|  |
| --- |
|  |
| **EasySQL** |
| **PROYECTO DE DESARROLLO DE APLICACIONES MULTIPLATAFORMA** | | |
| CURSO 2017/18 |
| **ELABORADO POR:** ALEJANDRO ANTONIO DEL RÍO MORENO. | |
| 0EasySQL  [http://www.cenecmalaga.es/uploads/7/9/8/4/79848062/9917986.png](http://www.cenecmalaga.es/) | |

**Índice de contenido**

**[1. Descripción 3](#_Toc20060)**

**[1.1. Introducción 3](#_Toc30434)**

**[1.2. Motivación 5](#_Toc1573)**

**[2. Análisis del Mercado Actual 6](#_Toc24972)**

**[2.1. Estudio de Productos Existentes 7](#_Toc2230)**

**[2.2. Conclusiones del Estudio 10](#_Toc10316)**

**[3. Propuesta Innovadora 11](#_Toc1725)**

**[4. Estudio de Viabilidad 13](#_Toc32385)**

**[4.1. Recursos y Tecnologías a Emplear 13](#_Toc15839)**

**[4.2. Evaluación de Costes 18](#_Toc11255)**

**[4.3. Planificación del Tiempo 19](#_Toc15500)**

**[4.4. Alcance del Proyecto 20](#_Toc7142)**

**[5. Arquitectura de la Aplicación 21](#_Toc9593)**

**[5.1. Análisis y Diagramas UML 23](#_Toc16651)**

**[5.2. Análisis y Diagramas E/R 33](#_Toc30694)**

**[6. Futuras Mejoras y Conclusiones 33](#_Toc24398)**

**[7. Pruebas y Resultados 33](#_Toc19227)**

**[8. Manual Usuario 33](#_Toc4051)**

**[9. Bibliografía 33](#_Toc7871)**

**[10. Índice de imágenes 33](#_Toc14958)**

**[Anexo I.- Manual del Código 33](#_Toc12023)**

1. **Descripción**

En esta memoria pretende ser un sumario en el que plasmar y dejar reflejado el trabajo realizado a lo largo del desarrollo del proyecto de final de Formación Profesional de Grado Superior, así de sus tecnologías usadas.

* 1. **Introducción**

EasySQL es una herramienta pensada para facilitar el aprendizaje del lenguaje SQL al usuario, por medio de una interfaz gráfica amigable y simple, el manejo de datos de los dos grandes sistemas gestores de bases de datos que existen en la actualidad.



Imagen 1. Logotipo de EasySQL.

Esto permite un acercamiento más acogedor a un usuario nobel o inexperto, que pueda llegar a verse abrumado por la cantidad de herramientas y opciones que poseen las completas soluciones que ofrecen hoy en día los principales desarrolladores, pensadas para grandes empresas o complejos proyectos, con una curva de aprendizaje mucho más dura.

El mundo del aprendizaje siempre ha sido un mundo en constante evolución, con el tiempo van apareciendo nuevas metodologías y maneras de enseñanza para adaptarse a los nuevos tiempos. Sobre todo en la informática, campo en el que normalmente unos contenidos de hace 5 años quedarían obsoletos en la actualidad.

En la actualidad, estamos viviendo la transición en el ámbito de la educación dentro de las centros del tradicional formato de papel y bolígrafo, aquellos que nos han acompañado desde largo tiempo atrás, a métodos más interactivos y tecnológicos, como el caso de la implantación de tablets y ordenadores para los alumnos, que aunque muy poco a poco pero en mayor medida en campos como la informática estamos viendo que cada vez ocupan una mayor importancia dentro de las aulas. Por ello, es necesario que tanto docentes como alumnos cuenten siempre con las mejores herramientas para favorecer un rápido aprendizaje, más en los tiempos que corren, que cada minuto cuenta y hay tanta materia y conocimiento que cualquier mejora en la manera de transmitir información puede ser muy bien recibida.

Es por ello que gracias a nuestra formación, debemos crear herramientas que aprovechen la disponibilidad de recursos que existen y ayuden a allanar el terreno a aquellos alumnos que quedan por venir.

La herramienta, especialmente pensada para los estudiantes, permite al usuario con una forma visual y fácil ir realizando las operaciones básicas de creación y manejo de datos, además de poder ver al instante el resultado de éstas, así como la sentencia SQL equivalente.

Normalmente, un usuario debe tener cierta preparación e instrucción a la hora de poder empezar a utilizar sistemas gestores de bases de datos. Este programa intenta acortar el tiempo necesario de preparación previa de dicho usuario y animarlo a ver resultados rápidos sin tener que lidiar con la exigente sintaxis del lenguaje SQL, permite un acercamiento básico y seguro.

* 1. **Motivación**

Dado que la informática es una rama de la ciencia en la que continuamente se está avanzando, creando nuevos productos, surgiendo necesidades y adaptando tecnologías a ellas, los desarrolladores de aplicaciones son aquellos que deben cargar con el peso de esto. Al contrario que otras ciencias como la medicina, el mundo de la informática avanza muy rápido, constantemente se están produciendo cambios, y nosotros debemos adaptarnos a ellos.

Es por ello que cuando uno se enfrenta a una tecnología, que, aunque en este caso tenga ya largo recorrido y esté bastante establecido como es el Lenguaje de Consulta Estructurada, del inglés **Structured Query Language** o por sus siglas, SQL, siempre es de agradecer tener una herramienta para los primeros momentos del aprendizaje, aquellos en los que el usuario se pueda sentir más perdido y sea más propenso a fallos y errores como aquellos producidos por, en este caso, el punto y coma (;), o sentencias peligrosas de ejecutar ya sea por pérdidas masivas de datos sin que el usuario sea consciente de ellas.

Otra de las motivaciones que me llevaron a hacer el proyecto sobre esta tecnología en concreto, es la robustez y lo sorprendentemente bien que ha aguantado los años dicha tecnología sin convertirse en obsoleta. La primera versión del lenguaje data de inicios de 1974, y a día de hoy sigue siendo el lenguaje número uno en manejo de datos a través de Sistemas Gestores de Bases de Datos, sin previsión de que esto vaya a cambiar en corto plazo, ya que constantemente el estándar se va actualizando para adaptarse a las nuevas necesidades que van surgiendo con el tiempo.

1. **Análisis del Mercado Actual**

La principal apuesta del proyecto es el uso del sólido y consolidado lenguaje SQL, utilizado por las principales y empresas a nivel mundial.

A día de hoy, está emergiendo una prometedora nueva tecnología llamada NoSQL, que aún teniendo gran recepción por parte del público de desarrolladores indies y compañías pequeñas, está teniendo problemas en acercarse al ámbito de las grandes empresas, ya que un cambio de tecnología supondría el rehacer y trasladar los datos de las ya robustas y fiables bases de datos SQL tradicionales a estas más modernas.



Imagen 2. Comparación proporción de uso actual entre diferentes tecnologías y Sistemas Gestores de Bases de Datos (gráfico no excluyente).

