**UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA**

FACULTAD DE INGENIERÍA ARQUITECTURA Y DISEÑO



**PERMISOS UABC**

DESCRIPCIÓN DEL DESARROLLO DEL PROYECTO: SISTEMA DE SOLICITUD DE PERMISOS DE SALIDA UABC (SIPES).

**EQUIPO:**

Elías David López Mora (344324)

Gabriel Ezequías Melchor Sánchez (342122).

Omar Jovany Hernández Sánchez (344187).

FECHA

8 DE JUNIO DE 2017

**IMPLEMENTACIÓN, MÓDULOS Y DESCRIPCIÓN DE DESARROLLO**

Lo primero que se hizo fue crear las cuatro capas necesarias para el correcto funcionamiento del sistema: la capa de datos, entidades, negocios y de presentación:

* En la capa de **datos** se estableció la conexión a la base de datos a trabajar con la única clase ConnectionDB, en ella se establece la conexión SQL con una codena de conexión definida.
* En la capa de **negocios** se crearon todas las clases necesarias para la comunicación entre la base de datos y la presentación al usuario, para ello se tiene el apoyo de la clase ConnectionDB que se encuentra en la capa de datos.
* En capa de entidades se encuentra todas las clases necesarias para crear cada tipo de objeto dependiendo de la acción que se deba hacer. Por ejemplo, cuando el usuario solicita un nuevo permiso, se utiliza la clase E\_Solicitudes, cuando el usuario inicia sesión, toda su información se administra con la clase E\_Usuario y si una solicitud ha sido rechazada, se tiene la clase E\_SolicitudRechazada, que contiene toda la información necesaria para registrar y mostrarle motivos de rechazo al usuario final.
* Finalmente está la capa de presentación, en ella se encuentra toda la parte visual, lo que el usuario verá. Esta es la capa más superficial del sistema en general.

A continuación se describirá detalladamente cada una de las implementaciones del proyecto, dividiéndolo en dos partes: el lado de la programación de C# en Visual Studio y los stored procedures del lado del servidor SQL.

**DESCRIPCIÓN DE LAS CLASES DE CADA CAPA**

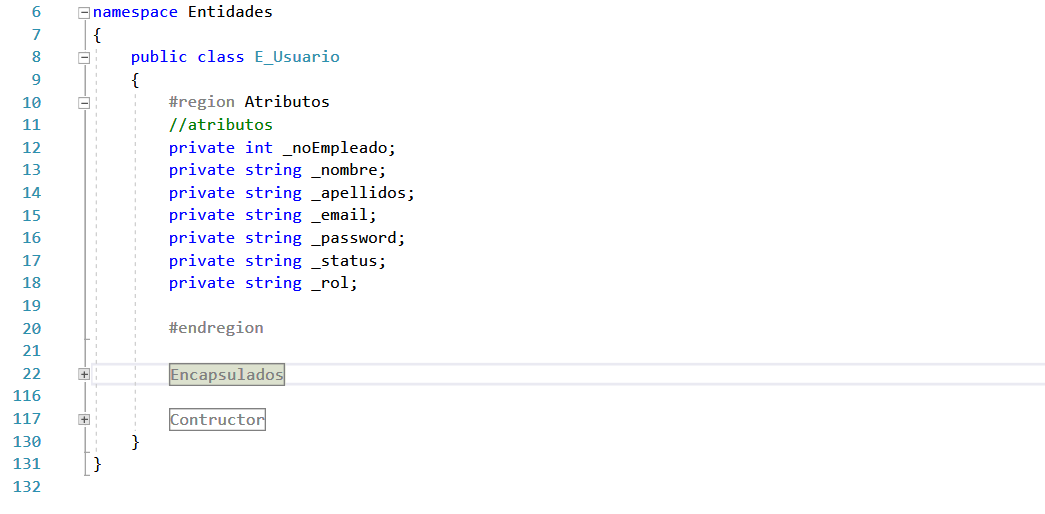
**C# en Visual Studio**

**CAPA DE DATOS: ConnectionDB.cs**

Esta se encarga de establecer una conexión a la base de datos SQL a trabajar, indicado por la connection string tomada del archivo Web.config. Esta clase no contiene ningún método interno, sólo su propio constructor.

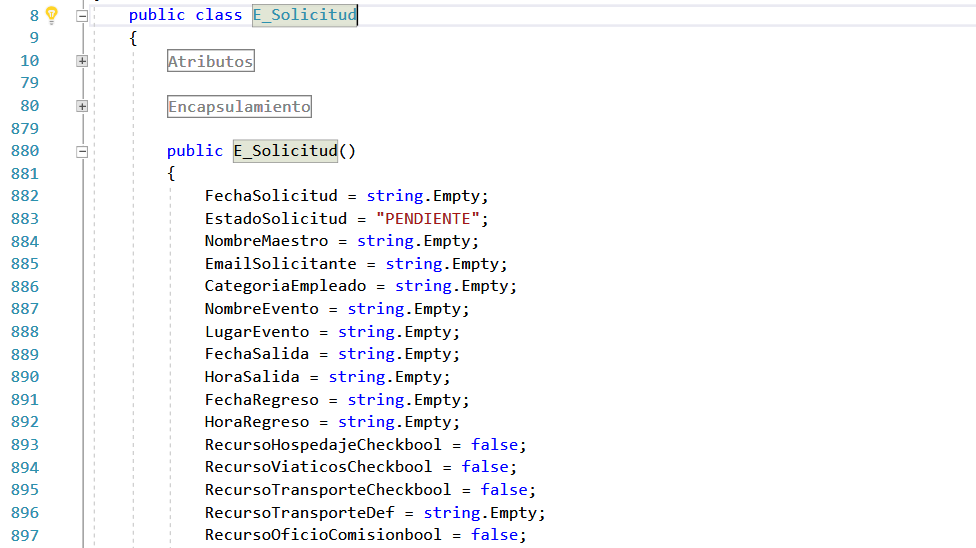


**CAPA DE ENTIDADES: E\_Usuario.cs**

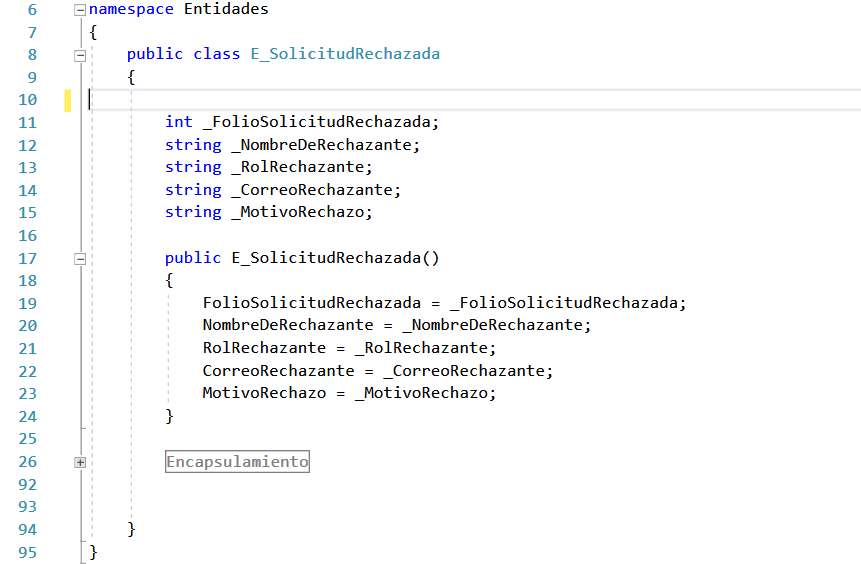
Esta clase tiene la información completa del usuario requerida para identificarlo. En su estructura contiene una serie de atributos y métodos Getter y Setter de cada uno, así como su constructor. Al igual que la clase ConnectionDB esta entidad no tiene métodos internos, sólo es usada para la administración de información de usuario.

**CAPA DE ENTIDADES: E\_Solicitud.cs**

Utilizada para la administración de la información de una solicitud, como la información requerida, información de solicitantes y responsables. Esta no contiene métodos internos mas que su constructor.



**CAPA DE ENTIDADES: E\_SolicitudRechazada.cs**

A diferencia de E\_Solicitud.cs, esta se utiliza para administrar la información de una solicitud rechazada.

**CAPA DE NEGOCIOS: SolicitudesDALC.cs**

Esta clase contiene todos los métodos necesarios para la comunicación entre la capa de presentación y el servidor. Servirá de intermediario para envío de la información de la solicitud a la base de datos, así como en el aviso de resultados al usuario final. A diferencia de las clases expuestas anteriormente, esta si contiene métodos con una funcionalidad distinta. Se apoya de la clase *E\_Solicitud* de la capa de Entidades, ya que necesita reconocer los parámetros de una solicitud para administrarlos, así como de la clase *ConnectionDB* de la capa de Datospara establecer la conexión a la base de datos.

