30 de mayo de 2018

Esteban coto alfaro – Johan torres creed - Danny Xie

Instituto tecnologico de costa rica

Documentación fase ii

Proyecto 2

**Índice**

Introducción ……………………………………….……………………………………………………….. 2

Descripción del problema…………………….………………………………………………………… 2

Documentación del Web service……………………………………………………………………… 4

Documentación del app………………………………………………………………………………….. 5

Conclusiones……………………………..………………………………………………………………….. 5

**Introducción**

El siguiente problema, el cual se presenta a ser desarrollado mediante la utilización del modelo relacional-objeto en Oracle, es un sistema para soportar las operaciones diarias de la Universidad Tecnológica (UT), donde se permita que la base de datos pueda contener información sobre los campus, facultades, edificios, personal, grados, materias ofrecidas entre sus derivados de la UT. La oficina STI (Servicios de Tecnología de Información) la cual es responsable del sistema de base de datos en la universidad, ha decidido utilizar un modelo relacional-objeto sobre Oracle para la implementación de la base de datos, lo cual se explicará en la descripción el problema.

Además, la segunda fase incluye el manejo de RESTful Web APIs (Application Programming Interfaces) sobre una aplicación Web MVC, permitiendo que dentro de la misma se pueda comunicar con la base de datos en Oracle y así realizar las operaciones sobre esa base de datos, las cuales incluyen inserción, lectura, actualización y borrado en una tabla determinada.

**Descripción del problema**

Actualmente la universidad Tecnológica usa un sistema de base de datos relacional para realizar sus operaciones cotidianas. De la cual este posee registros acerca de los campus, facultades, personal, edificios, grados, entre otros registros de diferentes entidades. La oficina de Servicios de Tecnología de Información (STI), es el encargado de las bases de datos de la universidad, este nos describe el problema de utilizar base de datos modelos relacional-objeto usando el motor de Oracle.

Se describe que la universidad posee ocho campus en todo el país. De la cual la entidad Campus tiene relación con las entidades Edificio y Persona, además no existe relación directa sobre las entidades cursos de diferentes grados y las facultades.

La universidad posee cinco facultades, cada facultad está compuesta de distintos departamentos, escuelas y centros de investigación. Este se implementa con objetos. Usar tablas anidadas en estas entidades descritas anteriormente.

Los datos School\_Prof y Dept\_Prof de las entidades facultad, escuela, departamento y centros de investigación son entidades. Además, el atributo Unit de centro de investigación son varias unidades.

Cada campus posee varios edificios, cada uno están compuestos por distintas habitaciones que esta puede ser oficinas, salas de clase y laboratorios. La facultad puede ocupar varios edificios. Un edificio sólo puede ser asignado a una facultad.

Cada facultad ofrece a los estudiantes una cantidad de grados, cada grado particular sólo puede ser asignado por una facultad particular.

Los datos del personal, este se puede categorizar en dos tipos: estudiantes y staff. El staff se puede categorizar en administrador, técnico, profesor, tutor. Un lecturer puede ser categorizado como associate lecturer o senior lecturer. Además, un tutor puede ser un estudiante.

Cada estudiante tendrá una carrera, este puede llevar más de un degree. Además, el estudiante está enlazado con una materia, además la materia está relacionada con un profesor que es el encargado de la materia.

Para la segunda fase del problema se va a desarrollar una aplicación web y un web service RESTful que será consumida por la aplicación web.

**Documentación de la implementación de la aplicación**

La aplicación se implementó en una plataforma web en el lenguaje C# usando la arquitectura Model View Controller.

En la siguiente sesión de la aplicación está el registro del personal de la universidad. En esta sesión permite el registro del personal tales como el registro de estudiantes, profesores, empleados, estudiantes tutores, empleados tutores, técnicos, administradores y profesores asociados.

A continuación, en la figura 1 se muestra la pantalla principal de registro de personal de la universidad. En la barra superior se muestra el personal que se podrá registrar en el sistema, como se describió anteriormente. Cada sección este compuesto por 4 módulos, para realizar operaciones sobre cada registro, por ejemplo, visualizar, eliminar, actualizar y agregar registros.

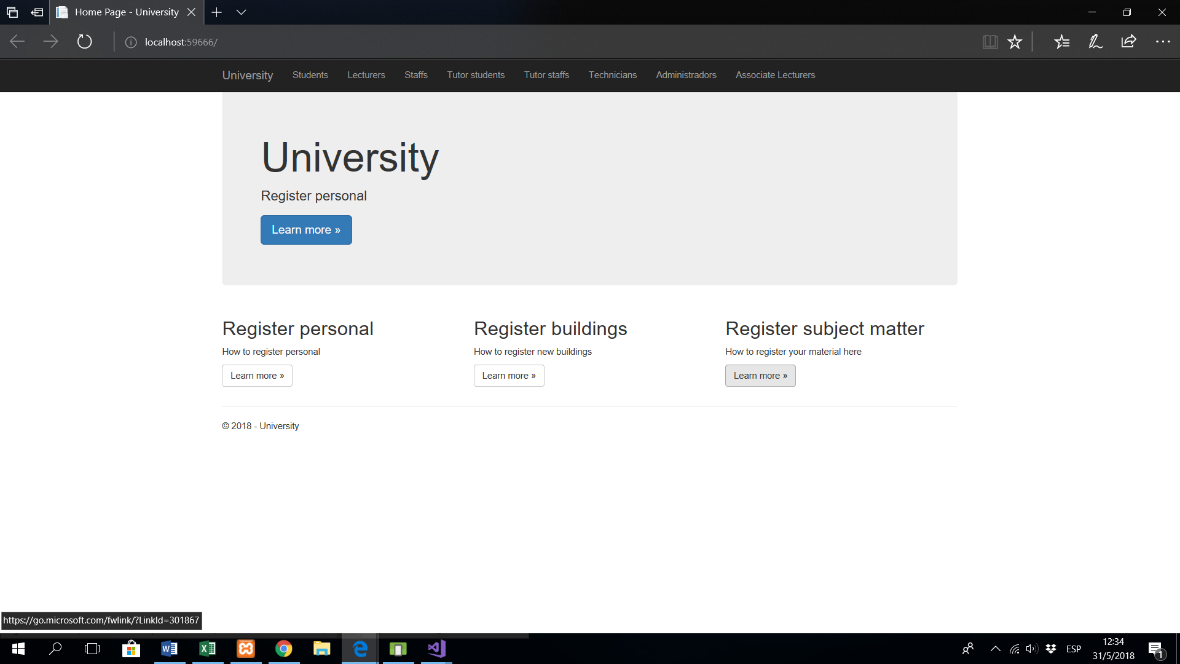


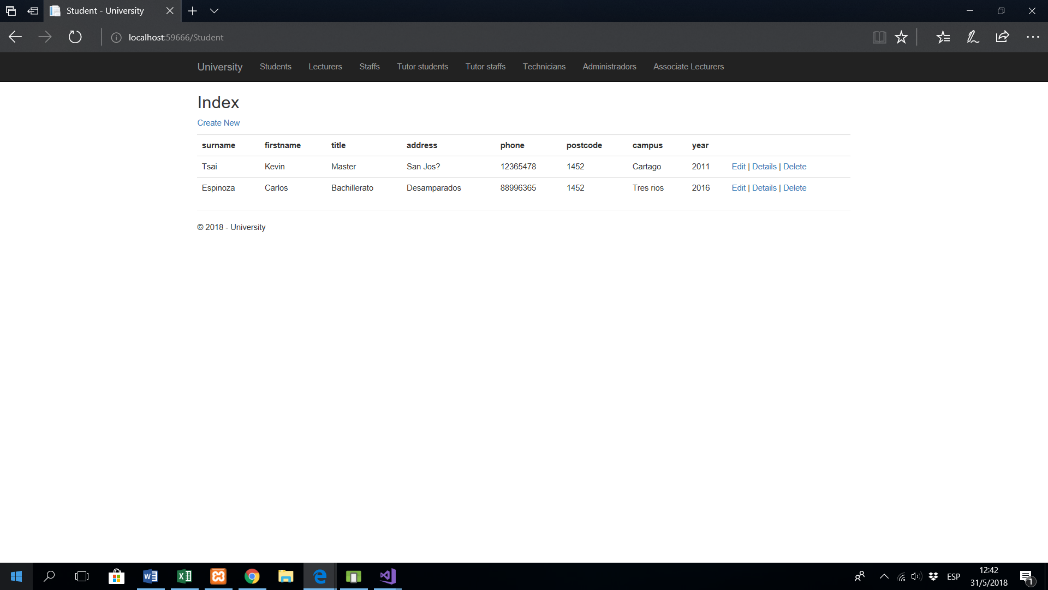
Figura 1

A continuación, se describirá cada sección de registro del personal:

* Sección Student

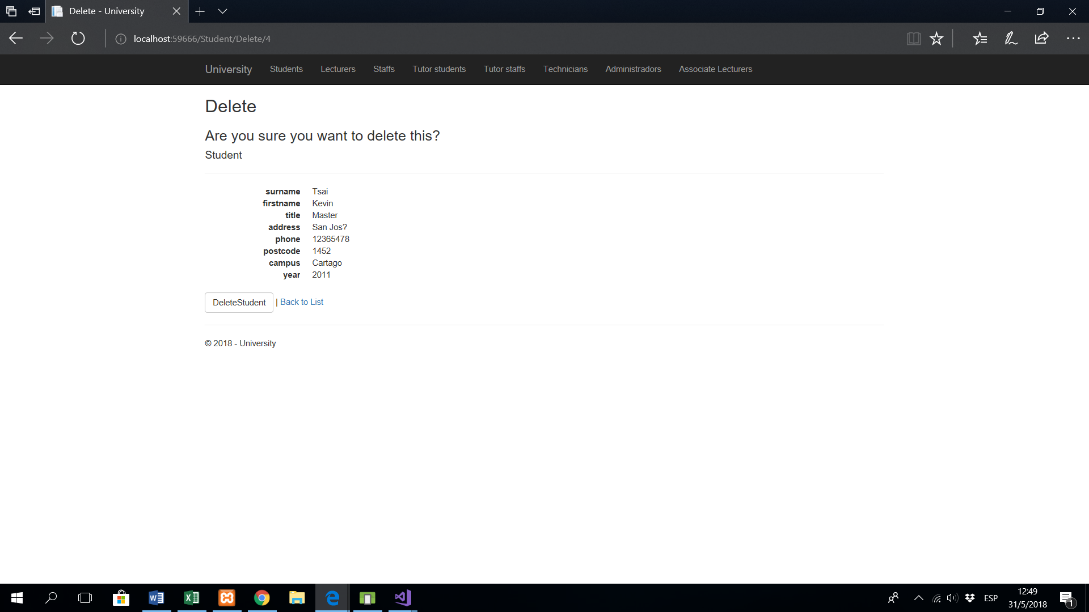
1. **Visualizar registros**

En esta sección se podrá visualizar los registros de los estudiantes guardados en el sistema.



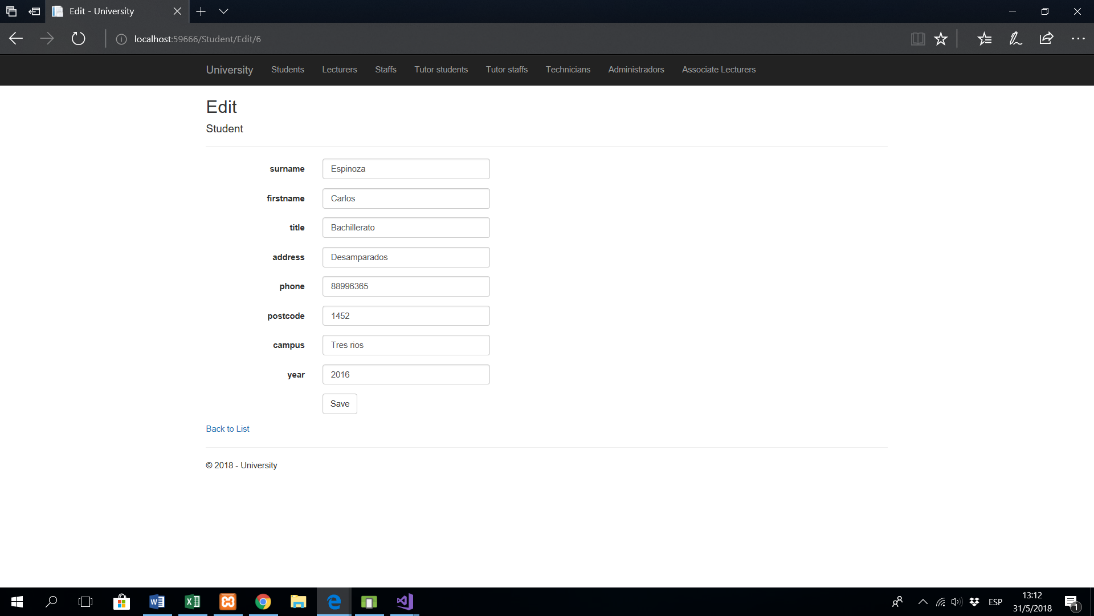
1. **Eliminar registro**

Después de hacer click en un registro en la opción de delete se le mostrrá la siguiente vista para eliminar el registro seleccionado. Si está seguro de eliminar el registro debe tocar el botón *DeleteStudent* o sino *back To List* para regresar a la sección anterior.



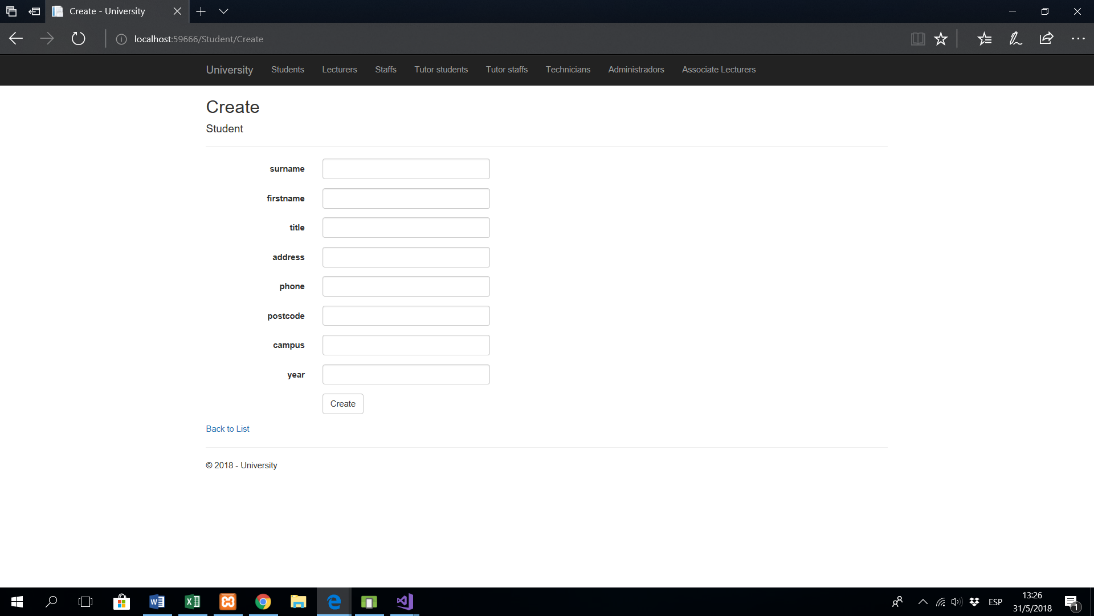
1. **Actualizar registro**

Al tocar el botón de editar este podrá actualizar los datos del registro en el sistema. Este podrá actualizar los registros, después de actualizarlos deberá tocar el botón de *save* para guardar los cambios en el sistema.



1. **Registrar un registro**

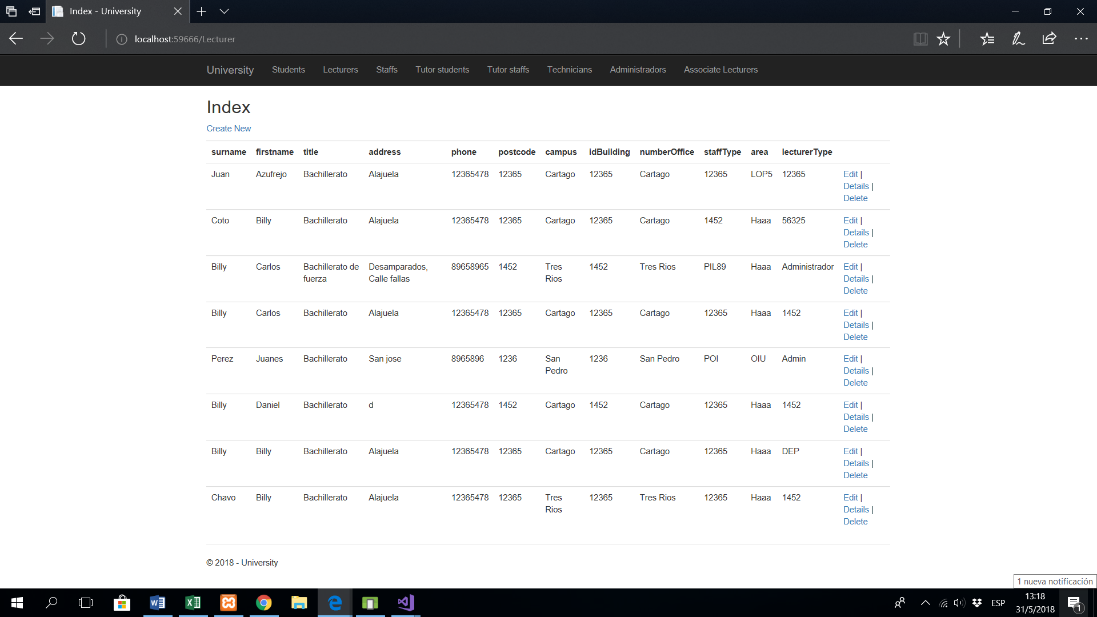
Al tocar *Create New*, se le mostrará los campos que debe llenar para poder registrar el registro en el sistema. AL terminar de llenar los datos deberá tocar el botón de *Create* para poder registrar el nuevo registro en el sistema.



* Sección Lecturer

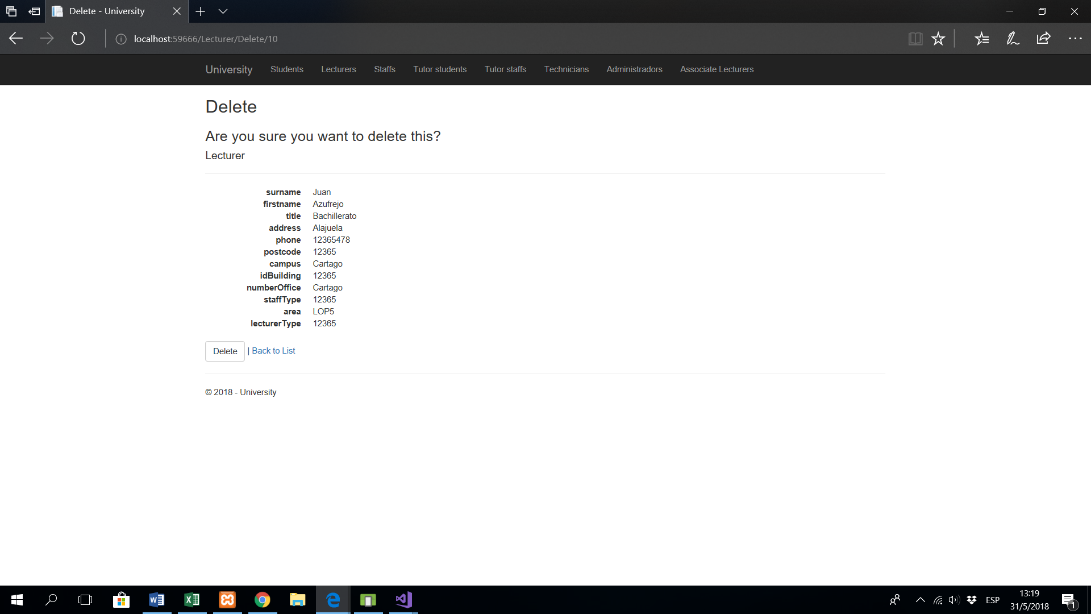
1. **Visualizar registros**

En esta sección se podrá visualizar los registros de los profesores guardados en el sistema.

