MANUAL DE SISTEMAS

INTRODUCCIÓN

El Sistema Automatizado de Gestión Administrativa en Entorno Web (proceso de inducción de anteproyecto, proyecto, reporte de proyecto y urbanismo, Calculo de aranceles, estadística descriptiva y reporte de regresión lineal simple, para el Centro de Ingenieros del Estado Nueva Esparta (CIENE)", ubicado en La Asunción, Municipio Arismendi, Estado Nueva Esparta. Esta diseñado con el fin de gestionar, resguardar y lograr un control de la informacion manejada por la Oficina coordiandora del ejercicio profesional (OCEPRO). Este sistema ofrece a los usuarios realizar sus labores de manera muy sencilla , brindandole una interfaz gráfica amigable, intuitiva y de facil uso.

Como parte de la formación y entrenamiento de los usuarios que haran uso del sistema, se presenta el manual de sistema detallando los pasos a seguir para modificación ó ampliaciones que se deseen realizar al sistema .

Cabe destacar que solo el usuario master, tendrá acceso a dicho manual. Siendo este requisito obligatorio exigido por Universidad de Margarita (UNIMAR) para optar por el titulo de Ing. De Sistemas.

USUARIOS

Directorio: gestor-ciene/app/controllers/usuarios_controllers.rb

- **Registro de usuario:** El registro de usuario se realizó por medio de la gema 'devise', evitando que el código sea manipulable por el usuario.

- **Editar usuario:** En la siguiente imagen se muestra el código para la edición o modificación de usuario.

```
def update
  respond_to do |format|
    if @usuario.update(usuario_params)
      format.html { redirect_to usuarios_path, notice: 'El usuario fue actualizado exitosamente.' }

    else
      format.html { render :edit }
      format.json { render json: @usuario.errors, status: :unprocessable_entity }
    end
  end

  authorize @usuario
end
```

Figura 1. Fuente propia 2016.

- **Eliminar usuario:** En la presente imagen se muestra el código para la destrucción del usuario de manera leve ('soft destroy').

```
def destroy
  @usuario = Usuario.find(params[:id])
  @usuario.soft_delete

  respond_to do |format|
    if @usuario.save
      format.html { redirect_to usuarios_path, notice: 'La cuenta fue eliminada satisfactoriamente.' }
    else
      format.html { redirect_to usuarios_path, notice: 'No fue posible eliminar el usuario.' }
    end
  end
end
```

Figura 2. Fuente propia 2016.

- **Listado usuario:** El código a continuación, busca una lista de usuarios a través del modelo 'usuario' y aplica los filtros seleccionados en la vista.

```ruby
def index
  @usuarios_params = parametros_filtro_usuario
  @usuarios = policy_scope(Usuario).where('deleted_at is null').filter(@usuarios_params)
  authorize @usuarios
end
```

Figura 3. Fuente propia 2016.

```ruby
scope :usuario, -> (usuario) {where(usuario: usuario)}
scope :nombres, -> (nombres) {where('nombres like ?', '%'+nombres+'%')}
scope :cedula, -> (cedula) {where(cedula: cedula)}
scope :civ, -> (civ) {where(civ: civ)}
scope :email, -> (email) {where(email: email)}
scope :agremiados, -> {where(role: 3)}
```

Figura 4. Fuente propia 2016.

- **Métodos validar:** Las siguientes funciones se encargan de hacer las validaciones de duplicidad de datos al momento de registrar o modificar usuarios.

```ruby
def validar_usuario
  usuario = Usuario.select('usuario').where('usuario = ?',params[:usuario][:usuario])
  respond_to do |format|
    format.json { render json: !usuario.present?}
  end
end

def validar_mail
  usuario = Usuario.select('email').where('email = ?',params[:usuario][:email])
  respond_to do |format|
    format.json { render json: !usuario.present?}
  end
end

def validar_ci
  usuario = Usuario.select('usuario').where('cedula = ?',params[:usuario][:cedula])
  respond_to do |format|
    format.json { render json: !usuario.present?}
  end
end

def validar_civ
  usuario = Usuario.select('usuario').where('civ = ?',params[:usuario][:civ])
  respond_to do |format|
    format.json { render json: !usuario.present?}
  end
end

def validar_mail_v2
  usuario = Usuario.select('email').where('email = ?',params[:usuario][:email]).where('id != ?', @usuario.id)
  respond_to do |format|
    format.json { render json: !usuario.present?}
  end
end

def validar_ci_v2
  usuario = Usuario.select('usuario').where('cedula = ?',params[:usuario][:cedula]).where('id != ?', @usuario.id)
  respond_to do |format|
    format.json { render json: !usuario.present?}
  end
end

def validar_civ_v2
  usuario = Usuario.select('usuario').where('civ = ?',params[:usuario][:civ]).where('id != ?', @usuario.id)
  respond_to do |format|
    format.json { render json: !usuario.present?}
  end
end
```