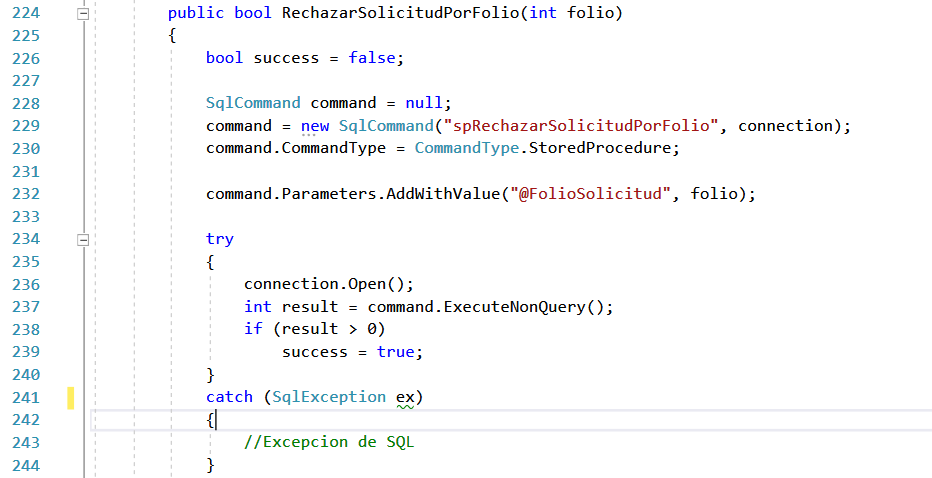
A diferencia de las otras clases esta si contiene métodos internos, expuestas a continuación:

**SolicitudesDALC: AgregarSolicitud.**

Este método se utiliza para parametrizar y enviar la información de una solicitud a la base de dato. En el método se especifica el tipo de comando a utilizar para la conexión (en este caso siempre es stored procedure).

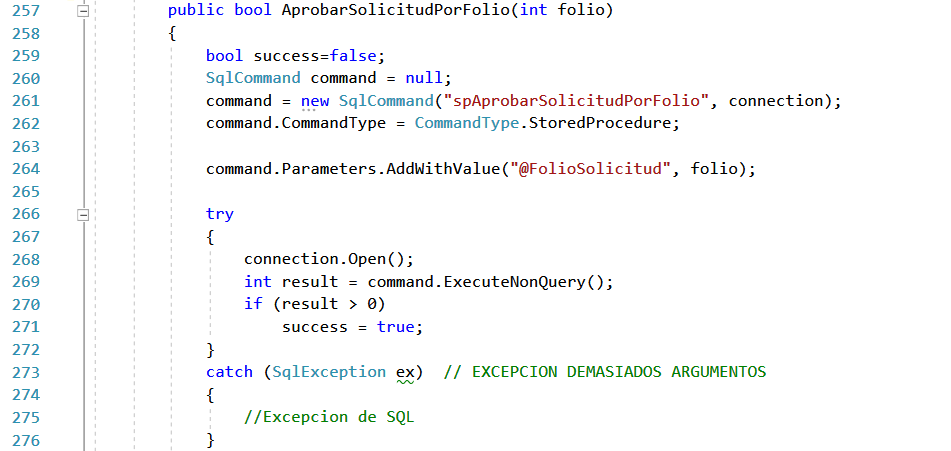


**SolicitudesDALC: RechazarSolicitud.**

Esta función se utiliza para establecer el estado de una solicitud en rechazado. Recibe como único parámetro el folio de la solicitud a rechazar, requiere del atributo connection de la clase ConnectionDB para funcionar.

**SolicitudesDALC: AprobarSolicitud.**

Este método se utiliza para establecer el estado de una solicitud como aprobada, recibe como para funcionar.



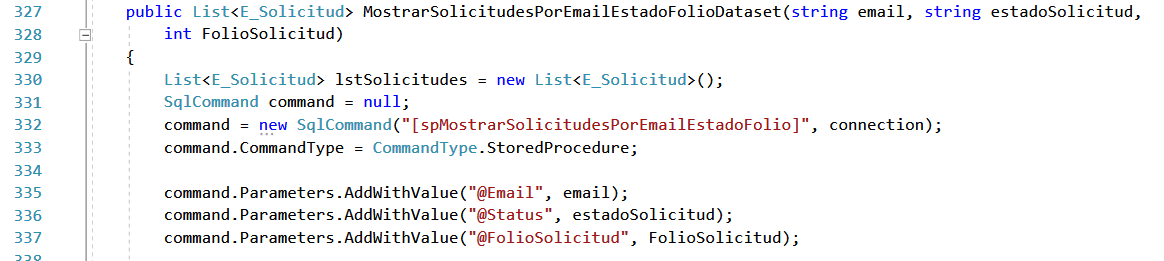
**SolicitudesDALC: MostrarSolicitudesPorEmailEstadoDataset**

Este método se utiliza para realizar la transferencia entre la base de datos y el sistema en C#, convierte un objeto de tipo DataSet recibo por la base de datos, en una lista de entidades de tipo E\_Solicitud, de tal forma que sea manipulable en C#.



Adicionalmente, para que el método MostrarSolicitud cumpla su cometido requiere paralelamente del método popule, que convierte cada registro de tipo DataRow de las tablas de cada DataSet en atributos de la clase *E\_Solicitud*. Este método se ejecuta en cada registro o renglón de la tabla en un DataSet. El método Mostrar y el popule trabajan juntos para formar la lista de entidades de tipo *E\_Solicitud* para después retornarla.

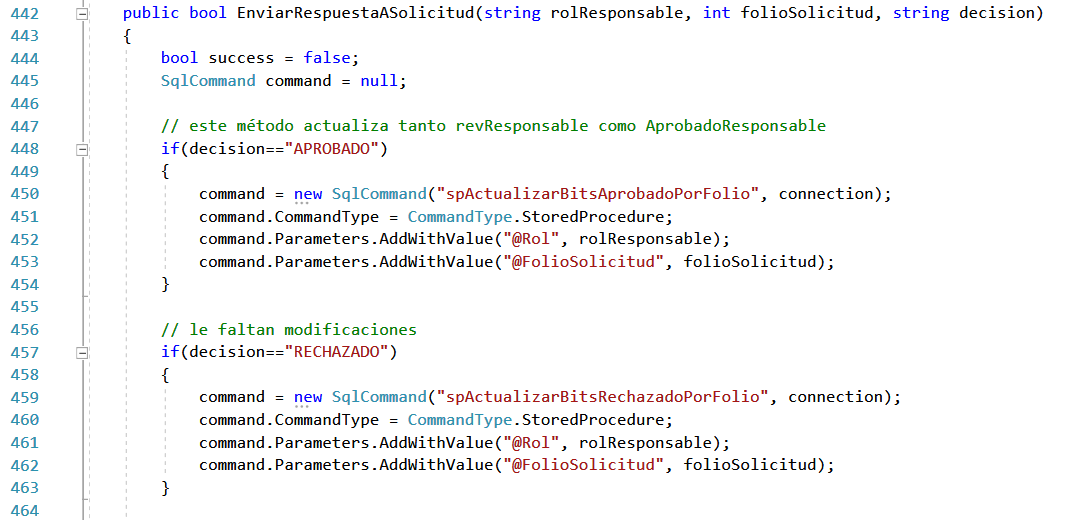
**SolicitudesDALC: MostrarSolicitudesPorEmailEstadoFolioDataset**

Este método realiza la misma actividad que la anterior, sólo que esta recibe un email, estado y folio como parámetros para funcionar.

**SolicitudesDALC: EnviarRespuestaASolicitud.**

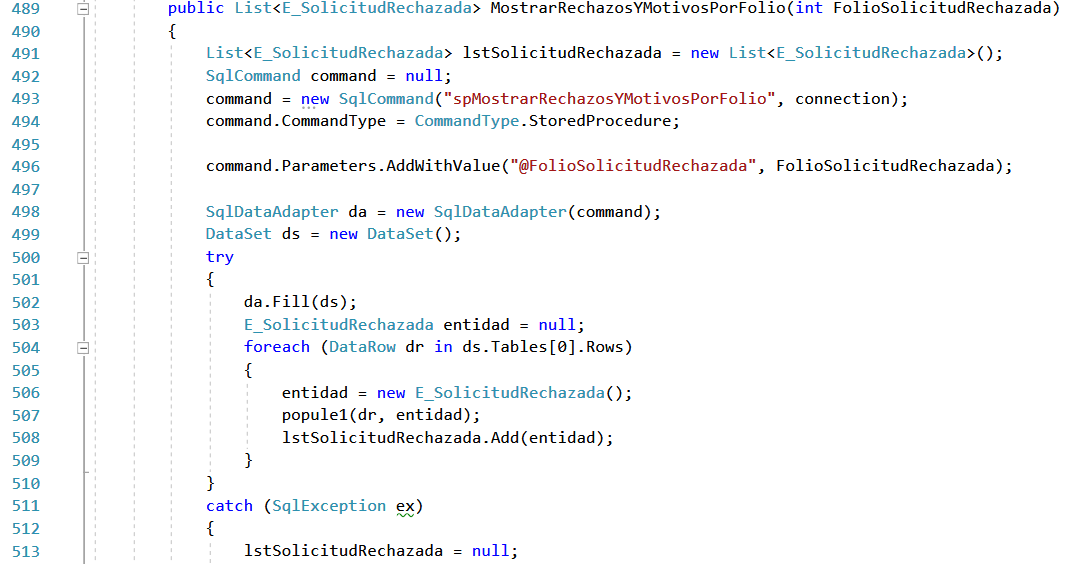
Se utiliza para enviar la respuesta de cada responsable a la solicitud indicada por folio. Este recibe el rol del responsable que se encuentre conectado, el folio de la solicitud a revisar y la decisión que tomó el responsable. Para ello existen dos posibles resultados: que el responsable acepte o que rechacé:

* Si la decisión es APROBADO, se manda a llamar el stored procedure *spActualizarBitsAprobadoPorFolio*.
* En cambio si la decisión es RECHAZADO, se utiliza el stored procedure *spActualizarBitsRechazadoPorFolio*.