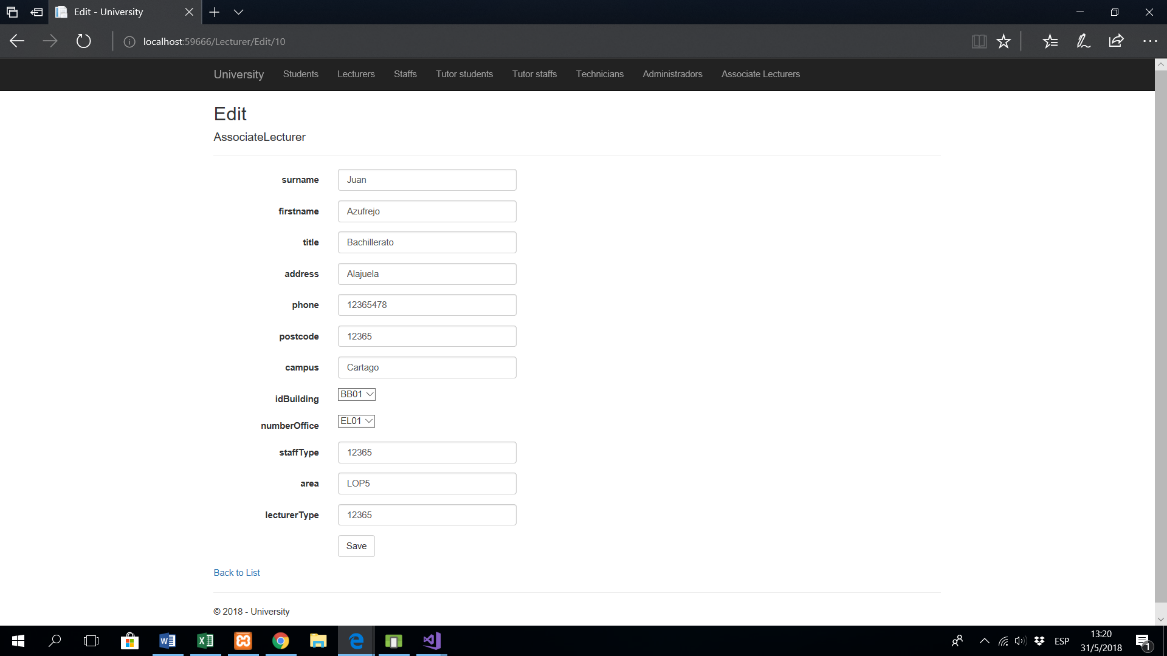


1. **Eliminar registro**

Después de hacer click en un registro en la opción de delete se le mostrrá la siguiente vista para eliminar el registro seleccionado. Si está seguro de eliminar el registro debe tocar el botón *Delete* o sino *back To List* para regresar a la sección anterior.

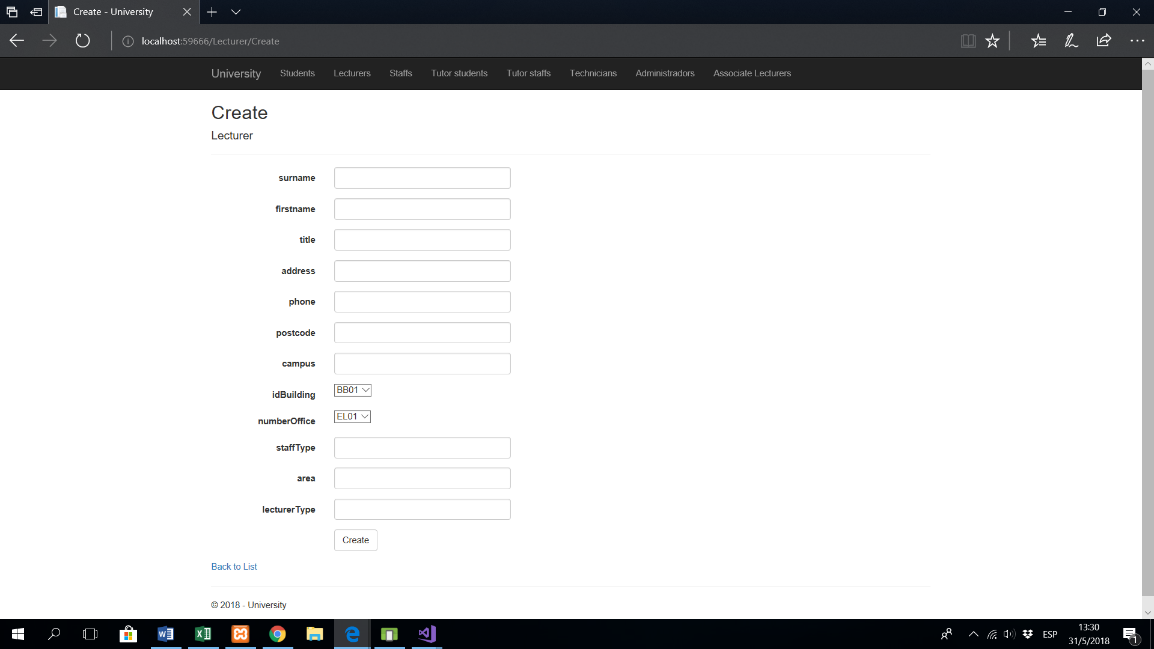


1. **Actualizar registro**

Al tocar el botón de editar este podrá actualizar los datos del registro en el sistema. Este podrá actualizar los registros, después de actualizarlos deberá tocar el botón de *save* para guardar los cambios en el sistema.

1. **Crear registro**

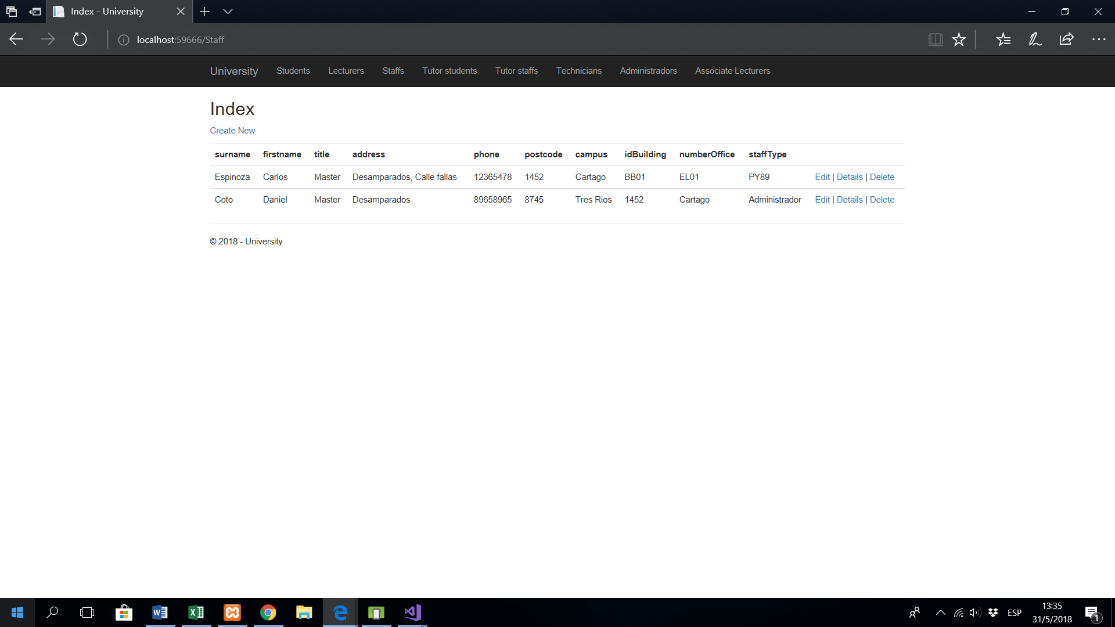
Al tocar *Create New*, se le mostrará los campos que debe llenar para poder registrar el registro en el sistema. AL terminar de llenar los datos deberá tocar el botón de *Create* para poder registrar el nuevo registro en el sistema.



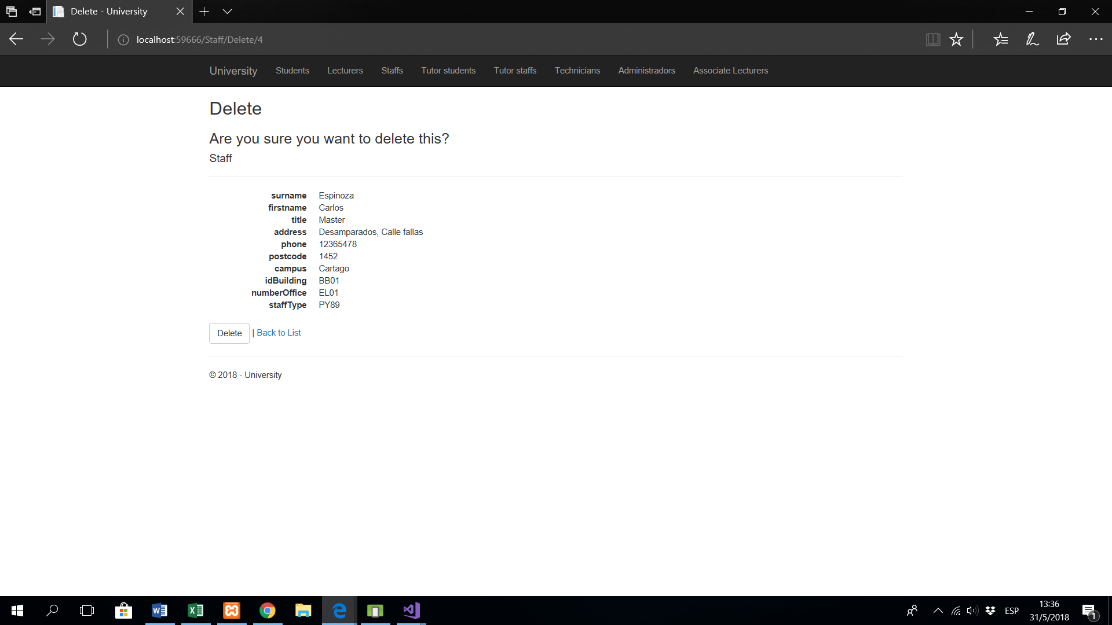
* Sección Staff

1. **Visualizar registros**

En esta sección se podrá visualizar los registros de los personal guardados en el sistema.

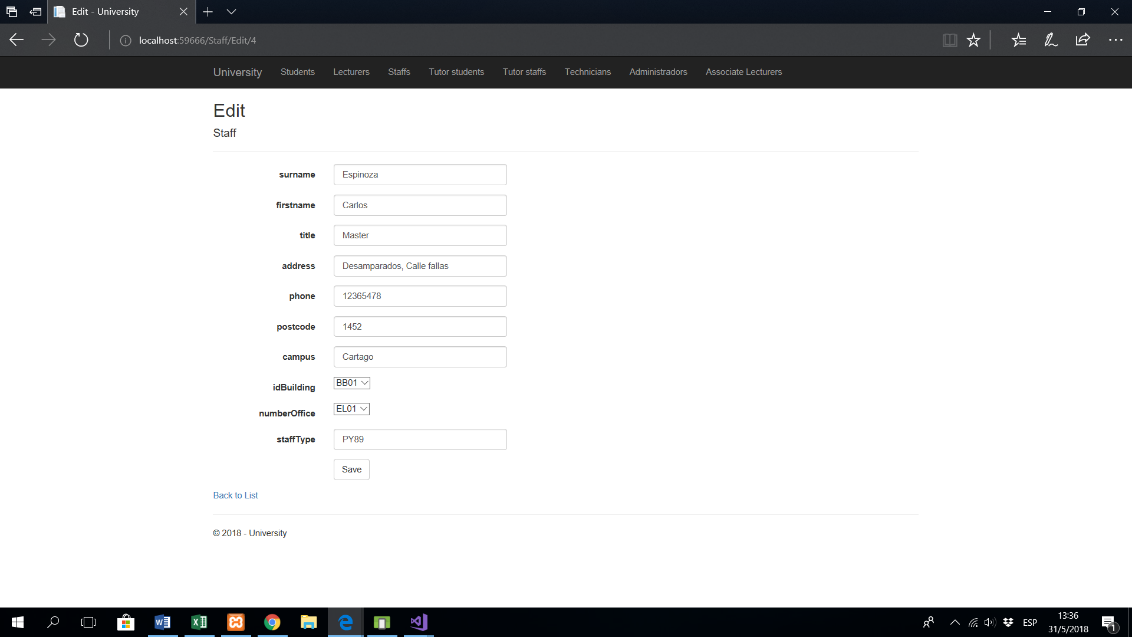


1. **Eliminar registro**

Después de hacer click en un registro en la opción de delete se le mostrrá la siguiente vista para eliminar el registro seleccionado. Si está seguro de eliminar el registro debe tocar el botón *Delete* o sino *back To List* para regresar a la sección anterior.

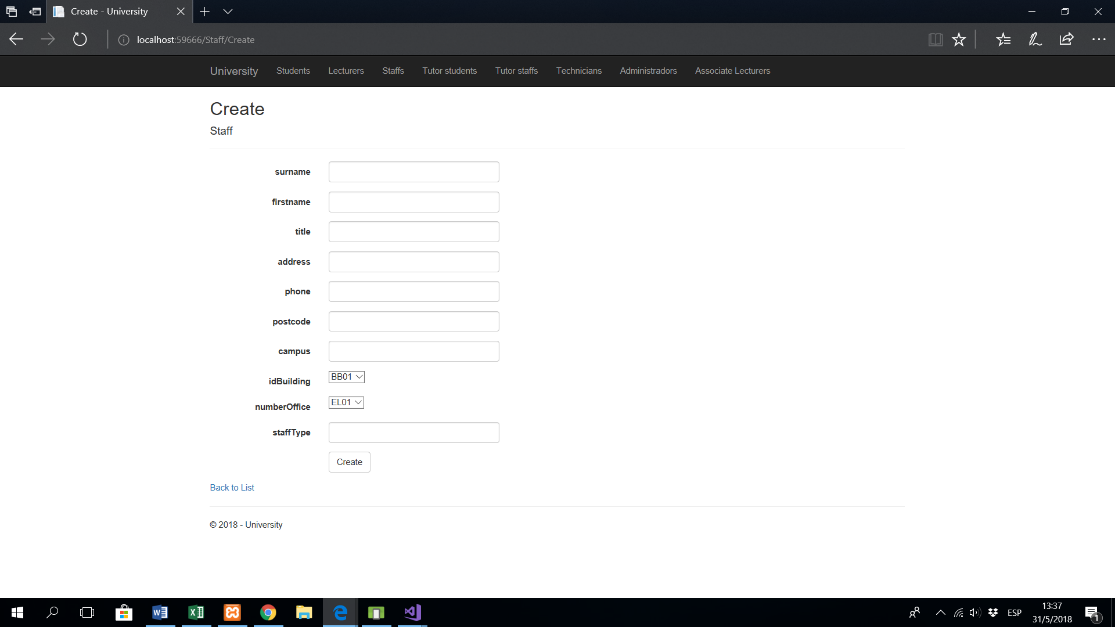
1. **Actualizar registro**

Al tocar el botón de editar este podrá actualizar los datos del registro en el sistema. Este podrá actualizar los registros, después de actualizarlos deberá tocar el botón de *save* para guardar los cambios en el sistema.



1. **Crear registro**

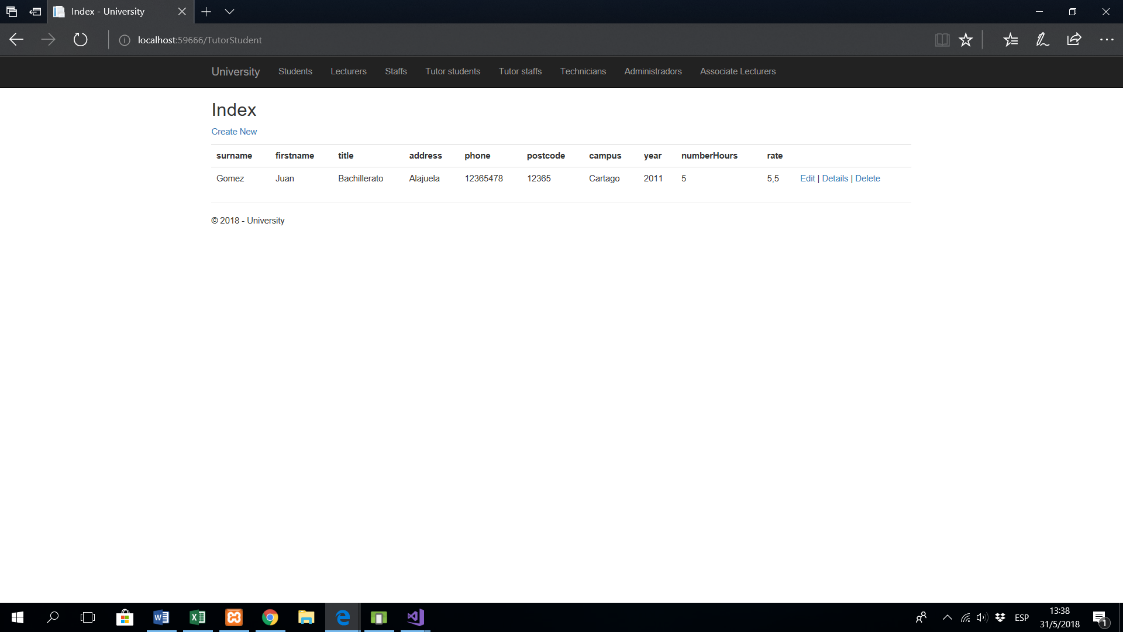
Al tocar *Create New*, se le mostrará los campos que debe llenar para poder registrar el registro en el sistema. AL terminar de llenar los datos deberá tocar el botón de *Create* para poder registrar el nuevo registro en el sistema.



* Sección Tutor Student

1. **Visualizar registros**

En esta sección se podrá visualizar los registros de los tutor student guardados en el sistema.



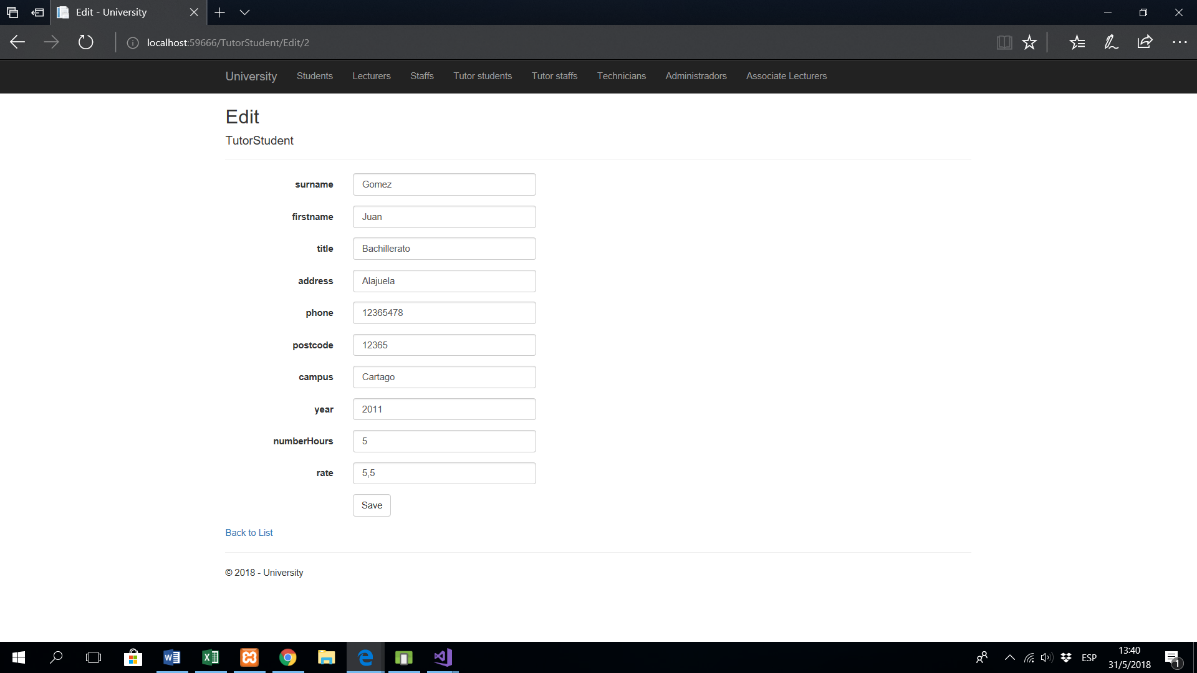
1. **Eliminar registro**

Después de hacer click en un registro en la opción de delete se le mostrará la siguiente vista para eliminar el registro seleccionado. Si está seguro de eliminar el registro debe tocar el botón *Delete* o sino *back To List* para regresar a la sección anterior.



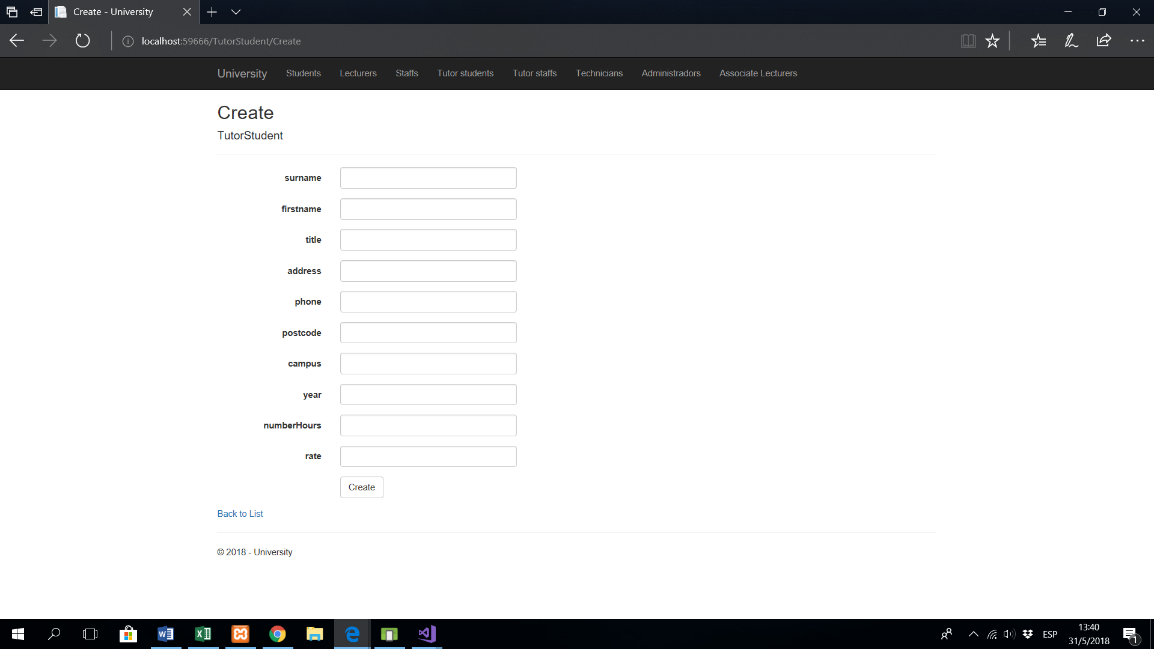
1. **Actualizar registro**

Al tocar el botón de editar este podrá actualizar los datos del registro en el sistema. Este podrá actualizar los registros, después de actualizarlos deberá tocar el botón de *save* para guardar los cambios en el sistema.