Figura 5. Fuente propia 2016.

- **Definición de roles de usuarios:** En la siguiente imagen se mostrara la acción donde se define los rol de usuarios.

```ruby
class Usuario < ActiveRecord::Base
  # Include default devise modules. Others available are:
  # :confirmable, :lockable, :timeoutable and :omniauthable
  include Filterable
  devise :database_authenticatable, :registerable,
         :recoverable, :rememberable, :trackable, :validatable

  enum role: [:system, :admin, :operador, :agremiado]
  has_many :obras, class_name: :Obra, foreign_key: 'coordinador_proyecto_id'


  after_initialize :set_default_role, :if => :new_record?

  def set_default_role
    self.role ||= :agremiado
  end

  def roll_usuarios
    if tipo_usuario
      if tipo_usuario == "AGREMIADO"
        3
      elsif tipo_usuario == "OPERADOR"
        2
      elsif tipo_usuario == "SUPERVISOR"
        1
      end
    end
  end
```

Figura 6. Fuente propia 2016.

- **Rol por defecto:** A continuación, se muestra el método donde se define q el rol por defecto al no seleccionar el tipo de usuario será agremiado.

```ruby
def set_default_role
  self.role ||= :agremiado
end
```

Figura 7. Fuente propia 2016.

CoffeeScripts

Visado_obras.js.coffee

Directorio: gestor-ciene/app/assets/javascripts/ Visado_obras.js.coffee

- **Definición de variables:** En la presente imagen se muestra el listado de variables globales declaradas para el proceso de cálculo de visado y asignación de número de expediente.

```
constructor: (options)->
  @container                      = $(options.container)
  @selector_tipo_visado           = $('#tipo_visado')
  @selector_de_uso_container      = $('#selector_de_uso')
  @selector_de_uso                = @selector_de_uso_container.find('select')
  @area_bruta                     = $('#area_bruta')
  @costo_estimado_proyecto_input  = $('#costo_estimado_proyecto')
  @boton_guardar                  = $('#guardar')
  @boton_cancelar                 = $('#cancelar')
  @es_repeticion                  = $('#es_repeticion')
  @es_repeticion_checkbox         = $('#pollito')
  @cantidad_repeticiones_input    = $('#num_repeticiones')
  @arancel_visado_input           = $('#arancel_de_visado')
  @complejidades                  = $('#complejidad')
  @resultado                      = $('#costo_construccion')
  @resultado_estimado             = $('#costo_construccion_estimado')
  @total                          = $('#costo_completo')
  @tablita                        = $('#honorarios_minimos')
  @boton_oceprone                 = $('#boton_oceprone')
  @numero_oceprone                = $('#numero_ocepro')
  @comentario_obra                = $('#comentario_obra')
  @boton_comentario               = $('#boton_comentario')
  console.log 'BOTON Y NUMERO SALUD', @boton_oceprone, @numero_oceprone
  @indice_complejidad = null
  @descarga_pdf                   = $('#descarga_pdf')
  @arancel_visado_val = null

  @costos                     = @container.data('costos')
  @costos_url                 = @container.data('costos-url')
  @obra_url                   = @container.data('obra-url')
  @indice_costo_complejidad   = @container.data('indice-costos-complejidad')
  @exteriores_url             = @container.data('exteriores-url')
  @cuota_civ                  = @container.data('cuota-civ')
  @honorario_minimo_url       = @container.data('honorarios-minimos-costo-url')
  @obra_id                    = @container.data('obra-id')
  @honorarios_url             = @container.data('honorarios-url')
  @visados_url                = @container.data('visados-url')
```

Figura xx. Fuente propia 2016.

