Hola, mi nombre es Felipe Gavilán y te doy la bienvenida al curso de Blazor donde aprenderemos

a desarrollar aplicaciones de punto net que van a correr en el navegador.

También veremos cómo interactuar con una base de datos a través de un web API en ese punto.

Net Core.

En este primer módulo comenzaremos hablando de Blazor.

Veremos sus ventajas y lo que significa utilizar velas.

Más adelante hablaremos de los tres modelos de alojamiento de Blazor.

El modelo del lado del cliente o Blazor o Web Assembly.

El modelo del lado del servidor y el modelo híbrido para aplicaciones desktop y móviles.

Luego veremos dónde podemos utilizar Play Store y cuál es el criterio para saber si nuestros clientes

podrán correr aplicaciones de ellas en Play Store.

Finalmente prepararemos nuestro ambiente de desarrollo, ya sea que queramos utilizar Visual Studio

o Visual Studio Code.

Con el Dot Net Cli.

Blazor es un framework que nos permite crear aplicaciones interactivas en sitios web que van a ser

utilizadas a través de un navegador web.

Históricamente, si queríamos crear una aplicación que corriera en el navegador, estábamos forzados

a utilizar JavaScript.

Ahora, gracias a Web Assembly, podemos utilizar otros lenguajes de programación en el navegador.

Entre ellos podemos utilizar Sharp y es Blay sobre el framework que nos la pone cómoda para escribir

código de Sharp en el navegador utilizando un paradigma de componentes.

La pregunta es qué ventajas nos trae Blazor o en general utilizar punto net en el navegador?

Pues primero una de las principales es poder aprovechar el ecosistema de punto net para nuestras aplicaciones

front end.

Segundo, poder utilizar Sharp, el cual es un lenguaje bastante poderoso.

Esto incluye LINQ y programación asíncrona con A5 tuits.

Tercero, tener la habilidad de compartir código entre el front end y el backend de nuestra aplicación

de ASP.NET Core.

Y 4.º, podemos trabajar con componentes.

Estas y otras ventajas las exploraremos a lo largo del curso.

Sin embargo, estas son las más importantes a mi consideración.

Además de utilizar código de Sharp, si lo queremos, podemos llamar código de JavaScript desde nuestras

aplicaciones de Play Store.

Y claro, desde el código de JavaScript podemos invocar código de Sharp.

No todas las aplicaciones de Blazor son iguales.

Existen tres modelos de alojamiento, los cuales definen la manera en que tu aplicación va a funcionar:

* Estos son los modelos del lado del cliente,
* del lado del servidor
* el modelo híbrido.

Comenzaremos hablando del modelo del lado del cliente.

Como dijimos con Blazor podemos tener aplicaciones de .net corriendo en el navegador.

Esto aplica principalmente a las aplicaciones que utilizan el modelo del lado del cliente.

A estas aplicaciones les llamamos **Blazor web Assembly**.

Utilizar el modelo del lado del cliente implica descargar el runtime de .net en el navegador junto con las

DLL necesarias para coger tu aplicación y cogerlo todo dentro del navegador.

Cuando estamos creando un proyecto de blazor web Assembly tenemos dos opciones.

La primera es solamente crear el lado del cliente en ésta solamente se van a utilizar archivos estáticos

cuando creas un proyecto de esta manera solamente tienes un proyecto de front end.

En ese proyecto puedes tener tu código de sillar que se va a ejecutar en el navegador y claro desde

este proyecto vas a poder realizar peticiones HTTP hacia un Web API realizado en cualquier tecnología

ya sea este .net Core, Go, node.js, entre otros.

La segunda opción es alojar esta aplicación asp.net core.

De esta manera se nos crea un proyecto de front end y un proyecto de back end el cual va a servir como

un web API para nuestra aplicación.

Esta configuración es ideal cuando vas a tener un front end back ensillar los cuales quieres desplegar

de manera simultánea.

En este modelo podemos compartir código entre el front end y el Bakken en este curso nos vamos a enfocar

en el modelo de Blazor web Assembly alojado en asp.net Core ya que éste nos va a ayudar a desarrollar

aplicaciones completas que vayan desde el front end hacia un back end conectado con una base de datos

en el modelo del lado del servidor.

También podemos hacer esto pero tienen unas implicaciones que veremos en el próximo vídeo.