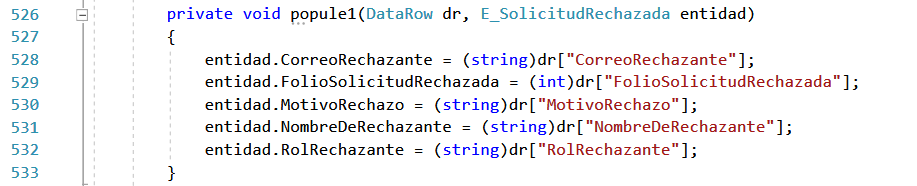


**SolicitudesDALC: MostrarRechazosYMotivosPorFolio**

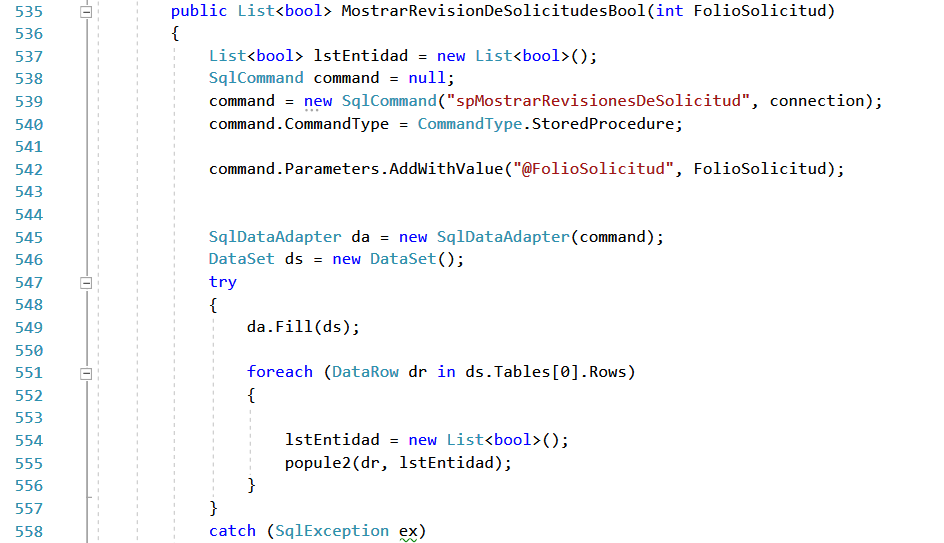
Este método retorna una lista de *E\_SolicitudRechazada*, necesario para mostrarle al usuario los motivos de rechazo de la solicitud indicada por el estado de folio que pide como parámetro. Al igual que los anteriores, este método requiere del método popule para funcionar.



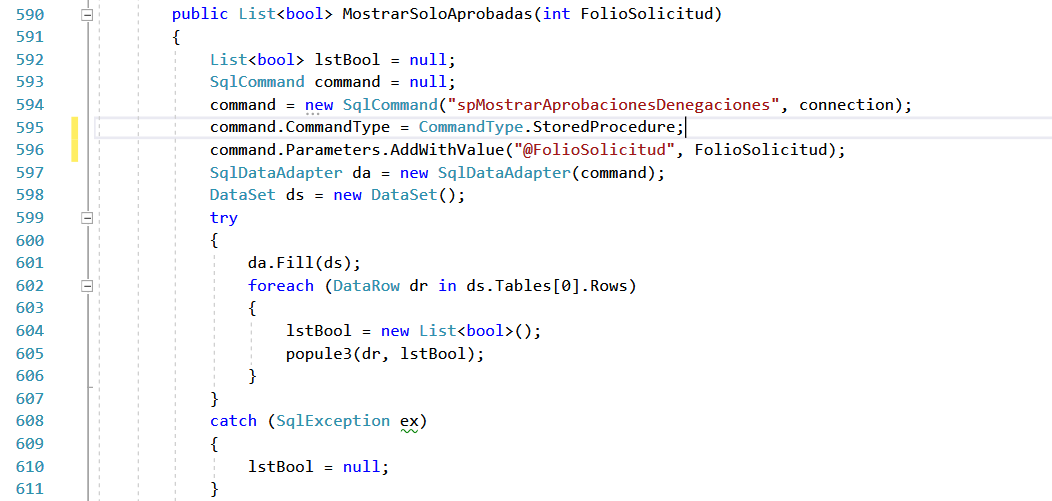
**Método popule de MostrarRechazosYMotivosPorFolio**

****

**SolicitudesDALC: MostrarRevisionDeSolicitudesBool**

****Este método se utiliza para saber el estado de las revisiones de una solicitud en específico, con ella se sabe quiénes han revisado la solicitud y quiénes no. Recibe el folio de la solicitud como parámetro y requiere del atributo connection de la clase *ConnectionDB* para funcionar.

**SolicitudesDALC: MostrarSoloAprobadas**

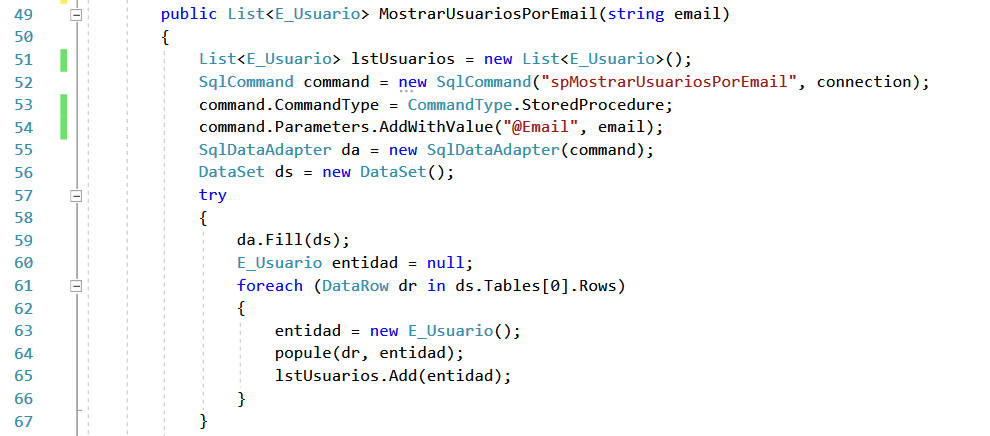
****A diferencia del método anterior, este sólo retorna una lista de las revisiones que sean aprobatorias.

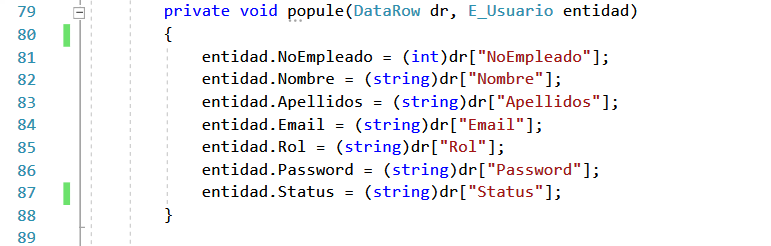
**CAPA DE NEGOCIOS: UsuariosDALC.cs**

En esta clase se tienen todos los métodos necesarios para la administración de usuarios, necesita de la clase *ConnectionDB* para establecer la conexión a la base de datos y de *E\_Usuario* para reconocer la información de usuario necesaria. Esta clase tiene sólo dos métodos internos:

**UsuariosDALC: MostrarUsuariosPorEmail.**

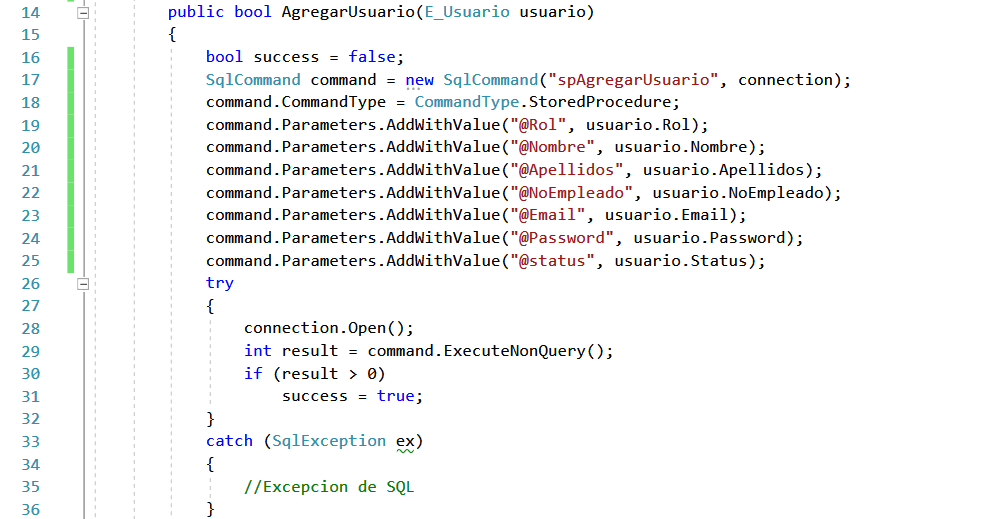
Retorna una lista con sólo una entidad de tipo *E\_Usuario*, correspondiente al email que recibe como parámetro, este requiere del atributo connection de la clase *ConnectionDB* para funcionar así como de su método popule correspondiente.