- **Evento 'change' en @selector_tipo_visado:** Este evento se genera al seleccionar un tipo de visado. Se encarga de limpiar el formulario y adaptarlo para solicitar solo los datos necesarios del visado seleccionado. Al seleccionar ANTEPROYECTO o PROYECTO se visualizara la opción 'repeticiones', si se selecciona 'URBANISMO' o 'EXTERIORES' se visualizara la tabla de los diferentes estudios con sus porcentajes asociados.

```
bind: ()->
  @selector_tipo_visado.on 'change', (evt) =>
    tipo_visado = @selector_tipo_visado.val()
    @selector_de_uso.val("")
    $('.tipo_visado_label').text(tipo_visado)
    @colocar_unidades_de_medida(tipo_visado)
    @area_bruta.val('')
    @area_bruta_val = null
    @costo_estimado_proyecto_input.val('')
    $('#costo_construccion_estimado').val('')
    if ((tipo_visado == 'PROYECTO') ||  (tipo_visado == 'ANTEPROYECTO'))
      @costo_construccion = null
      $('#costo_construccion').val('')
      @es_repeticion.show()
      @honorarios_minimos_distancia = null
      $('.proyantep').show()
      $('.exturb').hide()
      @buscarHonorariosMinimosPorTipoVisado(tipo_visado)
    else
      @buscarCostoConstruccion(tipo_visado)
      @es_repeticion.hide()
      $('.exturb').show()
      $('.proyantep').hide()
```

Figura xx. Fuente propia 2016.

```
buscarCostoConstruccion: (descripcion) =>
  $.ajax @costos_url+"?descripcion=#{descripcion}",
    type: 'GET'
    dataType: 'json'
    error: (jqXHR, textStatus, errorThrown) ->
      console.log "Falle "
    success: (data, textStatus, jqXHR) =>
      @costo_construccion = data.costos
      $('#costo_construccion').val(BSF+@numberToDecimal(@costo_construccion))
```

Figura xx. Fuente propia 2016.

- **Evento 'blur' en @area_bruta.val():** Al ingresar un valor en el campo de área bruta se colocara el delimitador y separador a los campos y ejecutara la función calcular_costo_construccion_estimado.

```
@area_bruta.on 'blur', (evt) =>
  console.log @area_bruta.val(), @area_bruta.val().replace(",",".") , parseFloat(@area_bruta.val()), parseFloat(@area_bruta.val().replace(",","."))
  @area_bruta_val = parseFloat(@area_bruta.val().replace(",","."))
  console.log 'area bruta',@area_bruta_val
  @area_bruta.val(@numberToDecimal(@area_bruta_val))
  @calcular_costo_construccion_estimado()
```

Figura xx. Fuente propia 2016.

- **Evento 'change' en @selector_de_uso:** Al seleccionar el uso de la obra, obtendremos el índice de complejidad e índice de costo por construcción, datos que permiten ejecutar el cálculo de costos por construcción estimado.

```
@selector_de_uso.on 'change', (evt) =>
  tipo_uso = @selector_de_uso.val()
  tipo_visado = @selector_tipo_visado.val()
  for indice_complejidad in @indice_costo_complejidad
    if tipo_uso == indice_complejidad.vivienda_uso
      @indice_complejidad = indice_complejidad
      $('#complejidad').val(@numberToDecimal(indice_complejidad.indice_complejidad))
      if ((tipo_visado == 'PROYECTO') ||  (tipo_visado == 'ANTEPROYECTO'))
        $('#costo_construccion').val(BSF+@numberToDecimal(indice_complejidad.indice_costo))
        @costo_construccion = indice_complejidad.indice_costo
        @calcular_costo_construccion_estimado()
```

Figura xx. Fuente propia 2016.