Como podemos observar en la gráfica de la página anterior, el gran grueso del mercado actual utiliza las clásicas bases de datos SQL. Los porcentajes del gráfico no son excluyentes, esto quiere decir que en la práctica las compañías según sus necesidades utilicen varias de estas tecnologías o Sistemas Gestores de Bases de datos para aprovechar las particularidades y beneficios de cada uno. También se puede observar que, dentro de las bases de datos tradicionales, las dos grandes bazas son la apuesta de Microsoft con Microsoft SQL Server, y la apuesta de Sun Microsystems (ahora parte de Oracle Corporation).



Imagen 3. Las dos principales marcas dominantes del mercado.

Estas dos grandes marcas unidas copan una gran cuota de mercado entre ellas. Es por ello que el proyecto se centra en dar soporte a ellas.

Además, ambas cuentan con versiones gratuitas específicas para que en caso de ser estudiante o pequeño desarrollador, podamos utilizarlas y aprender con ellas sin tener que asumir costes iniciales.

* 1. **Estudio de Productos Existentes**

Para los Sistemas Gestores de Bases de Datos a utilizar, existen dos completas y complejas soluciones en el mercado para su manejo y utilización: MySQL WorkBench y SQL Server Management. Las aplicaciones fueron creadas y son a día de hoy mantenidas por los mismos creadores del dichos Sistemas Gestores, los mencionados anteriormente Microsoft y Sun Microsystems.

La primera aplicación a comentar es MySQL WorkBench. A destacar sobre su competidora, esta dispone de dos versiones; una Community, licenciada sobre GPL, de código abierto, gratuita y almacenada en GitHub para que cualquier usuario pueda contribuir a su desarrollo, y otra versión Standard de código cerrado, destinada a empresas, con soporte y diferentes planes de pago según las necesidades del negocio, así como añade algunas series de plugins y módulos de código cerrado.



Imagen 4. Interfaz de usuario de MySQL WorkBench (MacOs)

Esta herramienta dispone de un potente editor gráfico específico para diseñar estructuras de bases de datos, tablas y relaciones entre ellas. Desde su introducción se ha vuelto muy popular dentro de la comunidad MySQL. Actualmente es el segundo producto más descargado desde la web de MySQL con más de 250.000 descargas al mes, a parte de haber sido calificado por revistas open-source como el mejor manejador de bases de datos de código abierto existente.

SQL Server Management Studio es la respuesta de Microsoft a la necesidad de un software de manejo sus bases de datos, ya que la primera versión del Servidor SQL Server fue lanzada en 1989 pero no sería hasta 2005 cuando sacaron una aplicación gráfica para su manejo, unos años más tarde que MySQL. En este caso, todas las versiones de este producto están bajo licencia propietaria, y en ningún caso el código está disponible al usuario.



Imagen 5. Interfaz de usuario de SQL Server Management Studio (Windows)

La gran ventaja de esta herramienta es que si estás desarrollando en un lenguaje .NET y usas el entono de desarrollo Visual Studio tienes gran integración en ambas, encontrarás a mano la mayoría de las herramientas que puedas necesitar usar para tu proyecto .NET, como podrías encontrar en Visual Studio. Provee de métodos y maneras de acceder a los datos de manera segura y probada, y ahorra al desarrollador de tener que escribir sus propias funciones para el acceso o manipulación de datos.

* 1. **Conclusiones del Estudio**

En los tiempos que corren, el poder consultar, almacenar y modificar información es algo totalmente necesario para cualquier empresa, esto se traduce como que en la práctica mayoría de empresas de una envergadura considerable se necesita personal dedicado al tratamiento de grandes datos.

Constantemente se generan nuevos puestos de trabajo relacionados con este ámbito, y para poder rellenar esos huecos necesitan personas que previamente reciban una formación al respecto.

Imagen 6. Logotipo de SQL

A día de hoy las herramientas destinadas para este fin pecan de los mismos fallos. Aún habiendo visto las dos más importantes del mercado, las demás soluciones en mayor o menor medida tienen un mismo denominador común: son bastante grandes y complejas, requieren de una buena formación previa para poder empezar a ejecutar operaciones sencillas (casi todo debe hacerse por comandos escritos sin mucha ayuda por el entorno) y en definitiva son poco intuitivas y complejas de usar para un usuario nobel, que suele esperar ver resultados rápidos y avances en poco tiempo.

Es por ello, unido a la larga vida y el gran futuro de esta tecnología, de la que se tiene la necesidad constante de aprendizaje por parte de nuevos usuarios de ella, que existe un hueco para este nicho de usuarios del que esta aplicación puede sacar provecho para establecerse y ser una herramienta útil para un primer acercamiento del usuario al mundo del tratamiento de datos por medio de consultas con el lenguaje SQL.

1. **Propuesta Innovadora**

En la actualidad, docentes y alumnos carecen de una herramienta de fácil manejo, portable, sin instalación, gratuita, de código abierto y casi nula necesidad de formación previa como la desarrollada en este proyecto.

Es por ello que, esta aplicación puede entrar en un mercado específico que apenas está cubierto, y atraer a cierta cantidad de usuarios ya que no tiene competidores directos.

La herramienta, está pensada para los primeros momentos de la enseñanza de un alumno, por lo que se prevee que éste en un periodo de tiempo no muy largo, cuando avance en su aprendizaje y tenga más comprensión del lenguaje SQL, pase a utilizar herramientas más sofisticadas y que ofrezcan soluciones más completas.

Es por ello que se ha tenido en cuenta el peso de esta, reduciendo a mínimos el tamaño que ocupa el ejecutable, y la sencillez de su comienzo de uso, así como lo fácil de deshacerse de ella una vez el usuario lo estime conveniente. La aplicación final para el alumno consta de un archivo ejecutable .exe que solo necesita abrir y listo, nada de instaladores o desinstaladores ni tampoco necesidad de dependencias de terceros programas u librerías de componentes.

Aun así, el usuario final terminará comprobando que para ciertas operaciones rápidas y sencillas de consulta e inserción de datos este programa sigue siendo mucho más rápido de usar, ya que los tiempos de carga de la aplicación, como por tiempos de creación de un comando son bastante menores en comparación con las grandes soluciones de los fabricantes, que en equipos modestos suelen tardar bastante en hacer la carga inicial.

Otro punto importante es el hecho que este proyecto desde el minuto 0 de su creación ha estado siendo almacenado y actualizado en la plataforma online número uno en cantidad de usuarios y proyectos de código abierto y colaborativos conocida como GitHub.



Imagen 7. Logotipo de GitHub

Esto, dado que es una aplicación de desarrolladores para desarrolladores, es una ventaja importante frente a las alternativas de código cerrado y propietarias, ya que permite que el proyecto se pueda mantener a lo largo del tiempo gracias a contribuciones o aportaciones de personas anónimas que quieran ayudar de manera desinteresada con tan sólo descargarse la solución para Visual Studio completa, que contiene los archivos de código fuente sin compilar, imágenes de prueba, ejemplos, archivos documentados y scripts SQL de pruebas. Todo ello se encuentra alojado en la misma web para una mayor comodidad para el desarrollador final, que encontrará con la documentación y ejemplos una manera sencilla de continuar con la labor de desarrollo.

Aquel alumno que aprenda y se beneficie con el uso de esta herramienta también podría, si está interesado, observar y mejorar el funcionamiento interno y aprender cómo la aplicación trabaja en su interior, esto es: las llamadas a bases de datos, la manera de gestionar las conexiones, las diferencias entre los conectores MySQL y SQL Server... También podría tener un efecto llamada y animar a esa persona a acercarse al mundo de las tecnologías .NET como WPF y el lenguaje C# con los que está construida la aplicación.