**Método popule de Mostrar Usuario.**

**UsuariosDALC: AgregarUsuario.**

Se utiliza para insertar un nuevo usuario a la base de datos, parametrizando cada atributo de la clase E\_Usuario. Necesita del atributo connection de la clase ConnectionDB para funcionar.



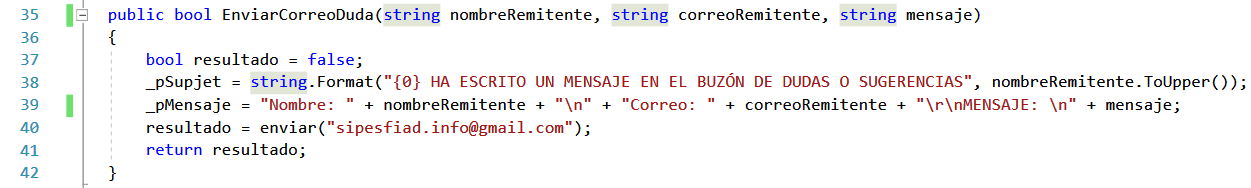
**CAPA PRESENTACIÓN: EnviaCorreoClass.cs**

Por conveniencia, se ha creado esta clase en la capa presentación. Esta contiene una serie de métodos indispensables para el envío de correos en diferentes situaciones. Para esta clase fue necesario instalar el paquete NuGet iTextSharp para el envío de correo electrónicos.

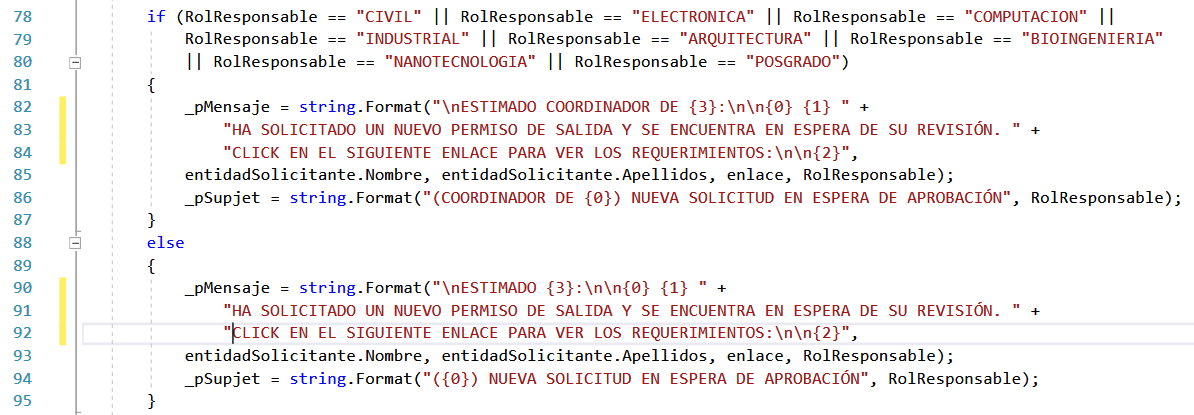
Contiene una serie de atributos para realizar su cometido y un constructor en donde se inicializan el asunto y el mensaje en vacío y el Email fuente y contraseña con la cuenta del sistema creado. A continuación se expondrán sus métodos:

**EnviaCorreoClass: EnviarCorreoDuda.**

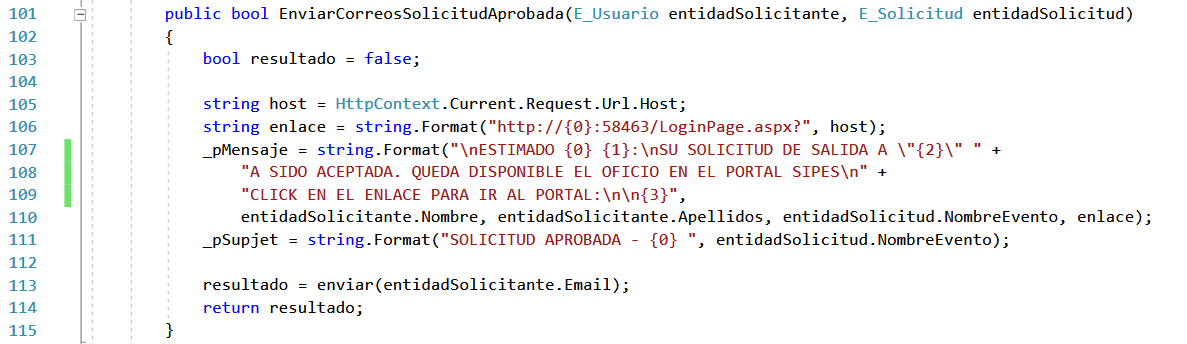
Se utiliza para enviar el correo de la sección de dudas del portal de SIPES sin sesión., esta recibe como parámetros el nombre del remitente, su correo y el mensaje que se enviará.

****

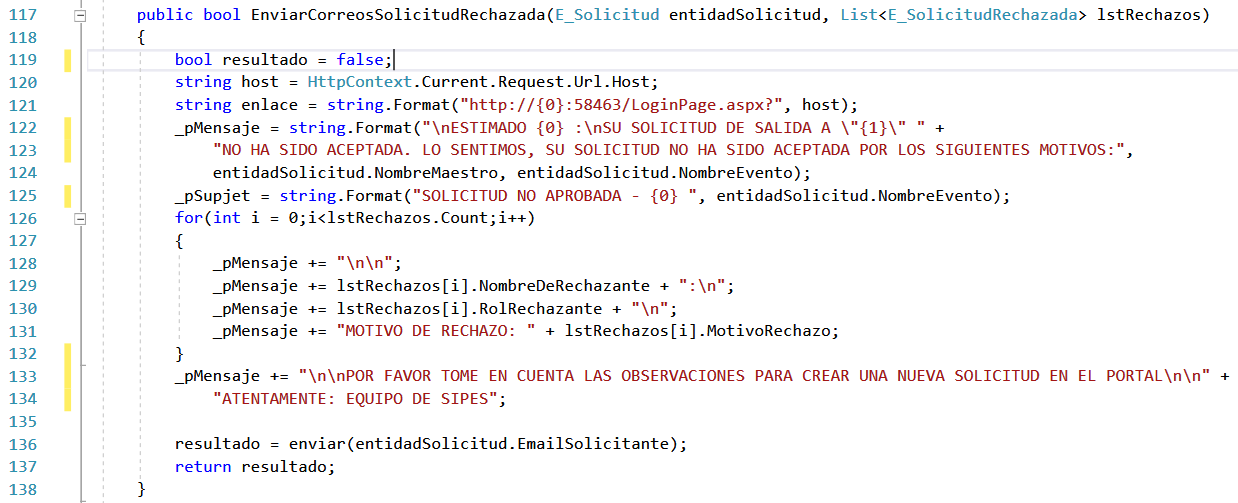
**EnviaCorreoClass: EnviarCorreoResponsable**

Funciona similar a EnviarCorreoDuda, sólo que este es utilizado para enviarles correos a los responsables, avisando cada vez que se solicite un nuevo permiso.

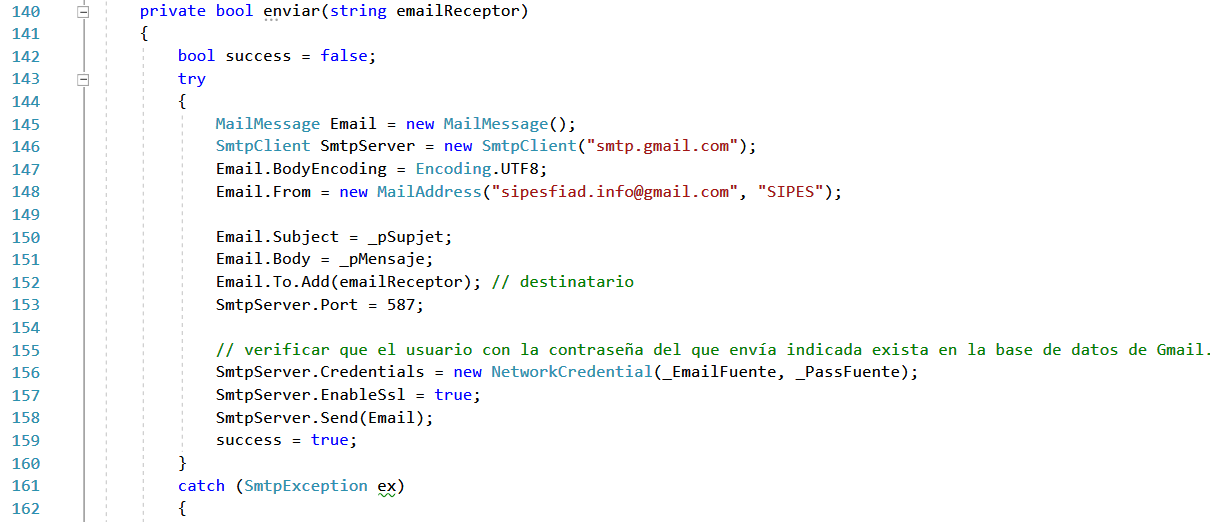
**EnviaCorreoClass: EnviarCorreoSolicitudAprobada.**

Para notificarle al solicitante cuando su solicitud haya sido aprobada.

