**UNIVERSIDAD VERACRUZANA**

SERVICIO SOCIAL

**Sistema de Experiencia Recepcional**

Presenta:

Omar Baltazar Islas

Josué Capistrán García

Jefa de servicio:

Dra. María Karen Cortés Verdín

07 de febrero 2022

Xalapa, Ver.

# Ficha de documento

|  |  |
| --- | --- |
| **Título del proyecto** | Sistema de Experiencia Recepcional (SER) |
| **Autor** | Omar Baltazar Islas estudiante de séptimo semestre de la Lic. de Ingeniaría de Software UV campus Xalapa  Josué Capistrán García estudiante de noveno semestre de la Lic. De Ingeniería de software UV campus Xalapa |
| **Responsable** | Dra. María Karen Cortés Verdín |
| **Lugar de realización** | Xalapa, Veracruz |
| **Nombre de documento** | Documentación de avances de SER |
| **Fecha** | 03 de febrero de 2022 |

# Control de versiones

|  |  |
| --- | --- |
| **Versión** | **Descripción del cambio** |
| V1.0 | Versión inicial de documentación |

# Introducción

El presente escrito constituye la documentación básica del proyecto Sistema de Experiencia Recepcional (SER), el cual ha sido un proyecto llevado a cabo como servicio social en la facultad de Estadística e Informática, teniendo como responsable directa a la Dra. María Karen Cortes Verdín quien también ha fungido como la principal stakeholder para el sistema.

1. Generalidades sobre este documento:
   * Es el punto de unión de la documentación básica del proyecto.
   * Contiene toda la información relevante del proyecto para facilitar la lectura, comprensión y continuación del proyecto.
   * Su misión es mostrar las soluciones adoptadas y describir de forma general el objeto del proyecto para quien continúe su desarrollo.
   * Se ha procurado redactar de forma clara y comprensible, no sólo para los profesionales sino para cualquier miembro implicado, y en particular, para quien dé continuación al proyecto. La información más detallada y con mejor visibilidad, como diagramas, se encuentra en archivos diferentes.
2. Contenido de este documento
   * Contiene la descripción de todos los elementos necesarios para una correcta comprensión de quien continúe el proyecto.
   * Describe también los elementos que se deben utilizar para el proyecto y así garantizar una correcta descarga y ejecución de este.

# Objetivo del proyecto

El proyecto SER es un sistema web pensado para la gestión del historial de los alumnos de la facultad de Estadística e Informática (FEI) durante la Experiencia Educativa (EE) de Proyecto guiado y Experiencia Recepcional. Con este sistema, se busca facilitar el seguimiento del trabajo de los alumnos desde la EE de proyecto guiado y su continuación en la EE de experiencia Recepcional. De igual forma, se busca que el sistema permita un correcto registro, así como una fácil gestión, de profesores, sinodales y alumnos, así como los trabajos recepcionales que se registren y se asocien a cada uno de ellos. Lo anterior, permitirá monitorear el avance del alumno en su trabajo y registrar el estado en el que se encuentra este.

# Herramientas y Modelos

En este apartado se incluyen las tecnologías utilizadas para el desarrollo del proyecto, así como una breve justificación de la selección. De igual forma, se anexan los modelos utilizados para el desarrollo del proyecto SER.

## Tecnologías utilizadas

### ASP.NET Core

ASP.NET Core es un framework de código abierto y multiplataforma que permite la creación de aplicaciones web y APIs Web. Las aplicaciones con este framework pueden ser desarrolladas y ejecutadas en Windows, Mac y Linux, en otras palabras, no está asociada a un sistema operativo en especial. Uno de los beneficios más importantes otorgados por este framework es su mayor rendimiento.

Con las mejoras y actualizaciones que posee, el código se optimiza mucho más, lo que se traduce en un rendimiento mejorado. Una de las partes que más resalta en este beneficio es que no es necesario cambiar el código, puesto que el compilador de ASP.NET Core eventualmente optimizará todo el código cada vez que el código se vuelva a compilar. Además de lo anterior, esta tecnología contiene funciones propias de ASP.NET. Esto significa que los desarrolladores pueden optimizar fácilmente la estructura del código escribiendo declaraciones menores, lo que se refleja en menos horas para la creación de una funcionalidad y a su vez permite dar mantenimiento con mayor facilidad. Al hablar de desarrollo de aplicaciones web, es importante asegurarse de que la aplicación sea realmente compatible con todas las plataformas. Como se mencionaba al inicio de esta sección, ASP.NET Core es en realidad multiplataforma que le permite crear fácilmente aplicaciones web que se ejecutan en Windows, Linux y Mac, es decir, todo el backend usará el mismo código C #. Lo anterior ayuda a que el código pueda ser reutilizado para el desarrollo de aplicaciones en las distintas plataformas.