Por ahora sigamos hablando del modelo del lado del cliente tal como dijimos en una aplicación de Bloy

sólo del lado del cliente.

Descargamos el frontend en punto net hacia el navegador y ejecutamos el código de Sharp desde ahí utilizando

web Assembly.

Esto es algo que ocurre automáticamente de lo cual Blazor se encarga.

Esto es completamente transparente para nosotros sólo nos tenemos que limitar a programar como siempre

lo hemos hecho y Bloise para el resto es claro que Bloy será el cliente solamente puede funcionar en

navegadores donde se pueda utilizar web Assembly.

Lo bueno es que cualquier navegador moderno incluso los navegadores de celulares soportan web Assembly.

Esto significa que no podemos utilizar este modelo de Play Store en navegadores antiguos como Internet

Explorer el modelo del lado del cliente tienen la ventaja de que es altamente escalable a través de

éste es posible servir a muchos usuarios de manera simultánea.

Esto es así porque nuestra aplicación al menos el front end corre en el navegador del usuario lo que

nos permite quitarle dicha carga al servidor.

Una de las desventajas más notables de este modelo es que se necesita descargar el runtime de .net

y las librerías necesarias para acoger toda aplicación de .net en el navegador.

Esto podría implicar una descarga de incluso unos 2 megabytes de acuerdo con el tamaño de la aplicación.

Destacamos que esta descarga solo se debe de hacer una vez ya que muchos de estos archivos se van a

guardar en caché.

El segundo modelo de alojamiento que vamos a ver es el del lado del servidor.

En el caso de placer del lado del servidor, la aplicación de Blaze or corre en el servidor y el cliente

interactúa con ella a través de una conexión de signalR.

Consigna lar Manejamos comunicación en tiempo real.

Esto hace entonces que el cliente no tenga que descargar el runtime de punto net, sino que simplemente

interactúe con la aplicación de manera remota.

Es claro que el caso de las aplicaciones de Blazor del lado del servidor no tenemos la inherente

descarga de los dos megabytes para correr la aplicación, lo que hace que la aplicación servida del

lado del servidor cargue más rápido.

El no tener que cargar las aplicaciones de Blazor en el navegador implica que incluso dispositivos con

menos recursos deberían poder correr la aplicación sin problemas.

Pues el trabajo pesado queda delegado al servidor.

En consecuencia de lo anterior, las limitaciones del Blazor sobre el lado del servidor se relacionan

con el servidor.

Por ejemplo, siempre vamos a necesitar tener un servidor disponible para correr en nuestra aplicación

de Blazor.

Esto quiere decir que la aplicación deja de funcionar si algo ocurre en el servidor, dado que el servidor

maneja la aplicación y cada interacción con ella.

Esto significa que puede ser difícil servir a muchos usuarios desde un servidor de recursos ilimitados.

Además, si cada interacción del usuario con la aplicación debe ser manejada por el servidor, entonces

esto quiere decir que habrá latencia, lo cual puede hacer que tu aplicación se sienta un poco lenta.

En el caso de que la conexión de Internet del cliente no sea muy buena.

El modelo híbrido de Blazor nos permite tener aplicaciones que se ejecuten de manera nativa en ambientes desktop y móviles.

Básicamente lo que se hace es tener una aplicación nativa, la cual utiliza un control de Web View Para

poder visualizar la aplicación de Blazor, podemos utilizar Blazor híbrido con distintas tecnologías

de punto NET como Win Form, WPF y MAUI con WinForm y WPF.

Podemos crear aplicaciones desktop y utilizar componentes de Blazor en ellas.

Con MAUI podemos crear también aplicaciones desktop y aplicaciones móviles, es decir, aplicaciones para celulares.

Como te lo puedes imaginar, la ventaja de usar Blazor Híbrido es poder compartir código entre aplicaciones web, desktop y móviles.

También está el hecho de poder reutilizar tus habilidades de desarrollo web en aplicaciones desktop y móviles.

Otra cosa es que las aplicaciones híbridas tienen acceso completo a las capacidades nativas del dispositivo.

Sin embargo, existen unas limitantes que vale la pena mencionar.

A diferencia de una página web, cuando hacemos una app nativa, tenemos que desplegarla con un proceso aparte.