**EnviaCorreoClass: EnviarCorreoSolicitudRechazada.**

****Utilizada para notificarle al usuario por correo cuando su solicitud no haya sido aceptada, también se le muestran los motivos del rechazo.

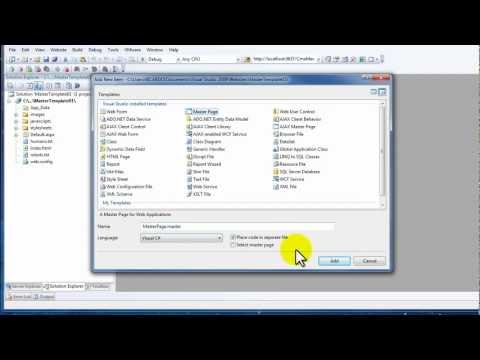
Todos los métodos de EnviaCorreoClass requieren del método enviar() para funcionar, este encuentra junto a los demás métodos y es el encargado de autentificar el usuario y crear la conexión a la base de datos de servicio de correo.



**CAPA PRESENTACIÓN**

En la capa presentación se encuentra la interfaz gráfica, intuitiva, amigable al usuario e incluso ajustable en dispositivos móviles, que le permitirá solicitar sus permisos ya sea desde su teléfono o móvil de una manera cómoda, rápida y eficiente.

Cabe mencionar que el diseño fue tomado de una página de templates, este fue migrado de HTML a ASP.NET apoyándonos del siguiente video:

[https://www.youtube.com/watch?v=MH\_ZztGR4-Y](https://www.youtube.com/watch?v=MH_ZztGR4-Y)

Tal y como se aprendió, se pasó el template a masterpage, nombrados como MP\_SinSesion y MP\_ConSesion, respectivamente. De ellas se derivaron las diferentes páginas que conforman la capa de presentación completa.

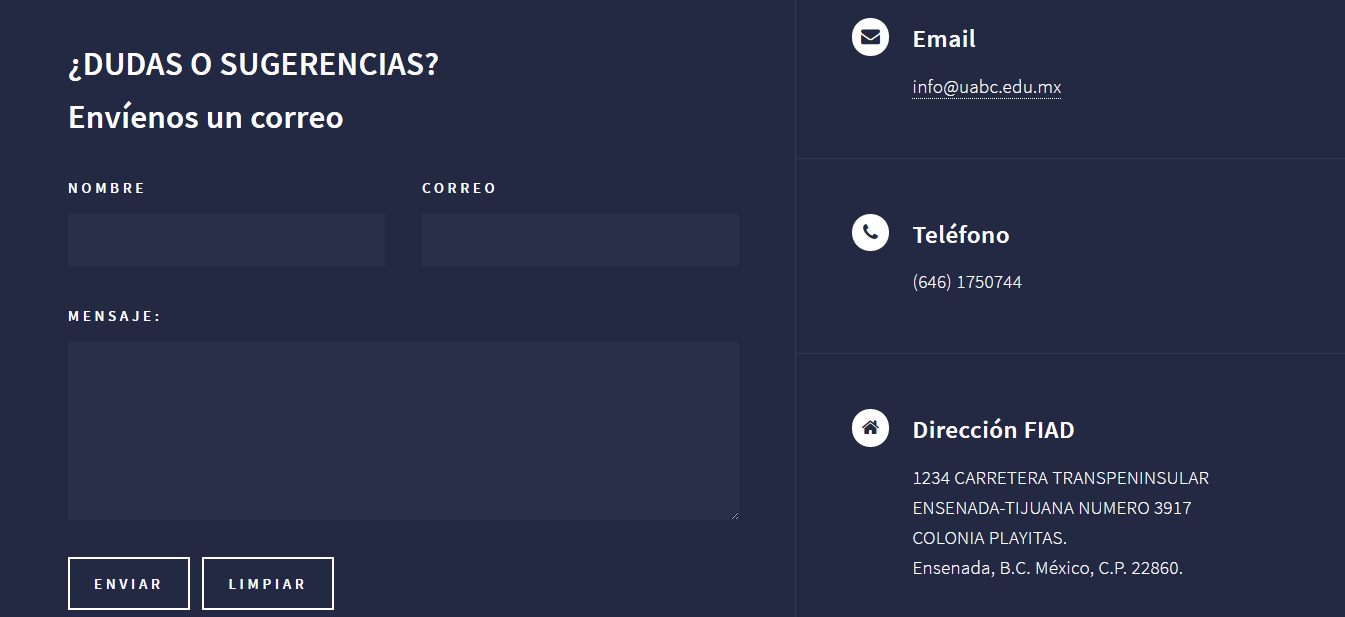
El template fue tomado de esta página web: <https://html5up.net/forty>

A continuación se presentará las vistas del sistema sin sesión y con sesión.

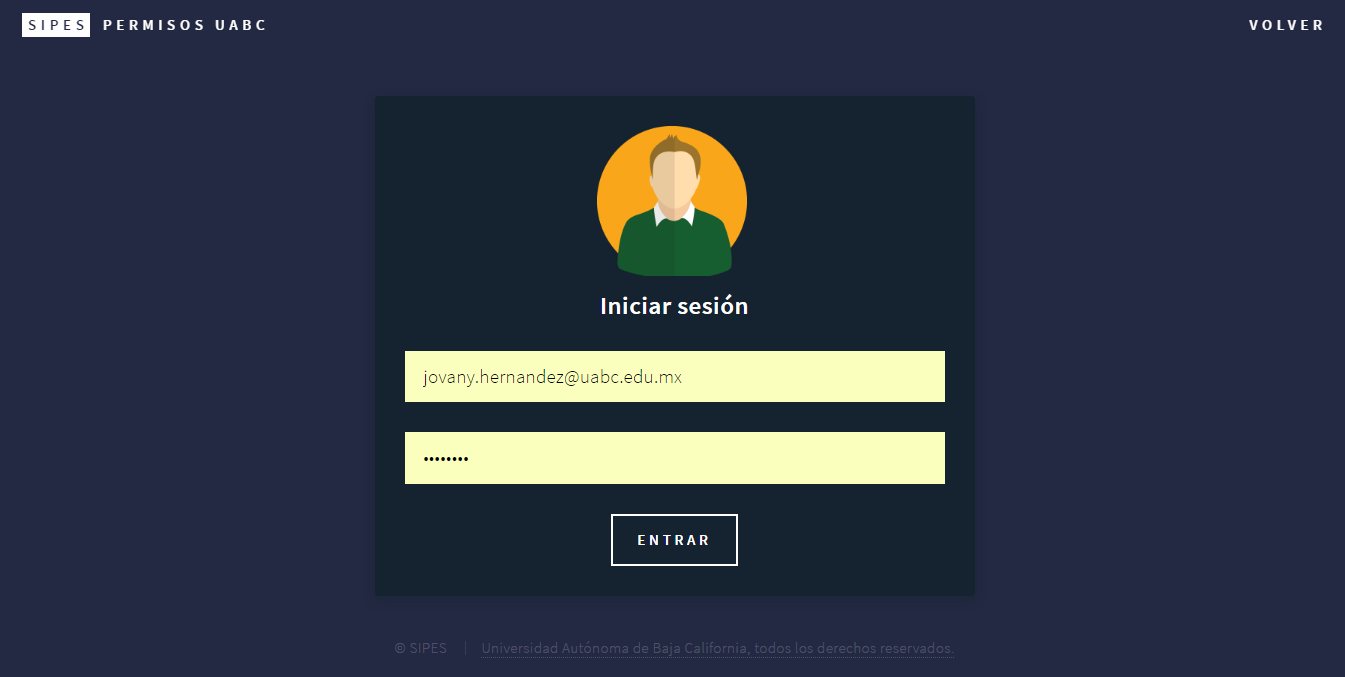
**SIPES: SIN SESIÓN.**

Como primera instancia se tiene la página de inicio, que le da la bienvenida al usuario y le muestra información acerca del sistema:

Si el usuario necesita más información acerca del cómo solicitar su permiso, puede enviar un correo al equipo de SIPES, quien en su debido momento le dirá exactamente qué hacer, dependiendo de sus necesidades.



Para solicitar un nuevo permiso, el interesado deberá iniciar sesión en el sistema, ingresando su correo facultativo y contraseña.