De forma particular al proyecto, este framework permite implementar fácilmente el patrón Model-View-ViewModel (MVVM) haciendo uso de sus Razor Pages. ASP.NET Core ofrece plantillas de inicio que reducen el tiempo de desarrollo del proyecto, lo cual fue una de las principales características que llevaron a su selección. De igual forma, su extensa documentación y sociedad son de bastante ayuda cuando se presenta un problema durante el desarrollo del proyecto, puesto que hay bastante información donde se puede encontrar la mejor solución.

### Microsoft SQL Server

Microsoft SQL Server es un sistema de gestión de datos relacional desarrollado como un servidor para otras aplicaciones que puede funcionar de forma local o conectado a través de una red. Una de las principales características de SQL Server es su alta disponibilidad al permitir un gran tiempo de actividad. Lo anterior sin la necesidad de sacrificar los recursos de memoria del sistema. Gracias a las funciones de memoria integradas directamente en el motor de SQL Server, mejora la flexibilidad y facilita su uso. Para fines del proyecto, se seleccionó este gestor ya que, para la conexión con la aplicación desarrollada con ASP.NET Core este framework ofrecía varias ventajas entre las cuales se encuentra su Object Relatinonal Mapper (ORM) Entity Framework. Haciendo uso de ese ORM, se pudo mapear de forma automática la base de datos a la solución y así poder realizar operaciones a ella de forma sencilla y segura. Otra de las razones por las cuales se seleccionó este gestor es la posibilidad de crear vistas de forma gráfica y sencilla haciendo uso de Microsoft SQL Server Management Studio, con ellas y los procedimientos almacenados, se puede reducir significativamente la carga al código del proyecto.

### Docker

Docker es una plataforma que permite a los usuarios empaquetar, distribuir y administrar aplicaciones dentro de contenedores. En otras palabras, es un proyecto de código abierto que automatiza la implementación de aplicaciones dentro de contenedores de software. Para el caso del proyecto, de momento únicamente se tiene la base de datos en un contenedor que más adelante se explicara, pero se pretende que toda la solución se introduzca a un contenedor para una ejecución automatizada de todo el entorno necesario para la misma.

