |
| C\_USUARIO\_MODI | VARCHAR2(100 BYTE) | No | Código del usuarioque modifica el registro |
| FI\_VIGENCIA | DATE | No | Fecha en que el registro es creado |
| FF\_VIGENCIA | DATE | Yes | Fecha en que el registro deja de tener vigencia |
| F\_MODIFICA | DATE | Yes | Fecha en que el registro fue modificado |
| N\_COMPORTAMIENTO | NUMBER(6,0) | Yes | Comportamiento: 0-al final, 1-al inicio, 2- en medio, 3 por peso |
| N\_PESO | NUMBER(6,0) | Yes | peso de la reasignacion |
| N\_PROM\_ESPERA | NUMBER | Yes | Promedio real (en segundos) de espera en cola |
| N\_PROM\_ATENCION | NUMBER | Yes | Promedio real (en segundos) de atención en escritorio |

**Tabla 8.9** GC\_CONF\_TRAMITE

|  |  |  |  |
| --- | --- | --- | --- |
| GC\_MULTIMEDIA  Nombre de secuencia: SEQ\_GC\_MULTIMEDIA | | | |
| Nombre del atributo | **Tipo de dato** | **Null** | **Descripción** |
| N\_MULTIMEDIA\_ID | NUMBER(38,0) | No | Llave primaria de la tabla |
| S\_TIPO | VARCHAR2(256 BYTE) | No | Tipo de contenido multimedia |
| N\_DURACION | NUMBER | No | Tiempo en minutos en que el contenido multimedia se reproducirá |
| FI\_MULTIMEDIA | DATE | No | Fecha de inicio de emisión del contenido |
| FF\_MULTIMEDIA | DATE | No | Fecha en que finaliza la emisión del contenido |
| D\_MULTIMEDIA | VARCHAR2(256 BYTE) | Yes | Descripción del registro |
| C\_USUARIO\_CREA | VARCHAR2(100 BYTE) | No | Código del usuarioque crea el registro |
| C\_USUARIO\_MODI | VARCHAR2(100 BYTE) | No | Código del usuarioque modifica el registro |
| FI\_VIGENCIA | DATE | No | Fecha en que el registro es creado |
| FF\_VIGENCIA | DATE | Yes | Fecha en que el registro deja de tener vigencia |
| F\_MODIFICA | DATE | Yes | Fecha en que el registro fue modificado |
| B\_ACTIVA | NUMBER(1,0) | No | Bandera que indica si el registro esta activo o no |
| S\_RUTA | VARCHAR2(1024 BYTE) | Yes | Directorio donde se encuentra el contenido multimedia a mostrar |
| X\_ARCHIVO | BLOB | Yes | Archivo con información multimedia. |
| S\_MULTIMEDIA | VARCHAR2(256 BYTE) | No | Nombre del archivo del contenido multimedia |

**Tabla 8.10** GC\_MULTIMEDIA

|  |  |  |  |
| --- | --- | --- | --- |
| GC\_PREGUNTAS  Nombre de secuencia: SEQ\_GC\_PREGUNTAS | | | |
| Nombre del atributo | **Tipo de dato** | **Null** | **Descripción** |
| N\_PREGUNTA\_ID | NUMBER | No | Llave primaria de la tabla |
| C\_UNIDAD\_RECEP | VARCHAR2(5 BYTE) | No | Código del centro de servicio |
| S\_PREGUNTA | VARCHAR2(256 BYTE) | No | Texto de la pregunta |
| N\_PONDERACION | NUMBER | No | Cuantificación de valor de la pregunta con respecto a la nota global de la evaluación |
| D\_PREGUNTA | VARCHAR2(512 BYTE) | Yes | Descripción del registro |
| C\_USUARIO\_CREA | VARCHAR2(100 BYTE) | No | Código del usuarioque crea el registro |
| C\_USUARIO\_MODI | VARCHAR2(100 BYTE) | No | Código del usuarioque modifica el registro |
| FI\_VIGENCIA | DATE | No | Fecha en que el registro es creado |
| FF\_VIGENCIA | DATE | Yes | Fecha en que el registro deja de tener vigencia |
| F\_MODIFICA | DATE | Yes | Fecha en que el registro fue modificado |