- **Evento 'click' en @es _repetición_checkbox:** Al seleccionar el checkbox de repetición, automáticamente mostrara el input donde el usuario colocara la cantidad de repeticiones que posee la obra.

```
@es_repeticion_checkbox.on 'click', (evt) =>
  if @es_repeticion_checkbox.prop("checked")
    $('.cantidad_repeticion').show()
  else
    $('.cantidad_repeticion').hide()
```

Figura xx. Fuente propia 2016.

- **Evento 'blur' @cantidad_repeticiones_input:** Al ingresar cantidad repeticiones se procede a realizar el cálculo de costos del visado tomando en cuenta la fórmula proporcionada por el C.I.E.N.E , donde se aplican diferentes porcentajes en base al monto de costo completo.

```
@cantidad_repeticiones_input.on 'blur', (evt) =>
  if @cantidad_repeticiones_input.val()
    cantidad = parseInt(@cantidad_repeticiones_input.val())
    porcentaje = @honorarioPorRepeticion(cantidad)
    costo_completo_repeticion = @costo_completo * (@indice_complejidad.indice_complejidad + porcentaje)
    if costo_completo_repeticion > 10000000
      costo_de_visado = (costo_completo_repeticion * 0.01) + (6 * @cuota_civ)
    else
      costo_de_visado = (costo_completo_repeticion * 0.0120) + (2 * @cuota_civ)


    $('#costo_completo_repeticion').val(BSF+@numberToDecimal(costo_completo_repeticion))
    @costo_completo_repeticion = costo_completo_repeticion
    @arancel_visado_input.val(BSF+@numberToDecimal(costo_de_visado))
    @arancel_visado_val = costo_de_visado
  else
    if @costo_completo > 10000000
      costo_de_visado = (@costo_completo * 0.01) + (6 * @cuota_civ)
    else
      costo_de_visado = (@costo_completo * 0.0120) + (2 * @cuota_civ)

    @total.val(BSF+@numberToDecimal(@costo_completo))
    @arancel_visado_input.val(BSF+@numberToDecimal(costo_de_visado))
    @arancel_visado_val = costo_de_visado
    @costo_completo_repeticion = 0
```

Figura xx. Fuente propia 2016.

- **Evento 'change' @selector_honorario:** Al seleccionar un honorario, se solicita al sistema el honorario minimo dependiendo del concepto seleccionado, tipo_visado y costo de construcción, dato q se usa para calcular el costo estimado del proyecto.

```
$('#selector_honorario').on 'change', (evt) =>
  concepto = $('#selector_honorario').val()
  tipo_visado = @selector_tipo_visado.val()
  @buscarHonorarioMinimo(tipo_visado, concepto)
```

Figura xx. Fuente propia 2016.

```
buscarHonorariosMinimosPorTipoVisado: (tipo_visado) =>
  console.log "TIPO VISADO ", tipo_visado, @honorarios_url, "#{@honorarios_url}?tipo_visado=#{tipo_visado}"
  $.ajax "#{@honorarios_url}?tipo_visado=#{tipo_visado}",
    type: 'GET'
    dataType: 'json'
    error: (jqXHR, textStatus, errorThrown) ->
      console.log "Falle "
    success: (data, textStatus, jqXHR) =>
      @crearSelectHonorarios(data)
```

Figura xx. Fuente propia 2016.

```
buscarHonorarioMinimo: (tipo_visado, concepto) =>
  $.ajax @honorario_minimo_url+"?tipo_visado=#{tipo_visado}&concepto=#{concepto}&costo=#{@costo_construccion_estimado}",
    type: 'GET'
    dataType: 'json'
    error: (jqXHR, textStatus, errorThrown) ->
      console.log "Falle "
    success: (data, textStatus, jqXHR) =>
      @calcularCostoEstimadoProyecto(data)
```

Figura xx. Fuente propia 2016.

- **Colocar_unidades_de_medida:** Función encargada de realizar el cambio de unidad a la etiqueta de los cuadro de texto costo construcción, costo de construcción estimado y área bruta.

```
colocar_unidades_de_medida: (tipo_visado) =>
  if tipo_visado == 'URBANISMO'
    $('.unidad_distancia').text('Ha')
    $('.unidad_monetaria_distancia').text('Bs/Ha')
  else
    $('.unidad_distancia').text('mts2')
    $('.unidad_monetaria_distancia').text('Bs/mts2')
```

Figura xx. Fuente propia 2016.