**SIPES: CON SESIÓN.**

Una vez iniciado sesión se le presenta la siguiente vista.

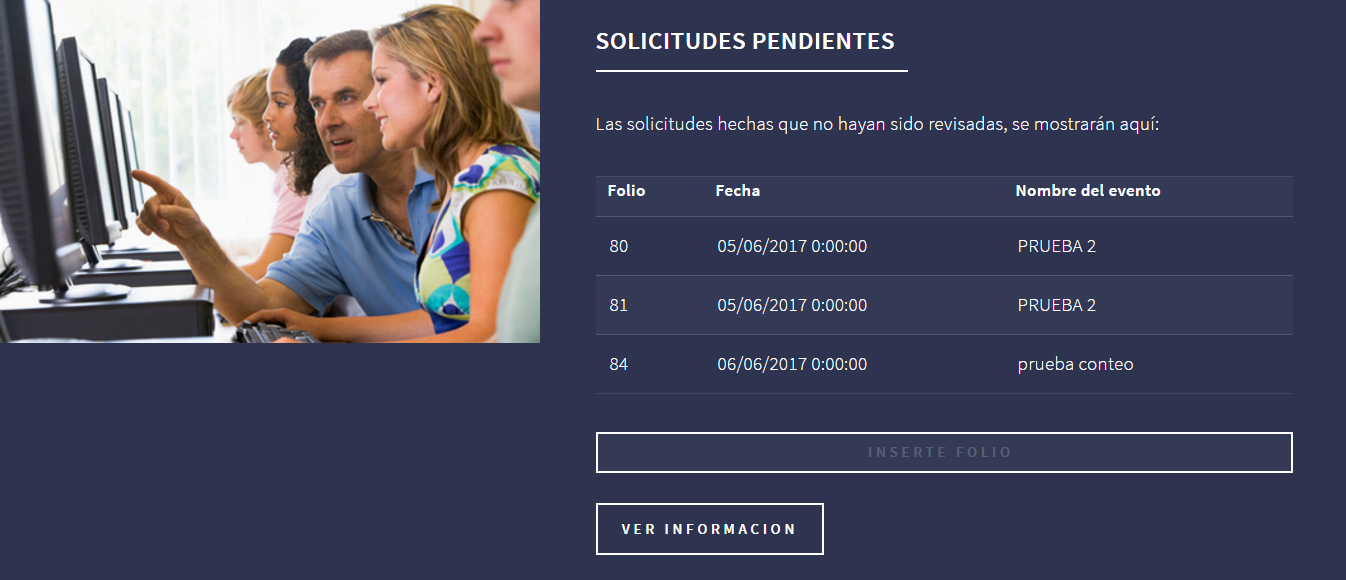
SECCIÓN DE BIENVENIDA:

Se muestra información del usuario, como su nombre y número de empleado.



SECCIÓN DE SOLICITUDES PENDIENTES.

En ella se muestran las solicitudes creadas que no hayan sido revisadas aún. Aquí, el usuario podrá revisar el estado de su solicitud, mostrándole quiénes la han revisado y quiénes faltan por hacerlo.



SECCIÓN DE SOLICITUDES APROBADAS.

En esta sección el usuario podrá ver el listado de las solicitudes aprobadas, así como imprimir, subir reportes y evidencias de las salidas.



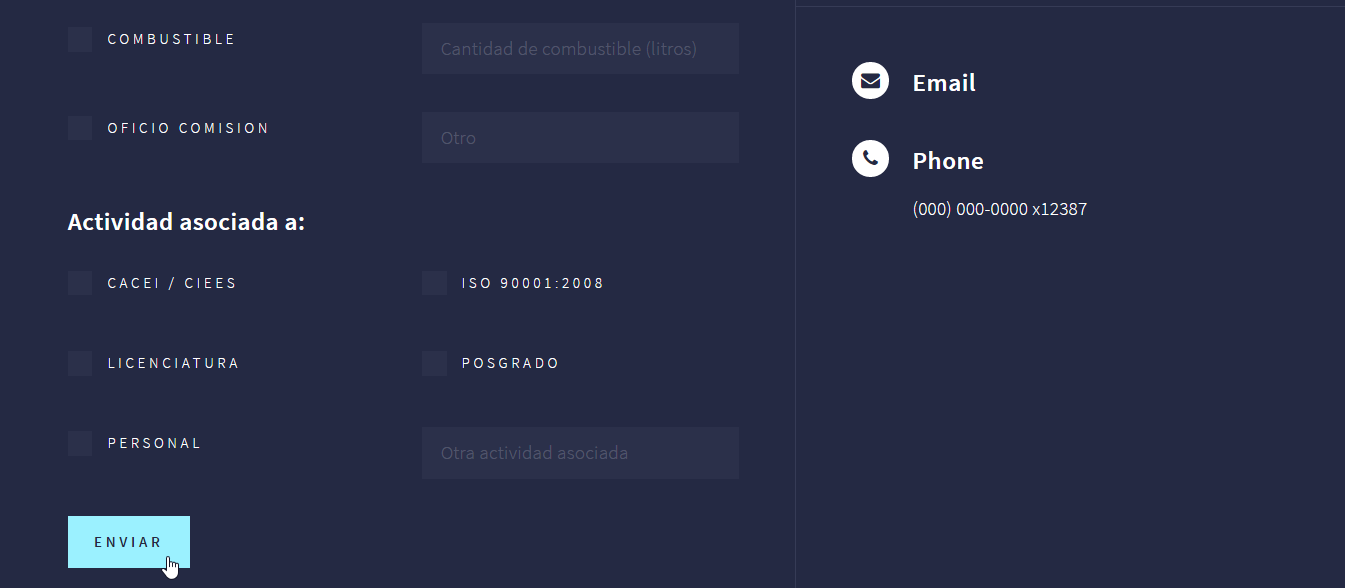
SECCIÓN DE SOLICITUDES RECHAZADAS.

En esta sección se alojan las solicitudes que no hayan sido aceptadas. Si alguna de las solicitudes se encuentra en este apartado, el usuario tendrá los motivos del rechazo especificados por medio de un correo electrónico enviado automáticamente por SIPES.



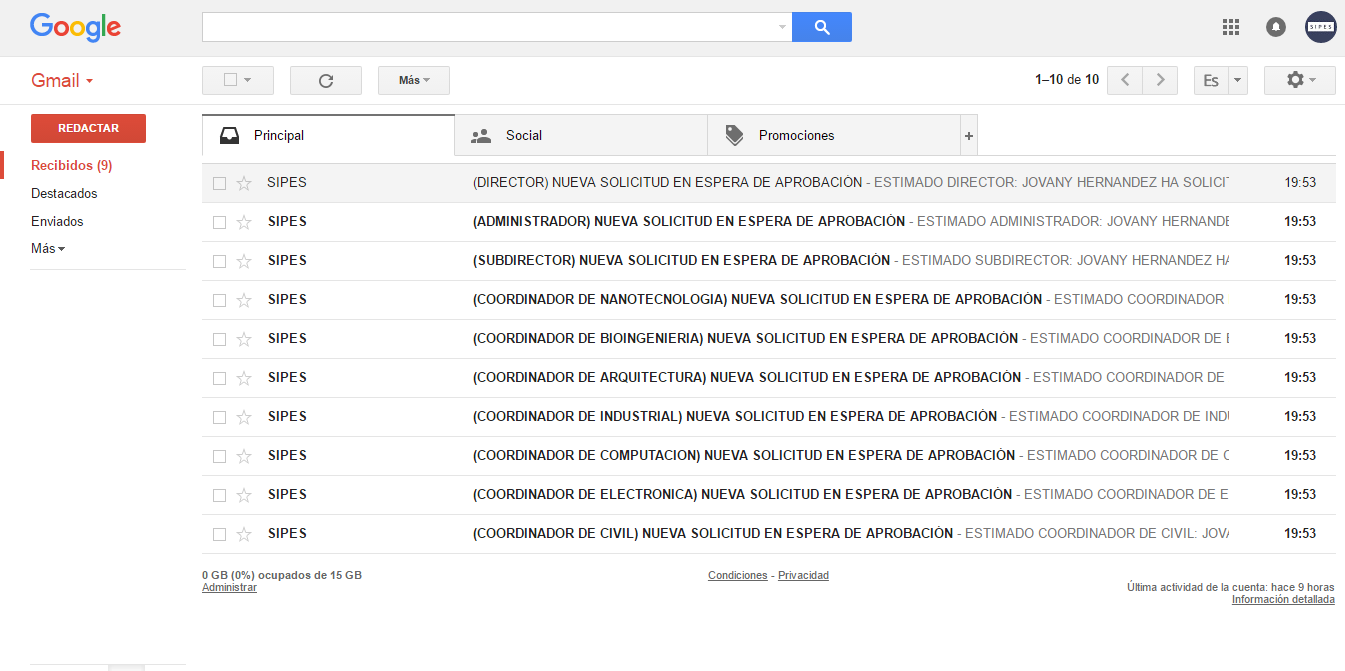
SECCIÓN SOLICITAR NUEVO PERMISO:

En este apartado, el usuario tendrá que llenar todos los requerimientos para realizar su solicitud.