**Tabla 8.11** GC\_PREGUNTAS

|  |  |  |  |
| --- | --- | --- | --- |
| GC\_PREGUNTAS  Nombre de secuencia: SEQ\_GC\_PREGUNTAS | | | |
| Nombre del atributo | **Tipo de dato** | **Null** | **Descripción** |
| N\_PREGUNTA\_ID | NUMBER | No | Llave primaria de la tabla |
| C\_UNIDAD\_RECEP | VARCHAR2(5 BYTE) | No | Código del centro de servicio |
| S\_PREGUNTA | VARCHAR2(256 BYTE) | No | Texto de la pregunta |
| N\_PONDERACION | NUMBER | No | Cuantificación de valor de la pregunta con respecto a la nota global de la evaluación |
| D\_PREGUNTA | VARCHAR2(512 BYTE) | Yes | Descripción del registro |
| C\_USUARIO\_CREA | VARCHAR2(100 BYTE) | No | Código del usuarioque crea el registro |
| C\_USUARIO\_MODI | VARCHAR2(100 BYTE) | No | Código del usuarioque modifica el registro |
| FI\_VIGENCIA | DATE | No | Fecha en que el registro es creado |
| FF\_VIGENCIA | DATE | Yes | Fecha en que el registro deja de tener vigencia |
| F\_MODIFICA | DATE | Yes | Fecha en que el registro fue modificado |

**Tabla 8.12** GC\_PREGUNTAS

|  |  |  |  |
| --- | --- | --- | --- |
| GC\_RESPUESTAS  Nombre de secuencia: SEQ\_GC\_RESPUESTAS | | | |
| Nombre del atributo | **Tipo de dato** | **Null** | **Descripción** |
| N\_RESPUESTA\_ID | NUMBER | No | Llave primaria de la tabla |
| N\_PREGUNTA\_ID | NUMBER | No | Referencia a la pregunta a la que pertenece la respuesta |
| S\_RESPUESTA | VARCHAR2(128 BYTE) | No | Texto de la respuesta |
| N\_ESCALA | NUMBER | No | Ponderación numérica que tiene la pregunta |
| C\_USUARIO\_CREA | VARCHAR2(100 BYTE) | No | Código del usuarioque crea el registro |
| C\_USUARIO\_MODI | VARCHAR2(100 BYTE) | No | Código del usuarioque modifica el registro |
| FI\_VIGENCIA | DATE | No | Fecha en que el registro es creado |
| FF\_VIGENCIA | DATE | Yes | Fecha en que el registro deja de tener vigencia |
| F\_MODIFICA | DATE | Yes | Fecha en que el registro fue modificado |

**Tabla 8.13** GC\_RESPUESTAS

|  |  |  |  |
| --- | --- | --- | --- |
| GC\_PRIORIDAD  Nombre de secuencia: SEQ\_GC\_PRIORIDAD | | | |
| Nombre del atributo | **Tipo de dato** | **Null** | **Descripción** |
| N\_PRIORIDAD\_ID | NUMBER | No | Llave primaria de la tabla |
| N\_PESO | NUMBER | No | Cuantificación de la prioridad |
| D\_PRIORIDAD | VARCHAR2(256 BYTE) | Yes | Descripción del registro |
| C\_USUARIO\_CREA | VARCHAR2(100 BYTE) | No | Código del usuarioque crea el registro |
| C\_USUARIO\_MODI | VARCHAR2(100 BYTE) | No | Código del usuarioque modifica el registro |
| FI\_VIGENCIA | DATE | No | Fecha en que el registro es creado |
| FF\_VIGENCIA | DATE | Yes | Fecha en que el registro deja de tener vigencia |
| F\_MODIFICA | DATE | Yes | Fecha en que el registro fue modificado |
| B\_ACTIVA | NUMBER(1,0) | No | Bandera que indica si el registro esta activo o no |
| B\_FILA\_ESP | NUMBER(1,0) | No | Si será asignado a la fila especial |
| S\_NOMBRE | VARCHAR2(256 BYTE) | Yes | Nombre de la prioridad |

**Tabla 8.14** GC\_PRIORIDAD

|  |  |  |  |
| --- | --- | --- | --- |
| GC\_SERVICIOS  Nombre de secuencia: SEQ\_GC\_PRIORIDAD | | | |
| Nombre del atributo | **Tipo de dato** | **Null** | **Descripción** |
| N\_SERVICIOS\_ID | NUMBER(38,0) | No | Llave primaria de la tabla |
| D\_SERVICIOS | VARCHAR2(256 BYTE) | Yes | Descripción del registro |
| C\_USUARIO\_CREA | VARCHAR2(100 BYTE) | No | Código del usuarioque crea el registro |
| C\_USUARIO\_MODI | VARCHAR2(100 BYTE) | No | Código del usuarioque modifica el registro |
| FI\_VIGENCIA | DATE | No | Fecha en que el registro es creado |
| FF\_VIGENCIA | DATE | Yes | Fecha en que el registro deja de tener vigencia |
| F\_MODIFICA | DATE | Yes | Fecha en que el registro fue modificado |
| N\_ORDEN | NUMBER | Yes | Orden en que se presentaran los servicios en pantalla |
| B\_ACTIVA | NUMBER(1,0) | No | Bandera que indica si el registro esta activo o no |
| S\_NOMBRE | VARCHAR2(256 BYTE) | Yes | Nombre del centro de servicio |