1. **Estudio de Viabilidad**

En este apartado se tienen en cuenta y se analizan los gastos materiales necesarios para la implantación del software de cara al usuario final. También se examinan las tecnologías usadas para la creación y desarrollo del proyecto.

Se tiene en cuenta que, en caso de tratar de analizar los costes de los recursos humanos, estos son soportados de manera íntegra por el autor del proyecto y no se cobrará ningún importe en la realidad, ya que se realiza sin ánimo de lucro, abierto y será cedido a la comunidad de desarrolladores de software libre por lo que no tendrá un beneficio económico.

* 1. **Recursos y Tecnologías a Emplear**

A la hora de analizar los medios empleados para el desarrollo y uso de este proyecto, podemos diferenciar dos grandes campos. Por un lado examinaremos los recursos utilizados en el lado del desarrollador, aquellos que han hecho posible que el proyecto cobre vida y haya llegado hasta donde está ahora, y por el otro indagaremos las tecnologías o bienes necesarios por parte del usuario final para poder hacer uso de este software.

Empezando por el lado del desarrollador, las siguientes tecnologías ahora listadas han sido utilizadas a lo largo del desarrollo de este producto:

* **Visual Studio**: es el entorno de desarrollo integrado (IDE) por excelencia para crear aplicaciones nativas para el Sistema Operativo Windows. Su desarrollador es Microsoft, autor del nombrado S.O. Soporta múltiples lenguajes de programación, y su uso no sólo se limita a aplicaciones para Windows, también tiene grandes capacidades a la hora de crear aplicaciones Web. Cuenta con un muy potente editor gráfico de ventanas que permite en poco tiempo obtener grandes resultados, abstrayendo al desarollador de escribir gran parte del código relacionado con la Vista de la aplicación. Tiene gran integración con bases de datos (sobretodo SQL Server) e incluye herramientas para el manejo de Git y GitHub. Existe una versión Community gratuita que es la que ha sido elegida para esta ocasión.

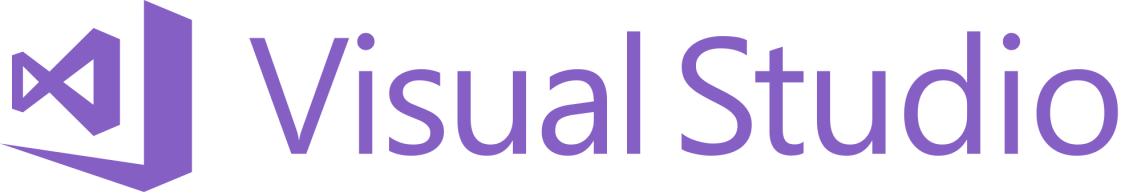


Imagen 8. Logo de la aplicación Visual Studio.

* **Windows Presentation Fundation**: También conocido como WPF, es una tecnología de Microsoft. Presentada junto con el sistema operativo Windows Vista, es el relevo de la antigua tecnología usada para crear ventanas en anteriores sistemas operativos Windows, conocida como Windows Forms, que aún sigue siendo soportada. La principal ventaja es que separa, usando el lenguaje de marcas XAML y los lenguajes .NET la interfaz de interacción de la parte del código, propiciando una arquitectura Modelo Vista Controlador. Esta tecnología permite crear unas interfaces de usuario que se adaptan independientemente de la resolución de la pantalla, y todos los gráficos se representan usando Direct3D, por lo que están optimizadas y ganan en velocidad por aceleración de la tarjeta gráfica del equipo.



Imagen 9. Logo de la tecnología WPF.

* **Git** con **GitHub**: Git es un software de control de versiones diseñado por el creador del kernel de Linux, Linus Torvalds. Su propósito es llevar registro de los cambios en archivos del proyecto y coordinar el trabajo en caso de que varias personas trabajen con los mismos archivos.

Permite al desarrollador ver un historial de los cambios producidos y volver en cualquier momento a un estado anterior del proyecto. También permite separar en ramas de desarrollo el código, por si se necesitan hacer pruebas o incorporar cambios que no se saben si pueden afectar a la estabilidad del software. Al estar configurado con GitHub, permite que el proyecto con su historial de cambios esté alojado en la nube, fuera de nuestro equipo local (a pesar de que se mantiene una copia local). Esto permite la posibilidad de trabajar en el código desde más de un ordenador, al mismo tiempo que al tener una copia siempre actualizada en la nube da la seguridad de que nuestro trabajo no va a desaparecer en ningún momento. Desde el primer momento del desarrollo de este software se lleva dejando constancia en la plataforma GitHub, para no dejar posibilidad a ninguna pérdida de datos que pudiese producirse en la copia local.

Una de las grandes ventajas que ofrece GitHub a la hora de depositar tu proyecto en su plataforma es la gran cantidad de usuarios activos existentes. Es la mayor plataforma de código abierto colaborativa que existe en la actualidad. Estos usuarios pueden colaborar y ser partícipes de tu proyecto, ayudando a mejorar el código, implementando nuevas funciones, o dándote consejos sobre cómo mejorar el software.



Imagen 10. Logo de la tecnología Git y la plataforma GitHub

* **SQL Server**: es un sistema gestor de base de datos relacional desarrollado por Microsoft , como se ha hablado de ella previamente en este manual. Se ha seleccionado por su gran integración y facilidad de uso en Visual Studio, que hace que ayuda al desarrollador de tener que lidiar con algunos de los problemas como seguridad de conexión y número de conexiones permitidas. Esta herramienta se utiliza a la hora de relacionar y almacenar con persistencia los usuarios con su contraseña para la función de guardado de conexiones de la aplicación.
* **BCrypt para C#**: es uno de los mejores algoritmos de hasheo de contraseñas que existen en la actualidad. Sus creadores fueron Niels Provos y David Mazières, y fue presentado en 1999. Es el algoritmo de contraseñas por defecto para el sistema operativo OpenBSD, así como muchas otras distribuciones Linux como SUSE Linux. Se diferencia del resto en la velocidad en la que hashea. Esta es conocida por tardar alrededor de 100ms por defecto en hashear una contraseña aunque se pueden configurar a más. Esto hace que a un posible atacante le sea inviable por el tiempo y costo computacional que conlleva para sacar una contraseña por fuerza bruta comparado con los otros algoritmos que obtienen resultados en tiempo mucho menores.

Se ha utilizado una implementación en forma de librería para C# escrita por Damien Miller.

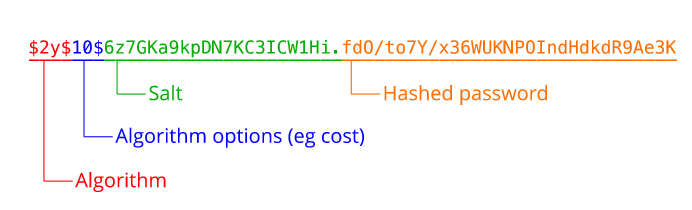


Imagen 11. Anatomía del hash devuelto por el algoritmo BCrypt.