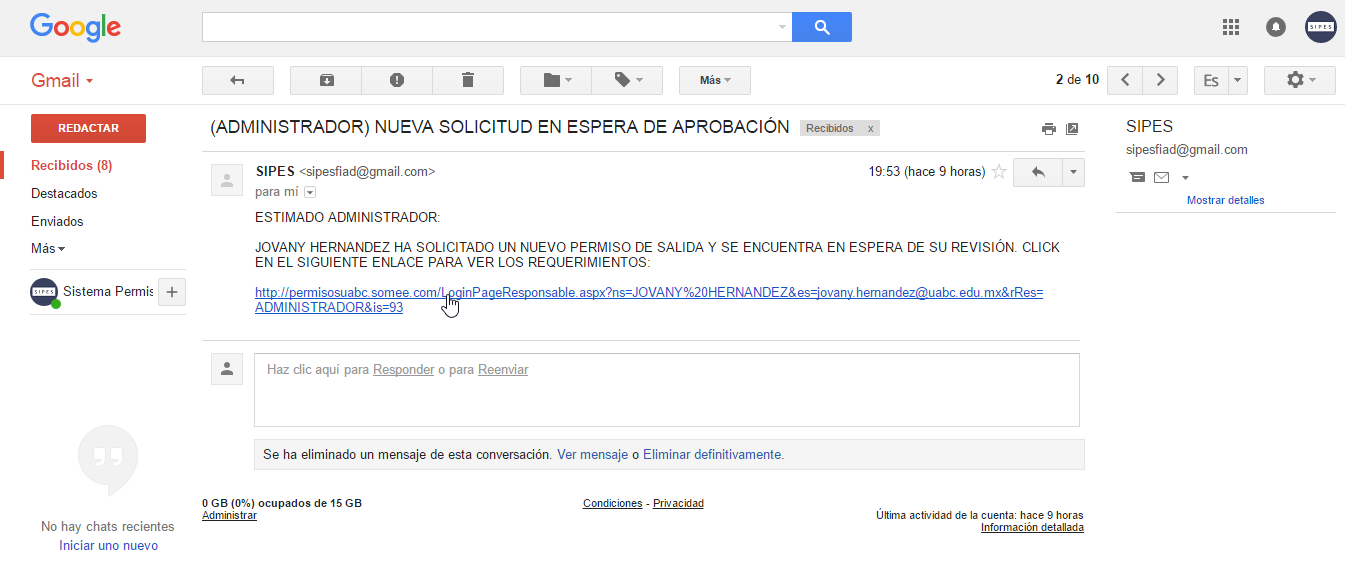


Cuando de Click en enviar, la notificación de nueva solicitud se le enviará a cada responsable dependiendo lo la descripción de la solicitud. El usuario deberá esperar a que estos responsables revisen y aprueben para poder imprimir su oficio comisión.

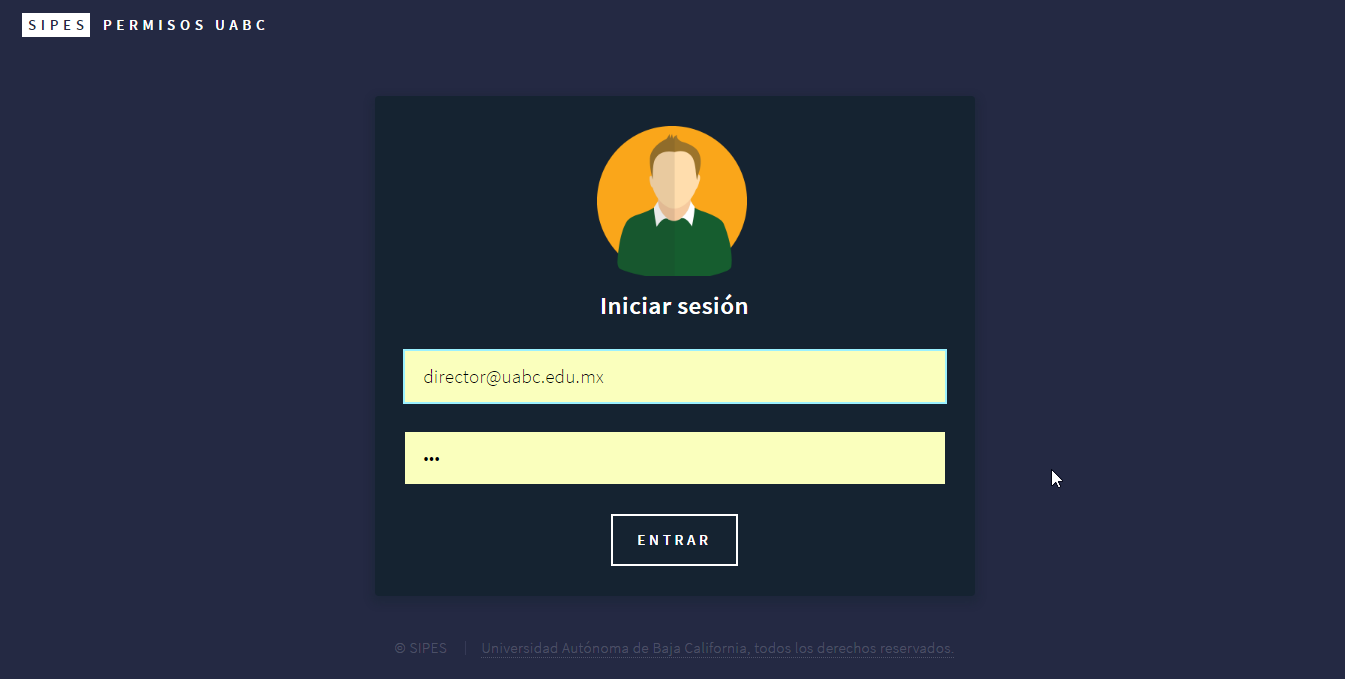
Cada responsable recibirá un correo personalizado donde se le notificará acerca de la nueva solicitud.



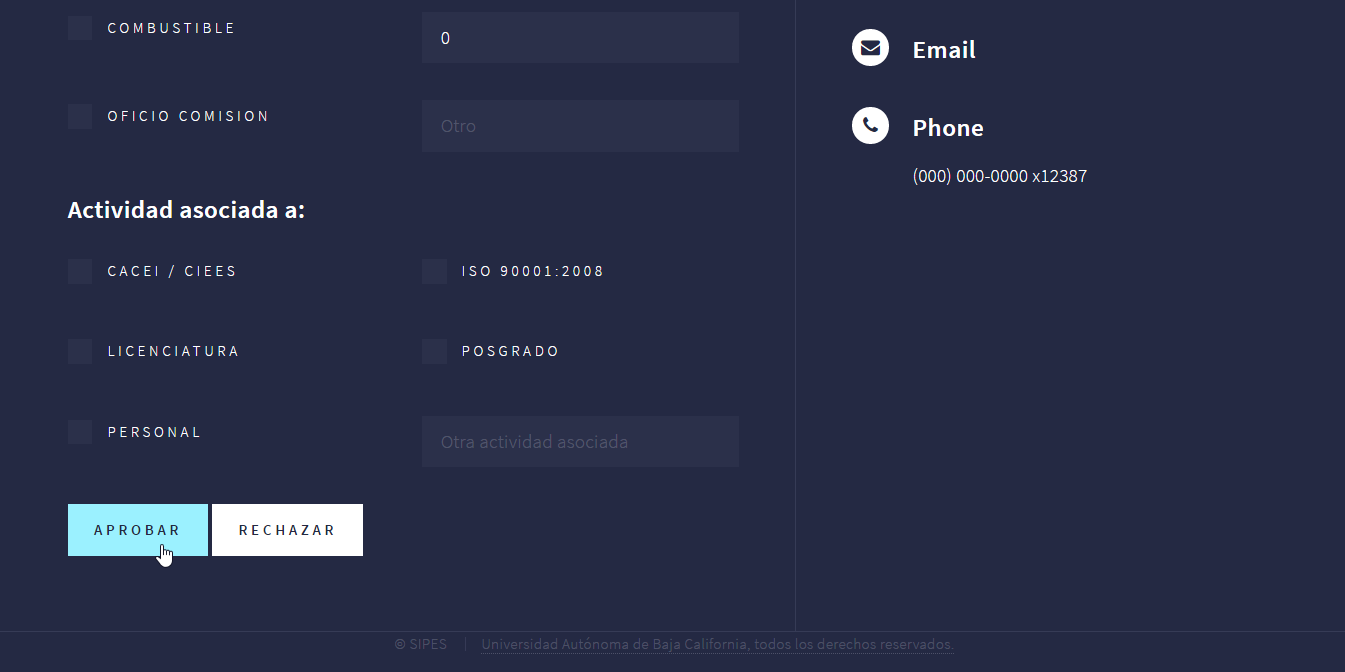
Adjunto al correo se le mostrará en enlace directo para ver la solicitud y dar su veredicto.



Cada responsable deberá iniciar sesión para revisar las solicitudes.



El responsable tendrá la facultad de negar o rechazar la solicitud.