**Tabla 8.15 GC\_SERVICIOS**

|  |  |  |  |
| --- | --- | --- | --- |
| GC\_TIPO\_EQUIPO  Nombre de secuencia: SEQ\_GC\_TIPO\_EQUIPO | | | |
| Nombre del atributo | **Tipo de dato** | **Null** | **Descripción** |
| N\_TIPO\_ID | NUMBER | No | Llave primaria de la tabla |
| D\_TIPO\_EQUIPO | VARCHAR2(512 BYTE) | Yes | Descripción del registro |
| S\_NOMBRE | VARCHAR2(256 BYTE) | Yes | Nombre del tipo de equipo |

**Tabla 8.16 GC\_TIPO\_EQUIPO**

**GC\_TIPO\_EQUIPO**

|  |  |  |  |
| --- | --- | --- | --- |
| GC\_TRAMITE  Nombre de secuencia: SEQ\_GC\_TRAMITE | | | |
| Nombre del atributo | **Tipo de dato** | **Null** | **Descripción** |
| N\_TRAMITE\_ID | NUMBER | No | Llave primaria de la tabla |
| N\_SERVICIOS\_ID | NUMBER | No | Referencia al servicio a que pertenece el trámite |
| N\_PESO | NUMBER | No | Cuantificación de la prioridad que debe tener el tramite luego de aplicarle todos los criterios |
| B\_NIT\_REQUERIDO | NUMBER(1,0) | No | Bandera que indica si es obligatorio proporionar el nit |
| B\_ESCALAMIENTO | NUMBER(1,0) | No | Bandera que indica si el tramite será escalado al supervisor |
| D\_TRAMITE | VARCHAR2(256 BYTE) | Yes | Descripción del registro |
| C\_USUARIO\_CREA | VARCHAR2(100 BYTE) | No | Código del usuarioque crea el registro |
| C\_USUARIO\_MODI | VARCHAR2(100 BYTE) | No | Código del usuarioque modifica el registro |
| FI\_VIGENCIA | DATE | No | Fecha en que el registro es creado |
| FF\_VIGENCIA | DATE | Yes | Fecha en que el registro deja de tener vigencia |
| F\_MODIFICA | DATE | Yes | Fecha en que el registro fue modificado |
| B\_ACTIVA | NUMBER(1,0) | No | Bandera que indica si el registro esta activo o no |
| S\_NOMBRE | VARCHAR2(256 BYTE) | No | Nombre del tramite |
| N\_ORDEN | NUMBER | Yes | Orden en que se presentaran los tramites en pantalla |

**Tabla 8.17 GC\_TRAMITE**

|  |  |  |  |
| --- | --- | --- | --- |
| GC\_CONF\_TRAMITE  Nombre de secuencia: SEQ\_GC\_CONF\_TRAMITE | | | |
| Nombre del atributo | **Tipo de dato** | **Null** | **Descripción** |
| N\_CONF\_TRA\_ID | NUMBER | No | Llave primaria de la tabla |
| C\_UNIDAD\_RECEP | VARCHAR2(5 BYTE) | No | Código del centro de servicio |
| N\_TRAMITE\_ID | NUMBER | No | Referencia al tramite a que se refiere la configuración |
| N\_ATENCION\_PROM | NUMBER | No | Tiempo promedio de atención |
| N\_TIEMPO\_ESPERA | NUMBER | Yes | Tiempo, en minutos, de espera máximo que debe tardarse en la cola |
| N\_TIEMPO\_HOLGURA | NUMBER | Yes | Tiempo, en minutos, que debe esperarse antes de incluirlo en la cola |
| C\_USUARIO\_CREA | VARCHAR2(100 BYTE) | No | Código del usuarioque crea el registro |
| C\_USUARIO\_MODI | VARCHAR2(100 BYTE) | No | Código del usuarioque modifica el registro |
| FI\_VIGENCIA | DATE | No | Fecha en que el registro es creado |
| FF\_VIGENCIA | DATE | Yes | Fecha en que el registro deja de tener vigencia |
| F\_MODIFICA | DATE | Yes | Fecha en que el registro fue modificado |
| N\_COMPORTAMIENTO | NUMBER(6,0) | Yes | Comportamiento: 0-al final, 1-al inicio, 2- en medio, 3 por peso |
| N\_PESO | NUMBER(6,0) | Yes | peso de la reasignacion |
| N\_PROM\_ESPERA | NUMBER | Yes | Promedio real (en segundos) de espera en cola |