## Modelos

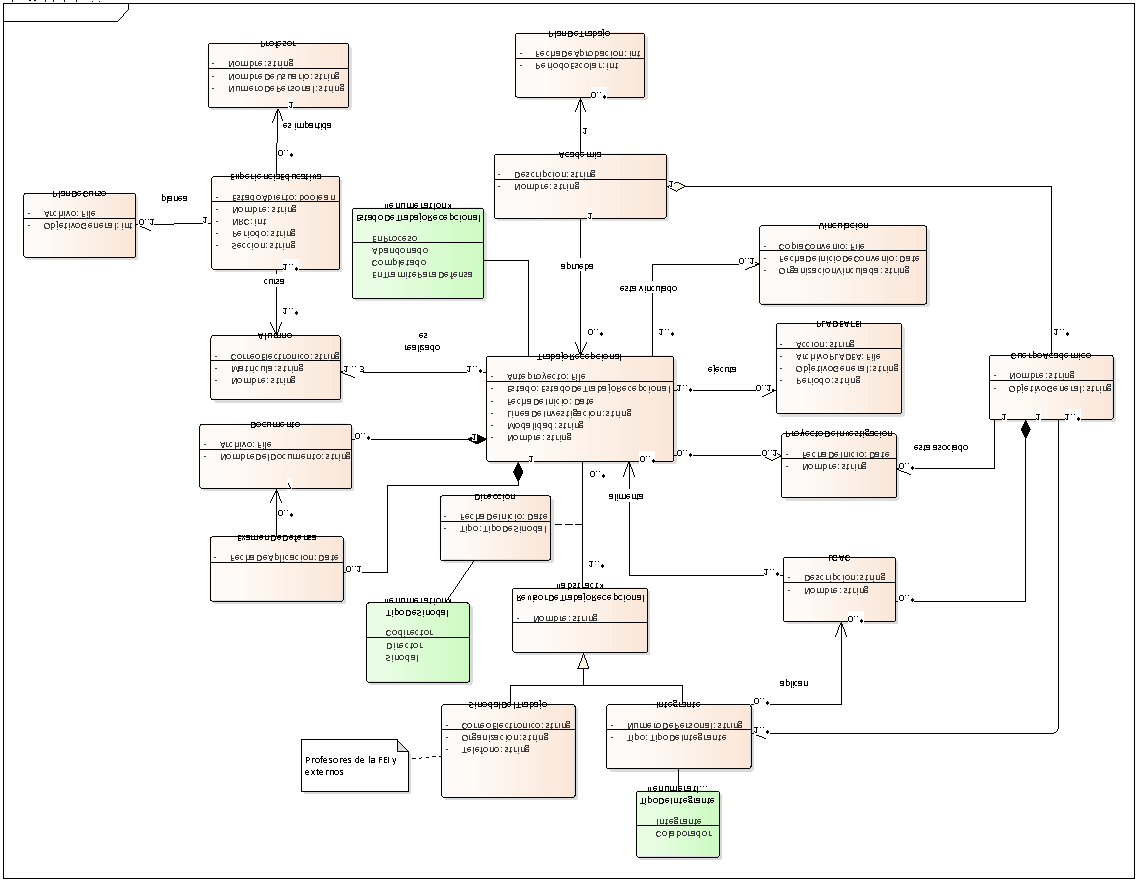
### Diagrama de paquetes

Gráfico, Gráfico de cajas y bigotes

Descripción generada automáticamente

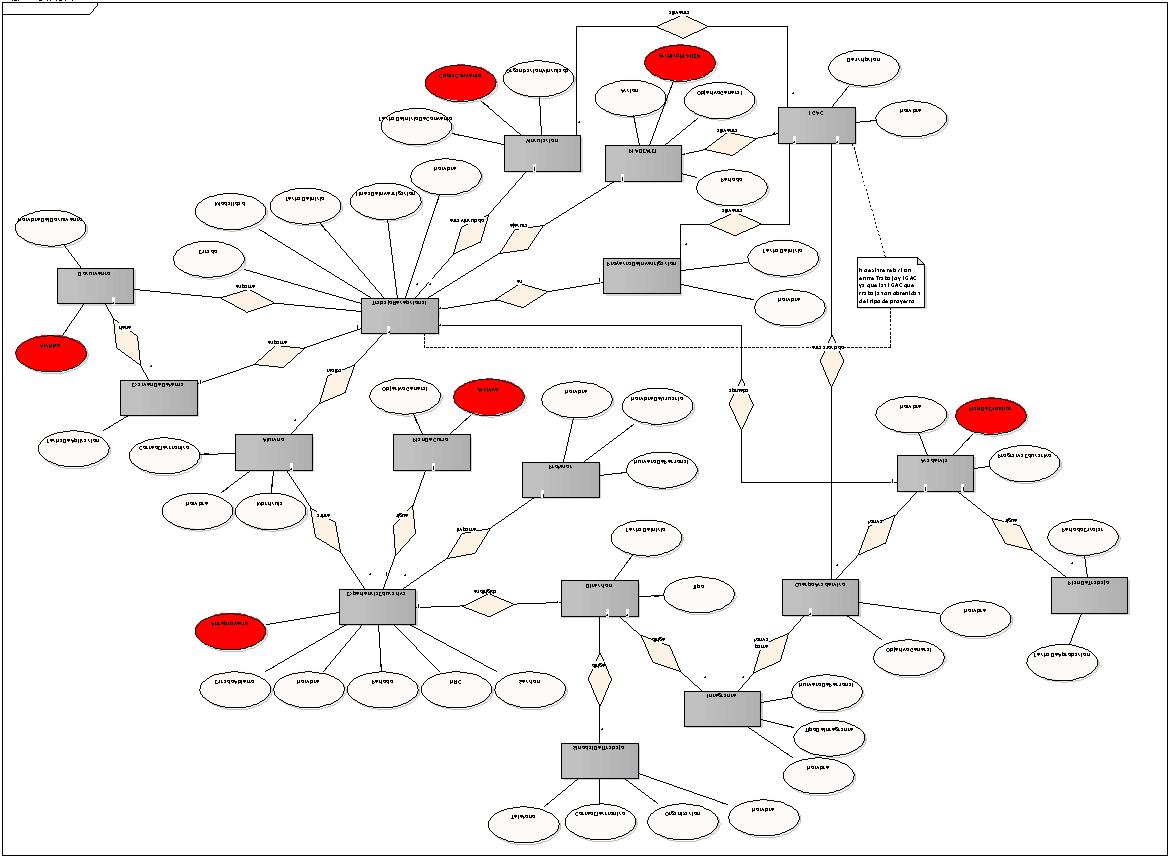
Este diagrama representa la arquitectura básica del proyecto. Como se puede observar, el patrón seleccionado es el Model-View-ViewModel. De forma general, este patrón indica que existe una vista (que es la interfaz gráfica), la cual está conectada con un modelo de vista el cual procesa la información y puede contactarse a los modelos, mismos que están conectados a la capa de datos.