* **SQL Server Management Studio, Bases de datos MySQL y MySQL WorkBench**: Las anteriores herramientas, que proveen de una interfaz gráfica para utilizar las bases de datos, se han utilizado a lo largo del desarrollo para probar que las funcionalidades de la aplicación de los comandos lanzados contra bases de datos han tenido una ejecución correcta. También han permitido crear y definir las propias bases de datos de pruebas y sus tablas.
* **Conector MySQL para .NET**: Propiedad de la compañía Oracle, se ha utilizado en el proyecto la nueva versión 8, recién salida este año, para asegurar tener el máximo rendimiento y soporte. Se encarga de proveer funcionalidades críticas a la hora de establecer conexiones y ejecutar comandos contra bases de datos MySQL, algo que las tecnologías .NET no proveen de forma nativa.
* **Otros**: durante el desarrollo del software se han utilizado otros recursos que se listan a continuación:
  + Ordenador con el Sistema Operativo Windows 10.
  + Suite ofimática WPS con la que se ha redactado la Memoria.
  + Notepad++ para el manejo de scripts SQL.

La lista de tecnologías que el usuario final debe tener para hacer uso de la aplicación es considerablemente menor, ya que éste sólo necesita un ordenador personal con una versión de Windows superior a Vista. En el caso de querer disponer de las funciones de almacenado de datos de conexiones para un determinado usuario, parte importante de la aplicación pero no imprescindible, también necesitaría tener instalada una instancia de SQL Server en su máquina o en un servidor externo que gestione los datos. También necesitaría como parte evidente la base de datos MySQL o SQL Server contra la que ejecutar los comandos que genere con la aplicación.

* 1. **Evaluación de Costes**

Podemos dividir la parte de la evaluación de costes en tres grandes ámbitos. El coste de los recursos humanos, el coste de las licencias, y el coste del mantenimiento futuro del proyecto.

Para empezar, ya que como se ha recalcado previamente, la aplicación ha sido en todo momento de código abierto y sin ánimo de lucro, no ha existido hasta el momento del lanzamiento del sotfware ningún tipo de coste humano. Todo el trabajo lo ha llevado el autor del proyecto (yo) sin recibir retribución.

Una vez comentado el tema del coste humano, se entra a valorar el tema de las licencias del software necesarios a utilizar. En este proyecto se ha optado por utilizar las versiones gratuitas que existían para las tecnologías usadas ya que perfectamente cumplen para nuestros requerimientos. No se ha tenido que pagar ninguna licencia. Con Microsoft Visual Studio se ha optado por la Community Edition. Esta versión es de licencia gratuita para desarrolladores. A pesar de no tener todos los contenidos, como los tiene la versión Professional, se podemos realizar todas las tareas de desarrollo que se han requerido. Con Sistema Gestor de Base de Datos de SQL Server se ha optado por la Developer Edition. Este tipo de licencias, no contienen soporte técnico pero ese no ha sido problema ya que todo el soporte y dudas que he ido podiendo necesitar las he resuelto con la ayuda de información de internet.

Para terminar se analiza el tema del mantenimiento futuro del producto. Ya que desde el principio se ha apostado por la filosofía open-source y de trabajo colaborativo sin ánimo de lucro, el proyecto no va a requerir de costes futuros. Siguientes futuras versiones y nuevas mejoras se esperan que sean entregadas de manera gratuita al público, así como que los costes de mantenimiento del alojamiento de los ficheros de código y la aplicación en sí se realizan de manera gratuita y de por vida por la plataforma GitHub.

* 1. **Planificación del Tiempo**

Una vez realizada la exposición inicial de ideas del proyecto al tutor de seguimiento, se comienza con su planificación. Esta planificación se ha realizado teniendo en cuenta que el autor de la aplicación está realizando unas prácticas de formación en centro de trabajo al mismo tiempo que elabora el proyecto.

De esta planificación inicial se obtienen los siguientes puntos importantes que resumen el tiempo de trabajo total que puede llevar tardar en realizarse.

* Reuniones con el tutor: se han dedicado unas 5 horas en todas las reuniones con el tutor, junto con otras 5 añadidas por el desplazamiento más 2 horas extras resolviendo dudas junto a otros profesores o alumnos. En total 12.
* Estudio y análisis del proyecto: se han dedicado alrededor de unas 15 horas en el estudio y análisis previo del proyecto.
* Diseño de la base de datos: un total de unas 2 horas se han utilizado para decidir las tecnologías apropiadas para la base de datos del programa así como su diseño interno.
* Creación de la base de datos: aproximadamente se han tardado 3 horas en la creación del esquema, tablas y sus relaciones en el sistema gestor de bases de datos utilizado, SQL Server.
* Diseño y codificación de la interfaz de usuario: esta es la segunda parte del proyecto que más tiempo me ha llevado, al rededor de 30 horas se han utilizado para el diseño y su implementación de todas las ventanas de la interfaz de usuario que tiene la aplicacion, ellas codificadas en XAML haciendo uso de la tecnología WPF.
* Diseño y codificación de la parte lógica de la aplicación: el grueso de la dedicación del tiempo se lo lleva, como no podía ser de otra manera, la parte del diseño y codificación de la lógica de negocio, todo aquello relacionado con el funcionamiento del software de manera interna. Aproximadamente un tiempo total de 60 horas.
* Instalación de software necesario: en este apartado se recoge el tiempo necesario utilizado para la instalación de las herramientas de software usadas para el desarrollo del proyecto. Unas 5 horas han sido invertidas entre el tiempo de descarga y tiempo de instalación de las tecnologías.
* Memoria del proyecto: por ultimo se tiene en cuenta el tiempo total utlizado para hacer esta memoria. Aproximadamente 20 horas se han necesitado para elaborar el contenido de la misma, sumando tiempos de análisis y búsqueda de datos así como el tiempo de plasmarlos y darle un formato correcto a este documento.

Según la planificación resultante, el proyecto ha tenido una duración de 177 horas. Teniendo en cuenta que para el desarrollo del proyecto se ha dispuesto de unos 3 meses de trabajo, Este se ha desarrollado con una dedicación aproximada de 12 horas semanales.

* 1. **Alcance del Proyecto**

En este apartado se evalúa el alcance actual y futuro a medio y largo plazo de este proyecto.

A la hora de entregar el proyecto, se tiene idea de que esté disponible para el usuario final en el mismo momento que la entrega. Este podrá hacer uso de él sin necesidad que el autor le asista personalmente e instruya a utilizarlo, ya que encontrará ejemplos de uso e instalación el servidor donde se encuentra la solución, en este caso GitHub.

A medio plazo se espera que con parte de ayuda de la comunidad de software libre y siendo liderado por el autor del proyecto, se sigan añadiendo mejoras, depurando funcionalidades y realizando optimizaciones que hagan de este producto uno más sencillo de usar si cabe. Es posible que algún alumno que utilice la aplicación se interese por el funcionamiento interna de esta y debe saber que tendrá la facilidad y la disponibilidad de aportar contenido y ayudar con el desarrollo en todo momento a través de GitHub.