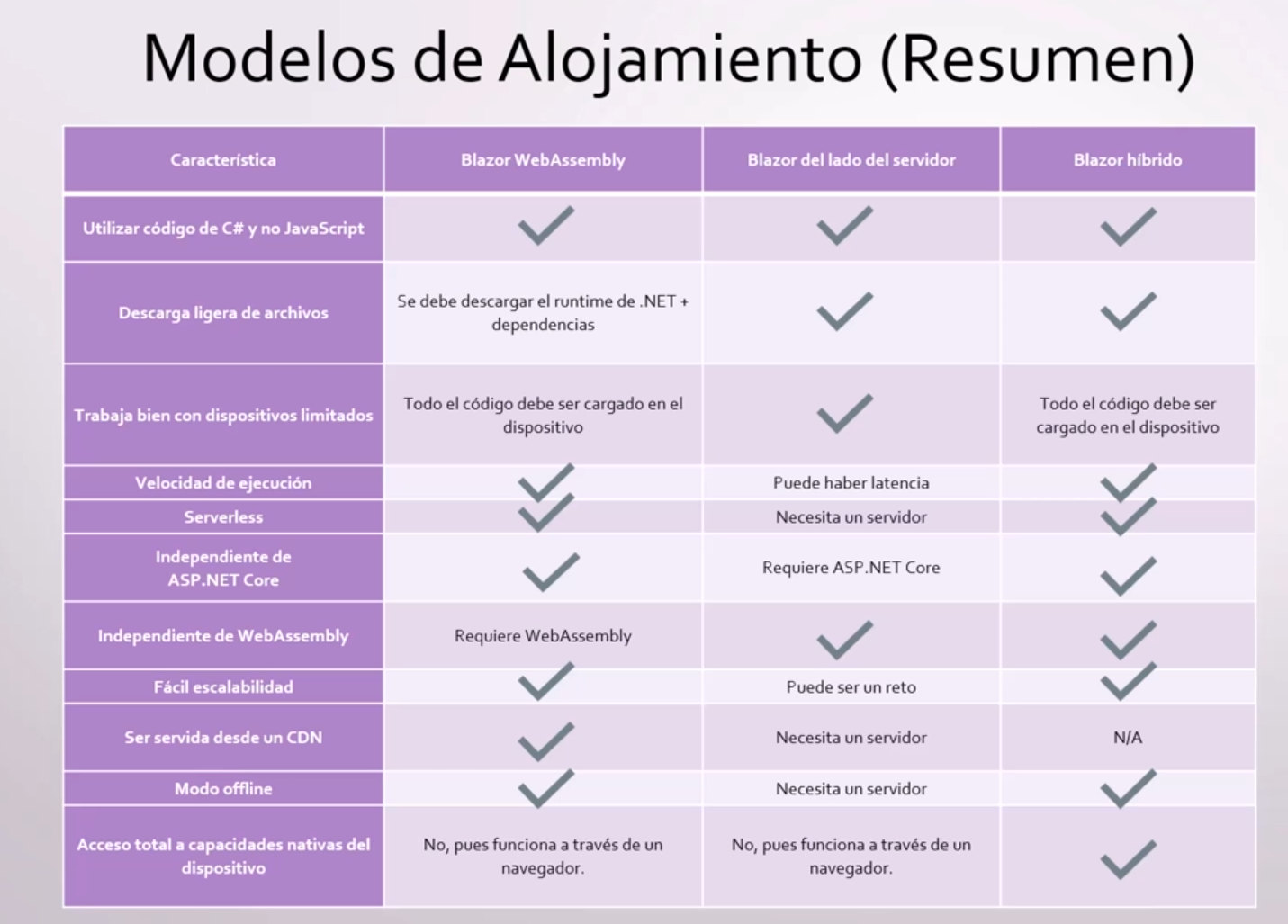
Por ejemplo, si lo que tenemos es una app de Android, pues tenemos que montarla en el Play Store.

A diferencia de una página web, la cual la montas una vez en un servidor web y listo.

Otra cosa es que es más lento para un usuario tener que buscar tu app, descargarla e instalarla para

usarla en el caso de una aplicación web.

El usuario simplemente debe ingresar a la aplicación utilizando una URL.



Cómo podemos ver cada modelo de alojamiento de Blazor tiene sus ventajas y desventajas.