**Tabla 8.17** GC\_CONF\_TRAMITE

|  |  |  |  |
| --- | --- | --- | --- |
| TB\_LISTAS\_VALOR  Nombre de secuencia: -- | | | |
| Nombre del atributo | **Tipo de dato** | **Null** | **Descripción** |
| C\_LISTA | VARCHAR2(50 BYTE) | No | Codigo de Parametro o Lista |
| C\_MODULO | VARCHAR2(50 BYTE) | No | Codigo de Modulo |
| D\_LISTA | VARCHAR2(255 BYTE) | Yes | Descripcion de la lista. |
| B\_STATUS | NUMBER(1,0) | Yes | Estado, Activo o Inactivo , 1 o 0 |
| C\_USUARIO | VARCHAR2(30 BYTE) | Yes | Usuario |
| F\_INGRESO | DATE | Yes | Fecha de ingreso |
| SYSTEM\_VALUE | NUMBER(1,0) | Yes | Es valor de sistema para 1/0, el usuario puede modificar el parametro cuando es 0 |

**Tabla 8.18** TB\_LISTAS\_VALOR

# Seguridad

Estos son los roles de seguridad usados en el Módulo de Declaracion Sugerida de Renta:

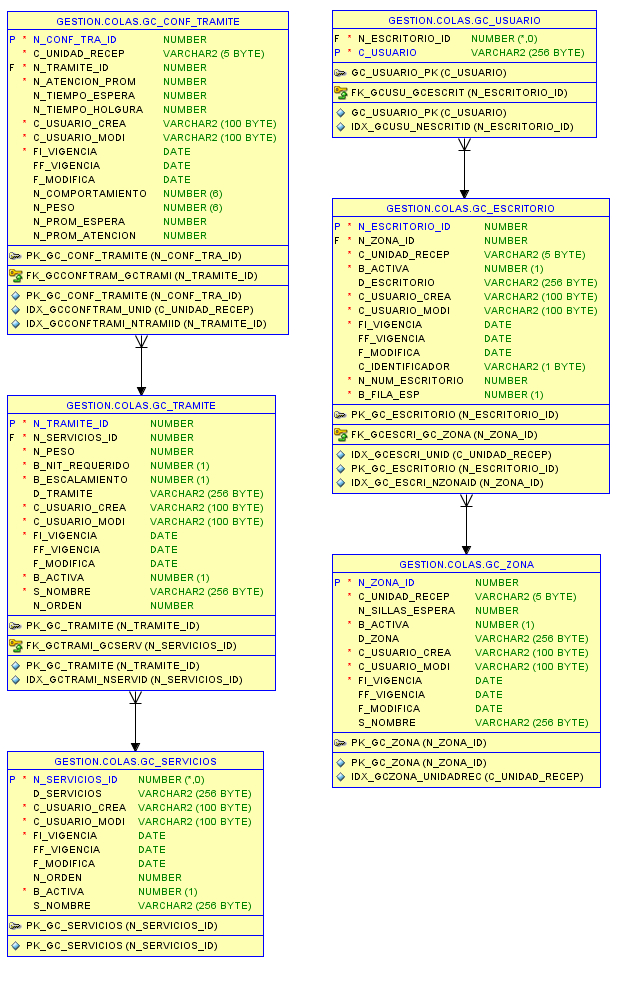
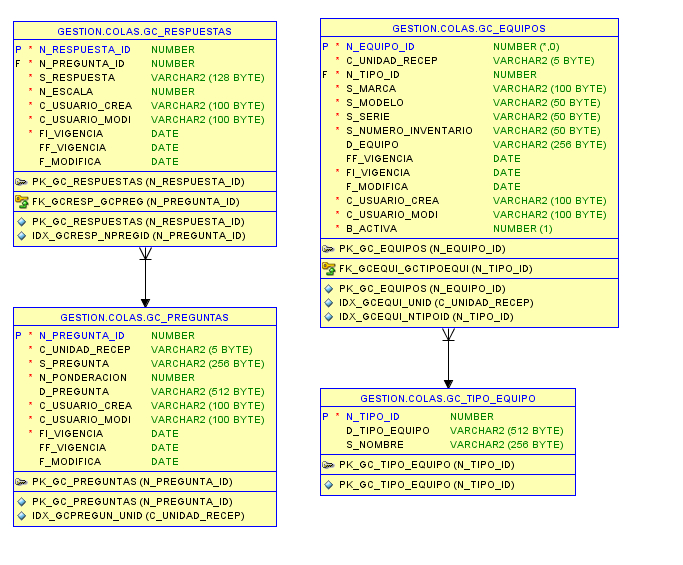
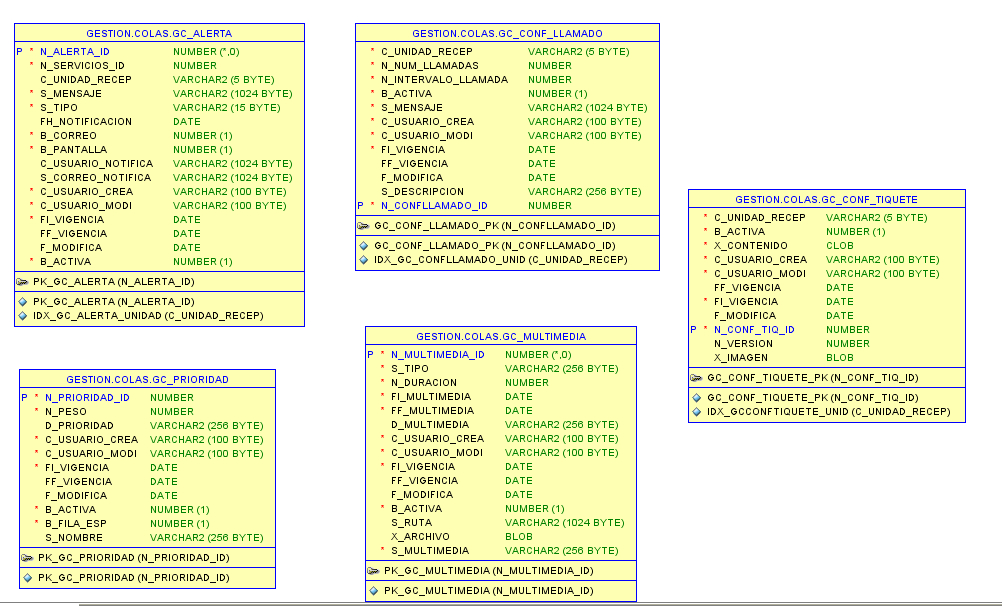
Seguridad y Roles para ambiente de desarrollo y test