- **Evento ‘click’ @boton_oceprone:** Al presionar el ‘boton_oceprone’ verifica si el campo se encuentra vacío o si el número de expediente se encuentra registrado.

```
  @boton_cancelar.on 'click', (evt) =>
    @limpiarFormulario()

  @boton_oceprone.on 'click', (evt) =>
    if @numero_oceprone.val() == ""
      sweetAlert("Error!", "El campo se encuentra vacio.", "error")
    else
      console.log 'presionado el boton'
      n_oceprone = @numero_oceprone.val()
      console.log n_oceprone
      obra =
        n_oceprone: n_oceprone
      console.log "DATA", obra
      $.ajax(
        url: @obra_url,
        method: 'PUT',
        data:{obra: {n_oceprone: n_oceprone}}
      ).done (resp) =>
        console.log 'RESPUESTA', resp
        swal
          type: 'success'
          title: 'El numero oceprone ha sido asignada exitosamente.'
          () ->
            window.location.reload()
      .fail () =>
        swal
          type: 'danger'
          tittle: 'Ha ocurrido un problema al asignar el numero de expediente.'
          () ->
            window.location.reload()
```

Figura xx. Fuente propia 2016.

```
class Validaciones_obras
    @bind: ()->
        if $(".registro-oceprone-helper").length > 0
            by_obras_url = $(".registro-oceprone-helper").data('by-obras-url')


            $(".ocepro-form").validate({
                rules:{
                    "obra[n_oceprone]":{
                        remote: by_obras_url
                    },

                },
                messages:{
                    "obra[n_oceprone]": {
                        remote: "Numero de expediente existente",
                    },

                },

            })

$ ()->
    Validaciones_obras.bind()
```

Figura xx. Fuente propia 2016.

- **GuardarVisado():** La presente función está encargada de guardar los cálculos realizados.

```
guardarVisado: () =>
  visado =
    visado:
      obra_id: @obra_id
      area_bruta: @area_bruta_val
      tipo: @selector_tipo_visado.val()
      complejidad: @indice_complejidad.indice_complejidad
      costo_construccion_estimado: @costo_construccion_estimado
      costo_estimado_proyecto: @costo_estimado_proyecto
      costo_completo: @costo_completo
      es_repeticion: @es_repeticion_checkbox.prop("checked")
      cantidad_repeticiones: @cantidad_repeticiones_input.val()
      costo_completo_repeticion: @costo_completo_repeticion
      arancel: @arancel_visado_val
      costo_construccion: @costo_construccion
      porcentaje_tabla_datos: @porcentaje_tabla_datos

  $.ajax @visados_url,
    type: 'POST'
    data: visado
    dataType: 'json'
    error: (jqXHR, textStatus, errorThrown) ->
      console.log "Falle "
    success: (data, textStatus, jqXHR) =>
      swal
        type: 'success'
        title: 'Guardado exitoso'
        () ->
          window.location.reload()
```

Figura xx. Fuente propia 2016.

- **LimpiarFormulario():** Se encarga de limpiar todos los input al presionar el botón 'cancelar' o posterior al guardado de datos.

```
limpiarFormulario: () =>
 @selector_tipo_visado.val("")
 @selector_de_uso_container.val("")
 @selector_de_uso.val("")
 @area_bruta.val("")
 @area_bruta_val = null
 @costo_estimado_proyecto_input.val("")
 @costo_estimado_proyecto = 0
 $('.cantidad_repeticion').hide()
 @es_repeticion_checkbox.prop('checked', false)
 @cantidad_repeticiones_input.val("")
 @arancel_visado_input.val("")
 @complejidades.val("")
 @resultado.val("")
 @resultado_estimado.val("")
 @total.val("")
 $('.proyantep').hide()
 $('.exturb').hide()
```

Figura xx. Fuente propia 2016.