Vamos a hacer una tabla comparativa a modo de resumen de las diferencias y similitudes entre los modelos

de alojamiento vistos anteriormente.

Primera característica utilizar código de sitio web y no JavaScript.

Todos los modelos cumplen con esta característica tanto placer de lado del cliente o blazer o de Blaisdell

del lado del servidor y blazer híbrido te permiten utilizar código de Sharp y no necesariamente utilizar

código de JavaScript para tener una aplicación interactiva.

Descarga ligera de archivos Blazer del lado del cliente funciona descargando una versión especializada

del runtime de punto NET y sus dependencias.

Por tanto, no hay una descarga ligera de archivos, lo que quiere decir que puede durar un poquito

para cargar una aplicación de blazer web Assembly.

Sin embargo, Blaze del lado del servidor no tiene este problema porque como corre en el servidor,

pues no hay nada que descargar.

Básicamente lo mismo ocurre con Blaze híbrido, ya que es una aplicación instalada, no hay que estar

descargando archivos para usarla.

Me refiero a que cada vez que vayan a usar la aplicación no haya que estar descargando archivos.

Por cierto, eso de estar descargando archivos por parte del cliente en el caso de Blazer Web Assembly

solamente es un problema para la primera carga de la aplicación, ya que en subsiguientes cargas se

utiliza el caché.

Trabaja bien con dispositivos limitados.

En el caso de Blazer Web Assembly, todo el código debe ser cargado en el dispositivo, lo cual requiere

que el dispositivo haga un trabajo, mientras que en el caso de Bless o del lado del servidor, esto

no es un problema porque la aplicación corre en el servidor.

En el caso de blazer híbrido es lo mismo que las pruebas en el sentido de que todo el código va a ejecutarse

en el dispositivo.

Velocidad de ejecución Blizzard web es bastante rápido.

Blazer del lado del servidor puede tener cierta latencia porque aunque tú estás visualizando la aplicación

en tu navegador, la aplicación se está ejecutando de manera remota y por tanto, cada interacción que

tú tengas con ella significa irse a un servidor a comunicarse con la aplicación y obtener una respuesta,

lo cual implica latencia.

Blazer híbrido, igual que el blazer web Assembly, tiene la máxima capacidad de velocidad de ejecución

server.

Es decir, funcionar sin un servidor web activo.

Blazer Web Assembly y Blazer híbrido.

Ambos pueden funcionar sin un servidor, mientras que blazer del lado del servidor siempre necesita

un servidor activo para ser utilizado.

Independiente de ASP.NET Core Blazer del lado del cliente o Web Assembly puede ser utilizado en un servidor

web sin problemas igualmente.

Blazer híbrido En ocasiones podemos utilizar Blazer Web Assembly con ese, pero no es obligatorio,

mientras que blazer del lado del servidor siempre requiere a spp.

Net Core independiente de Web Assembly Es claro que la Web Assembly no es independiente de Web Assembly,

es decir, siempre se necesita la capacidad de ejecutar código de huevas en Bleek.

Por otro lado, Blaze orientado al servidor y Blaze or híbrido no tiene ninguna dependencia con Ueva

Assembly.

Fácil escalabilidad.

Blazer Web Assembly se ejecuta en el navegador del usuario, mientras que híbridos se ejecuta en el

dispositivo del usuario de manera nativa.

Así que por tanto, tienen la máxima capacidad de escalabilidad posible porque 1 millón de usuarios

pueden utilizar sus respectivos dispositivos para ejecutar la aplicación, y esto no implica que la

aplicación va a ser lenta para los demás usuarios.

Por otro lado, Blazer del lado del servidor tiene un problema y es que como la aplicación corre en

un servidor.

Ese millón de usuarios estaré utilizando la aplicación en el mismo servidor, por lo que la escalabilidad

puede ser un reto en el caso de pasar del lado del servidor, ser servida desde un CDN.

CDN significa Content Delivery Network y esta es una estrategia que podemos utilizar para alojar archivos

estáticos.

Blazer Web Assembly puede ser servida como un conjunto de archivos estáticos, lo cual hace el hosting

mucho más barato y a veces hasta gratuito.

Por otra parte, Blaze Or del lado del servidor no puede ser servido como archivos estáticos y por tanto

no puede utilizar un CDN, lo que quiere decir que necesita un servidor.