A largo plazo se ha planteado un posible fork del proyecto hacia una versión Web, que contaría con los mismos principios que este proyecto. Debería ser gratuito, de código abierto y hosteado en GitHub y primar siempre la facilidad de uso. Esto permitiría utilizar el software desde cualquier dispositivo, sin tener en cuenta el sistema operativo (actualmente limitado a Windows) ni tampoco dónde se ejecuta (ahora sólo en ordenadores), ya que el dispositivo únicamente necesitaría de un navegador web.

1. **Arquitectura de la Aplicación**

La aplicación es una intermediaria gráfica entre el usuario que la maneje y el host final para peticiones que siguen la arquitectura de Cliente - Servidor.

Esta es un modelo de diseño de software por capas en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes.

Un usuario que haga uso de ella la utilizará para generar consultas SQL de manera sencilla, y ejecutarlas contra un servidor de bases de datos que devolverá una respuesta en forma de datos o en forma de información como el número de filas afectadas.

Algunos otros ejemplos de aplicaciones computacionales que usen el modelo cliente-servidor son el [correo electrónico](https://es.wikipedia.org/wiki/Correo_electr%C3%B3nico" \o "Correo electrónico), un [Servidor de impresión](https://es.wikipedia.org/wiki/Servidor_de_impresi%C3%B3n" \o ") y la [World Wide Web](https://es.wikipedia.org/wiki/World_Wide_Web" \o "World Wide Web).

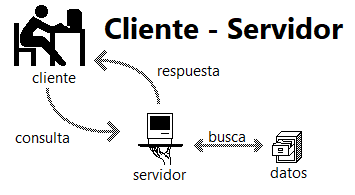


Imagen 12. Arquitectura Cliente - Servidor

A la hora del desarrollo y la manera de codificar la aplicación, se ha apostado por el uso del patrón de arquitectura de software llamado Modelo-vista-Controlador (MVC). Este patrón separa los datos y la lógica de negocio de una aplicación de la manera en la que se representa visualmente al usuario. Este patrón de arquitectura de software se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento.

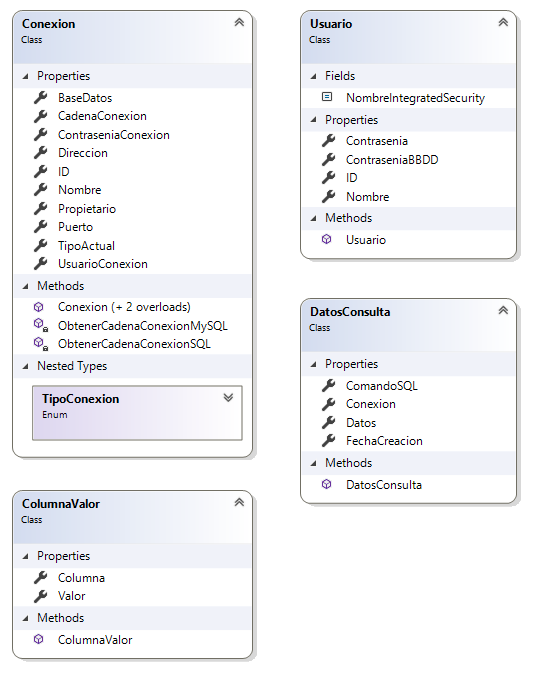


Imagen 13. Colaboración típica entre comoponentes en un patrón MVC.

* 1. **Análisis y Diagramas UML**

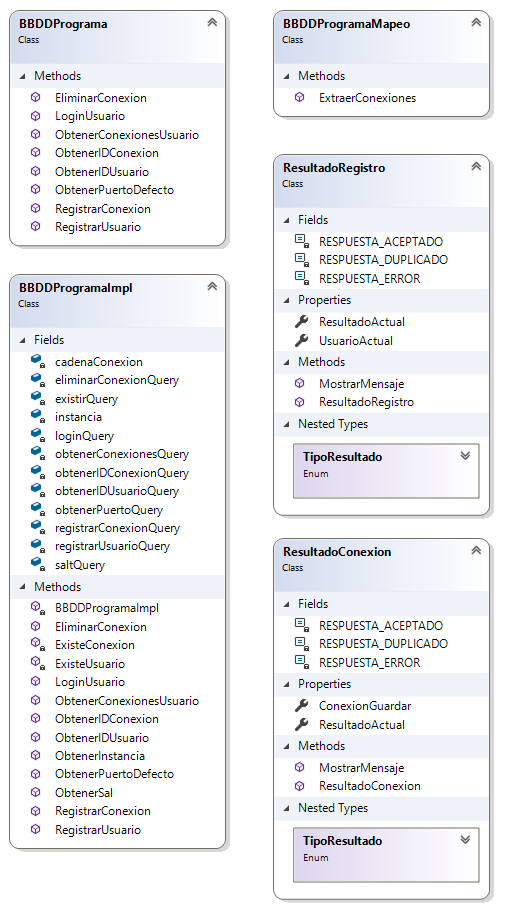
En las siguientes páginas se muestran en lista las tablas UML extraídas de las clases utilizadas durante el desarrollo del proyecto. Estas permiten ver de un vistazo rápido la manera en la que están compuestas las clases, y en ellas aparecen sus métodos y campos.

Las primeras tablas UML corresponden a los modelos de datos utilizados.



UML 1. Modelos de datos.

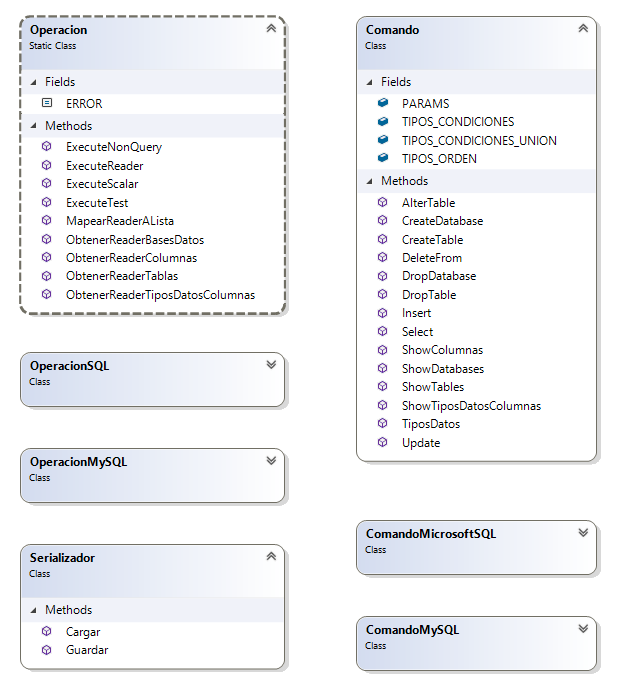
Las siguientes corresponden a las clases utilizadas para el manejo contra la base de datos utilizada por el programa para guardar usuarios y contraseñas.



UML 2. Manejo base de datos del programa.