- **CalcularCostoEstimadoProyectoExtUrb:** Se encarga de realizar el cálculo de arancel correspondiente a áreas EXTERIORES y URBANISMO.

```
calcularCostoEstimadoProyectoExtUrb: () =>
  console.log "Ahora calculare el costo estimado del proyecto"
  selected = $(":checkbox[name='honorario_minimos']:checked")
  porcentaje = 0
  for checkbox_selected in selected
    porcentaje += parseFloat($(checkbox_selected).attr('data-porcentaje'))

  @costo_estimado_proyecto = @costo_construccion_estimado * (porcentaje / 100)
  @porcentaje_tabla_datos = porcentaje
  @costo_estimado_proyecto_input.val(BSF+@numberToDecimal(@costo_estimado_proyecto))
  costo_completo = @costo_estimado_proyecto  * @indice_complejidad.indice_complejidad

  if costo_completo > 10000000
    costo_de_visado = (costo_completo * 0.01) + (6 * @cuota_civ)
  else
    costo_de_visado = (costo_completo * 0.0120) + (2 * @cuota_civ)

  @total.val(BSF+@numberToDecimal(costo_completo))
  @costo_completo = costo_completo
  @arancel_visado_input.val(BSF+@numberToDecimal(costo_de_visado))
  @arancel_visado_val = costo_de_visado
```

Figura xx. Fuente propia 2016.

- **Evento 'click' @boton_oceprone:** Realiza la asignación del número de expediente.

```
@boton_oceprone.on 'click', (evt) =>
  if @numero_oceprone.val() == ""
    sweetAlert("Error!", "El campo se encuentra vacio.", "error")
  else
    console.log 'presionado el boton'
    n_oceprone = @numero_oceprone.val()
    console.log n_oceprone
    obra =
      n_oceprone: n_oceprone
    console.log "DATA", obra
    $.ajax(
      url: @obra_url,
      method: 'PUT',
      data:{obra: {n_oceprone: n_oceprone}}
    ).done (resp) =>
      console.log 'RESPUESTA', resp
      swal
        type: 'success'
        title: 'El numero oceprone ha sido asignada exitosamente.'
        () ->
          window.location.reload()
    .fail () =>
      swal
        type: 'danger'
        tittle: 'Ha ocurrido un problema al asignar el numero de expediente.'
        () ->
          window.location.reload()
```

Figura xx. Fuente propia 2016.

- **Evento ‘click’ comentario:** Realiza el guardado de comentario referente a la obra revisada.

```
@boton_comentario.on 'click', (evt) =>
  if @comentario_obra.val() == ""
    sweetAlert("Error!", "El campo se encuentra vacio.", "error")
  else
    comentario = @comentario_obra.val()
    obra =
      comentario: comentario
    $.ajax(
      url: @obra_url,
      method: 'PUT',
      data:{obra: {comentario: comentario}}
    ).done (respuesta) =>

      swal
        type: 'success'
        title: 'Su comentario ha sido guardado exitosamente.'
        () ->
          window.location.reload()
    .fail () =>
      swal
        type: 'danger'
        tittle: 'Ha ocurrido un problema al guardar el comentario, intente nuevamente.'
        () ->
          window.location.reload()
```

Figura xx. Fuente propia 2016.

Validaciones.coffee

Directorio: gestor-ciene/app/assets/javascripts/validaciones.coffee

- **Validaciones Registro/Modificación de usuario:** Verifica la duplicidad de los datos.

```
if $(".registro-usuario-helper").length > 0
    by_usuario_url = $(".registro-usuario-helper").data('by-usuario-url')
    by_mail_url = $(".registro-usuario-helper").data('by-mail-url')
    by_ci_url = $(".registro-usuario-helper").data('by-ci-url')
    by_civ_url = $(".registro-usuario-helper").data('by-civ-url')

    $(".test").validate({
        rules:{
            "usuario[usuario]":{
                remote: by_usuario_url
            },
            "usuario[email]":{
                remote: by_mail_url
            },
            "usuario[cedula]":{
                checkci: true,
                remote: by_ci_url
            },
            "usuario[civ]":{
                remote: by_civ_url
            }
        },
        messages:{
            "usuario[usuario]":
                remote: "Usuario ya existe en el sistema",
            "usuario[email]":
                remote: "Correo existe en el sistema",
            "usuario[cedula]":
                checkci: "Formato de cédula no  válido"
                remote:"Número de cédula existe en el sistema",
            "usuario[civ]":
                remote:"Número de C.I.V existe en el sistema",
        },

    })
    $(".tipo-ci").on  'change', () ->
        $("#cedula").trigger('blur')
```