En el caso de blazer híbrido, esto no aplica ni siquiera modo offline, es decir, que se puede utilizar

sin Internet.

Blaster web se puede utilizar sin Internet.

Tú puedes descargar una aplicación de pruebas y si no tienes Internet porque se te fue por un momento,

no hay ningún problema.

Puede seguir utilizando la aplicación siempre que ésta sea programada para funcionar de manera offline.

Lo mismo con Blaze Or Híbrido, porque recordemos que sería una aplicación nativa tanto Desktop como

móvil.

Por otra parte, Blaze del lado del servidor siempre necesita una conexión a un servidor y por tanto no

puede ser utilizada en modo offline.

Acceso total a capacidades nativas del dispositivo Blazor web Asembly.

No puede hacer esto.

No puede acceder a toda la capacidad nativa del dispositivo porque funciona en el navegador.

Y el navegador básicamente funciona como una caja.

Una caja que impide que nuestra aplicación pueda hacer todo lo que quisiera hacer.

No puedo acceder a determinados recursos de la computadora, etc, etc.

Lo mismo ocurre con ordenador servidor.

Sin embargo, Blazer hibrido si tiene acceso total a las capacidades nativas del dispositivo, porque

Blazer Híbridos representa una aplicación nativa y por lo tanto tiene acceso a todas las capacidades

nativas del dispositivo.

Así que como podemos ver, el modelo de alojamiento que escojamos depende mucho de nuestras necesidades.

Me gusta la libertad que ofrece el modelo del lado del cliente, pues nos permite tener flexibilidad

en la tecnología del servidor que queramos utilizar.

No requerimos necesariamente ASP.NET Core.

Podemos incluso servir nuestra aplicación desde un CDN o desde cualquier servicio que nos permita alojar

archivos estáticos, lo cual, como hemos dicho, es mucho más económico que otros servicios.

En este curso nos enfocaremos principalmente en el desarrollo de aplicaciones con Web Assembly.

De todos modos, lo que aprendamos es aplicable tanto para Bless o al lado del servidor como para blazer

híbrido.

Puede haber una que otra diferencia, pero los cambios son mínimos y gran parte de lo que aprendas será

aplicable a cualquier modelo de alojamiento de BlazorPrincipio del formulario

Donde podemos utilizar las distintas versiones de Blazor, Blazor Web Assembly.

Solamente puede ser utilizado en navegadores que soporten Web Assembly.

Cuáles son los navegadores que soportan Web Assembly?

Pues utilizando el servicio Can Eus podemos ver que el soporte global de Play es del 96%.

Eso quiere decir que el 96% de los usuarios de internet utiliza navegadores que soportan Blaster Web

Assembly.

La gran mayoría de navegadores modernos soportan Blazer Web Assembly como Chrome, Edge, Safari, Firefox,

Opera, entre otros.

Los que vemos en pantalla que tienen la columna totalmente roja como Internet Explorer, Opera, Mini

Key Oes Browser, son aquellos que no soportan Web Assembly.

En el caso de Blazor, del lado del servidor, básicamente cualquier navegador moderno puede utilizarlo

porque como hemos visto, ni siquiera tenemos una dependencia con Web Assembly.

En el caso de Blazor del lado del servidor, porque ahí la aplicación se ejecuta en el servidor

y no en el navegador del usuario.

Sin embargo, la documentación oficial de Microsoft nos dice que de los navegadores viejos, Internet

Explorer 11 puede soportar Blaze del lado del servidor, pero tiene que utilizar Poly Fields, es decir,

solamente Internet Explorer 11 puede soportar blaster del lado del servidor.

Las versiones anteriores de Internet Explorer no soportan ni siquiera blazers del lado del servidor.

Ya luego de ahí, todos los navegadores modernos sí soportan blazers del lado del servidor.

Por otra parte, sabemos que podemos utilizar Bleu Shore en aplicaciones de stop de manera nativa y

también en aplicaciones móviles, también de manera nativa, utilizando Blazor Híbrido.

Así que si tú necesitas correr tus aplicaciones en ambientes de escritorio o móviles de manera nativa,

puedes utilizar Blazor Híbrido.

Final del formulario

Principio del formulario

Vamos a aprender a preparar el ambiente de desarrollo para aquellas personas que quieran utilizar Visual

Studio para este curso.