### Modelo de dominio



En la ilustración 27 podemos apreciar el modelo de dominio para el proyecto SER. Dicho modelo es una representación de las clases conceptuales del problema a resolver. En él se muestran las asociaciones necesarias para las distintas entidades involucradas. Así mismo, se muestran los atributos de cada una de las entidades del sistema.

### Diagrama de Entidad-Relación



En la ilustración 28 se aprecia el modelo de datos. En él se muestra la estructura lógica de la base, incluidas las relaciones y limitaciones que determinan cómo se almacenan los datos y cómo se accede a ellos. De igual forma, se muestran los datos que se almacenarán para cada entidad involucrada en el sistema.

# Situación actual del proyecto

Actualmente el proyecto cuenta con dos funcionalidades las cuales son el CU02 Registrar documento de proyecto guiado y el CU09 Registrar sinodal. Para ambos casos de uso se ha desarrollado su plan de pruebas el cual ha sido entregado con el resto de los artefactos del proyecto. Como se mencionó en el apartado de tecnologías, las funcionalidades fueron desarrolladas haciendo uso de ASP.NET Core, utilizando el modelo MVVM. Para lo anterior, se han utilizado las Razor Pages de ASP.NET Core. A continuación, se presentarán algunas recomendaciones para la continuación del proyecto.

## Recomendaciones para continuación del proyecto

1. Se recomienda seguir utilizando el patrón MVVM ya que permite facilita el manejo de la información sin tener un controlador acoplado únicamente a una entidad de la base de datos.
2. Se recomienda utilizar las vistas dentro de la base de datos para las consultas complejas. Por ejemplo, para el caso de uso 02 Registrar documento de proyecto guiado, se elaboró una vista la cual recupera en una tabla datos de los estudiantes que cursan proyecto guiado, además de sus trabajos y otros datos adicionales. En lugar de hacer una línea de código extensa desde ASP.NET Core, se elaboró una vista directamente en SQL Server. Esto permite reducir la carga del código.
3. Se recomienda utilizar procesos almacenados para registro de entidades asociativas. Existen algunas entidades asociativas como *AlumnoTrabajoRececpcional.* Estas entidades no se encuentran mapeadas de forma independiente como el resto de las entidades ya que Entity Framework las incluye como atributos *ICollection* dentro de las entidades que asocia. Es por ello, que se recomienda utilizar procesos almacenados para hacer registros que involucren a estas entidades, tal podría ser el caso del CU04 Registrar trabajo recepcional.
4. Se recomienda tener precaución al actualizar los modelos dentro de ASP.NET Core ya que si se tiene código (como algún método agregado por el desarrollador) al actualizar dichos modelos con el comando *Scaffold*, como más adelante se mencionará, el código podría eliminarse ya que este comando realiza un nuevo mapeo de la base de datos y por ende sobrescribe los modelos.
5. Se recomienda que cualquier cambio que se decida sobre la arquitectura o la interfaz sea plasmado en sus artefactos correspondientes (diagramas, prototipos, etc.), esto con el fin de mantener una buena documentación del proyecto y facilitar su comprensión a futuros involucrados.

# Instrucciones para ejecución del código

En este apartado se enlistan las instrucciones necesarias para descargar, ejecutar y así poder continuar el desarrollo del proyecto SER.

## Instrucciones para descarga y ejecución del proyecto SER

1. Instalar Visual Studio Community. Al momento de instalar seleccionar los siguientes paquetes:
   * ASP.NET Core
   * ASP.NET
   * HTML
   * JavaScript
   * contenedores
2. Instalar Docker Desktop. Este paso es opcional pero facilita la ejecución del contenedor en el cual se aloja la base de datos.
3. Clonar el repositorio de SERWebApplication en el siguiente link 0<https://github.com/Obaltazar2k/SERWebApplication.git>
4. Dentro de la carpeta principal del repositorio crear dos carpetas nuevas y vacías, una de ellas con el nombre de **logs** y otra con el nombre de **data**.