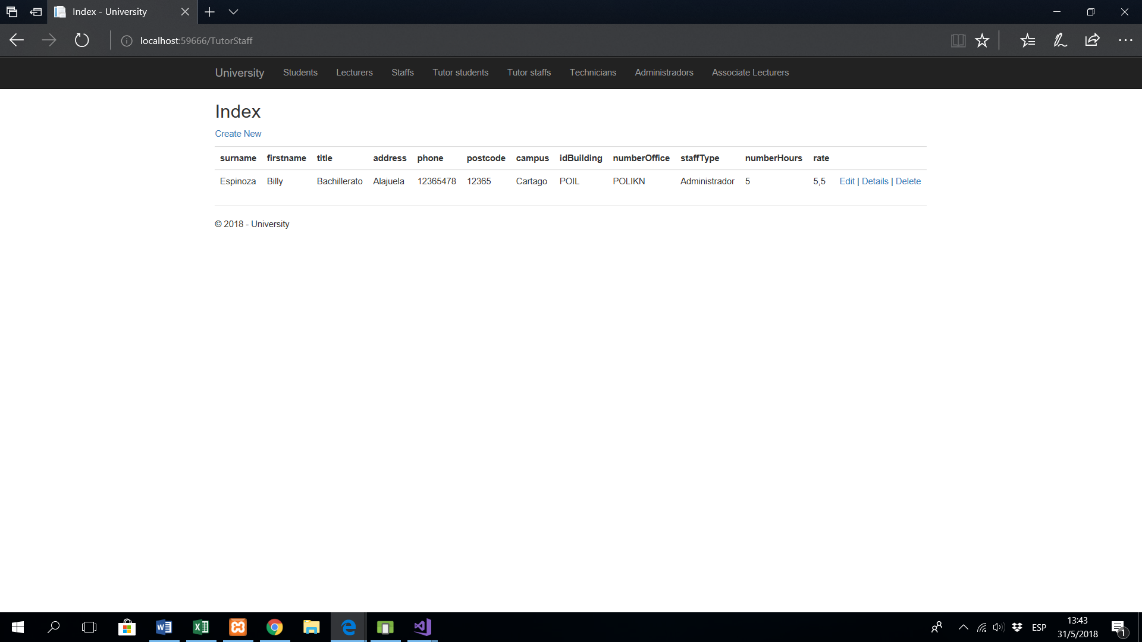
1. **Crear registro**

Al tocar *Create New*, se le mostrará los campos que debe llenar para poder registrar el registro en el sistema. AL terminar de llenar los datos deberá tocar el botón de *Create* para poder registrar el nuevo registro en el sistema.



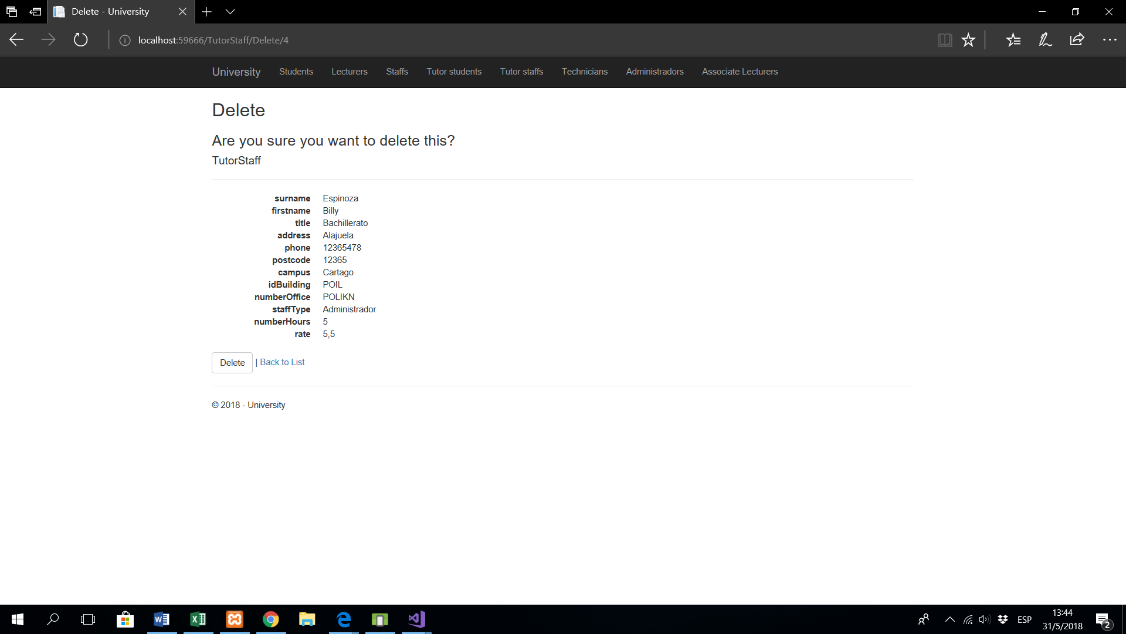
* Sección Tutor Staff

1. **Visualizar registros**

En esta sección se podrá visualizar los registros de los tutor staff guardados en el sistema.

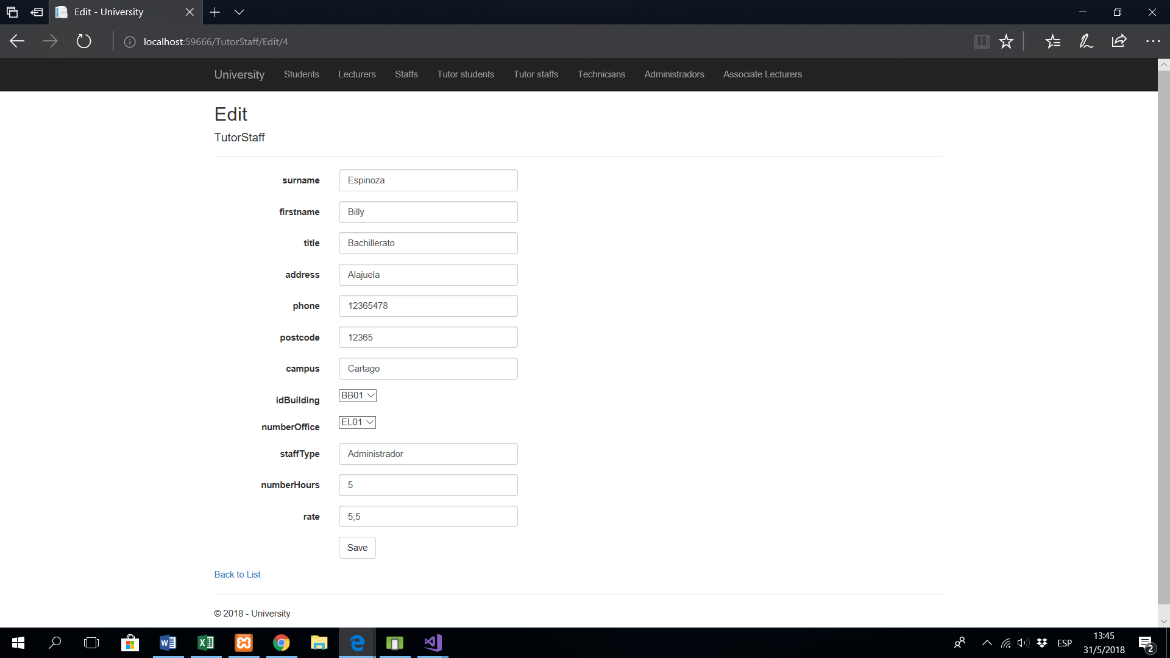
1. **Eliminar registro**

Después de hacer click en un registro en la opción de delete se le mostrará la siguiente vista para eliminar el registro seleccionado. Si está seguro de eliminar el registro debe tocar el botón *Delete* o sino *back To List* para regresar a la sección anterior.



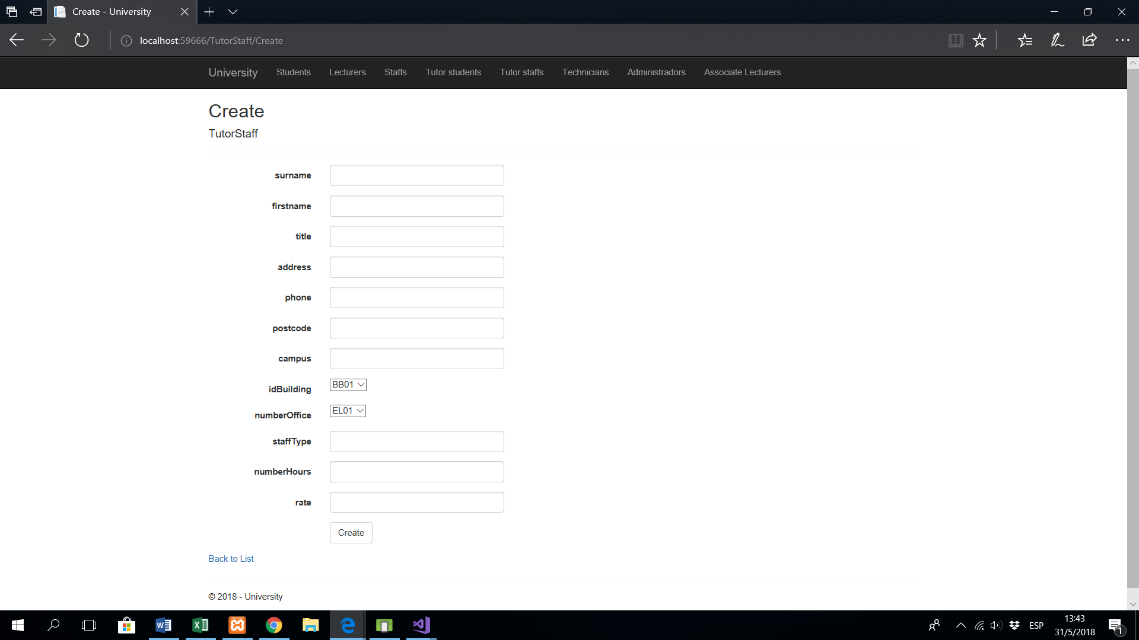
1. **Actualizar registro**

Al tocar el botón de editar este podrá actualizar los datos del registro en el sistema. Este podrá actualizar los registros, después de actualizarlos deberá tocar el botón de *save* para guardar los cambios en el sistema.



1. **Crear registro**

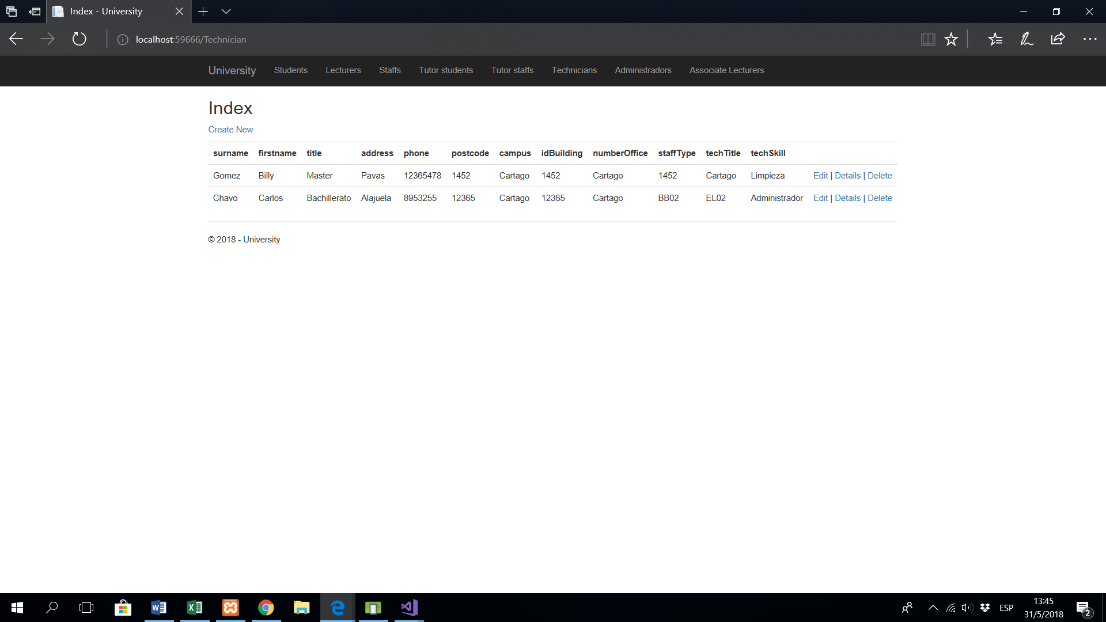
Al tocar *Create New*, se le mostrará los campos que debe llenar para poder registrar el registro en el sistema. AL terminar de llenar los datos deberá tocar el botón de *Create* para poder registrar el nuevo registro en el sistema.



* Sección Technicians

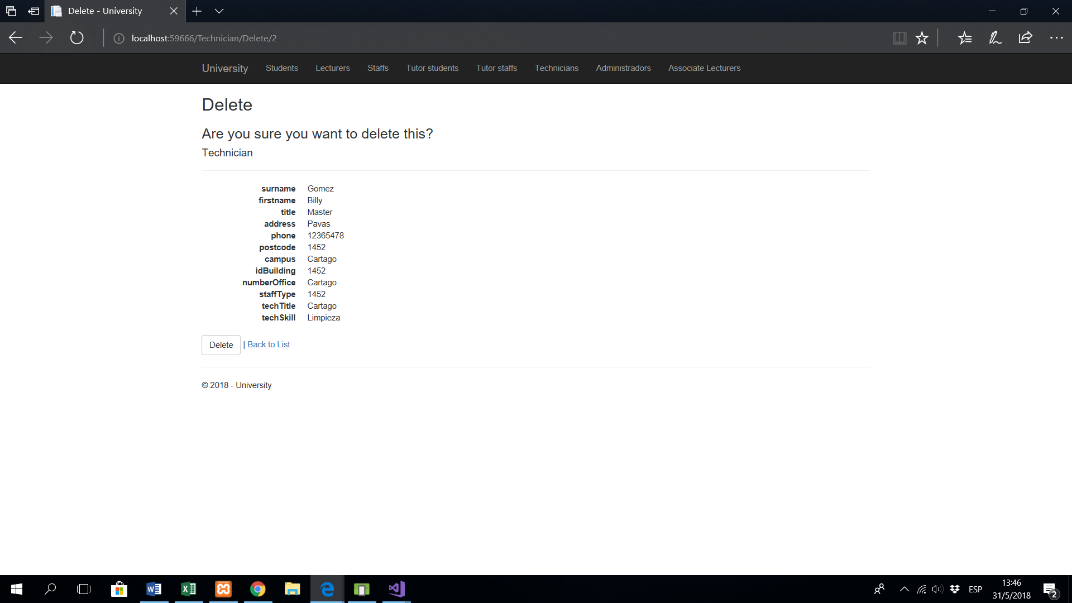
1. **Visualizar registros**

En esta sección se podrá visualizar los registros de los técnicos guardados en el sistema.



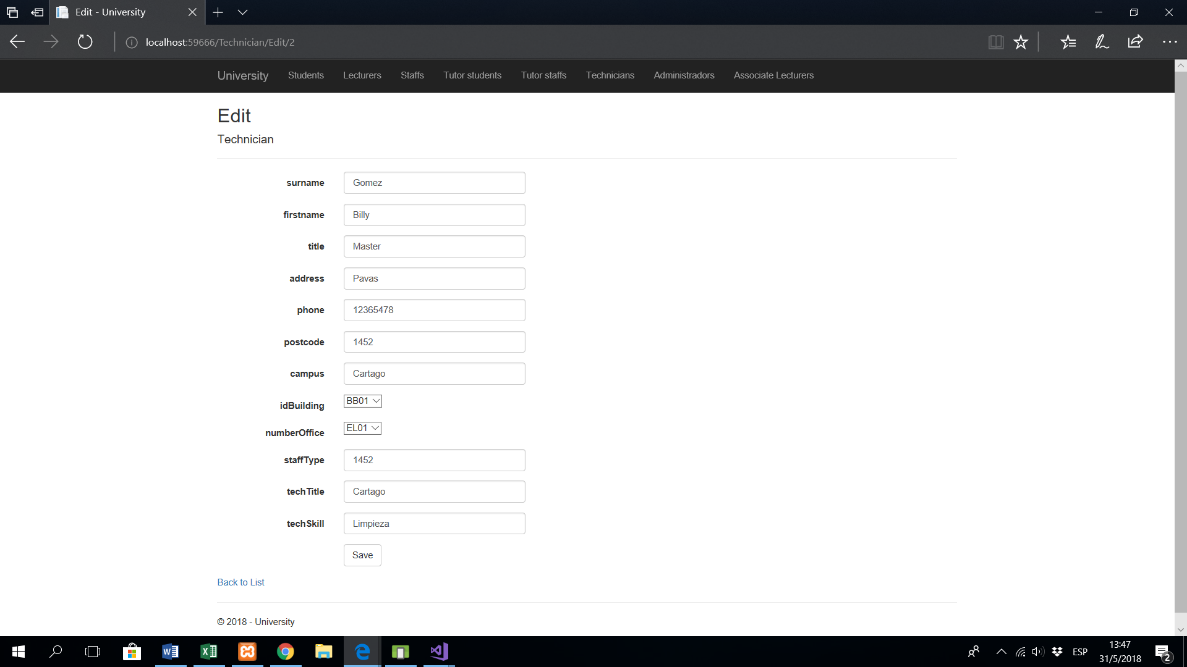
1. **Eliminar registro**

Después de hacer click en un registro en la opción de delete se le mostrará la siguiente vista para eliminar el registro seleccionado. Si está seguro de eliminar el registro debe tocar el botón *Delete* o sino *back To List* para regresar a la sección anterior.



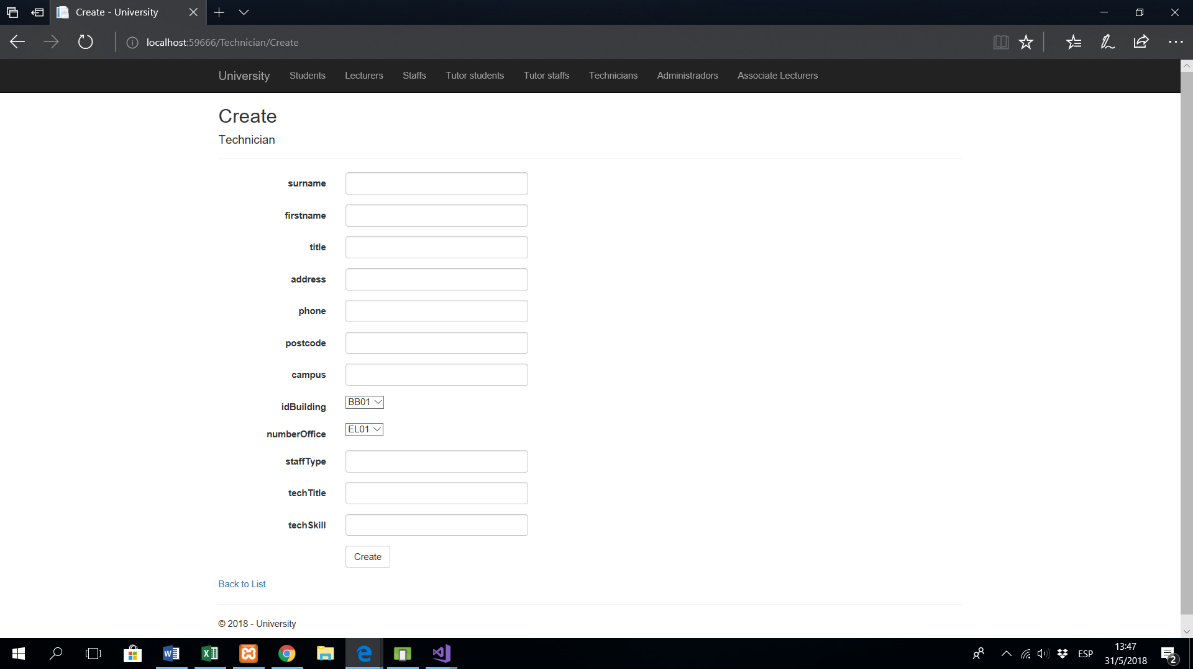
1. **Actualizar registro**

Al tocar el botón de editar este podrá actualizar los datos del registro en el sistema. Este podrá actualizar los registros, después de actualizarlos deberá tocar el botón de *save* para guardar los cambios en el sistema.