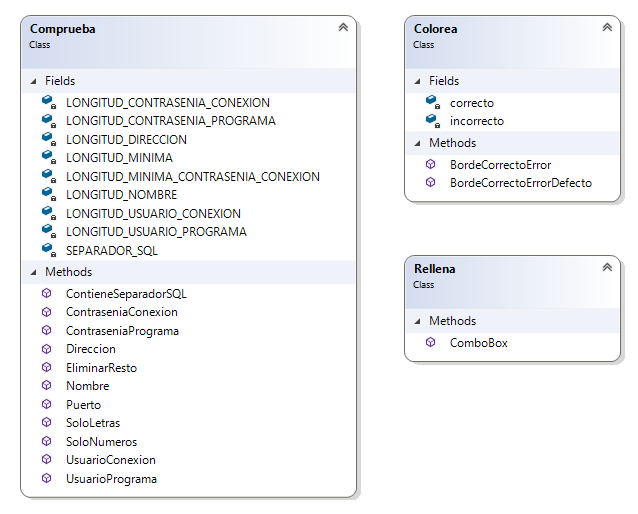
Ahora les toca el turno a aquellas usadas para lanzar consultas y comandos hacia las bases de datos que se quieran consultar. Estas clases son gran parte del núcleo del planteamiento inicial del proyecto. No se muestran aquellas llamadas \*MySQL y \*MicrosoftSQL ya que consisten únicamente en implementaciones específicas de la clase padre según la tecnología.

También se incluye la clase utilizada para serializar en disco de manera persistente los datos de las consultas en archivos con extensión \*.easy.



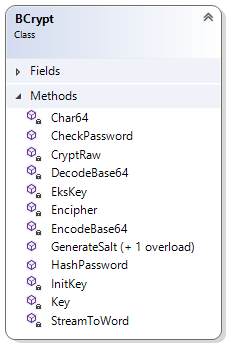
UML 3. Operaciones de manejo y consulta contra Bases de Datos externas.

Se sigue con estas clases de ayuda común para las interfaces gráficas.



UML 4. Clases del paquete Utils de código común.

A continuación, la librería utilizada para el hasheo de contraseñas en BBDD.

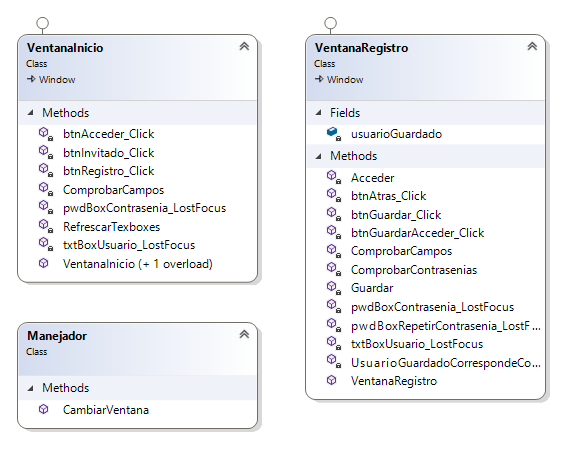


UML 5. Librería BCrypt.

Se prosigue con aquellas tablas UML relacionadas con la interfaz gráfica del usuario, o sea, las ventanas que se verán junto con su comportamiento.

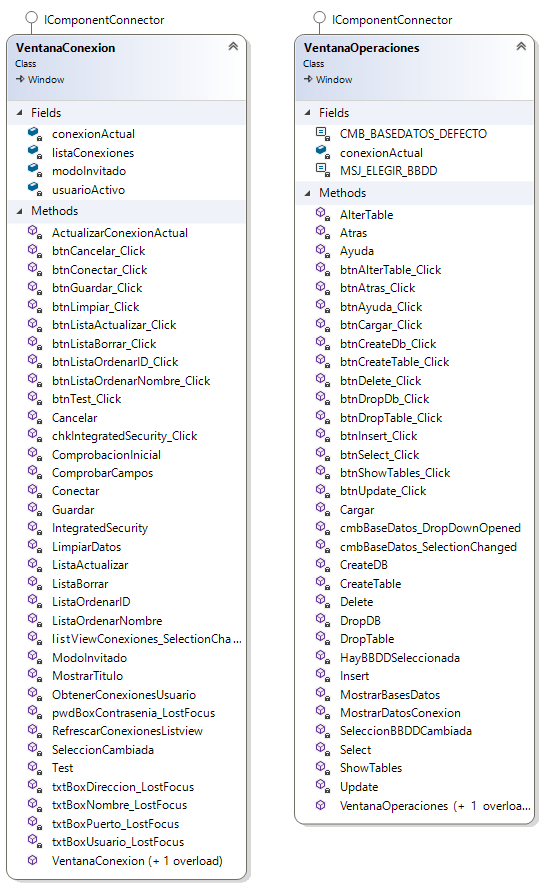
Estas se corresponden a las dos primeras ventanas que se muestran al usuario, la correspondiente con el Inicio y la ventana de Registro.

También se incluye la clase ayudante para gestionar el cambio de una ventana a otra durante toda la aplicación.



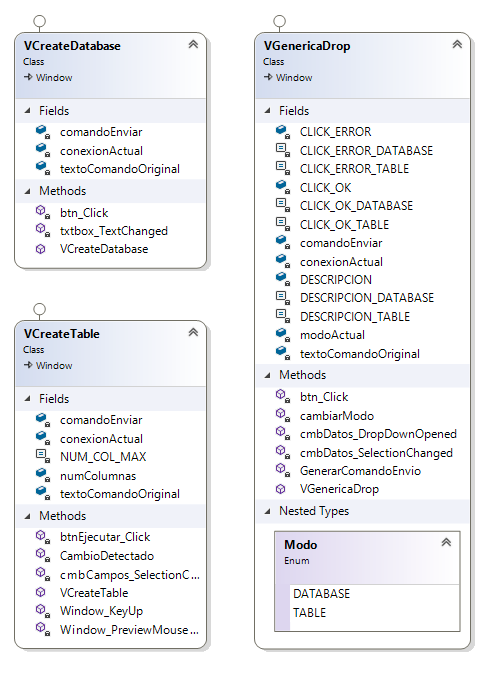
UML 6. Ventanas iniciales.

Estas corresponden a las dos grandes ventanas de entrada de datos del usuario.



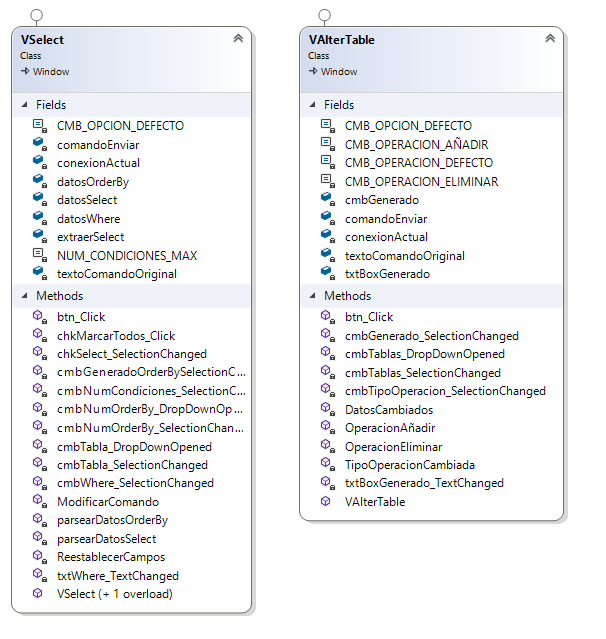
UML 7. Ventanas Conexión y Operaciones.

Se muestran ahora los UML relacionados con las ventanas relacionadas con la ejecución de comandos y consultas contra bases de datos.