Captura de pantalla de computadora

Descripción generada automáticamente

1. Una vez clonado el repositorio y creadas las carpetas, es necesario crear el contenedor utilizando el archivo **docker-composer.yaml**. Para ello, es necesario utilizar una terminal, ubicarse en la carpeta principal del proyecto, que es donde se encuentra el archivo, y ejecutar el comando docker-compose up.

**Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente**

Esto ejecutará el archivo el cual contiene las especificaciones para crear el contenedor, una vez que se levante dicho contenedor, desde Docker desktop podemos comprobar que se ejecuto el archivo con éxito.

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

1. Con el contenedor corriendo, ingresar a Microsoft SQL Server Management Studio para poder ejecutar el script de la base de datos que utiliza el proyecto. Se debe utilizar el servidor de **localhost**, la autenticación de SQL Server Authentication con el usuario: **sa** y la contraseña **qwerty\*1234**.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

1. Una vez dentro, sin crear una base de datos, se debe ejecutar el archivo **Script\_DataBase.sql** que se encuentra en la carpeta **Database**.

Captura de pantalla de un celular

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Sitio web

Descripción generada automáticamente

Esto creará la base de datos con todas las entidades y relaciones dentro del contenedor creado anteriormente. De esta forma, el proyecto podrá conectarse a esa base de datos para poder trabajar con ella.

1. Una vez creada la base de datos, se debe ejecutar el archivo **Script\_DatosSER.sql**, que se encuentra en la carpeta **Database**, para tener registros dentro de la base de datos y poder hacer pruebas con ellos durante el desarrollo del proyecto. Para ello es necesario asegurarse que la base de datos predeterminada es la de SER.

Captura de pantalla de un celular

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

1. Posteriormente debemos abrir el archivo del código en Visual Studio para corroborar que los paquetes NuGet estén instalados, en caso contrario se deben instalar los que se muestran a continuación.

Captura de pantalla de computadora

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Captura de pantalla de computadora

Descripción generada automáticamente

1. Por último solo queda ejecutar el programa para corroborar que no hay algún error.

Texto

Descripción generada automáticamente

## Actualización de modelos mapeados de la base de datos

Es posible que durante el desarrollo del proyecto sea necesario modificar la base de datos y por ende actualizar los modelos mapeados dentro de la solución de ASP.NET Core de Visual Studio. Para ello es necesario realizar lo siguiente:

1. Abrir la consola de del administrador de paquetes.

Captura de pantalla de un celular

Descripción generada automáticamente

1. Dentro de la consola se debe introducir el siguiente comando:

**Scaffold-DbContext "Data Source=localhost; Database=SER; User ID=sa;Password=qwerty\*1234;" Microsoft.EntityFrameworkCore.SqlServer -OutputDir Models/DB -f**

**Captura de pantalla de computadora

Descripción generada automáticamente**

1. Con lo anterior, se actualizarán de forma automática los modelos mapeados de la base de datos. No obstante es importante tener en cuenta que al hacer esto, lo que hay dentro de cada uno de los modelos se sobrescribe, por lo que si se tiene algún método o cambio que se haya agregado de forma manual a algún modelo, es probable que se pierda.

# Recomendaciones para trabajo futuro

En este apartado se enlistan algunas de las acciones que se deberían incluir en el desarrollo futuro del proyecto para un mejoramiento de este.

* Dockerizar la aplicación: Como se pudo observar durante la instalación, la base de datos se debe crearse cada que alguien nuevo descarga el proyecto, para evitar esto podría dockerizarse la aplicación completa para que únicamente al ejecutar el docker-compose, se cree todo el entorno necesario para el mismo.
* Elaborar pruebas dentro del código: Por cuestiones de tiempo, únicamente se realizaron pruebas de forma manual siguiendo el plan de pruebas redactado, sin embargo, sería una buena práctica realizar pruebas con selenium, Chrome driver o alguna otra herramienta para tener pruebas automatizadas del proyecto. De igual forma, podría considerarse crear workflows dentro de github para que se realicen pruebas automáticas al hacer commits o merges en las diferentes ramas.