Figura xx. Fuente propia 2016.

```
if $(".modificar-usuario-helper").length > 0
    console.log 'Helper???', $(".edit_usuario")
    by_mail_url = $(".modificar-usuario-helper").data('by-mail-url')
    by_ci_url = $(".modificar-usuario-helper").data('by-ci-url')
    by_civ_url = $(".modificar-usuario-helper").data('by-civ-url')
    console.log  by_mail_url,

    $(".edit_usuario").validate({
        rules:{
            "usuario[email]":{
                remote: by_mail_url
            },
            "usuario[cedula]":{
                checkci: true,
                remote: by_ci_url
            },
            "usuario[civ]":{
                remote: by_civ_url
            }
        },
        messages:{
            "usuario[email]":
                remote: "Correo existe en el sistema",
            "usuario[cedula]":
                checkci: "Formato de cédula no  válido"
                remote:"Número de cédula existe en el sistema",
            "usuario[civ]":
                remote:"Número de C.I.V existe en el sistema",
        },

    })
    $(".tipo-ci").on  'change', () ->
        $("#cedula").trigger('blur')
```

Figura xx. Fuente propia 2016.

Validaciones_usuario.coffee

Directorio: gestor-ciene/app/assets/javascripts/validaciones_usuario.coffee

- **Validacion_usuario:** Al seleccionar “G.-” o “J.-” muestra el selector de rif.

```
class Validacion_usuario
  @bind: ()->
    $('.tipoci').on 'change', () ->
      if $('.tipoci').val().trim() == "G.-" || $('.tipoci').val().trim() == "J.-"
        $('#ru').show()
      else
        $('#ru').hide()
        $('.ro').val("")
$ ()->
  Validacion_usuario.bind()
```

Figura xx. Fuente propia 2016.

Validacion_civ.js.coffee

Directorio: gestor-ciene/app/assets/javascripts/validacion_civ.js.coffee

- **Validacion_civ.js:** Al registrar usuario y seleccionar el rol ‘agremiado’ mostrara el input_civ como campo obligatorio.

```
class Validacion_civ
  @bind: ()->
    $('.role').on 'change', () ->
      if $('.role').val().trim() == "agremiado"
        $('#identidad').show()
      else
        $('#identidad').hide()
        $('.identificador').val("")
$ ()->
  Validacion_civ.bind()
```

Figura xx. Fuente propia 2016.

Mensaje.coffee

Directorio: gestor-ciene/app/assets/javascripts/mensaje.coffee

- **Evento ‘click’ $(“.multifirmoso”):** Muestra un mensaje emergente al descargar la planilla multifirma.

```
class MensajeMultifirma
  @bind: ()->

    $("#multifirmoso").on 'click', (evt) ->

      swal("Atención!", "La planilla debe ser anexada en el archivo memoria descritiva de no hacerlo su proyecto NO SERÁ
      APROBADO.", "info")

$ ()->
  MensajeMultifirma.bind()
```

Figura xx. Fuente propia 2016

ci_validations.coffee

Directorio: gestor-ciene/app/assets/javascripts/ci_validations.coffee

- **jQuery.validator.addMethod ("checkci"):** Valida el número de cedula en el registro, según sea su nacionalidad (Venezolano o Extranjero).

```
jQuery.validator.addMethod("checkci", ((value, elem) ->
    console.log value, elem
    tipo =  $(".tipo-ci").val().trim()
    value = parseInt(value)
    console.log tipo, value, typeof value,tipo == "V.-" ,  tipo == "E.-"
    if tipo == "V.-" && value >= 80000000
        console.log 'F', tipo
        return false
    else if tipo == "E.-" && value <= 80000000
        console.log 'F'
        return false
    else
        console.log 'T'
        return true

    ), "Formato de cédula no válido")
```

Figura xx. Fuente propia 2016.