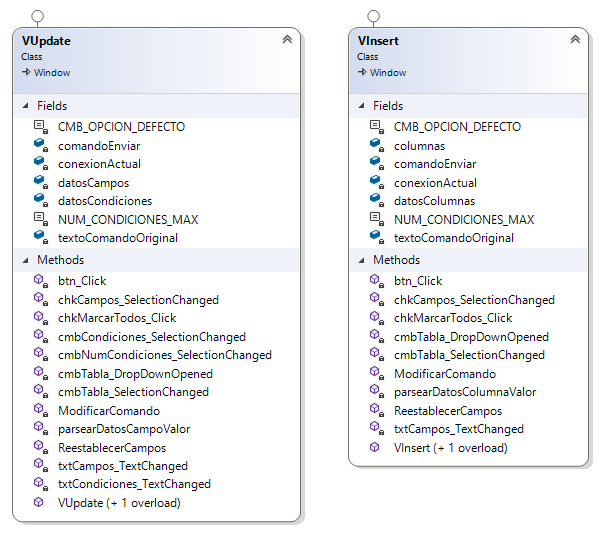


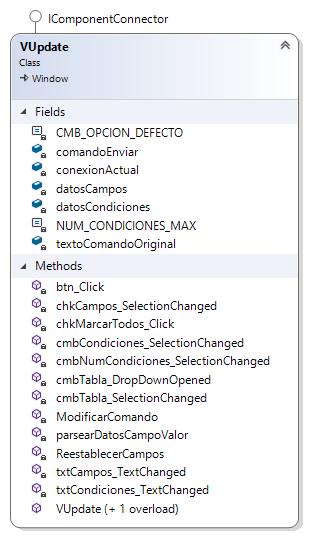
UML 8. Ventanas de comandos I.

Segunda parte de UML relacionados con ventanas de comandos



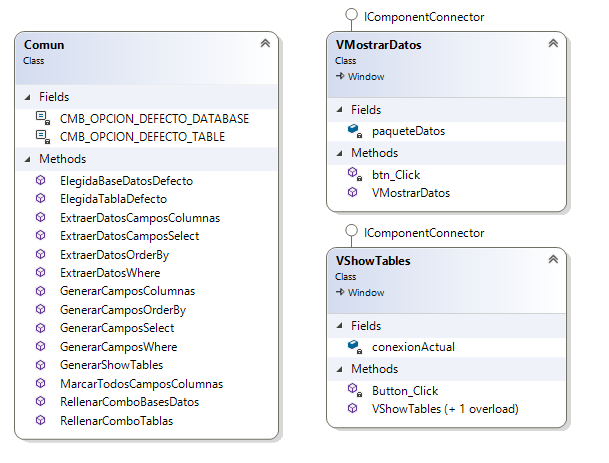
UML 9. Ventana comandos II.



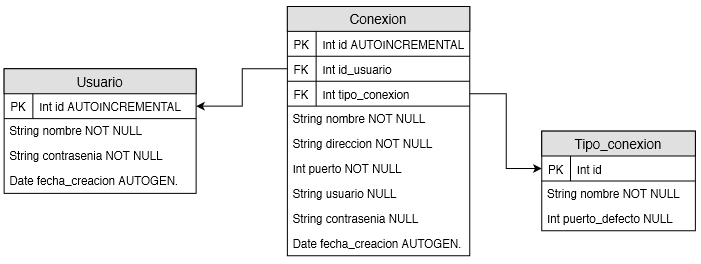


UML 10. Ventanas comandos III.

Por último se muestran las tablas UML de las dos últimas ventanas y de la clase Común que define métodos utilizados entre todas las ventanas. Al final de la página aparece el diagrama de tablas utilizado en la BBDD del programa.



UML 11. Ventanas comandos IV y Común.cs.



UML 12. BBDD del programa.

No se ha parado en entrar en detalle del comportamiento de las clases ya que estas están extensamente documentadas en el código fuente que se proporciona en las últimas páginas de esta Memoria.

* 1. **Análisis y Diagramas E/R**

La base de datos de este programa se ha reducido a mínimos ya que se usa sólo para una parte opcional de éste. Permite almacenar una lista de conexiones con sus datos de conexión a cada usuario. También permite conocer el tipo de puerto por defecto que debería tener una base de datos en caso del usuario no asignar uno.

Los usuarios y conexiones tienen almacenada de manera automática la fecha en la que se crearon para poder llevar con esos datos a cabo operaciones de estadística o ordenación por fecha de creación.

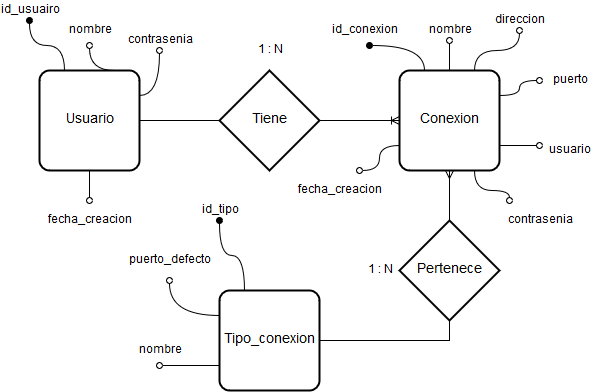


Imagen 14. Diagrama Entidad Relación de la BBDD.

1. **Futuras Mejoras y Conclusiones**

En este apartado se exponen las posibles futuras mejoras que el autor ha pensado que el proyecto pueda tener para versiones posteriores.

* Funcionalidad para añadir soporte de Foreing Keys. Estas, aunque son totalmente necesarias en bases de datos relacionales maduras, no se han decidido añadir al programa por creer que era una funcionalidad que queda más allá del ámbito de alcance de este software.

Se ha optado por incluir sólo aquellas funcionalidades básicas para la creación y manejo, para no liar al usuario final (alumnos) con conceptos más avanzados. Lo mismo se podría aplicar para constraints tipo checks, o funciones de agregación.

* Soporte para diferentes tipos de bases de datos. Aunque se ha decidido dar soporte a las dos más grandes del mercado, sería muy bien recibido el poder trabajar con otras marcas de bases de datos, como PostgreSQL.
* Añadir funcionalidad de almacenado de conexiones en modo local sin tener que estar registrado como usuario. Estas conexiones se serializarían y almacenarían en un fichero de datos.
* Posibilidad de almacenar la configuración de una consulta para no tener que repetirla de nuevo. Esto permitiría al usuario tener una lista con las consultas SQL más utilizadas y agilizaría su uso.

1. **Pruebas y Resultados**

En las siguientes páginas se muestran una serie de pruebas de control que se han realizado con el software. Estas muestran algunas de las implementaciones de seguridad y tolerancia a fallos que se han desarrollado.

El primer caso muestra un intento de registrar un usuario con un nombre que ya se encuentre registrado previamente. El software maneja la situación impidiendo crear dicho usuario y avisando a éste del fallo.