Para eso primero vamos a ir a Visual Studio Punto Makers o Puntocom y vamos a venir acá abajo y vamos

a ver que tenemos tres versiones de Visual Studio, la versión de Windows, la versión de Mac y la versión

de Visual Studio Code que funciona para Windows, MacOS y Linux.

Vamos a hablar de Visual Studio Kota en el siguiente vídeo.

Este vídeo es para aquellas personas que quieran utilizar Visual Studio.

Entonces tú puedes venir acá y descargar la versión Community si quieres una versión gratuita para trabajar.

Entonces tú le das aquí y se va a descargar.

Le das a Skip y le puedes dar acá para comenzar el proceso de instalación.

Le das a continuar y en mi caso ya yo tengo Visual Studio instalado.

Fíjate que es la versión 7,4, que es la más reciente al momento de grabar este video.

Si tú estás utilizando una versión posterior a esta, no hay ningún problema, pero como mínimo deberías

utilizar la 17.4.

Si tú no has instalado previamente Visual Studio en tu computadora, te va a salir de la siguiente pantalla.

Yo le voy a dar aquí a modificar, lo cual me permite venir esta pantalla de cargas de trabajo.

Estas cargas de trabajo son las que definen las herramientas que yo voy a tener disponible a la hora

de trabajar en Visual Studio.

En mi caso, para este curso yo necesitaría activar esta que hace punto net y Web Development, esto

para hacer desarrollo web también.

Yo tengo muchas opciones, por ejemplo aquí yo tengo mails, tengo para aplicaciones de esto, etc pero

como te decía para este curso necesitamos esta opción, tú la activas y le das por aquí a instalar,

aquí va a decir Instalar y así vas a comenzar el proceso de instalación tanto de Visual Studio como

de las herramientas necesarias para desarrollar aplicaciones con a ese punto.

NET.

Luego de eso, solamente para comprobar que ya está todo listo, tú puedes presionar la tecla de Windows

R y eso te va a abrir este pequeño cuadro de diálogo.

Escribes cmd, enter y eso te va a abrir una terminal.

Y tú dices aquí dot net, guión, guion, versión enter.

Y en mi caso yo tengo dos siete.

Si ya tienes esto quiere decir que tienes todo listo para comenzar.

Déjame entonces cerrar esto y yo lo que voy a hacer es que voy a abrir Visual Studio.

Déjame abrir una nueva instancia de Visual Studio aquí.

Yo tengo Visual Studio, así que le voy a dar a crear un nuevo proyecto y yo voy a escribir aquí Blaise

Sor y vamos a ver que tenemos varias versiones como Blaze or Server Blaze nuevas en Play.

En mi caso yo voy a escoger la que dice Blaise pruebas en play app, no la que dice Play Store o app

NC que quiere decir vacío.

No, yo quiero Blaise pruebas en play app porque esa me va a dar un poquito de código.

Para empezar, que vamos a utilizar de ejemplo a lo largo de nuestra aplicación.

Entonces yo le voy a dar a siguiente y le vamos a llamar Play Store películas a nuestro proyecto, le

vamos a dar a Next, vamos a escoger done siete.

Yo voy a dejar esto así en non vamos a utilizar https.

Yo voy a dejar esto Chequeado de ASP.NET CrossFit.

Esto es lo que me permite tener un web API acompañado a mi aplicación de Blazer y es en este web app

que yo voy a colocar mi código que por ejemplo va a acceder a una base de datos, así que necesito que

tengas esto chequeado y estas dos cosas sin chequear y le vamos a dar a crear.

Vamos a ver que tenemos nuestra aplicación de Play Store, tenemos muchas cosas que hablar de todo esto,

pero por ahora yo simplemente quiero que ejecutemos nuestra aplicación para ver que está todo en orden,

así que yo voy a presionar el control F5 sin hacer ningún tipo de cambio, lo cual va a ejecutar mi

aplicación y vamos a ver que tenemos nuestra aplicación acá y podemos navegar en ella.

Podemos utilizarla, pero aquí lo importante es que ya tenemos nuestro Visual Studio instalado y hemos

ejecutado nuestra aplicación, así que ya podemos comenzar a aprender acerca de Blazor

Final del formulario