Una vez terminada la revisión, se le notificará al solicitante vía correo electrónico

**AVISOS Y ADVERTENCIAS**

Con el fin de mejorar la experiencia de navegación, se han hecho una serie de avisos o advertencias de forma personalizada, para notificar cualquier error o suceso en el sistema, cada una de ellas expuestas a continuación:

INDIQUE FOLIO:

Se muestra cuando el usuario presiona el botón de ver información de solicitud, ver oficio comisión o subir reporte y evidencias, en SIPES con sesión sin haber indicado ningún folio válido.



FOLIO NO ENCONTRADO:

Se muestra cuando el usuario inserta un número de folio que no se encuentra registrado en la base de datos.



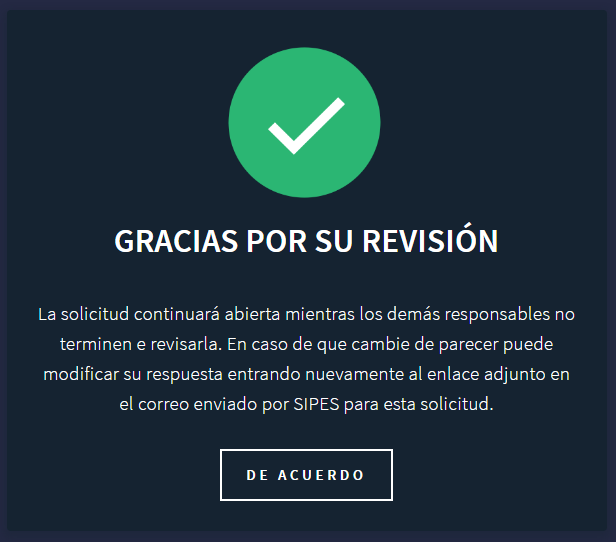
SOLICITUD ENVIADA

Se muestra cada vez que el usuario llena el formato de solicitud y lo registra en SIPES



GRACIAS POR SU REVISIÓN

Este se muestra cuando alguno de los responsables revisa la solicitud y la acepta.



GRACIAS POR SU REVISIÓN: ESPECIFICA MOTIVO DE RECHAZO.

En caso de que el responsable rechazo alguna solicitud, deberá especificar la razón de su rechazo.



EVALUACIÓN DE SOLICITUD TERMINADA:

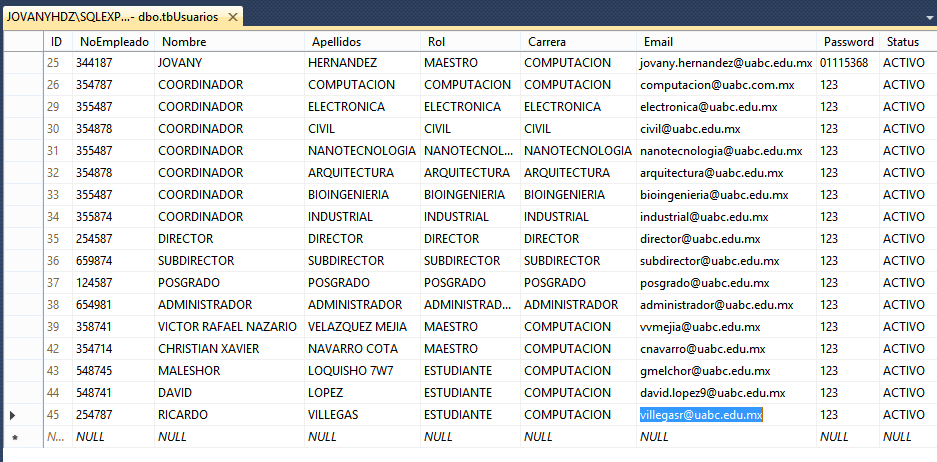
Este aviso se le mostrará a cualquier responsable que desee entrar nuevamente para ver los requerimientos de alguna solicitud que ya haya sido revisada por el resto de los demás responsables.

**DESCRIPCIÓN DE TABLAS Y MÉTODOS**

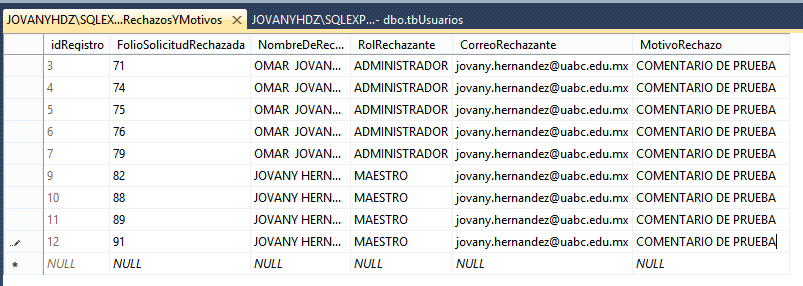
**SQL con Management Studio 2014**

Para el funcionamiento de nuestro sistema se empleó el uso de tres tablas principales: tbSolicitudes, tbSolicitudesRechazadas y tbUsuarios.

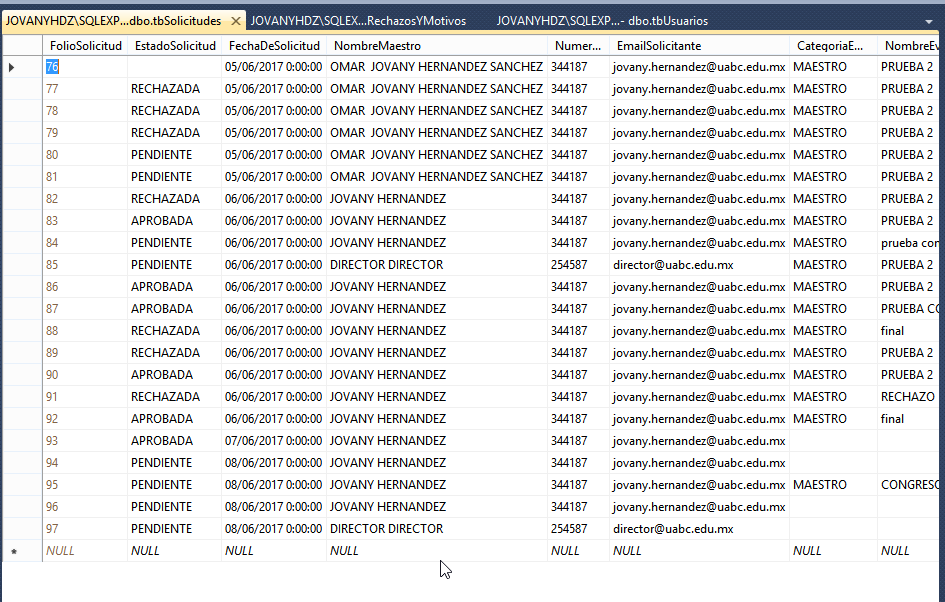
**tbUsuarios:** en ella se almacena toda la información relevante sobre los usuario registrados en el sistema.

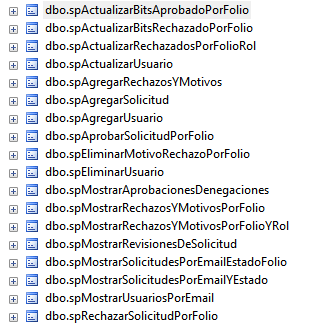


**tbSolicitudesRechazadas:** para almacenar información acerca de los motivos de rechazo de todas las solicitudes.



tbSolicitudes: en ella se almacenan todas las solicitudes hechas por todos los solicitantes:



**Este es el listado de los stored procedures utilizados.**

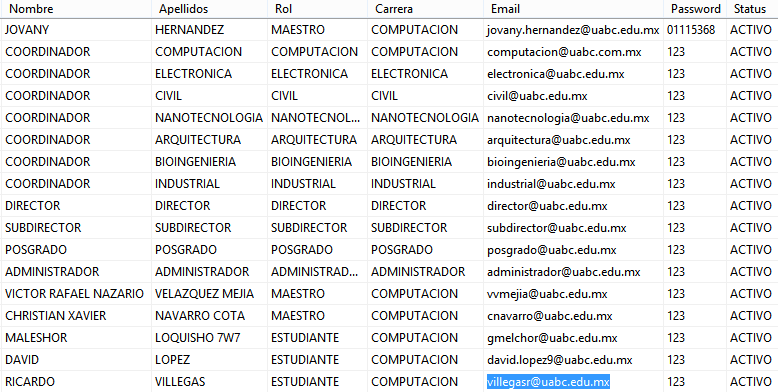
No son mas que un conjunto de métodos de SELECTE, INSERT, UPDATE Y DELETE, para cada caso de consulta requerida. Estos procedimientos son la base fundamental de la interacción con las tablas en la base de datos con toda la lógica de programación en el sistema escrito en C#.

Para una mejor experiencia, se ha subido el sistema a un servidor de ASP.NET, puede verlo directamente de este enlace:



<http://www.permisosuabc.somee.com/>

Puede acceder con cualquiera de estas cuentas, incluyendo su correo electrónico de la universidad.



También se deja a su disposición en enlace de descarga al código fuente del proyecto corregido:

<https://drive.google.com/file/d/0B3zmNnoWLkdfZHJCU1V0WW5Xa0U/view?usp=sharing>