Control de cambios – Reclutamiento

Desarrollados por José Antonio Navarro Cholico

REQ - Ligar la consulta de proyectos de la vacante al módulo de Prospectos

Habilitar que al dar Click en cualquiera de los datos que se presentan en el Grid se pueda abrir el módulo de Prospectos mandándole como parámetro el Id del Prospecto.

***Código generado:***

Vista: Vacante\_Index.cshrml

Línea: 43

Descripción: Se agrega un modal del prospecto a detalle con carga de vista parcial haciendo referencia a las vistas del prospecto llamada prospecto\_detalle.cshtml.

Línea: 1568

Descripción: Se agrega funcionalidad a la tabla del acordeon prospectos para que cuando se le de click en cualquier registro de la tabla se abra el modal.

Línea: 1619

Descripción: Se agrega función para obtener datos de la tabla de prospectos y enviarlos a los campos del detalle del prospecto.

Nota: Esta misma funcionalidad se agregó para todas las tablas dónde se muestran registros de prospectos.

REQ - En el Grid que presenta los Prospectos a seleccionar para asignarlos a la vacante, presentar también los prospectos ya seleccionados anteriormente.

***Código generado:***

Vista: Vacante\_Index.cshrml

Línea: 1796

Descripción: Se agrego la configuración de la columna del prospecto seleccionado.

Vista: Acordeon\_Prospectos.cshrml

Línea: 34

Descripción: Se agrego la columna del prospecto seleccionado.

Proyecto: Entidades

Descripción: Se agrego la entidad de seleccionado, así como en el campo dentro del procedimiento de consulta de prospectos y así mismo se mapeo con el campo agregado en la función de lectura de datos.

Vista: Vacante\_Index.cshrml

Se agrego el parámetro en el envío del método Ajax y el httpPost del backend de c#.

Se desarrollo la función utilizando el store procedure InsVacante\_Prospecto para ejecutar otros store procedures con validaciones, Insert o updates según sea el caso para evitar duplicidad de registros y mantener integridad de los datos.

REQ - Desvincular, el prospecto de la vacante.

***Código generado:***

Incluir un botón por cada prospecto que permita desvincular al prospecto de la vacante, de desvinculación será lógica y se creara un campo de estado en la tabla vacante\_prospecto, el proceso de desvinculación, se dará después de la confirmación del usuario. Y estos prospectos ya no se mostrarán en el Grid, al re cargarlo.

Base de datos: Se modifico la base de datos agregando un campo para mostrar si estaba asignado a una vacante, se añadió a los parámetros de store procedure tanto en consulta como en guardado.

Proyecto: Entidades

Descripción: Se añadieron los campos nuevos y se mapearon en las funciones de consulta.

Vista: Vacante\_Index.cshrml

Se añadió la columna desvincular al grid y se le agrego funcionalidad para eliminar el vinculo con la vacante, tanto en base de datos como en funciones de conexión con la base de datos y métodos httppost.

REQ - Darle la funcionalidad de poder marcar como finalista(s) a los prospectos y así mismo poder realizar invitaciones a los candidatos para participar en la vacante.

Vista: Vacante\_Index.cshrml

* En la tabla de prospectos se añadió el botón de finalistas, el cuál muestra un grid nuevo creado para seleccionar los prospectos que se marcarán como finalistas y así mismo mostrar la columna en la tabla prospectos, por lo que en la tabla principal de prospectos se añadió la columna de finalista.
* En la tabla de prospectos se añadió el botón de invitar, el cuál muestra un grid nuevo creado para seleccionar los prospectos que se desea invitar a la vacante. Se añadió el store procedure para realizar las invitaciones para los prospectos seleccionados. Así mismo se agrego una columna en la tabla de prospectos principales indicando las invitaciones realizadas para el prospecto en la vacante seleccionada.

**Notas técnicas**

Dentro de los store procedures se utilizó una función generalizada que puede servir para otros procedimientos llamada y que se muestra más abajo.

Nombre de la función: “InsVacante\_Prospecto”.

Problemática: Dentro de esta función contenía un prepare y execute que recibía un isert con varios registros, pero estos no eran validados. Para esto habría que llamar un store procedure aparte para validar los datos que se deben isnertar, dado que el prepare y execute sólo ejecuta una instrucción, se opto por desarrollar una nueva función que permita ejecutar un conjunto de instrucciones dentro de una misma cada.

Descripcón: Función que recorre una cadena, las separa por (;) punto y coma y ejecuta cada una de las instrucciones por separado.

DELIMITER $$

CREATE DEFINER=`root`@`localhost` PROCEDURE `InsVacante\_Prospecto`(IN `pIdVacante` INT, IN `pDetalles` MEDIUMTEXT)

BEGIN

DECLARE \_list MEDIUMTEXT DEFAULT NULL;

DECLARE \_next TEXT DEFAULT NULL;

DECLARE \_nextlen INT DEFAULT NULL;

DECLARE \_value TEXT DEFAULT NULL;

START TRANSACTION ;

SET \_list = pDetalles;

iterator:

LOOP

-- Sale del clico Loop si la cadena se queda vacia o es nula.

-- Esta validación es necesaria para salir del ciclo loop.

IF LENGTH(TRIM(\_list)) = 0 OR \_list IS NULL THEN

LEAVE iterator;

END IF;

-- Obtiene la siguiente instrucción de la cadena.

SET \_next = SUBSTRING\_INDEX(\_list,';',1);

-- Guarda el tamaño del texto en la cadena para despues eliminarlo.

SET \_nextlen = LENGTH(\_next);

-- Asignar el texto al comando a ejecutar.

SET @SQL = TRIM(\_next);

PREPARE stmt1 FROM @SQL;

EXECUTE stmt1;

DEALLOCATE PREPARE stmt1;

-- Reescribe en la cadena original, reemplazando por nada para quitar el texto

-- ya ejecutado.

-- Los argumentos son:

-- Cadena original, posición inicial, cuántos caracteres a remover y

-- que caracter usarás para reemplazar, en este caso

-- un caracter en blanco, que remueve el tamaño del texto + 1;

SET \_list = INSERT(\_list,1,\_nextlen + 1,'');

END LOOP iterator;

SELECT NULL AS ErrorMessage;

COMMIT;

END