1. **Crear registro**

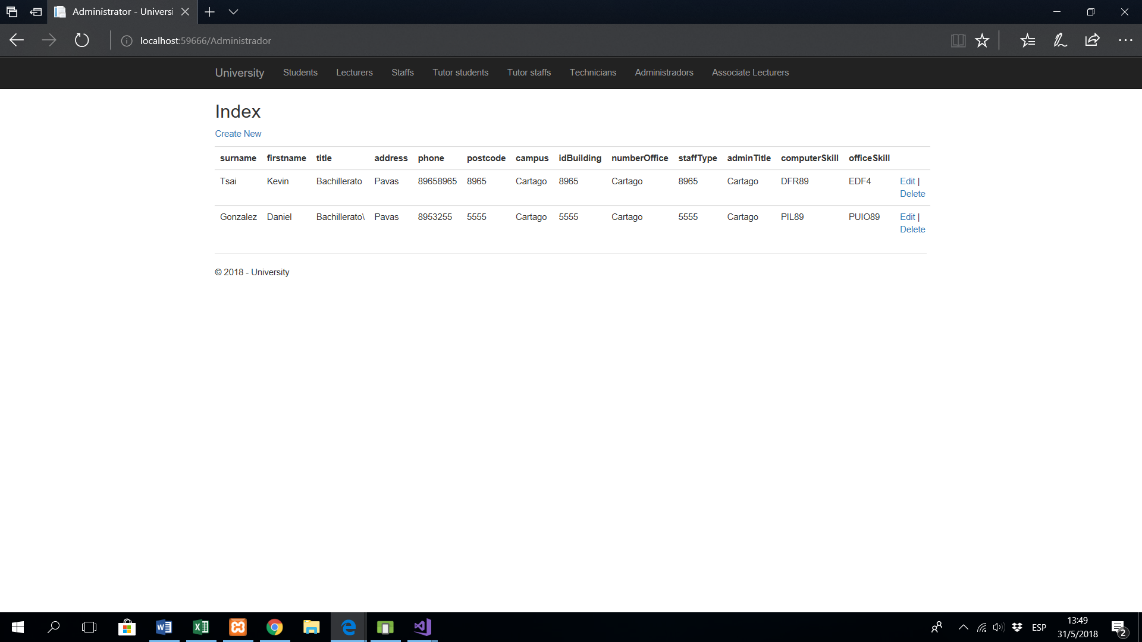
Al tocar *Create New*, se le mostrará los campos que debe llenar para poder registrar el registro en el sistema. AL terminar de llenar los datos deberá tocar el botón de *Create* para poder registrar el nuevo registro en el sistema.



* Sección Administrator

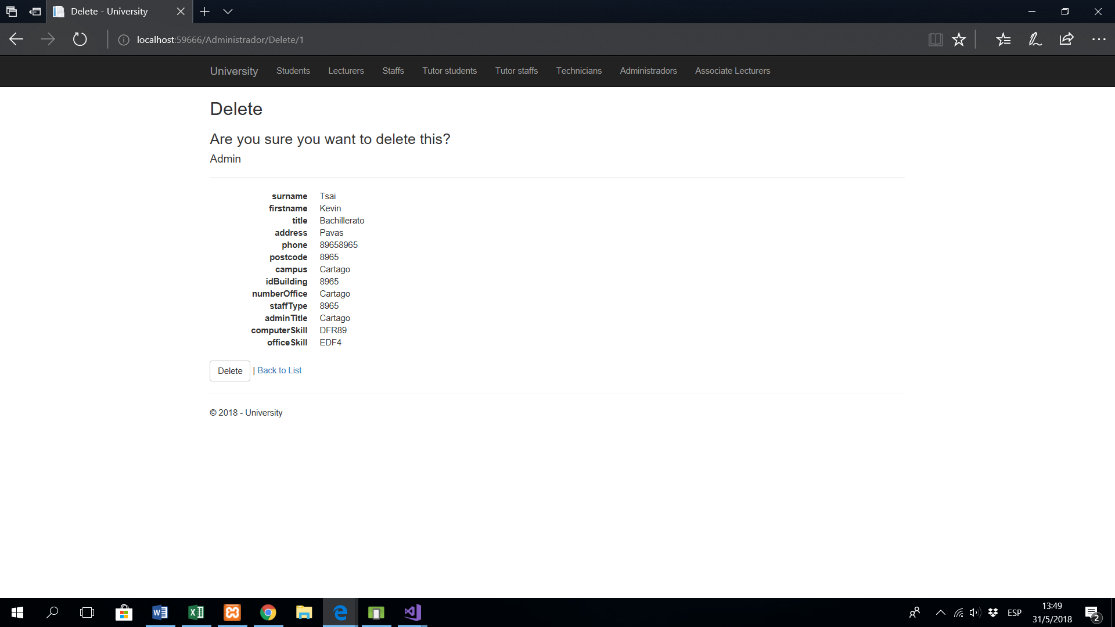
1. **Visualizar registros**

En esta sección se podrá visualizar los registros de los administradores guardados en el sistema.



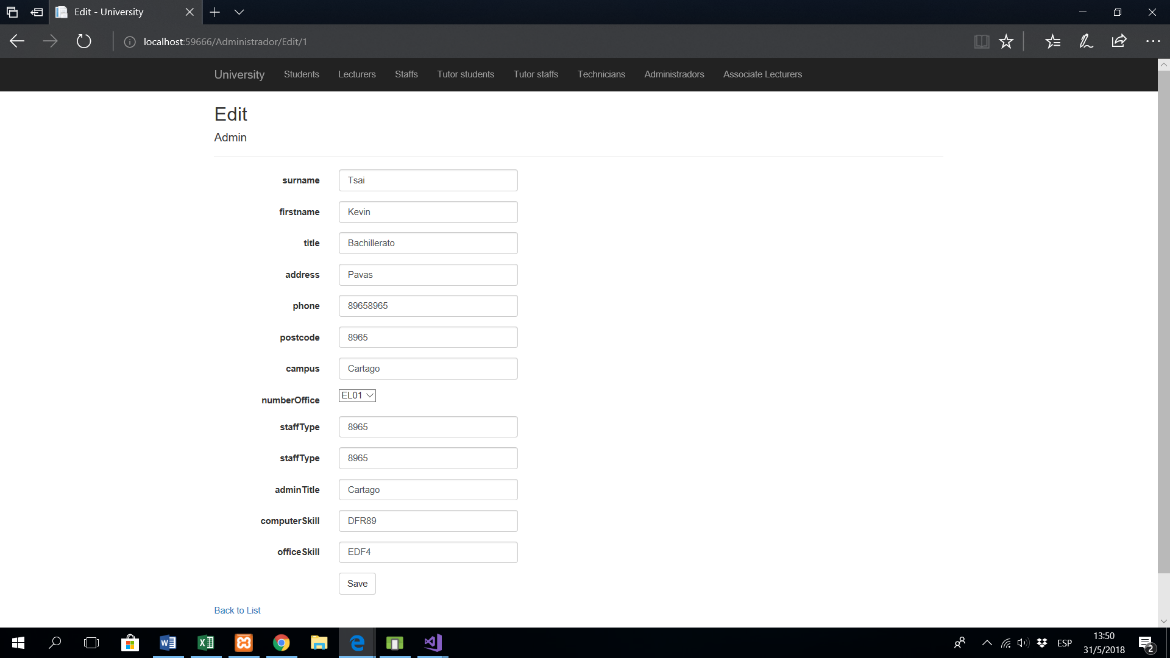
1. **Eliminar registro**

Después de hacer click en un registro en la opción de delete se le mostrará la siguiente vista para eliminar el registro seleccionado. Si está seguro de eliminar el registro debe tocar el botón *Delete* o sino *back To List* para regresar a la sección anterior.

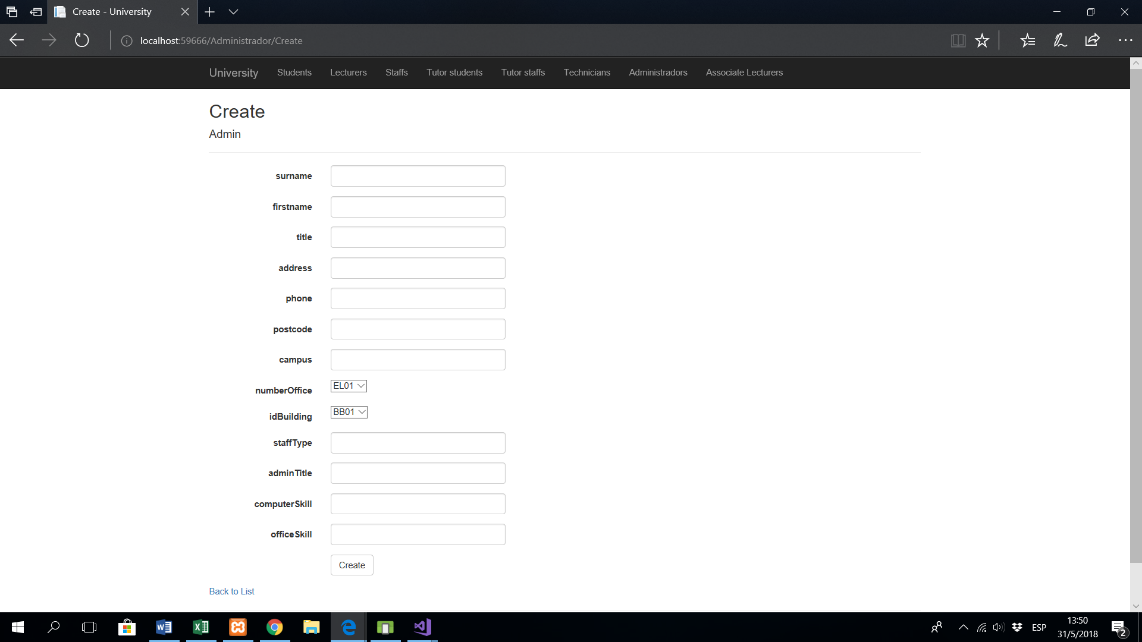


1. **Actualizar registro**

Al tocar el botón de editar este podrá actualizar los datos del registro en el sistema. Este podrá actualizar los registros, después de actualizarlos deberá tocar el botón de *save* para guardar los cambios en el sistema.



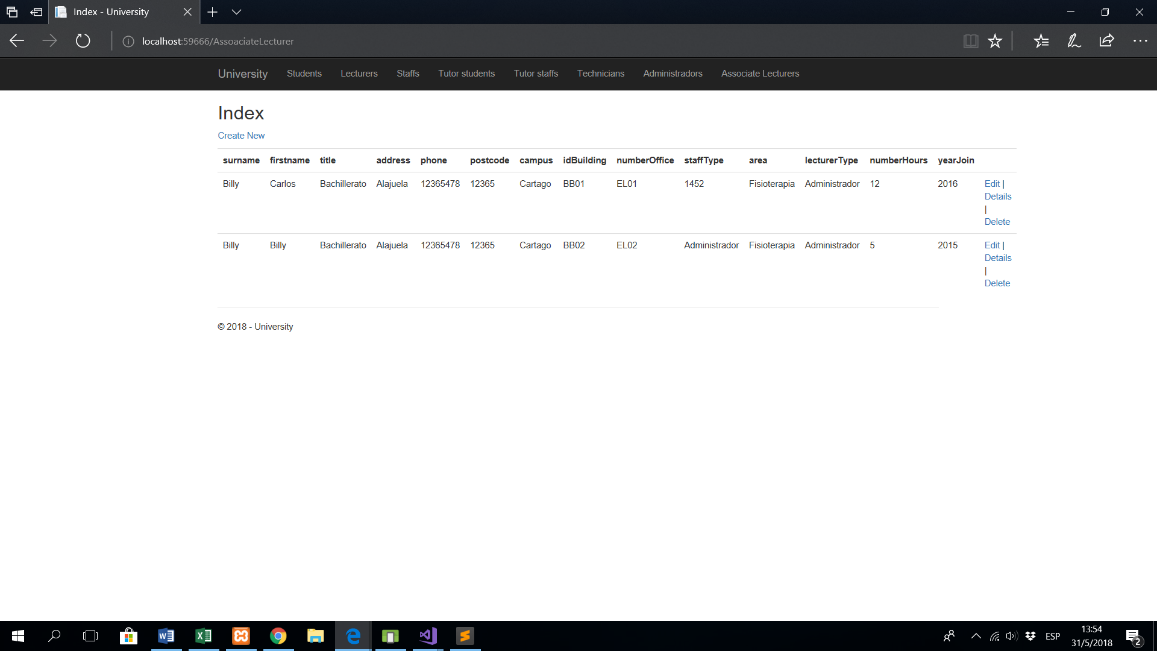
1. **Crear registro**

Al tocar *Create New*, se le mostrará los campos que debe llenar para poder registrar el registro en el sistema. AL terminar de llenar los datos deberá tocar el botón de *Create* para poder registrar el nuevo registro en el sistema.

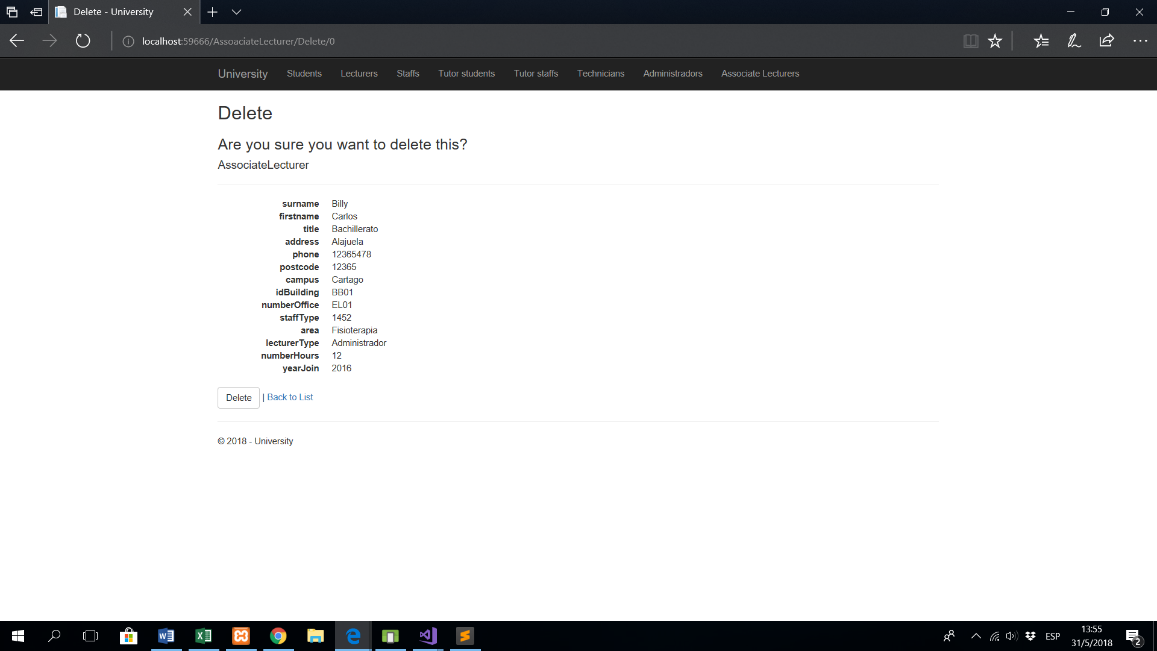
* Sección Profesores asociados

1. **Visualizar registros**

En esta sección se podrá visualizar los registros de los Profesores asociados guardados en el sistema.

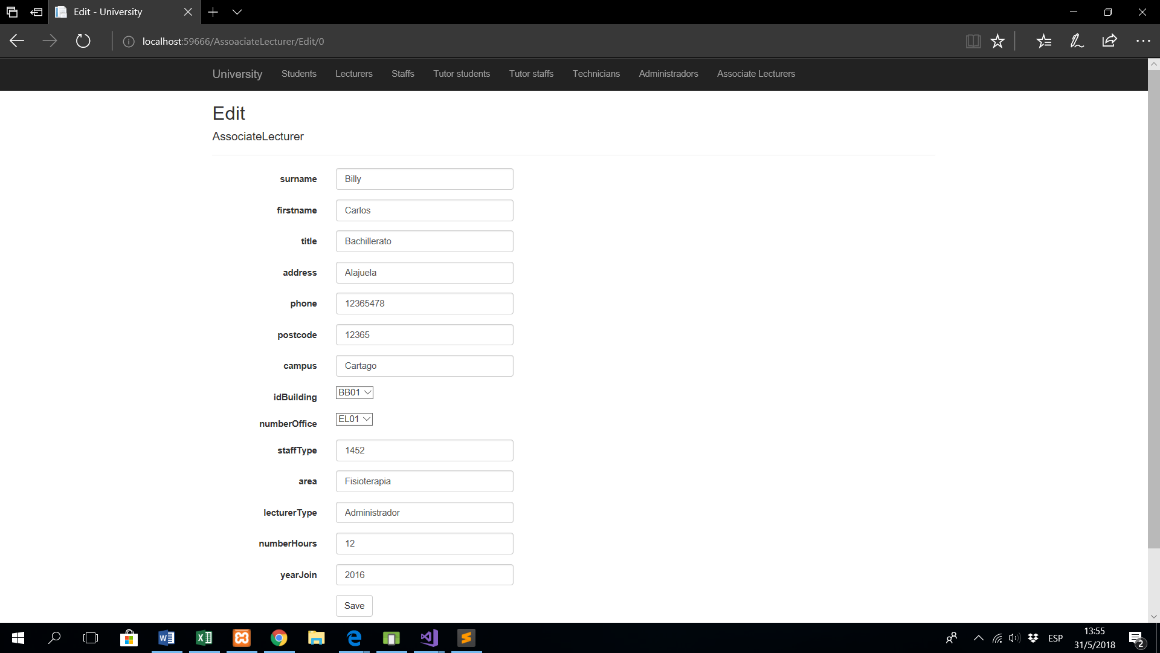


1. **Eliminar registro**

Después de hacer click en un registro en la opción de delete se le mostrará la siguiente vista para eliminar el registro seleccionado. Si está seguro de eliminar el registro debe tocar el botón *Delete* o sino *back To List* para regresar a la sección anterior.

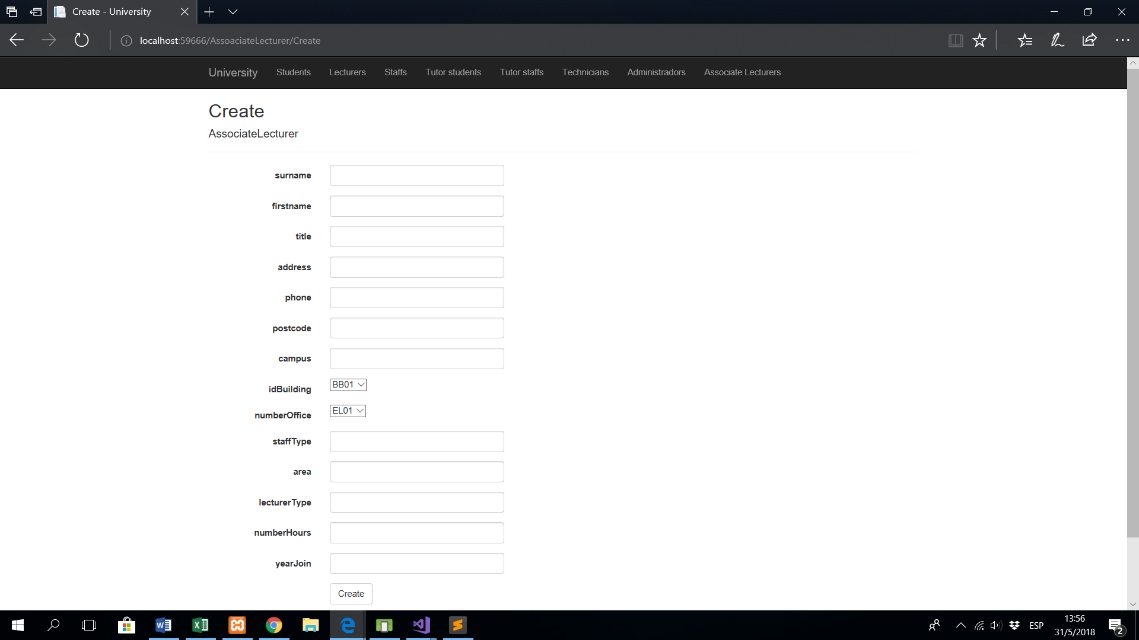
1. **Actualizar registro**

Al tocar el botón de editar este podrá actualizar los datos del registro en el sistema. Este podrá actualizar los registros, después de actualizarlos deberá tocar el botón de *save* para guardar los cambios en el sistema.



1. **Crear registro**

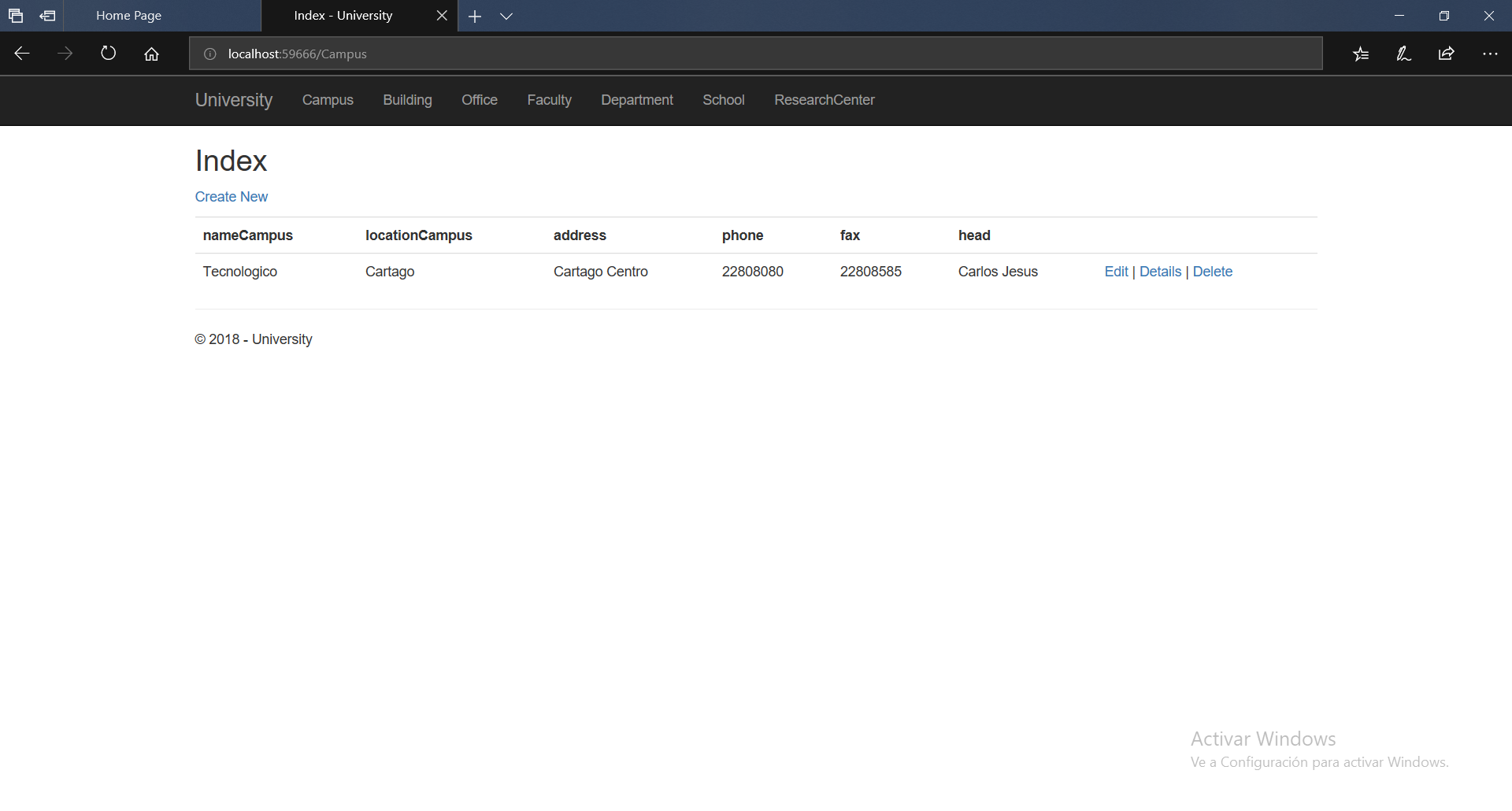
Al tocar *Create New*, se le mostrará los campos que debe llenar para poder registrar el registro en el sistema. AL terminar de llenar los datos deberá tocar el botón de *Create* para poder registrar el nuevo registro en el sistema.



* Sección Campus

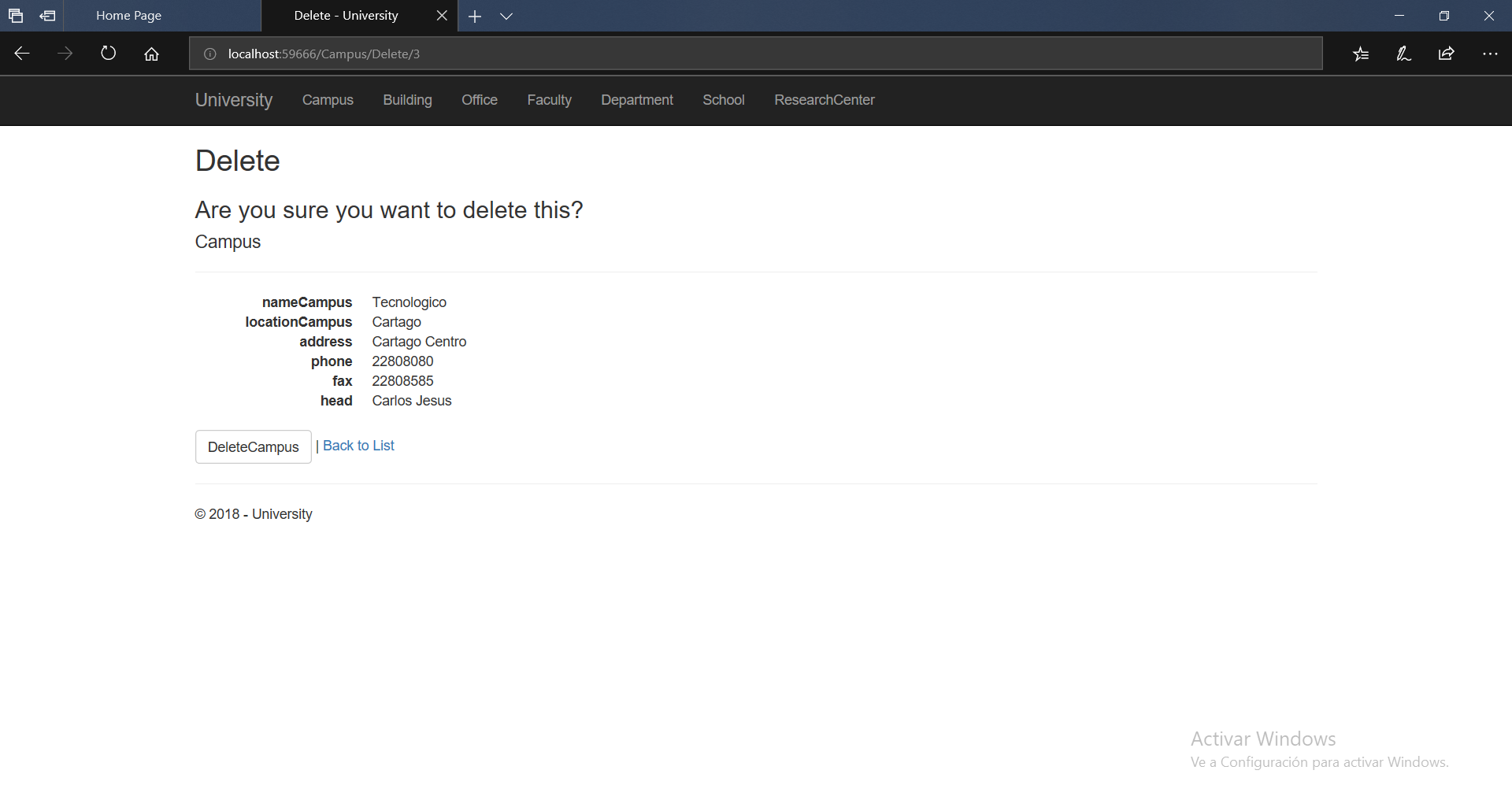
1. **Visualizar registro**

En esta sección se podrá visualizar los registros de los Campus guardados en el sistema.



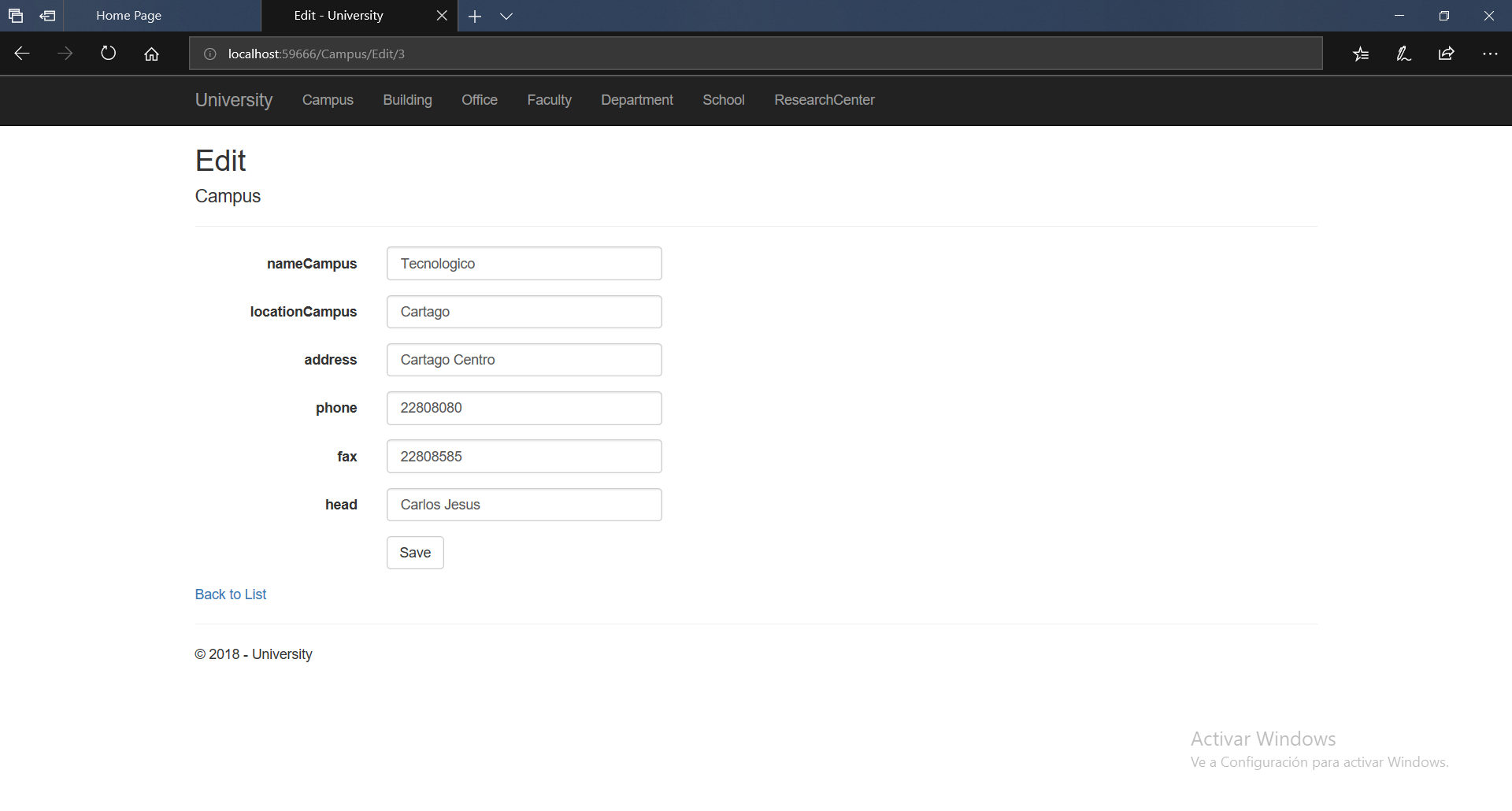
1. **Eliminar registro**

Después de hacer click en un registro en la opción de delete se le mostrará la siguiente vista para eliminar el registro seleccionado. Si está seguro de eliminar el registro debe tocar el botón *Delete* o sino *back To List* para regresar a la sección anterior.



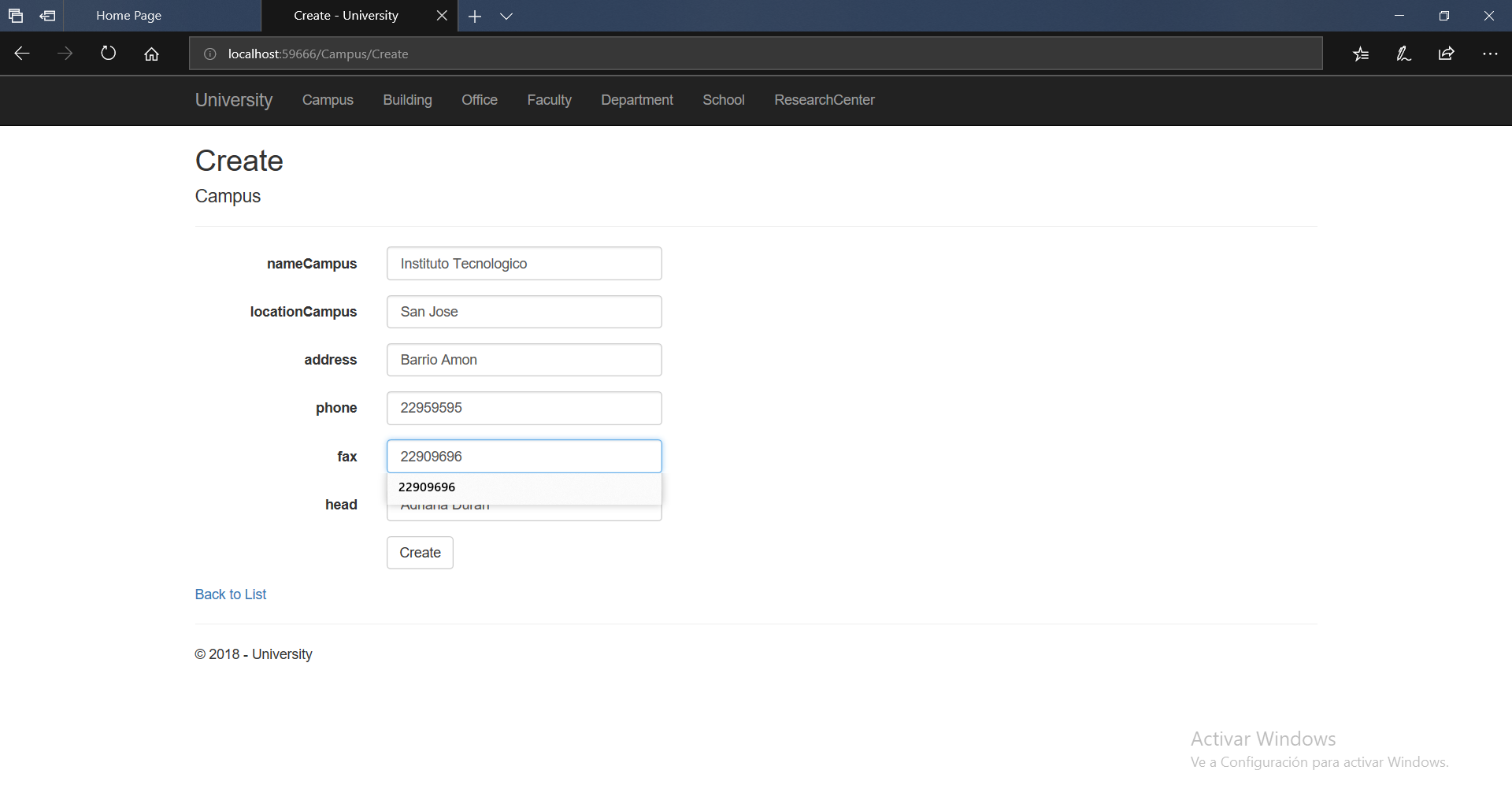
1. **Actualizar registro**

Al tocar el botón de editar este podrá actualizar los datos del registro en el sistema. Este podrá actualizar los registros, después de actualizarlos deberá tocar el botón de *save* para guardar los cambios en el sistema.



1. **Crear Registro**

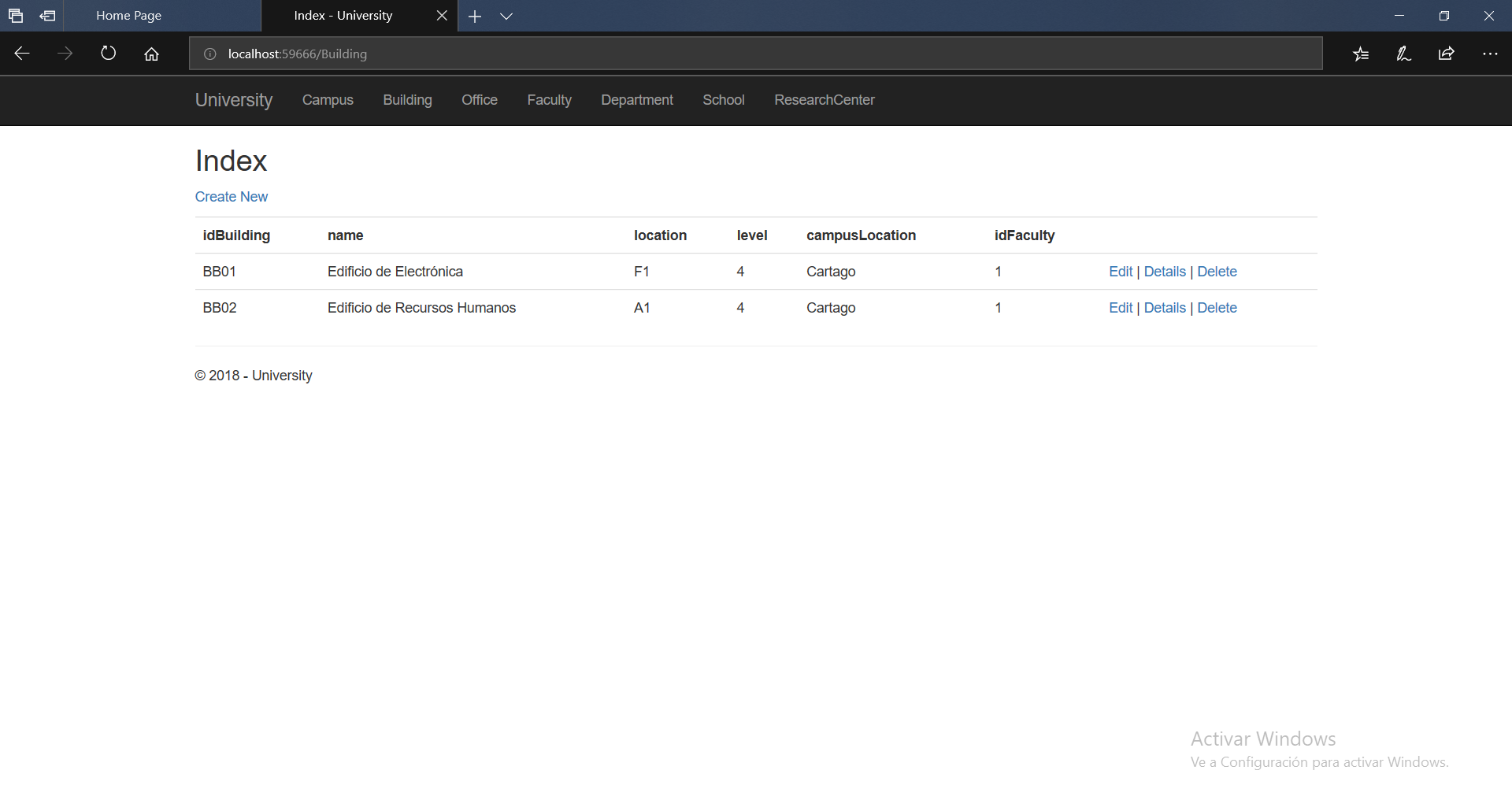
Al tocar *Create New*, se le mostrará los campos que debe llenar para poder registrar el registro en el sistema. Al terminar de llenar los datos deberá tocar el botón de *Create* para poder registrar el nuevo registro en el sistema.



* Sección Building

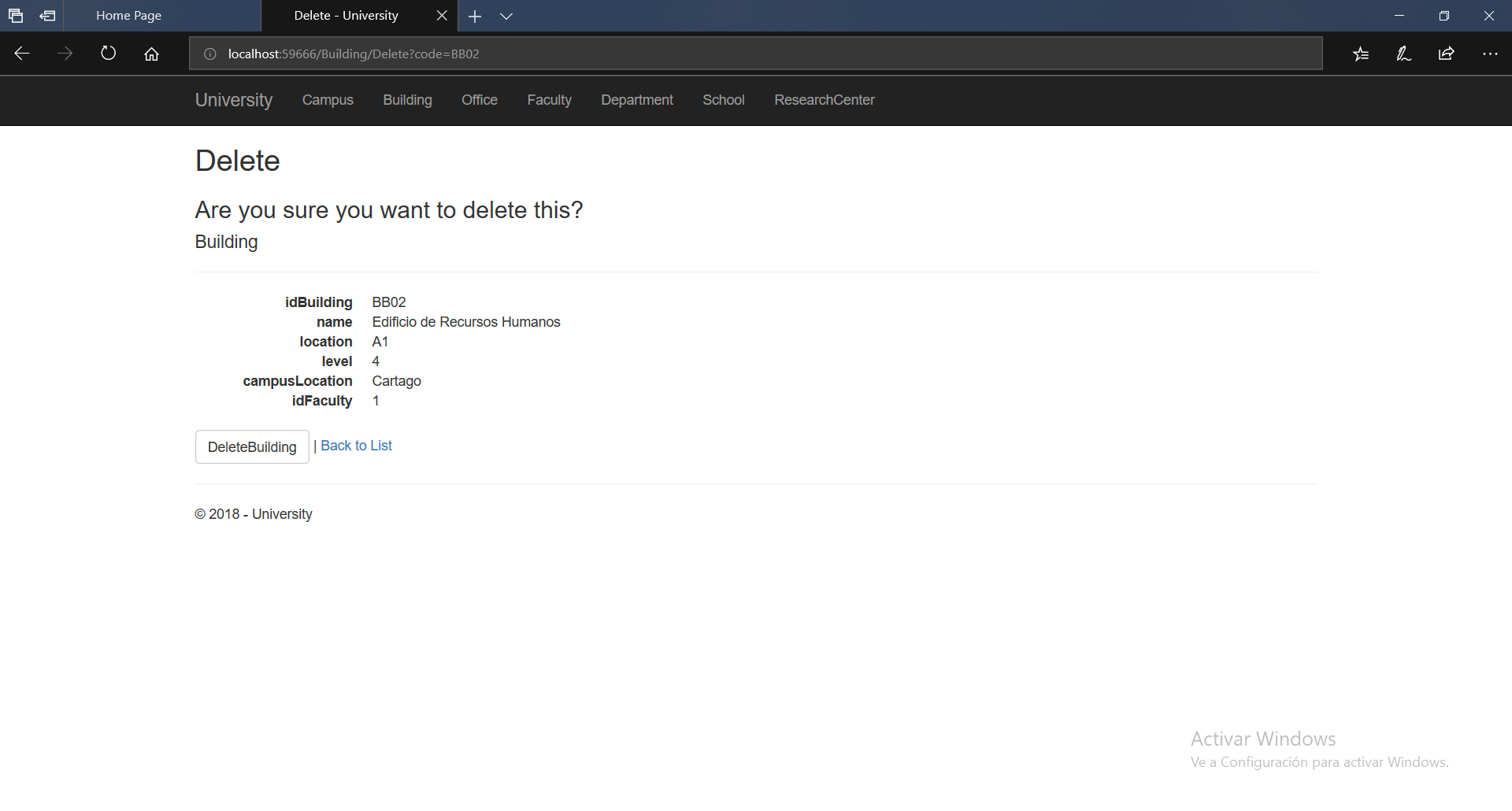
1. **Visualizar registro**

En esta sección se podrá visualizar los registros de los Edificios guardados en el sistema.



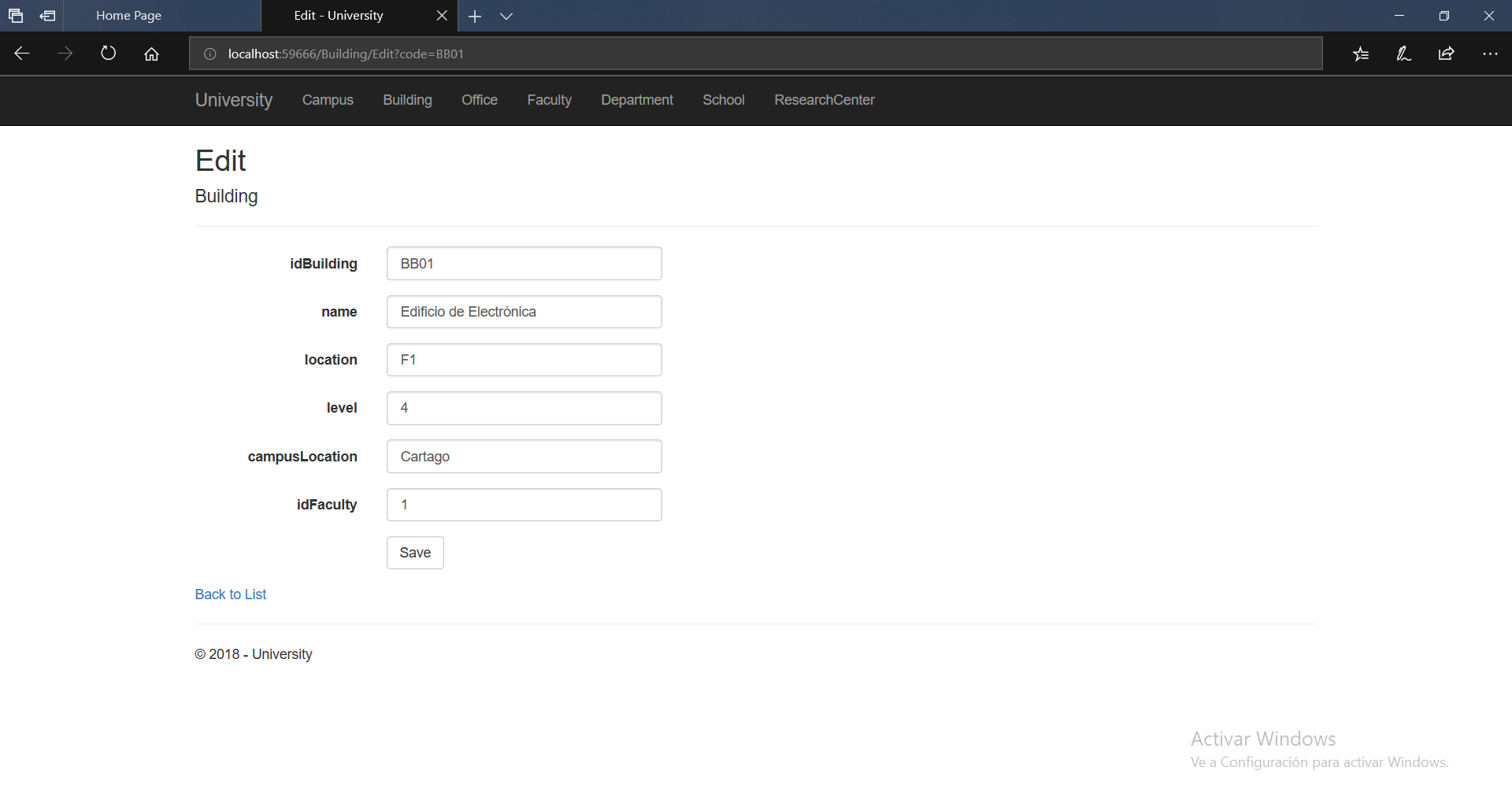
1. **Eliminar registro**

Después de hacer click en un registro en la opción de delete se le mostrará la siguiente vista para eliminar el registro seleccionado. Si está seguro de eliminar el registro debe tocar el botón *Delete* o sino *back To List* para regresar a la sección anterior.



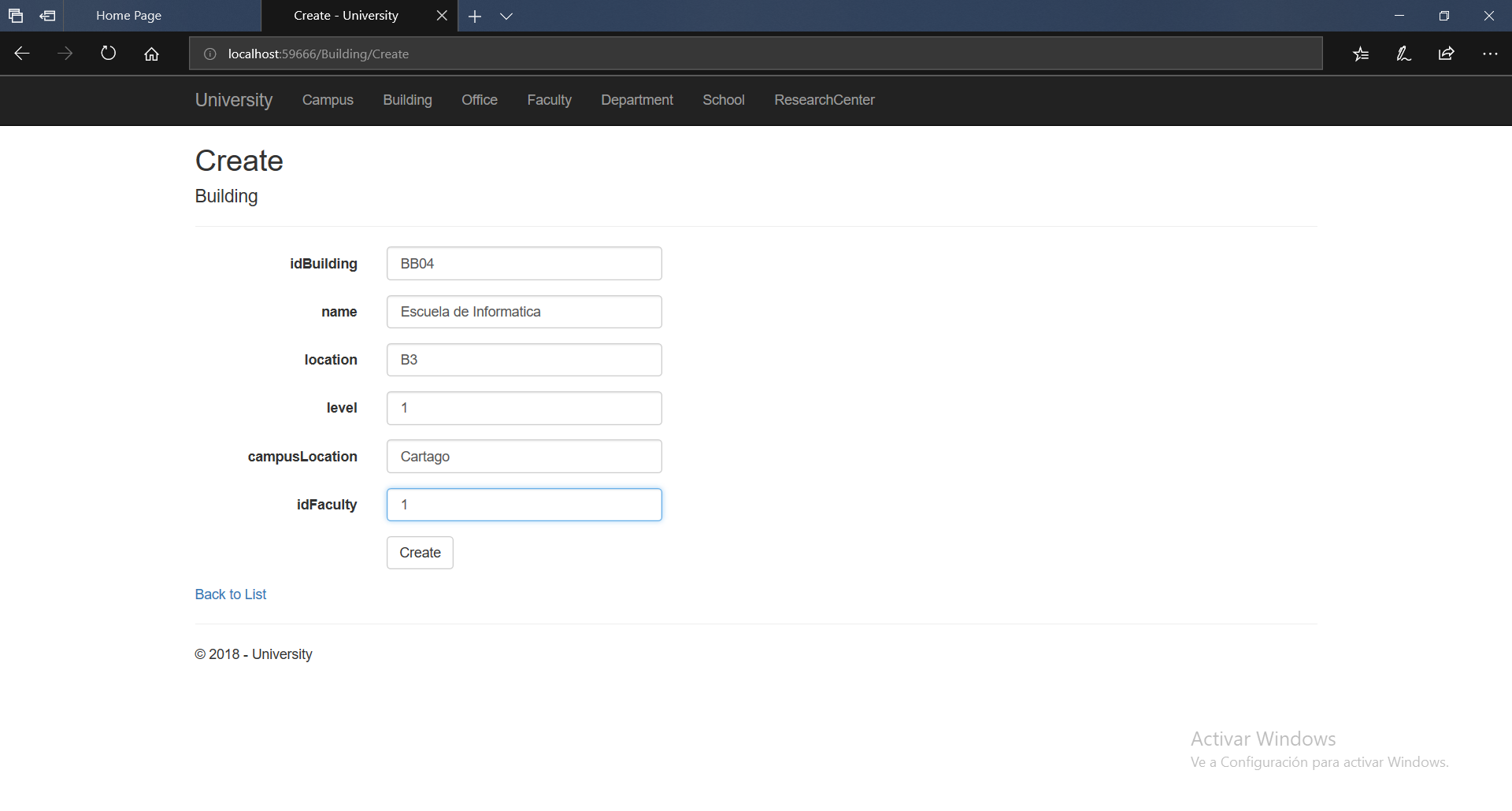
1. **Actualizar registro**

Al tocar el botón de editar este podrá actualizar los datos del registro en el sistema. Este podrá actualizar los registros, después de actualizarlos deberá tocar el botón de *save* para guardar los cambios en el sistema.



1. **Crear Registro**

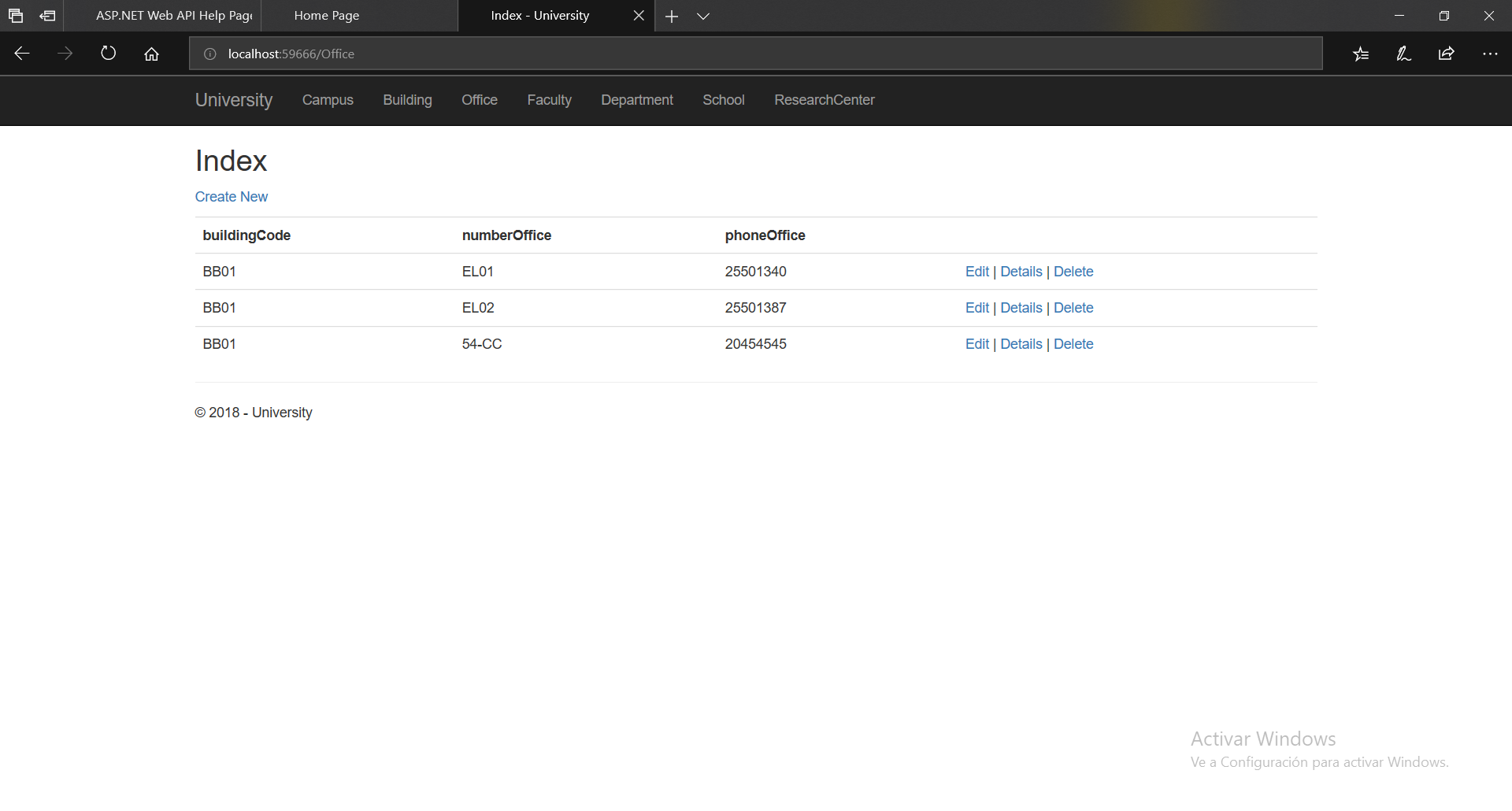
Al tocar *Create New*, se le mostrará los campos que debe llenar para poder registrar el registro en el sistema. Al terminar de llenar los datos deberá tocar el botón de *Create* para poder registrar el nuevo registro en el sistema.



* Sección Office

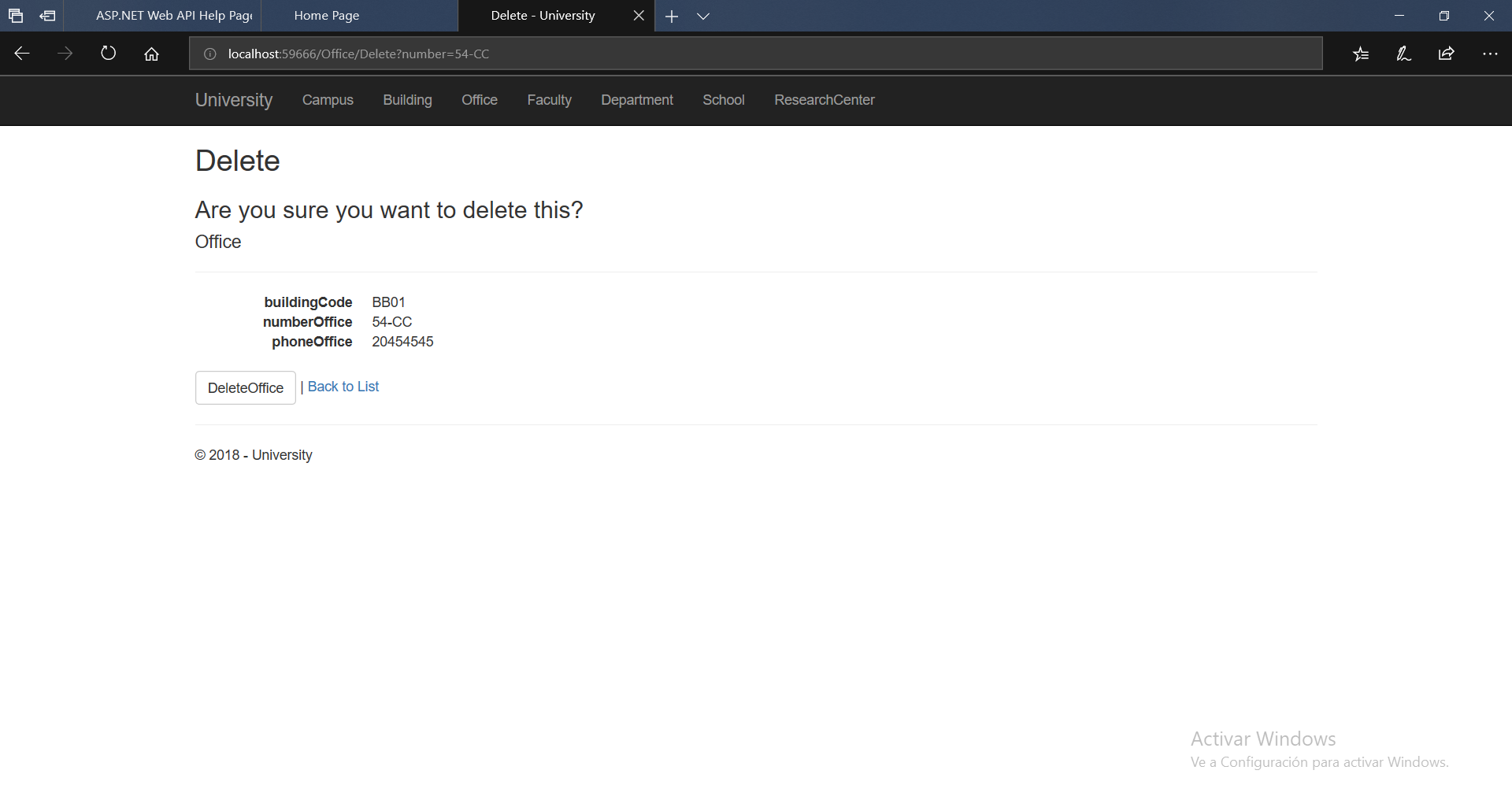
1. **Visualizar registro**

En esta sección se podrá visualizar los registros de las Oficinas guardadas en el sistema.



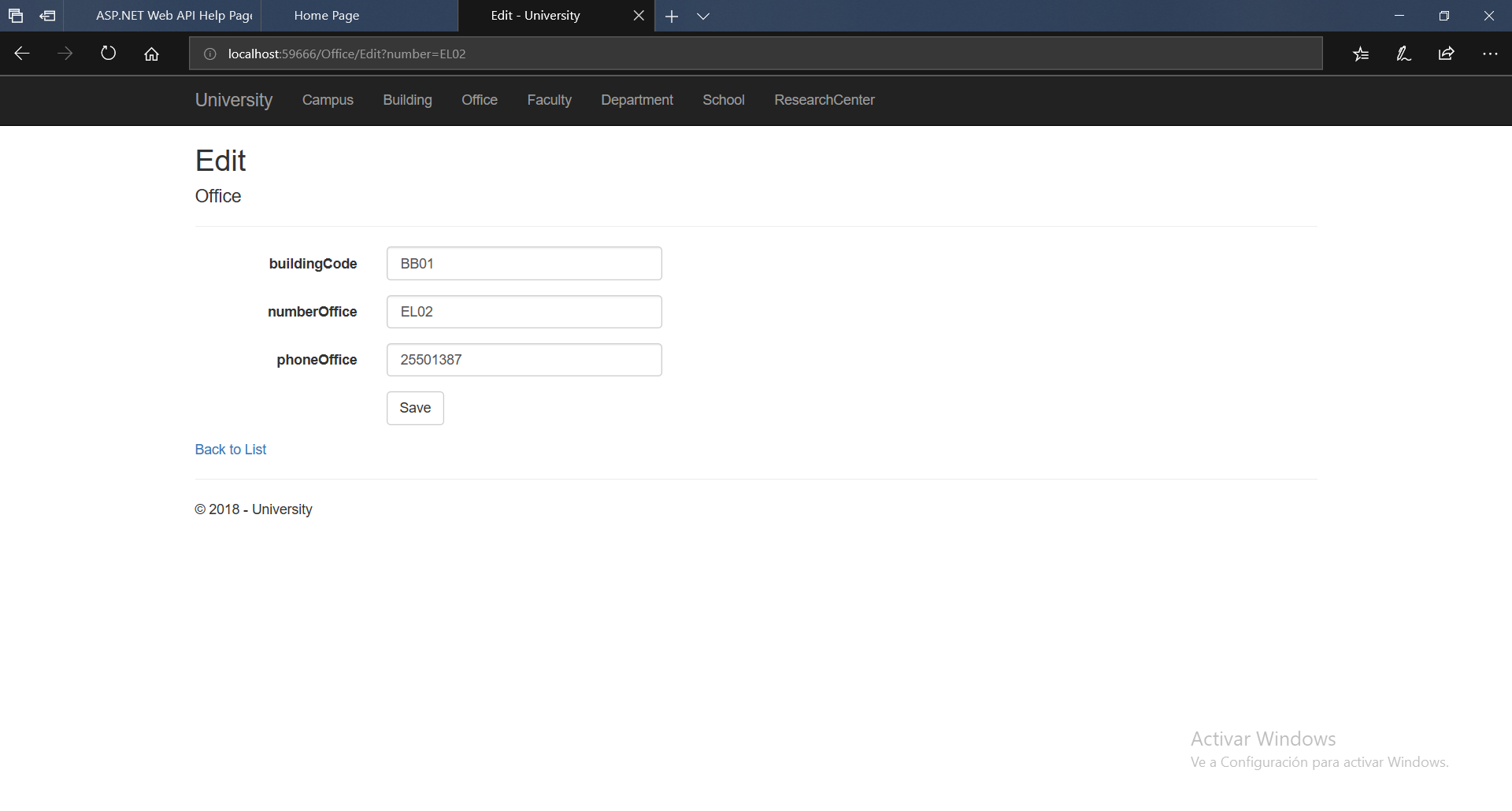
1. **Eliminar registro**

Después de hacer click en un registro en la opción de delete se le mostrará la siguiente vista para eliminar el registro seleccionado. Si está seguro de eliminar el registro debe tocar el botón *Delete* o sino *back To List* para regresar a la sección anterior.



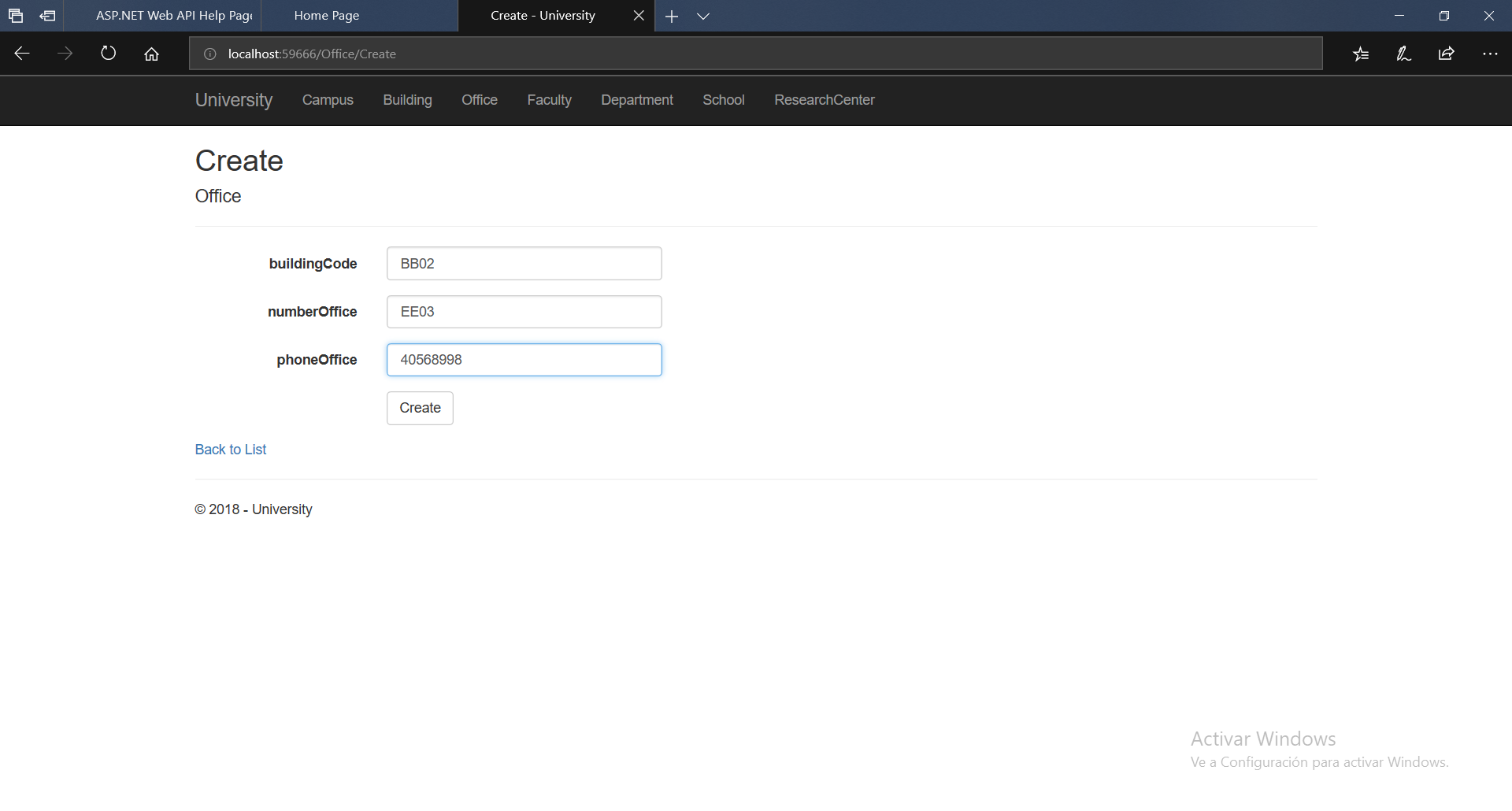
1. **Actualizar registro**

Al tocar el botón de editar este podrá actualizar los datos del registro en el sistema. Este podrá actualizar los registros, después de actualizarlos deberá tocar el botón de *save* para guardar los cambios en el sistema.



1. **Crear Registro**

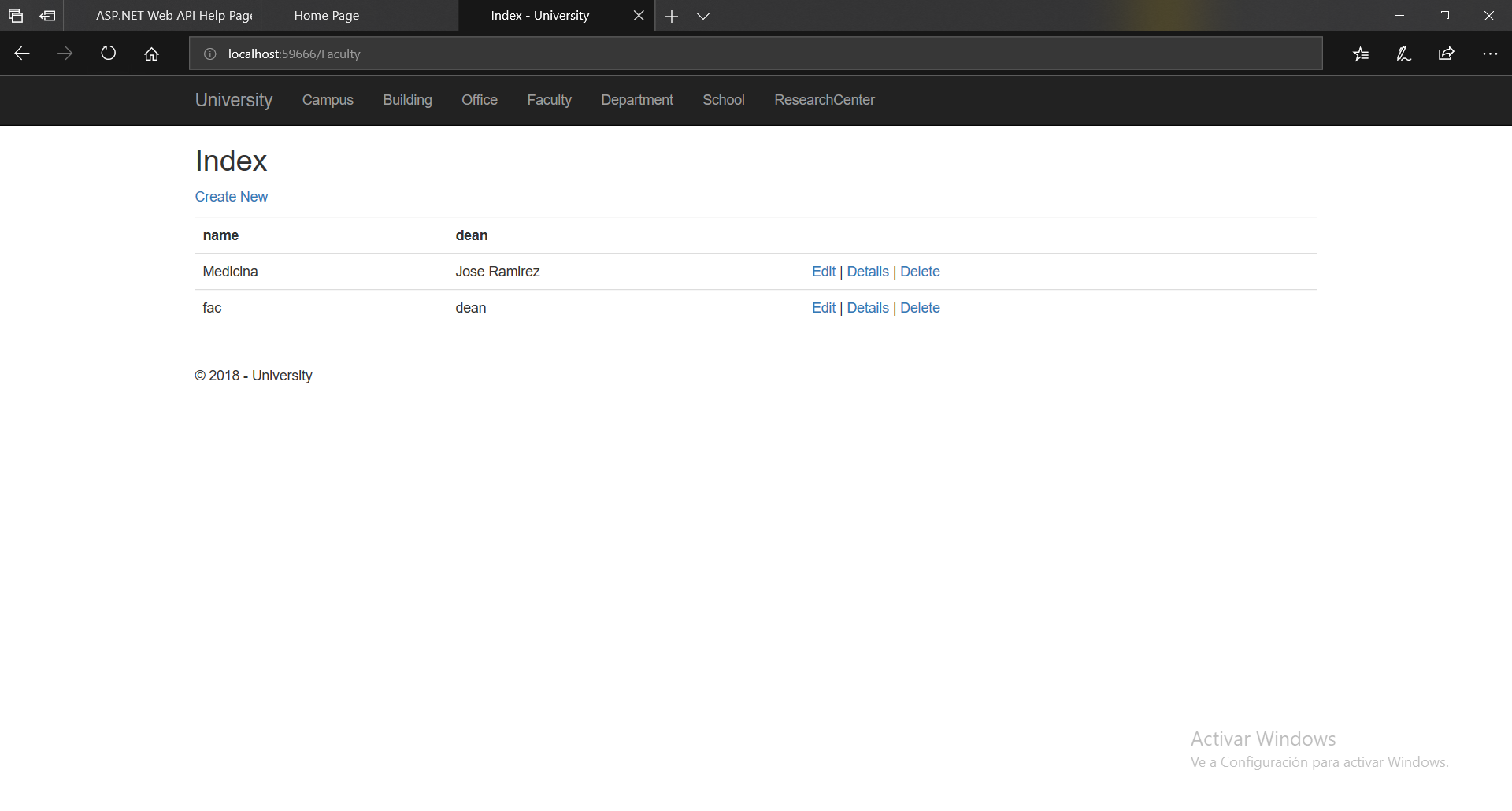
Al tocar *Create New*, se le mostrará los campos que debe llenar para poder registrar el registro en el sistema. Al terminar de llenar los datos deberá tocar el botón de *Create* para poder registrar el nuevo registro en el sistema.



* Sección Faculty

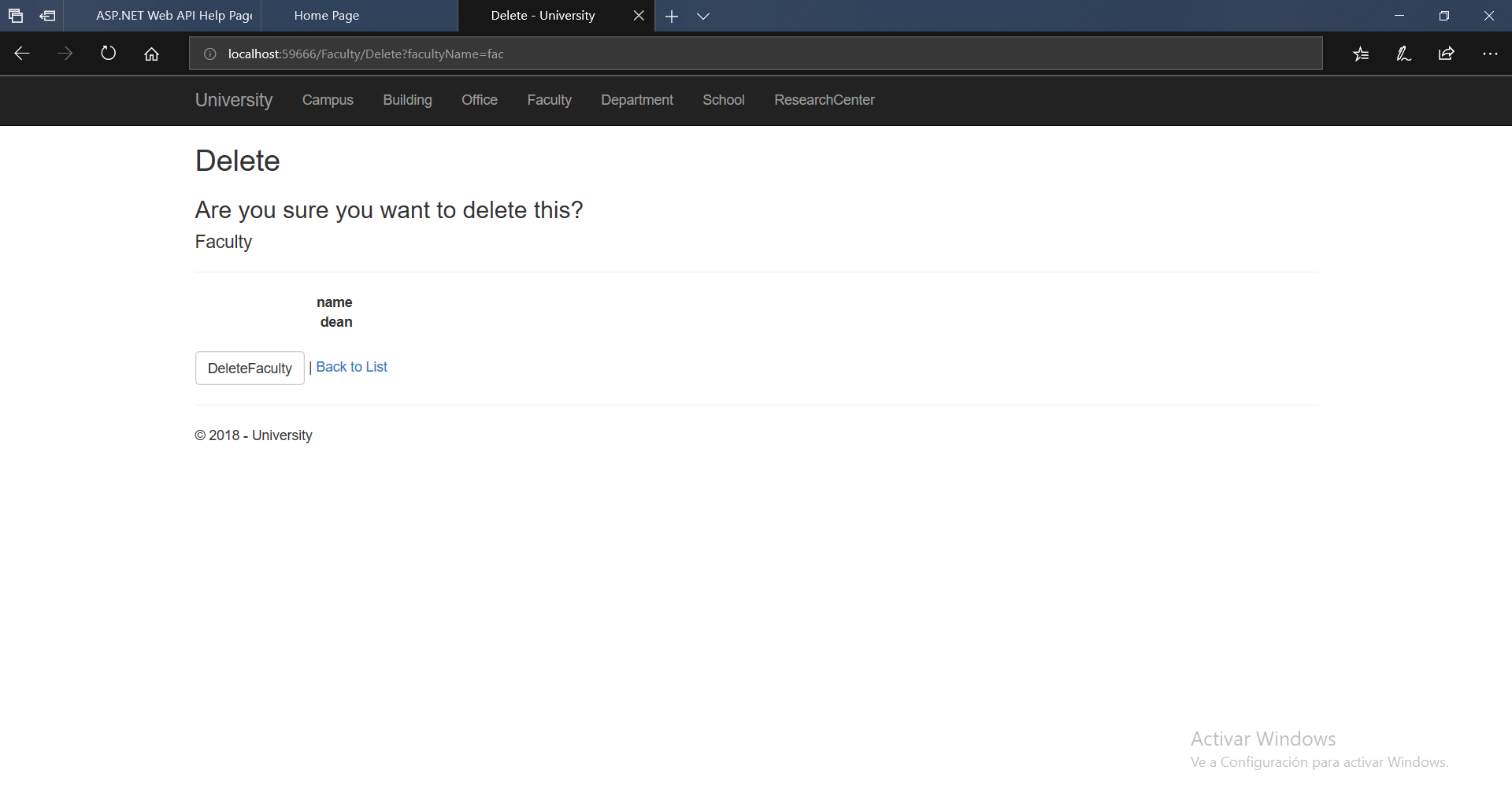
1. **Visualizar registro**

En esta sección se podrá visualizar los registros de las Facultades guardadas en el sistema.



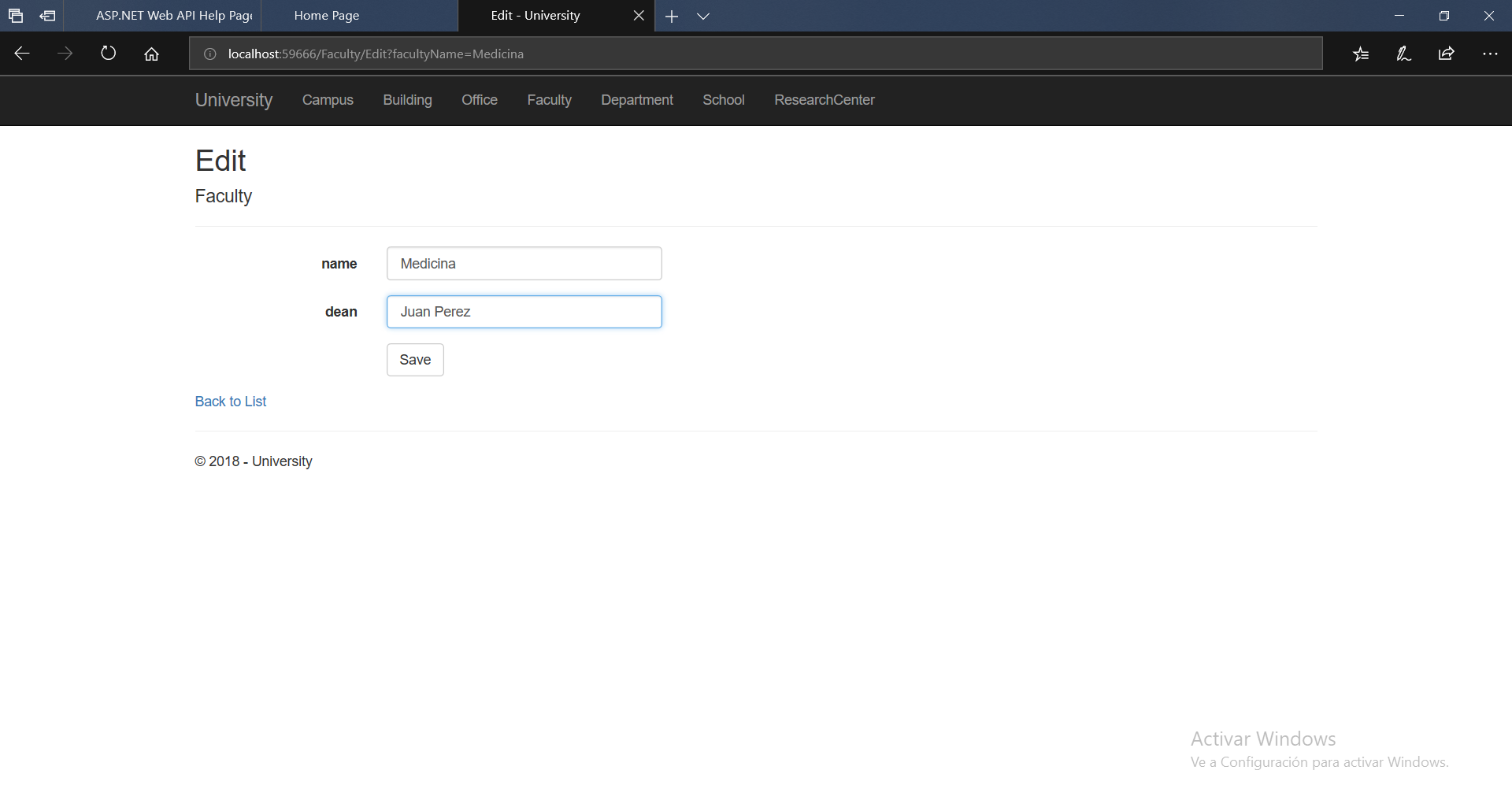
1. **Eliminar registro**

Después de hacer click en un registro en la opción de delete se le mostrará la siguiente vista para eliminar el registro seleccionado. Si está seguro de eliminar el registro debe tocar el botón *Delete* o sino *back To List* para regresar a la sección anterior.



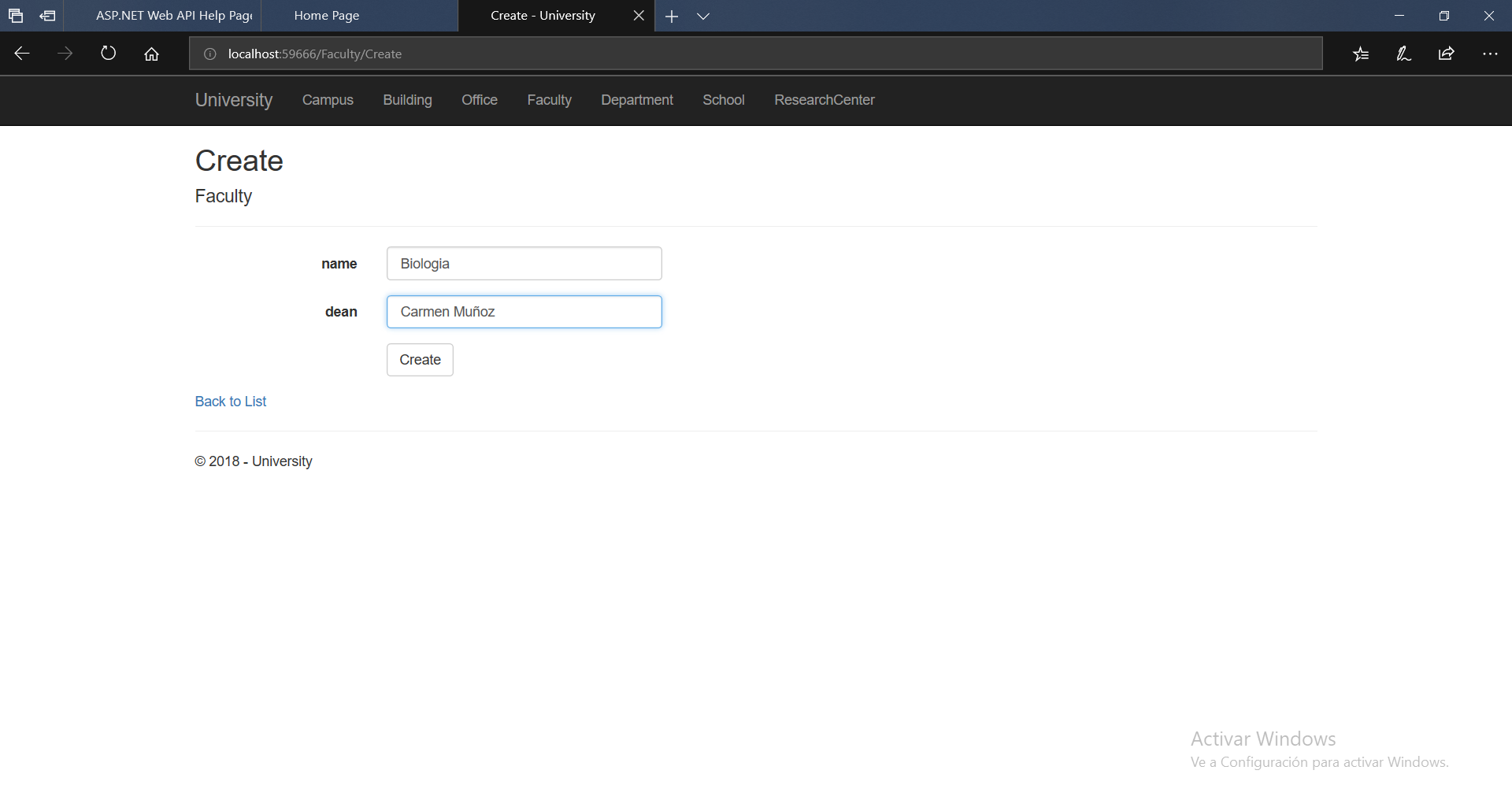
1. **Actualizar registro**

Al tocar el botón de editar este podrá actualizar los datos del registro en el sistema. Este podrá actualizar los registros, después de actualizarlos deberá tocar el botón de *save* para guardar los cambios en el sistema.



1. **Crear Registro**

Al tocar *Create New*, se le mostrará los campos que debe llenar para poder registrar el registro en el sistema. Al terminar de llenar los datos deberá tocar el botón de *Create* para poder registrar el nuevo registro en el sistema.



**Documentación del web service**

|  |  |  |
| --- | --- | --- |
| Admin |  |  |
| API | **Description** | **Example** |
| [GET api/Admin](http://localhost:62898/Help/Api/GET-api-Admin) | Método que recupera todos los datos de Admin de la base de datos. | <http://localhost:62898/api/Admin> |
| [POST api/Admin](http://localhost:62898/Help/Api/POST-api-Admin) | Método POST, que crea el Admin en la base de datos. | <http://localhost:62898/api/Admin> |
| [PUT api/Admin/{id}](http://localhost:62898/Help/Api/PUT-api-Admin-id) | Método que actualiza el Admin en la base de datos. | <http://localhost:62898/api/Admin/6> |
| [DELETE api/Admin/{id}](http://localhost:62898/Help/Api/DELETE-api-Admin-id) | Método que elimina el Admin en la base de datos. | <http://localhost:62898/api/Admin/5> |

|  |  |  |
| --- | --- | --- |
| AssociateLecturer |  |  |
| API | **Description** | **Example** |
| [GET api/AssociateLecturer](http://localhost:62898/Help/Api/GET-api-Admin) | Método que recupera todos los datos de AssociateLecturer de la base de datos. | <http://localhost:62898/api/AssociateLecturer> |
| [POST api/AssociateLecturer](http://localhost:62898/Help/Api/POST-api-Admin) | Método POST, que crea el AssociateLecturer en la base de datos. | <http://localhost:62898/api/AssociateLecturer> |
| [PUT api/AssociateLecturer/{id}](http://localhost:62898/Help/Api/PUT-api-Admin-id) | Método que actualiza el AssociateLecturer en la base de datos. | <http://localhost:62898/api/AssociateLecturer/6> |
| [DELETE api/AssociateLecturer/{id}](http://localhost:62898/Help/Api/DELETE-api-Admin-id) | Método que elimina el AssociateLecturer en la base de datos. | <http://localhost:62898/api/AssociateLecturer/5> |

|  |  |  |
| --- | --- | --- |
| SeniorLecturer |  |  |
| API | **Description** | **Example** |
| [GET api/SeniorLecturer](http://localhost:62898/Help/Api/GET-api-Admin) | Método que recupera todos los datos de SeniorLecturer de la base de datos. | [http://localhost:62898/api/SeniorLecturer](http://localhost:62898/api/AssociateLecturer) |
| [POST api/SeniorLecturer](http://localhost:62898/Help/Api/POST-api-Admin) | Método POST, que crea el SeniorLecturer en la base de datos. | [http://localhost:62898/api/SeniorLecturer](http://localhost:62898/api/AssociateLecturer) |
| [PUT api/SeniorLecturer/{id}](http://localhost:62898/Help/Api/PUT-api-Admin-id) | Método que actualiza el SeniorLecturer en la base de datos. | <http://localhost:62898/api/SeniorLecturer/6> |
| [DELETE api/SeniorLecturer/{id}](http://localhost:62898/Help/Api/DELETE-api-Admin-id) | Método que elimina el SeniorLecturer en la base de datos. | <http://localhost:62898/api/SeniorLecturer/5> |

|  |  |  |
| --- | --- | --- |
| TutorStaff |  |  |
| API | **Description** | **Example** |
| [GET api/TutorStaff](http://localhost:62898/Help/Api/GET-api-TutorStaff) | Método que recupera todos los datos de TutorStaff de la base de datos. | <http://localhost:62898/api/TutorStaff> |
| [POST api/TutorStaff](http://localhost:62898/Help/Api/POST-api-TutorStaff) | Método POST, que crea el TutorStaff en la base de datos. | <http://localhost:62898/api/TutorStaff> |
| [PUT api/TutorStaff/{id}](http://localhost:62898/Help/Api/PUT-api-TutorStaff-id) | Método que actualiza el TutorStaff en la base de datos. | <http://localhost:62898/api/TutorStaff/6> |
| [DELETE api/TutorStaff/{id}](http://localhost:62898/Help/Api/DELETE-api-TutorStaff-id) | Método que elimina el TutorStaff en la base de datos. | <http://localhost:62898/api/TutorStaff/5> |

|  |  |  |
| --- | --- | --- |
| TutorStudent |  |  |
| API | **Description** | **Example** |
| [GET api/TutorStudent](http://localhost:62898/Help/Api/GET-api-TutorStudent) | Método que recupera todos los datos de TutorStudent de la base de datos. | <http://localhost:62898/api/TutorStudent> |
| [POST api/TutorStudent](http://localhost:62898/Help/Api/POST-api-TutorStudent) | Método POST, que crea el TutorStudent en la base de datos. | <http://localhost:62898/api/TutorStudent> |
| [PUT api/TutorStudent/{id}](http://localhost:62898/Help/Api/PUT-api-TutorStudent-id) | Método que actualiza el TutorStudent en la base de datos. | <http://localhost:62898/api/TutorStudent/6> |
| [DELETE api/TutorStudent/{id}](http://localhost:62898/Help/Api/DELETE-api-TutorStudent-id) | Método que elimina el TutorStudent en la base de datos. | <http://localhost:62898/api/TutorStudent/5> |

|  |  |  |
| --- | --- | --- |
| Techinician |  |  |
| API | **Description** | **Example** |
| [GET api/Techinician](http://localhost:62898/Help/Api/GET-api-Techinician) | Método que recupera todos los datos de Techinician de la base de datos. | <http://localhost:62898/api/Techinician> |
| [POST api/Techinician](http://localhost:62898/Help/Api/POST-api-Techinician) | Método POST, que crea el Techinician en la base de datos. | <http://localhost:62898/api/Techinician> |
| [PUT api/Techinician/{id}](http://localhost:62898/Help/Api/PUT-api-Techinician-id) | Método que actualiza el Techinician en la base de datos. | <http://localhost:62898/api/Techinician/6> |
| [DELETE api/Techinician/{id}](http://localhost:62898/Help/Api/DELETE-api-Techinician-id) | Método que elimina el Techinician en la base de datos. | <http://localhost:62898/api/Techinician/5> |

|  |  |  |
| --- | --- | --- |
| Student |  |  |
| API | **Description** | **Example** |
| [GET api/Student](http://localhost:62898/Help/Api/GET-api-Student) | Método que recupera todos los datos de Student de la base de datos. | <http://localhost:62898/api/Student> |
| [POST api/Student](http://localhost:62898/Help/Api/POST-api-Student) | Método POST, que crea el Student en la base de datos. | <http://localhost:62898/api/Student> |
| [PUT api/Student/{id}](http://localhost:62898/Help/Api/PUT-api-Student-id) | Método que actualiza el Student en la base de datos. | <http://localhost:62898/api/Student/6> |
| [DELETE api/Student/{id}](http://localhost:62898/Help/Api/DELETE-api-Student-id) | Método que elimina el Student en la base de datos. | <http://localhost:62898/api/Student/5> |

|  |  |  |
| --- | --- | --- |
| Staff |  |  |
| API | **Description** | **Example** |
| [GET api/Staff](http://localhost:62898/Help/Api/GET-api-Staff) | Método que recupera todos los datos de Staff de la base de datos. | <http://localhost:62898/api/Staff> |
| [POST api/Staff](http://localhost:62898/Help/Api/POST-api-Staff) | Método POST, que crea el Staff en la base de datos. | <http://localhost:62898/api/Staff> |
| [PUT api/Staff/{id}](http://localhost:62898/Help/Api/PUT-api-Staff-id) | Método que actualiza el Staff en la base de datos. | <http://localhost:62898/api/Staff/6> |
| [DELETE api/Staff/{id}](http://localhost:62898/Help/Api/DELETE-api-Staff-id) | Método que elimina el Staff en la base de datos. | <http://localhost:62898/api/Staff/5> |

|  |  |  |
| --- | --- | --- |
| Lecturer |  |  |
| API | **Description** | **Example** |
| [GET api/Lecturer](http://localhost:62898/Help/Api/GET-api-Lecturer) | Método que recupera todos los datos de Lecturer de la base de datos. | <http://localhost:62898/api/Lecturer> |
| [POST api/Lecturer](http://localhost:62898/Help/Api/POST-api-Lecturer) | Método POST, que crea el Lecturer en la base de datos. | <http://localhost:62898/api/Lecturer> |
| [PUT api/Lecturer/{id}](http://localhost:62898/Help/Api/PUT-api-Lecturer-id) | Método que actualiza el Lecturer en la base de datos. | <http://localhost:62898/api/Lecturer/6> |
| [DELETE api/Lecturer/{id}](http://localhost:62898/Help/Api/DELETE-api-Lecturer-id) | Método que elimina el Lecturer en la base de datos. | <http://localhost:62898/api/Lecturer/5> |

|  |  |  |
| --- | --- | --- |
| Campus |  |  |
| API | **Description** | **Example** |
| [GET api/Campus](http://localhost:62898/Help/Api/GET-api-Lecturer) | Método que recupera todos los datos de Campus de la base de datos. | [http://localhost:62898/api/ Campus](http://localhost:62898/api/Lecturer) |
| [POST api/Campus](http://localhost:62898/Help/Api/POST-api-Lecturer) | Método POST, que crea el Campus en la base de datos. | [http://localhost:62898/api/ Campus](http://localhost:62898/api/Lecturer) |
| [PUT api/ Campus /{id}](http://localhost:62898/Help/Api/PUT-api-Lecturer-id) | Método que actualiza el Campus en la base de datos. | [http://localhost:62898/api/ Campus /5](http://localhost:62898/api/%20Campus%20/5) |
| [DELETE api/ Campus/{id}](http://localhost:62898/Help/Api/DELETE-api-Lecturer-id) | Método que elimina el Campus en la base de datos. | <http://localhost:62898/api/Campus/3> |

|  |  |  |
| --- | --- | --- |
| Building |  |  |
| API | **Description** | **Example** |
| [GET api/Building](http://localhost:62898/Help/Api/GET-api-Lecturer) | Método que recupera todos los datos de Building de la base de datos. | <http://localhost:62898/api/Building> |
| [POST api/Building](http://localhost:62898/Help/Api/POST-api-Lecturer) | Método POST, que crea el Building en la base de datos. | <http://localhost:62898/api/Building> |
| [PUT api/ Building?pOld={pOld}](http://localhost:62898/Help/Api/PUT-api-Lecturer-id) | Método que actualiza el Building en la base de datos. | <http://localhost:62898/api/Building/BB01> |
| [DELETE api/Building?idBuilding={idBuilding}](http://localhost:62898/Help/Api/DELETE-api-Lecturer-id) | Método que elimina el Building en la base de datos. | <http://localhost:62898/api/Lecturer/BB02> |

|  |  |  |
| --- | --- | --- |
| Office |  |  |
| API | **Description** | **Example** |
| [GET api/Office](http://localhost:62898/Help/Api/GET-api-Office) | Método que recupera todos los datos de Office de la base de datos. | <http://localhost:62898/api/Office> |
| [GET api/Office?idOffice={idOffice}](http://localhost:62898/Help/Api/GET-api-Office_idOffice) | Método que recupera un dato específico en Office de la base de datos. | <http://localhost:62898/api/Office> |
| [POST api/Office](http://localhost:62898/Help/Api/POST-api-Office) | Método POST, que crea el Office en la base de datos. | <http://localhost:62898/api/Office/6> |
| [PUT api/Office?pOld={pOld}](http://localhost:62898/Help/Api/PUT-api-Office_pOld) | Método que actualiza el Office en la base de datos. | <http://localhost:62898/api/Office/5> |
| [DELETE api/Office?officeNum={officeNum}](http://localhost:62898/Help/Api/DELETE-api-Office_officeNum) | Método que elimina el Office en la base de datos. | <http://localhost:62898/api/Office/5> |

|  |  |  |
| --- | --- | --- |
| Classroom |  |  |
| API | **Description** | **Example** |
| [GET api/Classroom](http://localhost:62898/Help/Api/GET-api-Office) | Método que recupera todos los datos de Classroom de la base de datos. | <http://localhost:62898/api/Classroom> |
| [GET api/Classroom?idClassroom={idClassroom}](http://localhost:62898/Help/Api/GET-api-Office_idOffice) | Método que recupera un dato específico de Classroom de la base de datos. | <http://localhost:62898/api/Classroom> |
| [POST api/Classroom](http://localhost:62898/Help/Api/POST-api-Office) | Método POST, que crea el Classroom en la base de datos. | <http://localhost:62898/api/Classroom/6> |
| [PUT api/Classroom?pOld={pOld}](http://localhost:62898/Help/Api/PUT-api-Office_pOld) | Método que actualiza el Classroom en la base de datos. | <http://localhost:62898/api/Classroom/5> |
| [DELETE api/Classroom?classroomNum={classroomNum}](http://localhost:62898/Help/Api/DELETE-api-Office_officeNum) | Método que elimina el Classroom en la base de datos. | <http://localhost:62898/api/Classroom/5> |

|  |  |  |
| --- | --- | --- |
| Lab |  |  |
| API | **Description** | **Example** |
| [GET api/Lab](http://localhost:62898/Help/Api/GET-api-Office) | Método que recupera todos los datos de Lab de la base de datos. | <http://localhost:62898/api/Lab> |
| [GET api/Lab?idLab={idLab}](http://localhost:62898/Help/Api/GET-api-Office_idOffice) | Método que recupera un dato específico de Lab de la base de datos. | <http://localhost:62898/api/Lab> |
| [POST api/Lab](http://localhost:62898/Help/Api/POST-api-Office) | Método POST, que crea el Lab en la base de datos. | <http://localhost:62898/api/Lab/6> |
| [PUT api/Lab?pOld={pOld}](http://localhost:62898/Help/Api/PUT-api-Office_pOld) | Método que actualiza el Lab en la base de datos. | <http://localhost:62898/api/Lab/5> |
| [DELETE api/Lab?labNum={labNum}](http://localhost:62898/Help/Api/DELETE-api-Office_officeNum) | Método que elimina el Lab en la base de datos. | <http://localhost:62898/api/Lab/5> |

|  |  |  |
| --- | --- | --- |
| Degree |  |  |
| API | **Description** | **Example** |
| [GET api/Degree](http://localhost:62898/Help/Api/GET-api-Office) | Método que recupera todos los datos de Degree de la base de datos. | <http://localhost:62898/api/Degree> |
| [GET api/Degree?idDegree={idDegree}](http://localhost:62898/Help/Api/GET-api-Office_idOffice) | Método que recupera un dato específico de Degree de la base de datos. | <http://localhost:62898/api/Degree> |
| [POST api/Degree](http://localhost:62898/Help/Api/POST-api-Office) | Método POST, que crea el Degree en la base de datos. | <http://localhost:62898/api/Degree/6> |
| [PUT api/Degree?idDegree={idDegree}](http://localhost:62898/Help/Api/PUT-api-Office_pOld) | Método que actualiza el Degree en la base de datos. | <http://localhost:62898/api/Degree/5> |
| [DELETE api/Lab?degreeCode={degreeCode}](http://localhost:62898/Help/Api/DELETE-api-Office_officeNum) | Método que elimina el Degree en la base de datos. | <http://localhost:62898/api/Degree/5> |

|  |  |  |
| --- | --- | --- |
| Subject |  |  |
| API | **Description** | **Example** |
| [GET api/Subject](http://localhost:62898/Help/Api/GET-api-Office) | Método que recupera todos los datos de Subject de la base de datos. | <http://localhost:62898/api/Subject> |
| [GET api/Subject?idSubject={idSubject}](http://localhost:62898/Help/Api/GET-api-Office_idOffice) | Método que recupera un dato específico de Subject de la base de datos. | <http://localhost:62898/api/Subject> |
| [POST api/Subject](http://localhost:62898/Help/Api/POST-api-Office) | Método POST, que crea el Subject en la base de datos. | <http://localhost:62898/api/Subject/6> |
| [PUT api/Subject?idSubject={idSubject}](http://localhost:62898/Help/Api/PUT-api-Office_pOld) | Método que actualiza el Subject en la base de datos. | <http://localhost:62898/api/Subject/5> |
| [DELETE api/Subject?nameSubject={nameSubject}](http://localhost:62898/Help/Api/DELETE-api-Office_officeNum) | Método que elimina el Subject en la base de datos. | <http://localhost:62898/api/Subject/5> |

|  |  |  |
| --- | --- | --- |
| EnrollsIn |  |  |
| API | **Description** | **Example** |
| [GET api/EnrollsIn](http://localhost:62898/Help/Api/GET-api-Office) | Método que recupera todos los datos de EnrollsIn de la base de datos. | <http://localhost:62898/api/EnrollsIn> |
| [GET api/EnrollsIn?idEnroll={idEnroll}](http://localhost:62898/Help/Api/GET-api-Office_idOffice) | Método que recupera un dato específico de EnrollsIn de la base de datos. | <http://localhost:62898/api/EnrollsIn> |
| [POST api/EnrollsIn](http://localhost:62898/Help/Api/POST-api-Office) | Método POST, que crea el EnrollsIn en la base de datos. | <http://localhost:62898/api/EnrollsIn/6> |
| [DELETE api/EnrollsIn?studentID={studentID}](http://localhost:62898/Help/Api/DELETE-api-Office_officeNum) | Método que elimina el EnrollsIn en la base de datos. | <http://localhost:62898/api/EnrollsIn/5> |

|  |  |  |
| --- | --- | --- |
| Takes |  |  |
| API | **Description** | **Example** |
| [GET api/Takes](http://localhost:62898/Help/Api/GET-api-Office) | Método que recupera todos los datos de Takes de la base de datos. | <http://localhost:62898/api/Takes> |
| [GET api/Takes?idTakes={idTakes}](http://localhost:62898/Help/Api/GET-api-Office_idOffice) | Método que recupera un dato específico de Takes de la base de datos. | <http://localhost:62898/api/Takes> |
| [POST api/Takes](http://localhost:62898/Help/Api/POST-api-Office) | Método POST, que crea el Takes en la base de datos. | <http://localhost:62898/api/Takes/6> |
| [DELETE api/Takes?studentID={studentID}](http://localhost:62898/Help/Api/DELETE-api-Office_officeNum) | Método que elimina el Takes en la base de datos. | <http://localhost:62898/api/Takes/5> |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| Faculty |  |  |
| API | **Description** | **Example** |
| [GET api/Faculty](http://localhost:62898/Help/Api/GET-api-Faculty) | Método que recupera todos los datos de Faculty de la base de datos. | <http://localhost:62898/api/Faculty> |
| [GET api/Faculty?idFaculty={idFaculty}](http://localhost:62898/Help/Api/GET-api-Faculty_idFaculty) | Método POST, que crea el Faculty en la base de datos. | <http://localhost:62898/api/Faculty> |
| [POST api/Faculty](http://localhost:62898/Help/Api/POST-api-Faculty) | Método que actualiza el Faculty en la base de datos. | <http://localhost:62898/api/Faculty/6> |
| [PUT api/Faculty?idFaculty={idFaculty}](http://localhost:62898/Help/Api/PUT-api-Faculty_idFaculty) | Método que elimina el Faculty en la base de datos. | <http://localhost:62898/api/Faculty/5> |
| [DELETE api/Faculty?name={name}](http://localhost:62898/Help/Api/DELETE-api-Faculty_name) | Método que elimina el Faculty en la base de datos. | <http://localhost:62898/api/Faculty/5> |

**Conclusiones**

Después de desarrollar la lógica requerida por el sistema de este proyecto, nos damos cuenta de que algunas tablas, como la de enrolado de estudiantes en un grado académico y la de promedios en una materia por estudiante, hacen uso de una referencia hacia el objeto respectivo, lo que ofrece un comportamiento similar al de una llave foránea en cuanto a recepción de valores desde la tabla de origen, aunque sin apuntar a la llave primaria de esa tabla. Además, en la consulta que almacena los resultados en la tabla Degree\_Records, se tuvo que utilizar la técnica de tabla.ref\_objeto.atributo para obtener cada uno de los valores cuyo código de grado dentro de la tabla de enrolados coincidía con cada uno de los códigos de grado almacenados en la tabla Degree.

En la fase anterior del proyecto solamente se implementó la capa lógica, o sea la correspondiente a los scripts DDL y las consultas en Oracle PL/SQL, ya que la capa de MVC (la referente a la interfaz de usuario, vistas y controladores) corresponde a la siguiente etapa, junto con los RESTful APIs (de servicios web) requeridos para acceder a la base de datos de Oracle alojada en la red de la Escuela de Computación del TEC.

Sin embargo, para esta segunda fase se incorporó dicha lógica de negocio al RESTful Web API necesario para la aplicación Web, además de implementar la arquitectura de MVC para la misma. Desarrollando esta fase del proyecto, descubrimos que la forma en cómo la aplicación maneja los WebServices es muy similar a la del primer proyecto (el cual se realizó sobre MEAN en MongoDB), aunque implementada de una manera completamente distinta, haciendo uso de Visual Studio 2017.