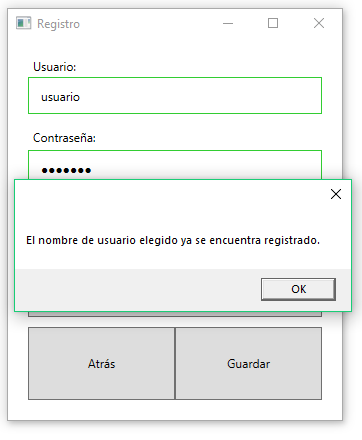


Imagen 15. Aviso de usuario ya registrado.

Este software, aun siendo una aplicación de escritorio tradicional, tiene métodos de validación que muestran de manera visual al usuario con colores como el verde o rojo cuando uno datos introducidos en los campos de tipo input tiene un fallo de formato.

Esta característica suele ser más propia de aplicaciones para móviles o web, que se centran en el diseño.

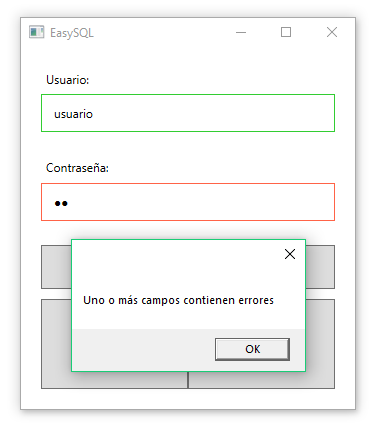


Imagen 16. Validación visual.

En la siguiente captura pantalla del software se pueden apreciar varios detalles. Por una parte, la comentada anterior validación visual. Por otro, podemos ver que al marcar el campo “Integrated Security”, una opción de autenticación para bases de datos SQL Server, inhabilita los campos usuario, contraseña y tipo de conexión, ya que todos ellos ya lo gestiona de manera automática la aplicación sin que el usuario haga nada. Por tanto, elimina la posibilidad de introducción de datos erróneos por parte del usuario.

Como mención especial, el valor del campo “Puerto” no se ha introducido, lo asigna por defecto el software teniendo en cuenta el tipo de conexión elegida.

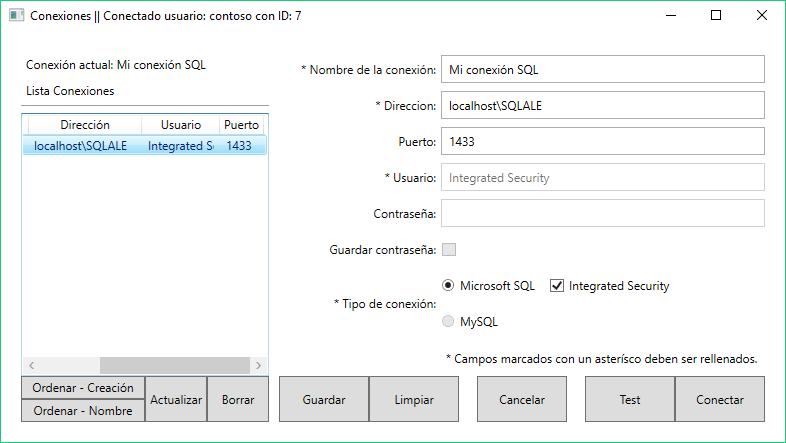


Imagen 17. Integrated Security y puerto por defecto.

A continuación, se pasa a explicar una característica que para el usuario final puede no tener demasiada importancia, pero a la hora del desarrollo de esta fue una de las partes con mayor dificultad de realizar.

Existe una característica con las bases de datos SQL Server y las aplicaciones .NET. Para agilizar y mejorar el rendimiento de las consultas, la primera vez que el software realiza una consulta contra una base de datos SQL Server, de manera transparente y automática mantiene la conexión abierta y un flujo de comprobación de estado constante corriendo. Esto provocaba que a la hora de intentar eliminar una instancia de base de datos, y en caso de haber realizado una consulta previamente contra ella, no se pudiese efectuar la operación debido a que existía una conexión abierta con ella. Esto llevó a la idea de una alerta para el usuario que mostrase un error y la pregunta de si se deseaba forzar el borrado. Al hacerlo, la aplicación manda unos comandos a la BBDD obligarla a cerrar las conexiones que estén activas en el momento con todos los usuarios. Después de eso, acaba borrando la BBDD de manera efectiva.

Como dato añadido, esto no ocurre en las bases de datos MySQL..

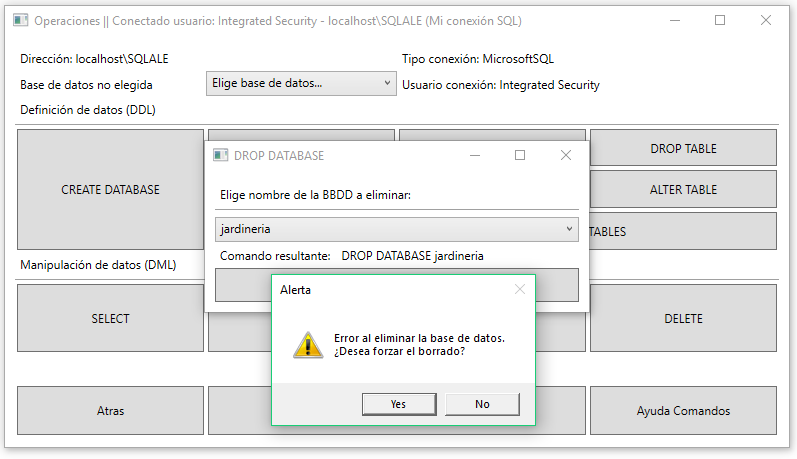


Imagen 18. Forzado de borrado SQL Server.

Ahora se expone un caso de uso y prueba. Una de las funcionalidades que tiene la aplicación es la de poder actualizar datos existentes en la BBDD. Esto se hace mediante el comando UPDATE.

Aquí se muestra un ejemplo en el que, a remarcar, informa al usuario de la cantidad de filas afectadas por el uso de la consulta.

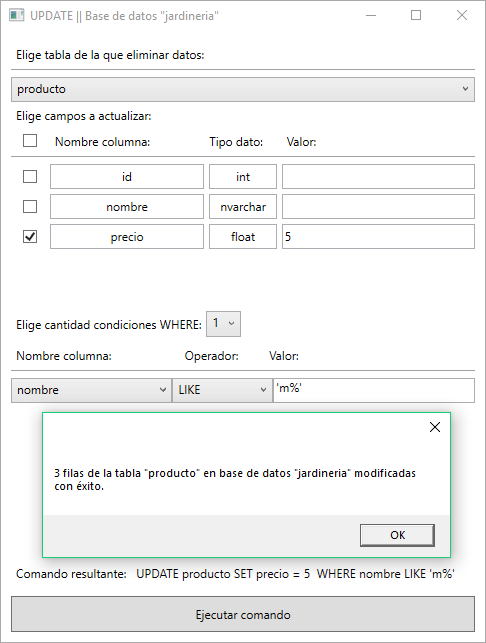


Imagen 19. Update e información de filas afectadas.

1. **Manual Usuario**
2. **Bibliografía**
3. **Índice de imágenes**

**Anexo I.- Manual del Código**

[](http://www.cenecmalaga.es/)**ELABORADO POR:** ALEJANDRO DEL RÍO MORENO.

**FECHA DE ENTREGA:** ##/06/2018.

**EMAIL:** alejandro.rio@outlook.es