| Nombre de la opción | Url | Roles |
| --- | --- | --- |
| Configuracion de Usuarios | /admin/ | ROLE\_GC\_R3 |
| Mantenimiento de Alertas | /alerta/ | ROLE\_GC\_R3 |
| Mantenimiento de Configuración de Llamado | /cllamado/ | ROLE\_GC\_R3 |
| Configuracion de Tiquete | /conftiquete/ | ROLE\_GC\_R3 |
| Mantenimiento de Configuración de Tramite | /ctramite/ | ROLE\_GC\_R3 |
| Mantenimiento de Equipos | /tequipo/ | ROLE\_GC\_R3 |
| Mantenimiento de Escritorios | /escritorio/ | ROLE\_GC\_R3 |
| Mantenimiento de Lista de Reproducción | /cllamado/ | ROLE\_GC\_R3 |
| Mantenimiento de Multimedia | /cllamado/ | ROLE\_GC\_R3 |
| Mantenimiento de Configuración - Evaluación | /adminpreguntas/ | ROLE\_GC\_R3 |
| Mantenimiento de Prioridades - Catálogo | /prioridad/ | ROLE\_GC\_R3 |
| Mantenimiento de Secciones - Catálogo | /servicio/ | ROLE\_GC\_R3 |
| Mantenimiento de Tipos de Equipos - Catálogo | /tequipo/ | ROLE\_GC\_R3 |
| Mantenimiento de Tramites - Catálogo | /ctramite/ | ROLE\_GC\_R3 |
| Mantenimiento de Zonas - Catálogo | /zona/ | ROLE\_GC\_R3 |
| Configuración de Reasignación | /creasignacion/ | ROLE\_GC\_R3 |

**Tabla 10.1 Seguridad y Roles**

# Anexos

Anexo 1



Anexo 2

url:

