**UAH**

16

**Desarrollo de aplicaciones multiplataforma**

**TG3**

**Contenido**

[1. Autores del trabajo, planificación y entrega 5](#_Toc450685108)

[1.1. Autores 5](#_Toc450685109)

[1.2. Planificación 5](#_Toc450685110)

[1.3. Entrega 5](#_Toc450685111)

[2. Restaurante Multiplataforma 6](#_Toc450685112)

[2.1. Requisitos funcionales 6](#_Toc450685113)

[2.2. Otros requisitos 7](#_Toc450685114)

[3. Criterios de comparación en la implementación 8](#_Toc450685115)

[3.1. Criterio 1: Horas empleadas en el desarrollo del sistema 8](#_Toc450685116)

[3.2. Criterio 2: Curva de aprendizaje 8](#_Toc450685117)

[3.3. Criterio 3: Herramientas externas 8](#_Toc450685118)

[3.4. Criterio 4: Estructura del código 8](#_Toc450685119)

[3.5. Criterio 5: Capacidad de adaptación 8](#_Toc450685120)

[3.6. Criterio 6: Rendimiento 8](#_Toc450685121)

[3.7. Criterio 7: Tiempo de arranque 8](#_Toc450685122)

[3.8. Criterio 8: Recursos existentes en la red 8](#_Toc450685123)

[4. Proyecto de implementación de un prototipo del sistema utilizando Ionic 9](#_Toc450685124)

[4.1. Documentación de diseño 9](#_Toc450685125)

[4.2. Documentación de construcción 10](#_Toc450685126)

[4.3. Documentación de pruebas 11](#_Toc450685127)

[*4.3.1.* Caso de prueba 1: Sintaxis del código. 11](#_Toc450685128)

[4.3.1.1. Requisitos a verificar en el caso del contenido. 11](#_Toc450685129)

[4.3.1.2. Descripción de la prueba: Sintaxis del código 11](#_Toc450685130)

[*4.3.2.* Caso de prueba 2: Funcionalidad e interfaz de la aplicación. 11](#_Toc450685131)

[4.3.2.1. Requisitos a verificar en el caso del contenido. 11](#_Toc450685132)

[4.3.2.2. Descripción de la prueba: Funcionalidad e interfaz de la aplicación. 11](#_Toc450685133)

[*4.3.3.* Caso de prueba 3: Navegación. 12](#_Toc450685134)

[4.3.3.1. Requisitos a verificar en el caso del contenido. 12](#_Toc450685135)

[4.3.3.2. Descripción de la prueba: Navegación 12](#_Toc450685136)

[*4.3.4.* Caso de prueba 4: Componentes. 12](#_Toc450685137)

[4.3.4.1. Requisitos a verificar en el caso del contenido. 12](#_Toc450685138)

[4.3.4.2. Descripción de la prueba: Componentes 12](#_Toc450685139)

[*4.3.5.* Caso de prueba 5: Configuración. 13](#_Toc450685140)

[4.3.5.1. Requisitos a verificar en el caso del contenido. 13](#_Toc450685141)

[4.3.5.2. Descripción de la prueba: Configuración 13](#_Toc450685142)

[*4.3.6.* Caso de prueba 6: Multiplataforma. 14](#_Toc450685143)

[4.3.6.1. Requisitos a verificar en el caso del contenido. 14](#_Toc450685144)

[4.3.6.2. Descripción de la prueba: configuración 14](#_Toc450685145)

[4.4. Incidencias detectadas en las pruebas 15](#_Toc450685146)

[4.5. Documentación de instalación 15](#_Toc450685147)

[4.6. Manual de usuario 18](#_Toc450685148)

[5. Proyecto de implementación de un prototipo del sistema utilizando Apache Cordova 21](#_Toc450685149)

[5.1. Documentación de diseño 21](#_Toc450685150)

[5.2. Documentación de construcción*.* 22](#_Toc450685151)

[5.3. Documentación de pruebas 23](#_Toc450685152)

[5.3.1. Caso de prueba 1: Sintaxis del código 23](#_Toc450685153)

[5.3.1.1. Requisitos a verificar en el caso de prueba del contenido 23](#_Toc450685154)

[5.3.1.2. Descripción de la prueba: Sintaxis del código 23](#_Toc450685155)

[5.3.2. Caso de prueba 2: Funcionalidad e interfaz de la aplicación 23](#_Toc450685156)

[5.3.2.1. Requisitos a verificar en el caso del contenido 23](#_Toc450685157)

[5.3.2.2. Descripción de la prueba: Funcionalidad e interfaz de la aplicación 24](#_Toc450685158)

[5.3.3. Caso de prueba 3: Navegación 24](#_Toc450685159)

[5.3.3.1. Requisitos a verificar en el caso del contenido 24](#_Toc450685160)

[5.3.3.2. Descripción de la prueba: Navegación 24](#_Toc450685161)

[5.3.4. Caso de prueba 4: Componentes 24](#_Toc450685162)

[5.3.4.1. Requisitos a verificar en el caso del contenido 24](#_Toc450685163)

[5.3.4.2. Descripción de la prueba: Componentes 25](#_Toc450685164)

[5.3.5. Caso de prueba 5: Configuración 25](#_Toc450685165)

[5.3.5.1. Requisitos a verificar en el caso del contenido 25](#_Toc450685166)

[5.3.5.2. Descripción de la prueba: Configuración 25](#_Toc450685167)

[5.3.6. Caso de prueba 6: Multiplataforma 26](#_Toc450685168)

[5.3.6.1. Requisitos a verificar en el caso del contenido 26](#_Toc450685169)

[5.3.6.2. Descripción de la prueba: configuración 26](#_Toc450685170)

[5.4. Incidencias detectadas en las pruebas 27](#_Toc450685171)

[5.5. Documentación de instalación 27](#_Toc450685172)

[5.6. Manual de usuario 29](#_Toc450685173)

[6. Comparación de las dos implementaciones 32](#_Toc450685174)

[6.1. Evaluación de los criterios en la implementación usando Ionic 32](#_Toc450685175)

[6.2. Evaluación de los criterios en la implementación usando Apache Cordova 32](#_Toc450685176)

[7. Comparación de la implementación de las tecnologías 33](#_Toc450685177)

[8. Conclusiones 34](#_Toc450685178)

# Autores del trabajo, planificación y entrega

## Autores

El grupo que desarrolla el trabajo que se presenta a continuación el es grupo 7 de laboratorio del turno de tarde; este grupo está formado por:

* Miguel Ángel Rodríguez Blanco (Coordinador del grupo).
* Luis Nueda García.
* Álvaro Laza Martín.
* Eduardo Dorado Pérez.

## Planificación

La planificación de proyecto ha sido realizada por el Coordinador del grupo utilizando la herramienta de diagramación GanttPro, tras una reunión con el grupo de trabajo en la que se han expuesto las tareas necesarias para la consecución del objetivo final y se ha realizado un reparto/asignación de tareas equitativo.

Este trabajo de planificación está disponible en la URL:

***https://github.com/MiguelRodriguezBlanco/TG3/blob/0bfdcf1a8894c41750958ea6ffbba2fedb13afa8/Planificaci%C3%B3n%20TG3.pdf***

## Entrega

El repositorio creado para la puesta en común de los avances del grupo y en el que se ha guardado el resultado de este trabajo es:

***https://github.com/MiguelRodriguezBlanco/TG3***

En este repositorio se encuentra, además de los archivos subidos a los largo del desarrollo del trabajo los archivos requeridos en la rama máster:

* Informe del trabajo: con el nombre TG3\_final.docx
* Presentación del trabajo: TG3\_final.pptx
* Prototipos obtenidos implementando cada una de las tecnologías:
  + Prototipo Ionic\_final.zip
  + Prototipo Cordova\_final.zip

# Restaurante Multiplataforma

Es una solución completa que se presenta como una aplicación móvil, y un potente back-end que permite actualizaciones de datos/contenido.

La aplicación se adapta a las necesidades de cualquier restaurante que quiera presenta su carta a sus clientes, proporcionando una interfaz de usuario sencilla, que permite a navegar a través de todas las categorías del menú, visualizar platos de la carta…La aplicación a desarrollar presentará un paquete de características de gran utilidad adaptadas a las necesidades y tecnologías de hoy en día, tales como compartir información a través de las redes sociales, informar sobre ofertas especiales y videos de YouTube.

Principales Características:

* Solo se necesitan conocimientos de HTML y CSS.
* Interfaz de usuario sencilla, limpia e intuitiva.
* Arquitectura modular que permite adaptación a las necesidades de cada cliente.
* Dispone de una amplia gama de colores depurada
* Instalación guiada paso a paso, rápida e intuitiva.
* Mantenimiento y actualización de contenidos sencillo
* Actualizaciones gratuitas: aplicación en constante evolución, surgen constantemente nuevas versiones depuradas y que aplican sus capacidades.
* Una base de código multiplataforma, compatible con Android y iOS.
* Soporte técnico rápido y eficiente.

## Requisitos funcionales

En la siguiente tabla se presentan los requisitos funcionales del prototipo

| **REQ.** | **DESCRIPCIÓN** |
| --- | --- |
| RF01 | Aplicación multiplataforma |
| RF02 | Desarrollo sobre JavaScript, CSS, HTML |
| RF03 | Dispondrá de Plantillas sobre la que presentar la carta |
| RF04 | Dispondrá de Plantillas con información sobre las ofertas especiales |
| RF05 | Aplicación fácil de usar |
| RF06 | Instalación paso a paso, guiada y sencilla |
| RF07 | Permitirá visualizar imágenes asociadas a cada uno de los platos de la carta. |
| RF08 | Desde la pantalla principal será posible acceder a las diferentes categorías de menú |
| RF09 | Desde cada submenú se puede avanzar hacia delante, atrás o acceder a la pantalla principal |
| RF10 | Será posible acceder a redes sociales a través de la app. |

## Otros requisitos

En la siguiente tabla se presenta el catálogo de requisitos no funcionales del sistema.

| **REQ.** | **DESCRIPCIÓN** |
| --- | --- |
| ***RI*** | ***Requisitos de Interfaz*** |
| RI01 | Interfaz sencilla e intuitiva |
| RI02 | Interfaz en color |
| RI03 | Visualización de la carta en color |
| ***RR*** | ***Requisitos de Rendimiento*** |
| R01 | Tiempos de respuesta rápidos |
| ***RSw*** | ***Requisitos Sw*** |
| RSw01 | Arquitectura modular |
| RSw02 | Funcionará al menos sobre los siguientes sistemas operativos:   * + Windows 8   + OsX   + Android   + iOS |
| ***RHw*** | ***Requisitos Hw*** |
| RHw01 | 256MB RAM mínima requerida para el dispositivo sobre el que se instala la app |
| RHw02 | 1 GB disco duro mínimo requerido para el dispositivo sobre el que se instala la app |
| RHw03 | Espacio de disco: 160 MB Libres |
| RHw04 | Mínimo un Pentium 4, a 1 GHz |
| RHw05 | Doble núcleo 2GHz |

# Criterios de comparación en la implementación

## Criterio 1: Horas empleadas en el desarrollo del sistema

**Descripción:** Tiempo requerido para el desarrollo de la aplicación.

**Tipo de valor:** Numérico (horas).

## Criterio 2: Curva de aprendizaje

**Descripción:** Promedio de tiempo que necesita un usuario para conocer las utilidades básicas de la aplicación.

**Tipo de valor:** Numérico (horas).

## Criterio 3: Herramientas externas

**Descripción:** Compatibilidad con el uso de herramientas externas.

**Tipo de valor:** Booleano (Si/No).

## Criterio 4: Estructura del código

**Descripción:** Número de paquetes que forman el código de la aplicación

**Tipo de valor:** Numérico.

## Criterio 5: Capacidad de adaptación

**Descripción:** Nivel de complejidad que presenta la aplicación para al usuario final a la hora de incorporar a la aplicación toda la información relativa su restaurante: menús, ofertas, imágenes….

**Tipo de valor:** Escala: 1…5 (1: muy fácil…5: muy complejo).

## Criterio 6: Rendimiento

**Descripción:** Tiempo de respuesta medio al ejecutar una utilidad de la aplicación

**Tipo de valor:** Numérico (segundos).

## Criterio 7: Tiempo de arranque

**Descripción:** Tiempo que tarda en arrancar la aplicación

**Tipo de valor:** Numérico (segundos)

## Criterio 8: Recursos existentes en la red

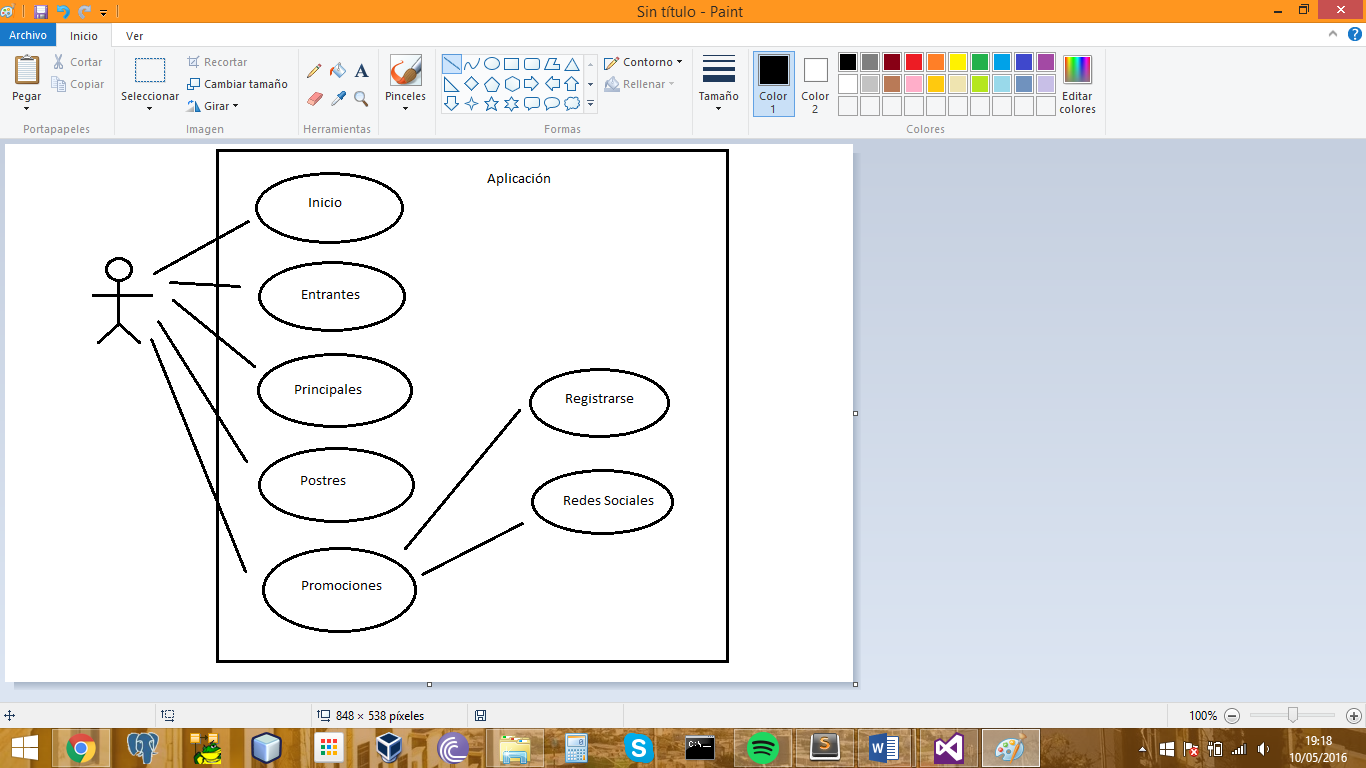
**Descripción:** Información existente en la red de soporte para el desarrollo de la aplicación.

**Tipo de valor:** Escala: escasa, media, abundante.

# Proyecto de implementación de un prototipo del sistema utilizando Ionic

## Documentación de diseño

Previo al desarrollo de la aplicación se plantea el caso de uso que presenta las interacciones necesarias entre el usuario y los diferentes submenús y uno de estos y las diferentes funcionalidades que incluye. Todas estas interacciones quedan identificadas y representadas como se indica en el caso de uso que se muestra a continuación.



A partir del caso de uso se desarrolla la aplicación que como se presenta al usuario a través de una pantalla principal un “Sidemenu”, con una SideBar, con diferentes pestañas.



## Documentación de construcción

El código de la aplicación se agrupa en diferentes paquetes, la estructura de paquetes que lo forman estos paquetes se presenta a continuación:



Dentro de esta estructura, los principales paquetes, aquellos que dan entidad a la aplicación se presentan son los que detallan a continuación:

* **img**: carpeta en la que se almacenan las imágenes que la aplicación incorpora asociadas a los platos del menú o las promociones.
* **templates**: plantillas html que dan formato a todas las categorías del menú de la aplicación.
* **js**: carpeta en la que se almacena el código Java Script.
* **css**: diferentes estilos que utiliza la aplicación.

## Documentación de pruebas

## Caso de prueba 1: Sintaxis del código.

## Requisitos a verificar en el caso del contenido.

A través de este caso de prueba se verificará el cumplimiento de los siguientes requisitos.

|  |  |
| --- | --- |
| RF01 | Aplicación multiplataforma |
| RF02 | Desarrollo sobre JavaScript, CSS, HTML |
| RF03 | Dispondrá de Plantillas sobre la que presentar la carta |
| RF04 | Dispondrá de Plantillas con información sobre las ofertas especiales |
| RSw01 | Arquitectura modular |

## Descripción de la prueba: Sintaxis del código

Este caso de prueba trata de verificar la sintaxis del código desarrollado recorriéndolo con el entorno de desarrollo, de modo que se demuestra que el código es claro y se ha definido correctamente.

| **Descripción** | **Resultado esperado** | **Resultado obtenido (Ok/Nok)** |
| --- | --- | --- |
| Recorrer el código con el entorno de desarrollo | Mostrar el código tabulado y bien estructurado, identificando por colores los diferentes clases, atributos etc. | **Ok** |

## Caso de prueba 2: Funcionalidad e interfaz de la aplicación.

## Requisitos a verificar en el caso del contenido.

A través de este caso de prueba se verificará el cumplimiento de los siguientes requisitos.

|  |  |
| --- | --- |
| RF05 | Aplicación sencilla, fácil de usar |
| RI01 | Interfaz sencilla e intuitiva |
| RI02 | Interfaz en color |
| RI03 | Visualización de la carta en color |

## Descripción de la prueba: Funcionalidad e interfaz de la aplicación.

A través de la ejecución de este caso de prueba, se trata de verificar la funcionalidad de la aplicación, así como la interfaz de usuario.

| **Descripción** | **Resultado esperado** | **Resultado obtenido (Ok/Nok)** |
| --- | --- | --- |
| Testeamos la aplicación desde ionic serve. | Ejecuta la aplicación con el diseño establecido. | **Ok** |

## Caso de prueba 3: Navegación.

## Requisitos a verificar en el caso del contenido.

A través de este caso de prueba se verificará el cumplimiento de los siguientes requisitos.

|  |  |
| --- | --- |
| RF08 | Desde la pantalla principal será posible acceder a las diferentes categorías de menú |
| RF09 | Desde cada submenú se puede avanzar hacia delante, atrás o acceder a la pantalla principal |
| R01 | Tiempos de respuesta rápidos |

## Descripción de la prueba: Navegación

Aplica casos de uso en el diseño de casos de prueba que ejerciten cada escenario de uso contra el diseño de navegación.

| **Descripción** | **Resultado esperado** | **Resultado obtenido (Ok/Nok)** |
| --- | --- | --- |
| Se navega a través de los diferentes menús disponibles en la aplicación, yendo hacia delante hacia detrás, saltando de uno a otro. | Es posible acceder a todos los menús y submenús disponibles, avanzando hacia delante y hacia atrás. | **Ok** |

## Caso de prueba 4: Componentes.

## Requisitos a verificar en el caso del contenido.

A través de este caso de prueba se verificará el cumplimiento de los siguientes requisitos.

|  |  |
| --- | --- |
| RF10 | Será posible acceder a redes sociales a través de la app. |
| RI01 | Interfaz sencilla e intuitiva |
| RI02 | Interfaz en color |
| RI03 | Visualización de la carta en color |

## Descripción de la prueba: Componentes

Ejercita el contenido y las unidades funcionales dentro de la web app, cada elemento de la arquitectura de la web app se prueba de manera unitaria en la medida de lo posible.

| **Descripción** | **Resultado esperado** | **Resultado obtenido (Ok/Nok)** |
| --- | --- | --- |
| Es posible acceder desde la aplicación a las aplicaciones de redes sociales disponibles en el dispositivo móvil así como publicar contenido de la app en ellas. | Desde la app es posible acceder a y visualizar correctamente el contenido de:   * Facebook, * Youtube | **Ok** |

## Caso de prueba 5: Configuración.

## Requisitos a verificar en el caso del contenido.

A través de este caso de prueba se verificará el cumplimiento de los siguientes requisitos.

|  |  |
| --- | --- |
| RF05 | Aplicación sencilla, fácil de usar |
| RF07 | Permitirá visualizar imágenes asociadas a cada uno de los platos de la carta. |
| RF08 | Desde la pantalla principal será posible acceder a las diferentes categorías de menú |
| RF09 | Desde cada submenú se puede avanzar hacia delante, atrás o acceder a la pantalla principal |
| RF10 | Será posible acceder a redes sociales a través de la app. |
| RI01 | Interfaz sencilla e intuitiva |
| RI02 | Interfaz en color |
| RI03 | Visualización de la carta en color |

## Descripción de la prueba: Configuración

Intentan descubrir errores específicos respecto de un determinado cliente o ambiente de servidores particulares. Las pruebas se encaminan a descubrir los errores asociados con cada posible configuración.

| **Descripción** | **Resultado esperado** | **Resultado obtenido (Ok/Nok)** |
| --- | --- | --- |
| Se instala la app para una configuración preestablecida se navega por ella verificando que se comporta y se visualiza el contenido correctamente | La app se instala de forma rápida y sencilla. Una vez instalada, cada pantalla de la app se visualiza correctamente permitiendo navegar por los diferentes menús disponibles sin errores. | **Ok** |

## Caso de prueba 6: Multiplataforma.

## Requisitos a verificar en el caso del contenido.

A través de este caso de prueba se verificará el cumplimiento de los siguientes requisitos.

|  |  |
| --- | --- |
| RF05 | Aplicación sencilla, fácil de usar |
| RF06 | Instalación paso a paso, guiada y sencilla |
| RF07 | Permitirá visualizar imágenes asociadas a cada uno de los platos de la carta. |
| RF08 | Desde la pantalla principal será posible acceder a las diferentes categorías de menú |
| RF09 | Desde cada submenú se puede avanzar hacia delante, atrás o acceder a la pantalla principal |
| RF10 | Será posible acceder a redes sociales a través de la app. |
| RI01 | Interfaz sencilla e intuitiva |
| RI02 | Interfaz en color |
| RI03 | Visualización de la carta en color |
| RSw02 | Funcionará al menos sobre los siguientes sistemas operativos:   * + Windows 8   + OsX   + Android   + iOS |
| RHw01 | 256MB RAM mínima requerida para el dispositivo sobre el que se instala la app |
| RHw02 | 1 GB disco duro mínimo requerido para el dispositivo sobre el que se instala la app |
| RHw03 | Espacio de disco: 160 MB Libres |
| RHw04 | Mínimo un Pentium 4, a 1 GHz |
| RHw05 | Doble núcleo 2GHz |

## Descripción de la prueba: configuración

Se verifica la característica multiplataforma de la app.

| **Descripción** | **Resultado esperado** | **Resultado obtenido (Ok/Nok)** |
| --- | --- | --- |
| Se instala la app en móviles con diferentes sistemas operativos con las siguientes características mínimas:   * 256MB RAM * 1 GB disco duro * 160 MB Libres * Pentium 4, a 1 GHz * Doble núcleo 2GHz | La app se instala de forma rápida y sencilla correctamente. Una vez instalada, se verifica el funcionamiento correcto. | **Ok** |

## Incidencias detectadas en las pruebas

|  |  |  |
| --- | --- | --- |
| **Caso de prueba** | **Incidencia** | **Acciones correctoras** |
| **6** | Cuando usamos un emulador para iniciar la aplicación, se generaban errores de dependencias y librerías no encontradas | Se soluciona eliminando el conjuntos de librerías y volviendo a instalarlas dentro del paquete de Visual Studio |

## Documentación de instalación

Antes de comenzar a crear una aplicación en Ionic hay que instalar y configurar algunos programas necesarios para crear y compilar la app que vamos a diseñar.

En primer lugar, hay que descargar nodeJS y una vez instalado hay que ejecutar el comando de instalación  de Ionic desde la terminal del sistema:

$ npm install -g cordova ionic

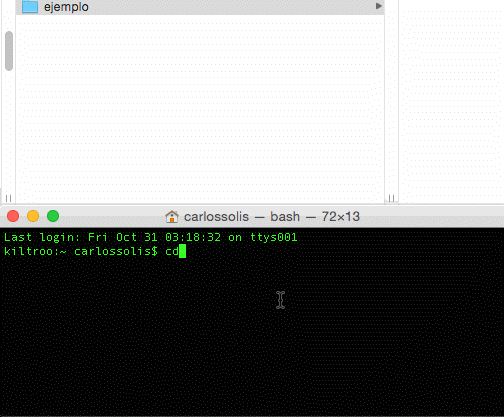
Aparte, debemos descargar el SDK de la aplicación en la que vamos a empaquetar, por ejemplo si quieres trabajar iOS hay que bajar Xcode, mientras que si trabajamos en Android hay que descargar el SDK de este sistema operativo.

Las aplicaciones Ionic se administran desde la terminal del sistema, es desde allí desde donde vamos a crear y enviar todos los comandos a Ionic.

Antes de comenzar a trabajar, debemos crear una carpeta donde almacenaremos la aplicación y una vez creada tenemos que abrirla desde la terminal. Lo haremos con el comando:

cd /DIRECCION\_DE\_TU\_CARPETA

Si no sabes cuál es la dirección de tu carpeta, basta con escribir en el terminal “cd ” y arrastrar la carpeta allí, el sistema se encargará de escribir la dirección.

[](http://revolucion.mobi/wp-content/uploads/2014/10/cap02.gif)

Ahora que estamos en la carpeta correcta, hay que ejecutar el comando para crear tu aplicación:

$ ionic start miAplicacion tabs

En este caso, el comando “ionic start” crea la aplicación, “miAplicacion” es el nombre de la aplicación y “tabs” es un comando de ayuda que indica a ionic que al crear nuestra aplicación, incluya automáticamente soporte para navegación por tabs.

[](http://revolucion.mobi/wp-content/uploads/2014/10/cap03.gif)

Luego de unos segundos nuestra aplicación habrá sido creada. Encontraremos una nueva carpeta con el nombre de la aplicación y una serie de archivos que componen el código de la futura aplicación.

Ahora que tenemos la aplicación creada hay que abrirla en la consola para controlarla, se puede hacer agregando el siguiente comando :

cd miAplicacion

En este caso el nombre “miAplicacion” es el nombre de nuestra aplicación y de la carpeta recién creada (habrá que adaptarlo según el nombre que demos a nuestra app)

Ya hemos creado la aplicación en su forma más, en este momento debemos agregarle soporte para diferentes plataformas como Android o iOs, por ejemplo. Para que la aplicación tenga una versión para Android usaremos el comando:

ionic platform add android

[](http://revolucion.mobi/wp-content/uploads/2014/11/cap05.gif)

Finalmente, para simular la aplicación en un dispositivo móvil, hay que insertar el siguiente comando que compilará el código y lo enviará al simulador:

ionic build android

[](http://revolucion.mobi/wp-content/uploads/2014/11/cap06.gif)

Para agregar y emular aplicaciones para iOs solo debemos modificar el término “android” por el de “ios” en los comandos que acabamos de usar.

El código base de la aplicación ya ha sido creado automáticamente por Ionic y contiene los elementos fundamentales para comenzar a trabajar.

En el proyecto encontraremos varias carpetas,  en la carpeta /www se encuentra el código de la aplicación y allí encontrarás un archivo con el nombre índex.html que es la vista principal de la app.

Ya podemos proceder a abrir el documento en el navegador y comenzar a depurar tu aplicación.

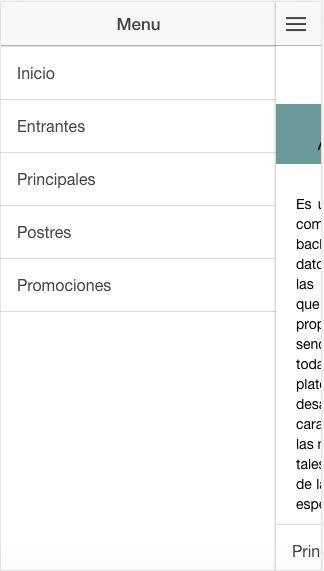
Ionic está basado en AngularJS y utiliza un sistema de plantillas para cargar las vistas según las vamos necesitando. Dentro de la carpeta /js/app.js podemos encontrar las instrucciones de uso de nuestra aplicación y además, si conoces este framework, observarás un código familiar que indica las vistas y manejo de controladores.

Por último, en la carpeta “/www/templates” encontramos las plantillas que generan las vistas de la aplicación.

Ya tenemos terminada nuestra primera aplicación creada con Ionic, solo queda guardar el documento y pruébalo en nuestro navegador.

## Manual de usuario

Al iniciar la aplicación encontramos en primer lugar una ventana con información primordial junto a un icono que nos permite visualizar el menú principal:

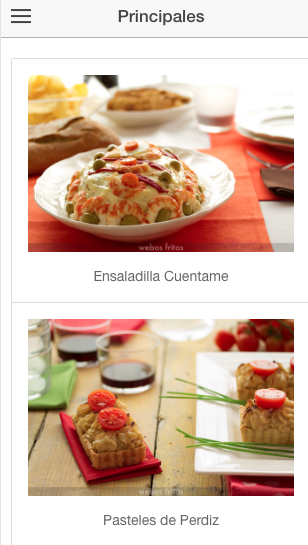
 

A través del menú principal podemos acceder al resto de pestañas de la aplicación:

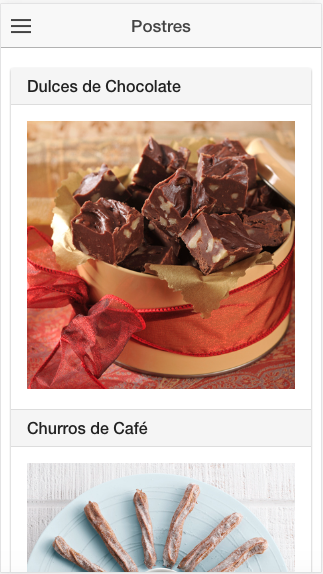
Entrantes:

En este espacio tenemos la iconografía de los entrantes que podemos encontrar en el restaurante, además de un breve pie de página que nos indica su nombre.

En la parte superior izquierda mantenemos el botón de hamburguesa para retornar al menú principal siempre que queramos. Dicho botón permanecerá siempre en el mismo lugar.

* Principales:

Aquí encontraremos los primeros platos que se pueden degustar en el restaurante. Mantenemos una estructura similar a la de la ventana anterior, con un breve pie de foto que incluye la descripción del plato.

* Postres:

En esta ventana podemos visualizar los postres de los que disponemos con su nombre a pie de foto. Mantenemos una estructura limpia y sencilla que resulta agradable a la vista del usuario.

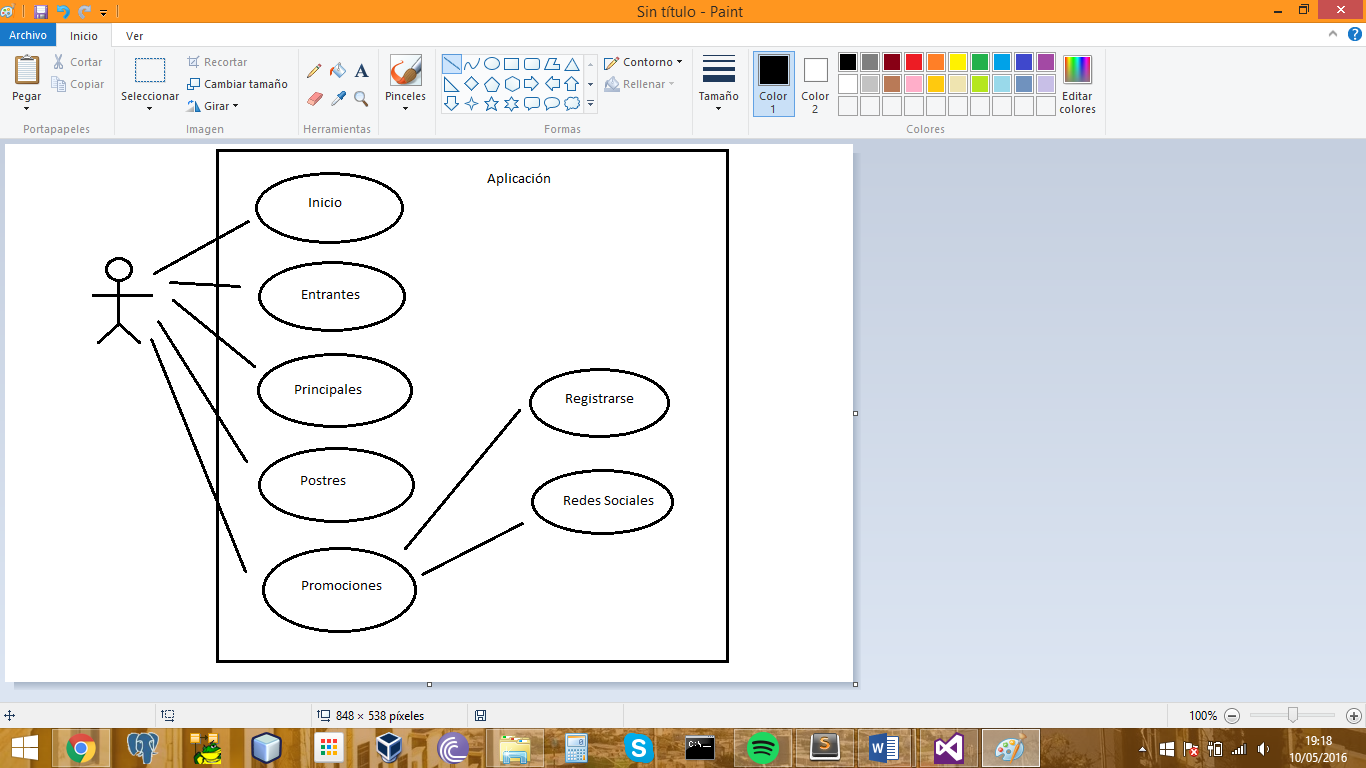
* Promociones:

Este apartado dispone información sobre los descuentos que hay vigentes en el restaurante. Para poder disfrutar de los mismos tenemos la opción de registrarnos en la app. Además, en la parte inferior incluimos el acceso a redes sociales.

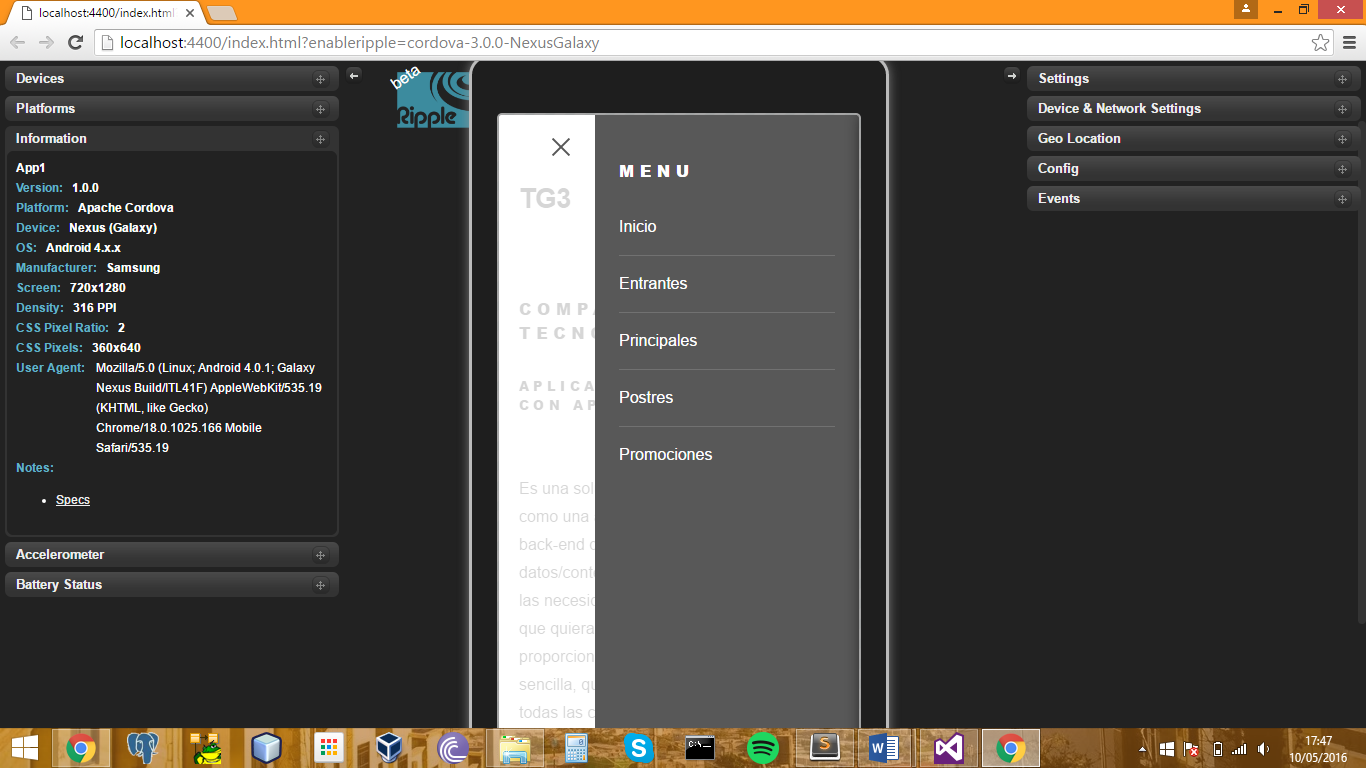
# Proyecto de implementación de un prototipo del sistema utilizando Apache Cordova

## Documentación de diseño

Previo al desarrollo de la aplicación se plantea el caso de uso que presenta las interacciones necesarias entre el usuario y los diferentes submenús y uno de estos y las diferentes funcionalidades que incluye. Todas estas interacciones quedan identificadas y representadas como se indica en el caso de uso que se muestra a continuación.

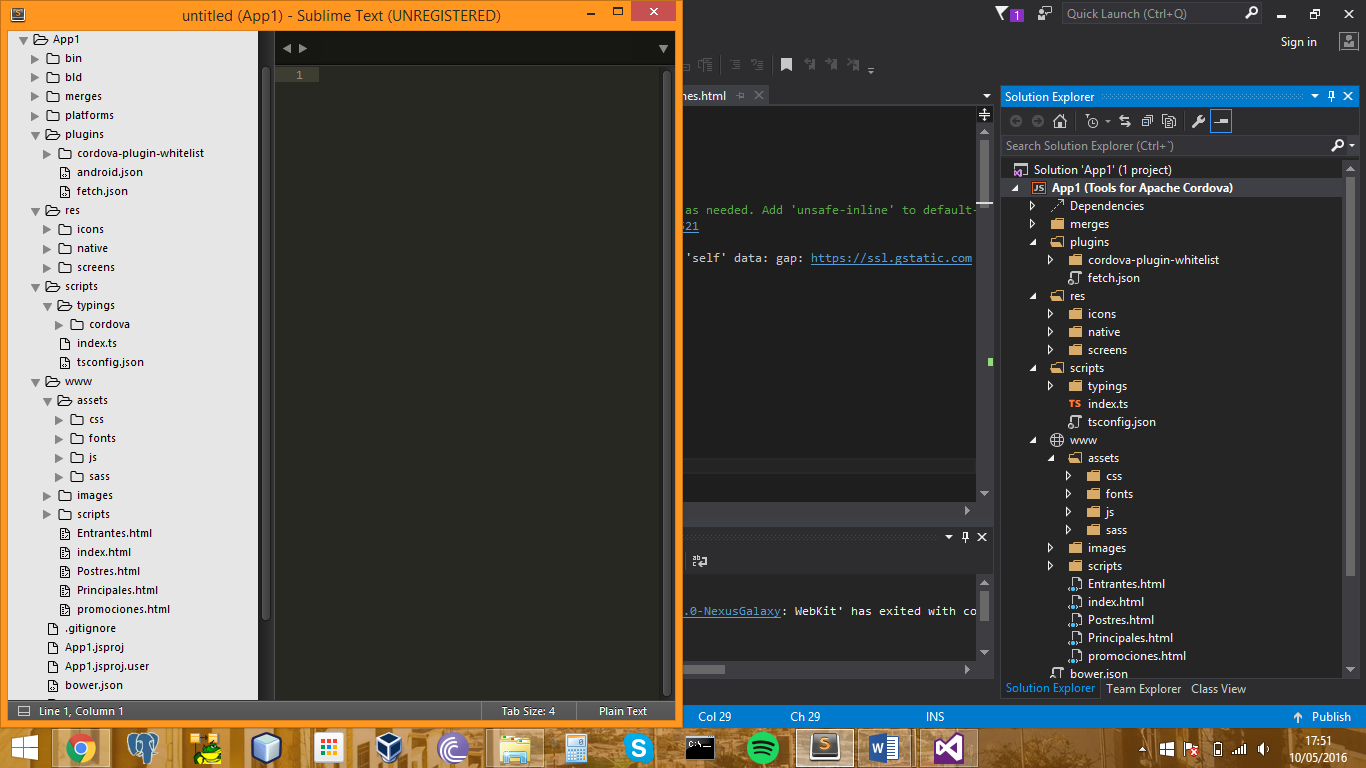


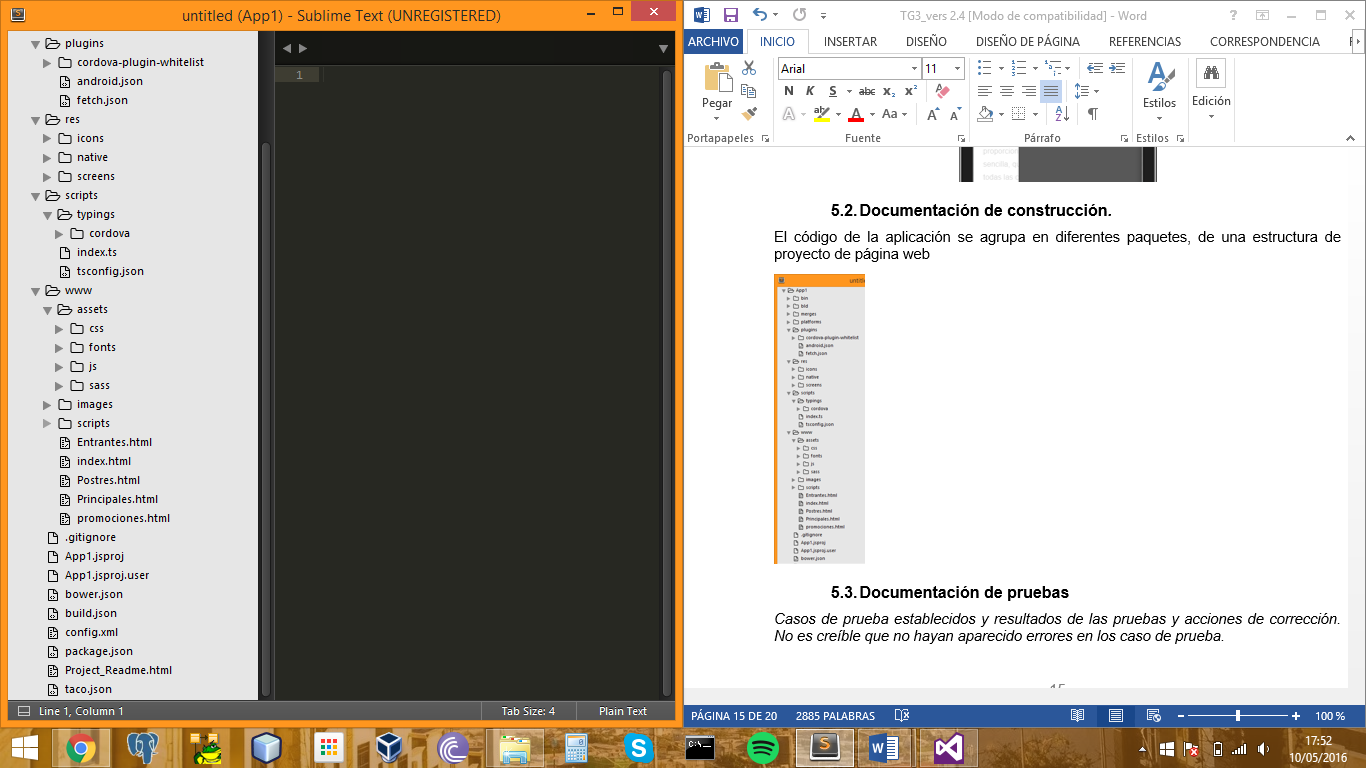
A partir del caso de uso se desarrolla la aplicación que como se presenta al usuario a través de una pantalla principal un “Sidemenu”, con una SideBar, con diferentes pestañas.



## Documentación de construcción*.*

El código de la aplicación se agrupa en diferentes paquetes, la estructura de paquetes que lo forman el proyecto de página web se presenta a continuación:





* **plugins:** Carpeta que contiene la información y los plugins de apache cordova que han sido instalados para la aplicación
* **res:** Archivos de configuración de apache cordova para las distintas plataformas (android, ios, wp). Contiene elementos gráficos y formato de dispositivos para su posterior emulación.
* **scripts:** Librería que contiene clases de los distintos plugins de apache cordova que pueden ser usados en nuestro proyecto
* **www:** Carpeta “principal” de nuestra aplicación.
  + **assets:** Contiene los archivos que sirven para dar estilo a nuestra aplicación (css; hoja de estilos, fonts; documento para iconos, js; contiene código JavaScript, sass; directorio que se complementa con css)
  + **images:** Incorpora las imágenes que usaremos en nuestra aplicación
  + **scripts:** Archivos de arranque de la aplicación. Función onDeviceReady

## Documentación de pruebas

## Caso de prueba 1: Sintaxis del código

## Requisitos a verificar en el caso de prueba del contenido

A través de este caso de prueba se verificará el cumplimiento de los siguientes requisitos.

|  |  |
| --- | --- |
| RF01 | Aplicación multiplataforma |
| RF02 | Desarrollo sobre JavaScript, CSS, HTML |
| RF03 | Dispondrá de Plantillas sobre la que presentar la carta |
| RF04 | Dispondrá de Plantillas con información sobre las ofertas especiales |
| RSw01 | Arquitectura modular |

## Descripción de la prueba: Sintaxis del código

Este caso de prueba trata de verificar la sintaxis del código desarrollado recorriéndolo con el entorno de desarrollo, de modo que se demuestra que el código es claro y se ha definido correctamente.

| **Descripción** | **Resultado esperado** | **Resultado obtenido (Ok/Nok)** |
| --- | --- | --- |
| Recorrer el código con el entorno de desarrollo | Mostrar el código tabulado y bien estructurado, identificando por colores los diferentes clases, atributos etc. | **Ok** |

## Caso de prueba 2: Funcionalidad e interfaz de la aplicación

## Requisitos a verificar en el caso del contenido

A través de este caso de prueba se verificará el cumplimiento de los siguientes requisitos

|  |  |
| --- | --- |
| RF05 | Aplicación sencilla, fácil de usar |
| RI01 | Interfaz sencilla e intuitiva |
| RI02 | Interfaz en color |
| RI03 | Visualización de la carta en color |

## Descripción de la prueba: Funcionalidad e interfaz de la aplicación

A través de la ejecución de este caso de prueba, se trata de verificar la funcionalidad de la aplicación, así como la interfaz de usuario.

| **Descripción** | **Resultado esperado** | **Resultado obtenido (Ok/Nok)** |
| --- | --- | --- |
| Testeamos la aplicación desde el emulador de Visual Studio de Apache Cordova. | Ejecuta la aplicación con el diseño establecido. | **Ok** |

## Caso de prueba 3: Navegación

## Requisitos a verificar en el caso del contenido

A través de este caso de prueba se verificará el cumplimiento de los siguientes requisitos.

|  |  |
| --- | --- |
| RF08 | Desde la pantalla principal será posible acceder a las diferentes categorías de menú |
| RF09 | Desde cada submenú se puede avanzar hacia delante, atrás o acceder a la pantalla principal |
| R01 | Tiempos de respuesta rápidos |

## Descripción de la prueba: Navegación

Aplica casos de uso en el diseño de casos de prueba que ejerciten cada escenario de uso contra el diseño de navegación.

| **Descripción** | **Resultado esperado** | **Resultado obtenido (Ok/Nok)** |
| --- | --- | --- |
| Se navega a través de los diferentes menús disponibles en la aplicación, yendo hacia delante hacia detrás, saltando de uno a otro. | Es posible acceder a todos los menús y submenús disponibles, avanzando hacia delante y hacia atrás. | **Ok** |

## Caso de prueba 4: Componentes

## Requisitos a verificar en el caso del contenido

A través de este caso de prueba se verificará el cumplimiento de los siguientes requisitos.

|  |  |
| --- | --- |
| RF10 | Será posible acceder a redes sociales a través de la app. |
| RI01 | Interfaz sencilla e intuitiva |
| RI02 | Interfaz en color |
| RI03 | Visualización de la carta en color |

## Descripción de la prueba: Componentes

Ejercita el contenido y las unidades funcionales dentro de la web app, cada elemento de la arquitectura de la web app se prueba de manera unitaria en la medida de lo posible.

| **Descripción** | **Resultado esperado** | **Resultado obtenido (Ok/Nok)** |
| --- | --- | --- |
| Es posible acceder desde la aplicación a las aplicaciones de redes sociales disponibles en el dispositivo móvil así como publicar contenido de la app en ellas. | Desde la app es posible acceder a y visualizar correctamente el contenido de:   * Facebook, * Youtube | **Ok** |

## Caso de prueba 5: Configuración

## Requisitos a verificar en el caso del contenido

A través de este caso de prueba se verificará el cumplimiento de los siguientes requisitos.

|  |  |
| --- | --- |
| RF05 | Aplicación sencilla, fácil de usar |
| RF07 | Permitirá visualizar imágenes asociadas a cada uno de los platos de la carta. |
| RF08 | Desde la pantalla principal será posible acceder a las diferentes categorías de menú |
| RF09 | Desde cada submenú se puede avanzar hacia delante, atrás o acceder a la pantalla principal |
| RF10 | Será posible acceder a redes sociales a través de la app. |
| RI01 | Interfaz sencilla e intuitiva |
| RI02 | Interfaz en color |
| RI03 | Visualización de la carta en color |

## Descripción de la prueba: Configuración

Intentar descubrir errores específicos respecto de un determinado cliente o ambiente de servidores particulares. Las pruebas se encaminan a descubrir los errores asociados con cada posible configuración.

| **Descripción** | **Resultado esperado** | **Resultado obtenido (Ok/Nok)** |
| --- | --- | --- |
| Se instala la app para una configuración preestablecida se navega por ella verificando que se comporta y se visualiza el contenido correctamente | La app se instala de forma rápida y sencilla. Una vez instalada, cada pantalla de la app se visualiza correctamente permitiendo navegar por los diferentes menús disponibles sin errores. | **Ok** |

## Caso de prueba 6: Multiplataforma

## Requisitos a verificar en el caso del contenido

A través de este caso de prueba se verificará el cumplimiento de los siguientes requisitos.

|  |  |
| --- | --- |
| RF05 | Aplicación sencilla, fácil de usar |
| RF06 | Instalación paso a paso, guiada y sencilla |
| RF07 | Permitirá visualizar imágenes asociadas a cada uno de los platos de la carta. |
| RF08 | Desde la pantalla principal será posible acceder a las diferentes categorías de menú |
| RF09 | Desde cada submenú se puede avanzar hacia delante, atrás o acceder a la pantalla principal |
| RF10 | Será posible acceder a redes sociales a través de la app. |
| RI01 | Interfaz sencilla e intuitiva |
| RI02 | Interfaz en color |
| RI03 | Visualización de la carta en color |
| RSw02 | Funcionará al menos sobre los siguientes sistemas operativos:   * + Windows 8   + OsX   + Android   + iOS |
| RHw01 | 256MB RAM mínima requerida para el dispositivo sobre el que se instala la app |
| RHw02 | 1 GB disco duro mínimo requerido para el dispositivo sobre el que se instala la app |
| RHw03 | Espacio de disco: 160 MB Libres |
| RHw04 | Mínimo un Pentium 4, a 1 GHz |
| RHw05 | Doble núcleo 2GHz |

## Descripción de la prueba: configuración

Se verifica la característica multiplataforma de la app.

| **Descripción** | **Resultado esperado** | **Resultado obtenido (Ok/Nok)** |
| --- | --- | --- |
| Se instala la app en móviles con diferentes sistemas operativos con las siguientes características mínimas:   * 256MB RAM * 1 GB disco duro * 160 MB Libres * Pentium 4, a 1 GHz * Doble núcleo 2GHz | La app se instala de forma rápida y sencilla correctamente. Una vez instalada, se verifica el funcionamiento correcto. | **Ok** |

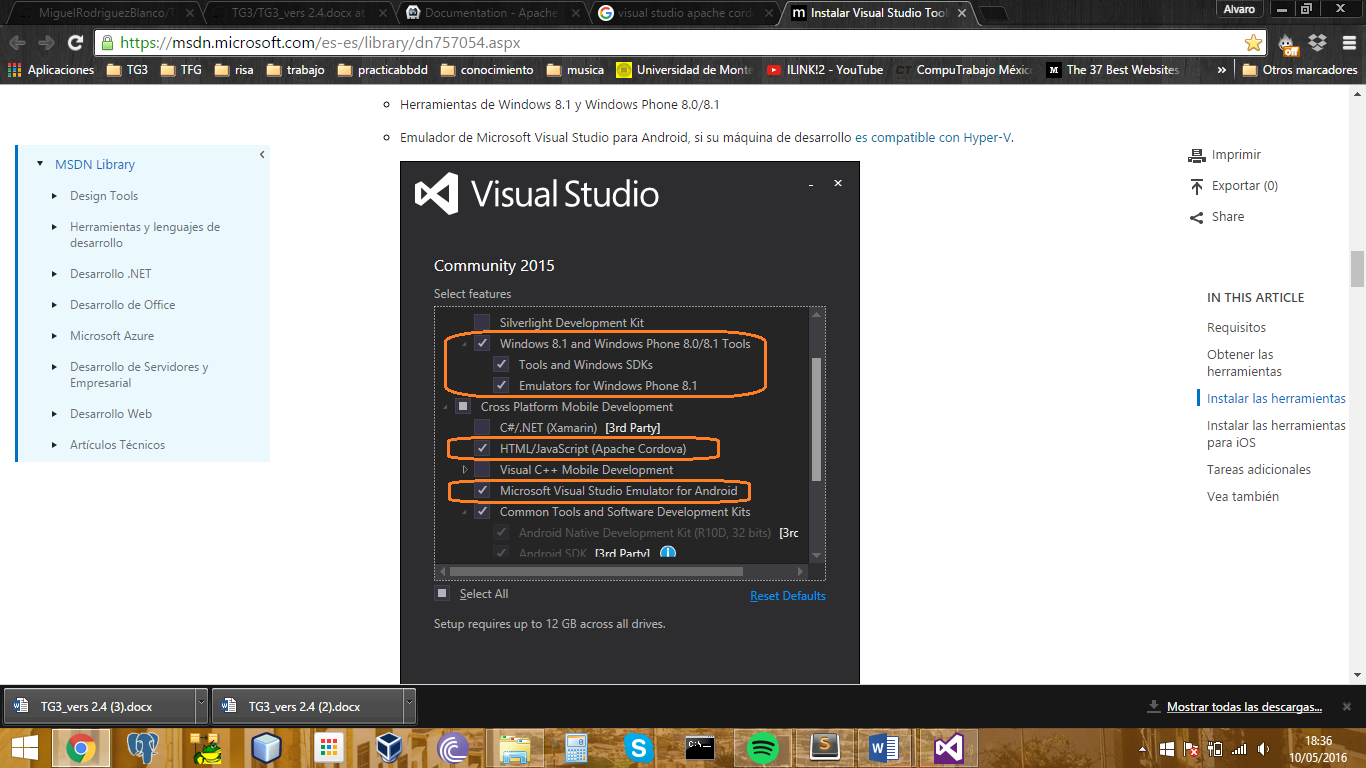
## Incidencias detectadas en las pruebas

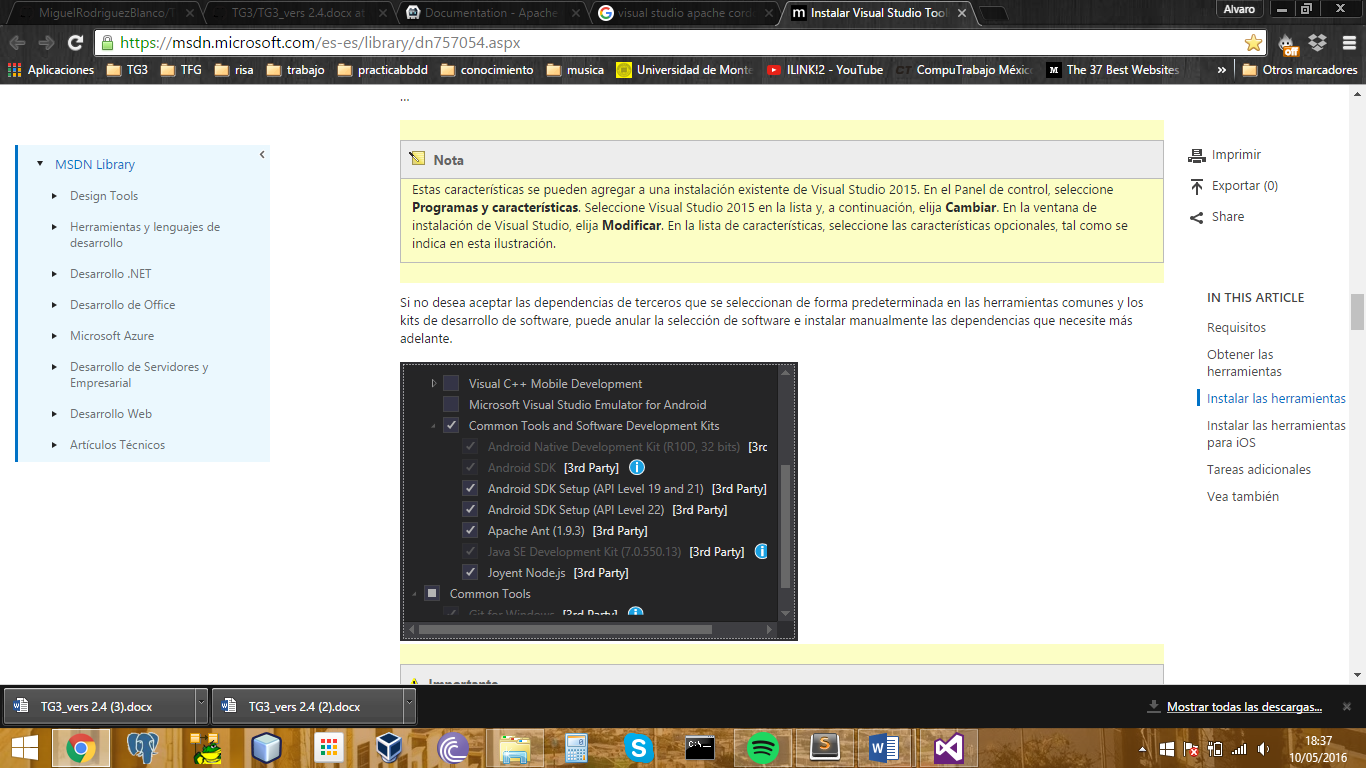
|  |  |  |
| --- | --- | --- |
| **Caso de prueba** | **Incidencia** | **Acciones correctoras** |
| **2** | Apache Cordova no cuenta con interfaz propia. A la hora de generar una, se encontraron problemas de dependencias | Creación de directorio de la interfaz incluyendo todos los paquetes necesarios (css, js y fonts) |
| **6** | A la hora de arrancar al app en un entorno virtual, se generaban errores de dependencias y paquetes mal instalados, imposibilitando la ejecución en la máquina virtual | Se procedió a desinstalar los elementos de Apache Cordova (apache ant, sdk android, node.js y apache), para volverlos a instalar dentro de la solución de Visual Studio |

## Documentación de instalación

Para la instalación y uso de Apache Cordova en un primer momento se decidió a instalar el software tal y como se explica en la documentación oficial de Apache Cordova (<https://cordova.apache.org/docs/en/latest/>). Durante el proceso de instalación no existieron problemas algunos, pero a la hora de depurar el programa o intentar emularlo en una imagen virtual nos encontramos con numerosos problemas como la falta de dependencias o la imposibilidad de conexión con dispositivos virtuales, por lo que se decidió instalar el complemento de desarrollo web y Apache Cordova para Visual Studio.

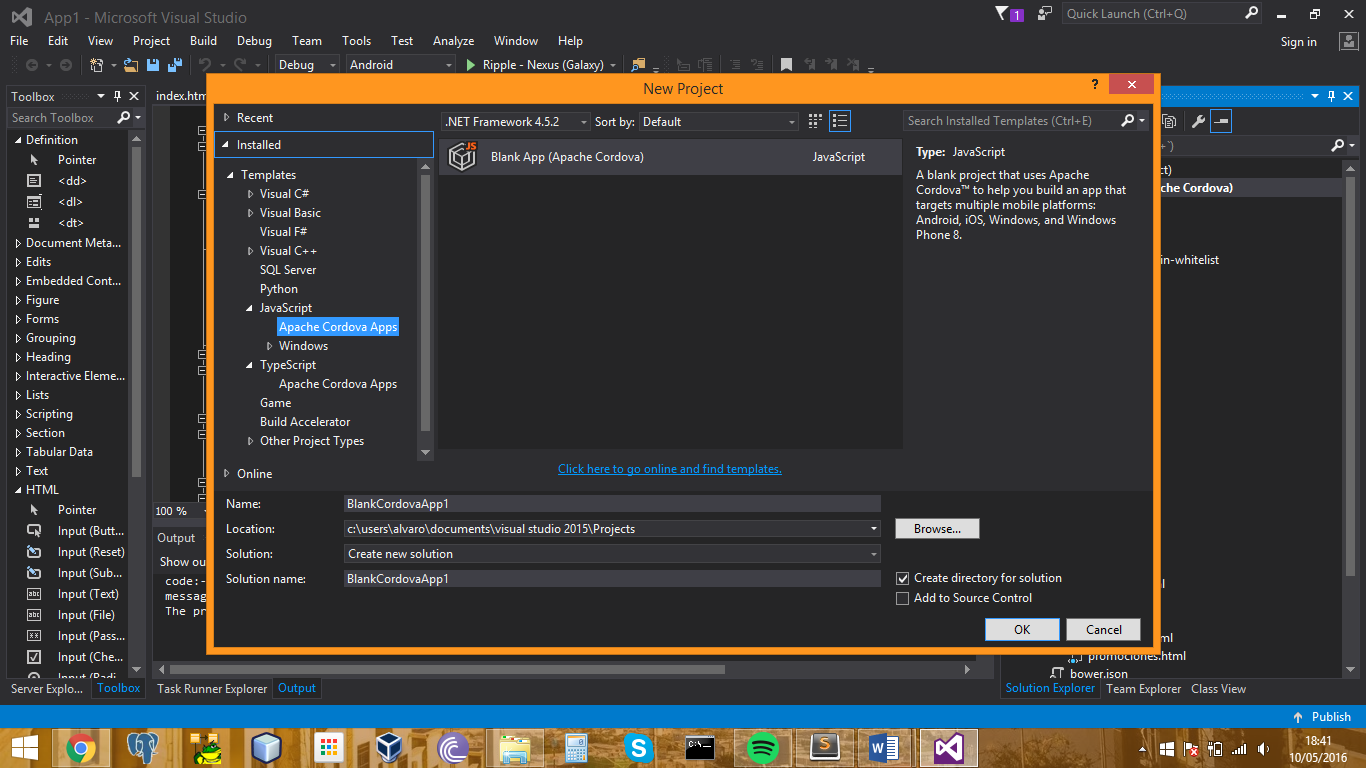
En el proceso de instalación de Visual Studio tenemos que incluir las siguientes características:



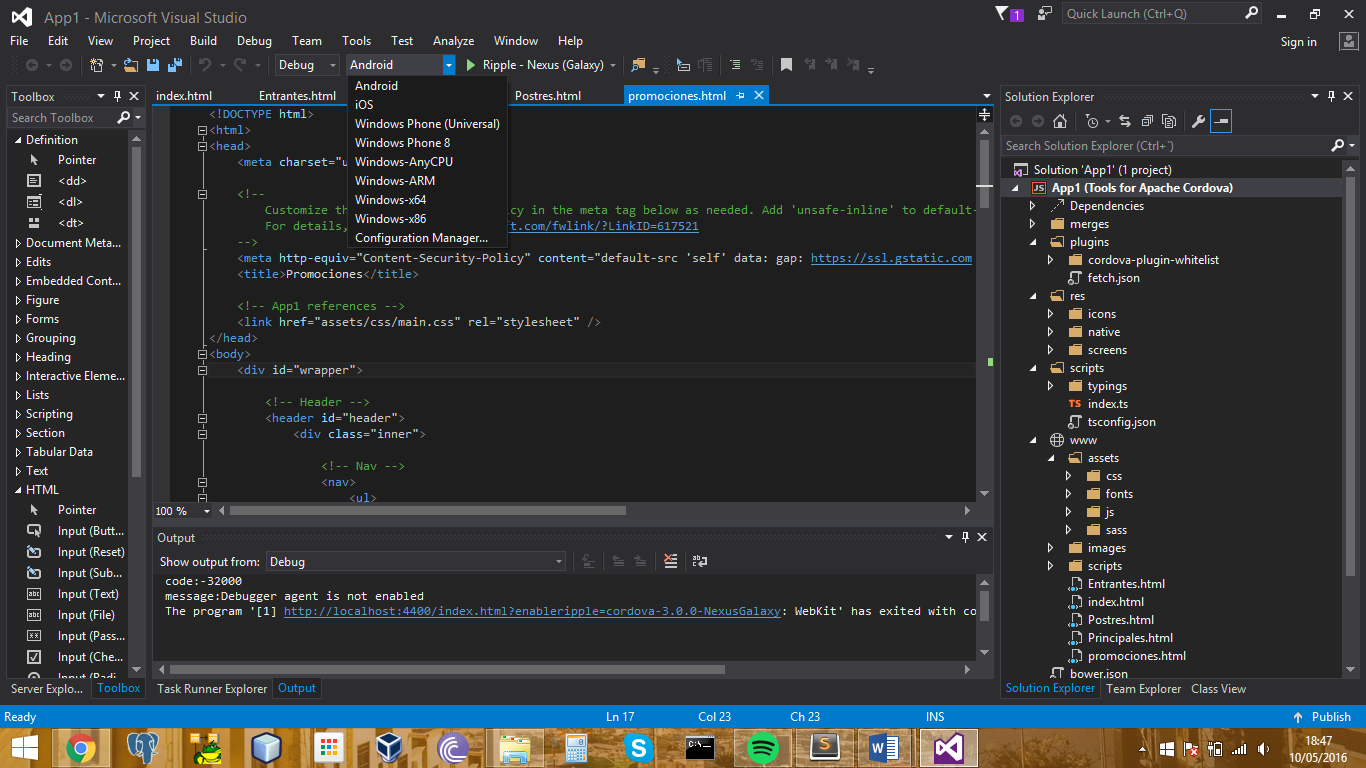


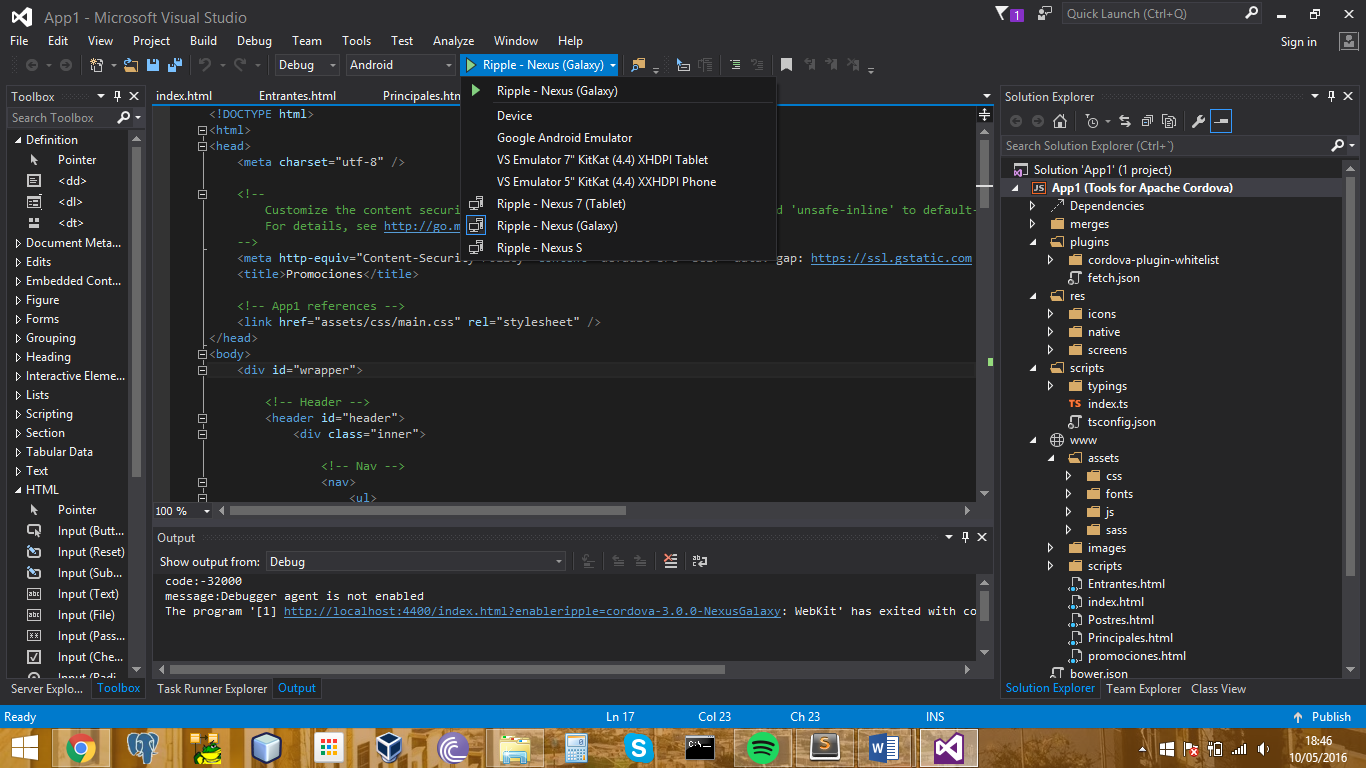
Al incluir todas las características anteriores, Visual Studio está instalando todos los complementos que necesitamos para la creación y emulación de proyectos para Apache Cordova, además de poder hacer uso de su interfaz que nos facilita el trabajo a la hora de programar.

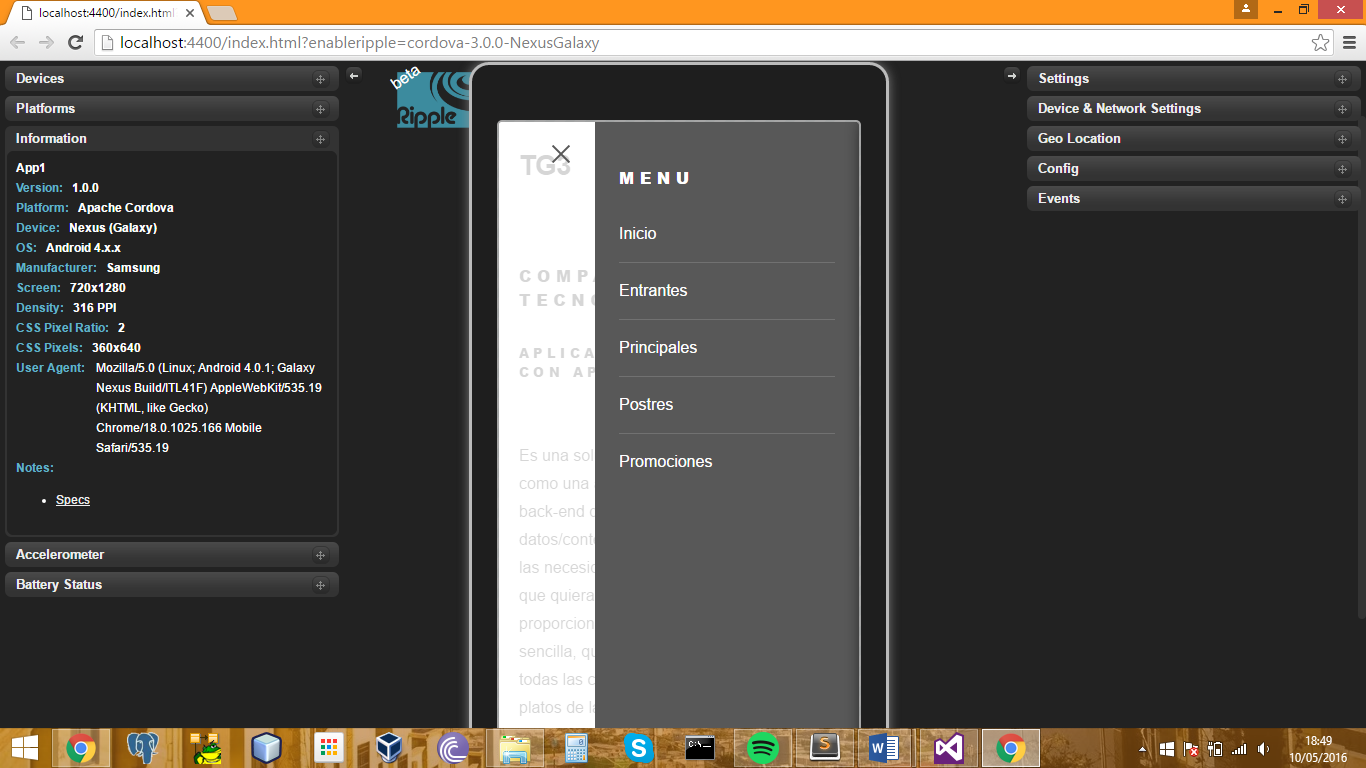
Una vez realizada la instalación, podemos crear proyectos de Apache Cordova.



Para poder emular el proyecto, o poder lanzarlo en un dispositivo hacemos uso de la solución que nos ofrece Visual Studio, pudiendo elegir que se realice una emulación entre varias plataformas y una serie de dispositivos, incluyendo la posibilidad de conectar nuestro propio dispositivo y lanzar la aplicación en el mismo.





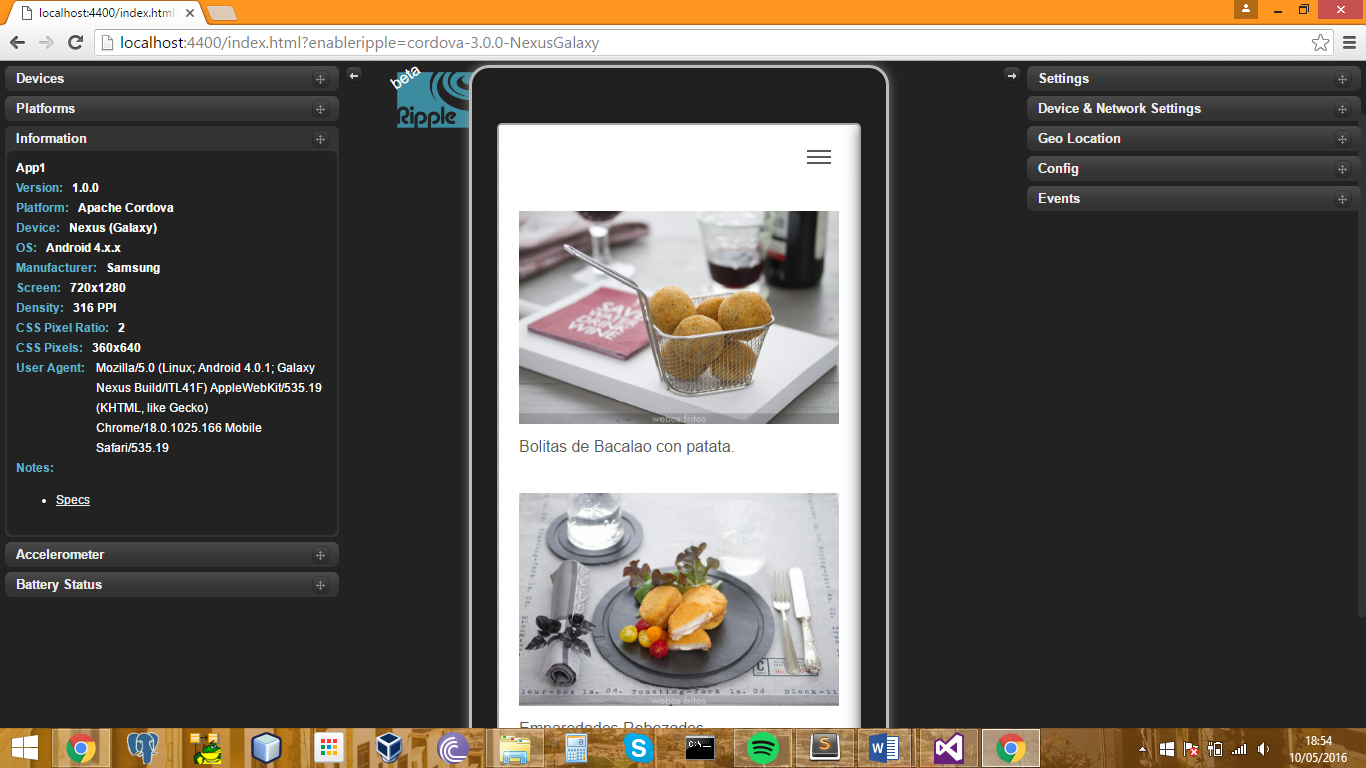


## Manual de usuario

La aplicación inicia con una primera ventana (Inicio), que contiene información principal junto con un icono que abre el menú principal

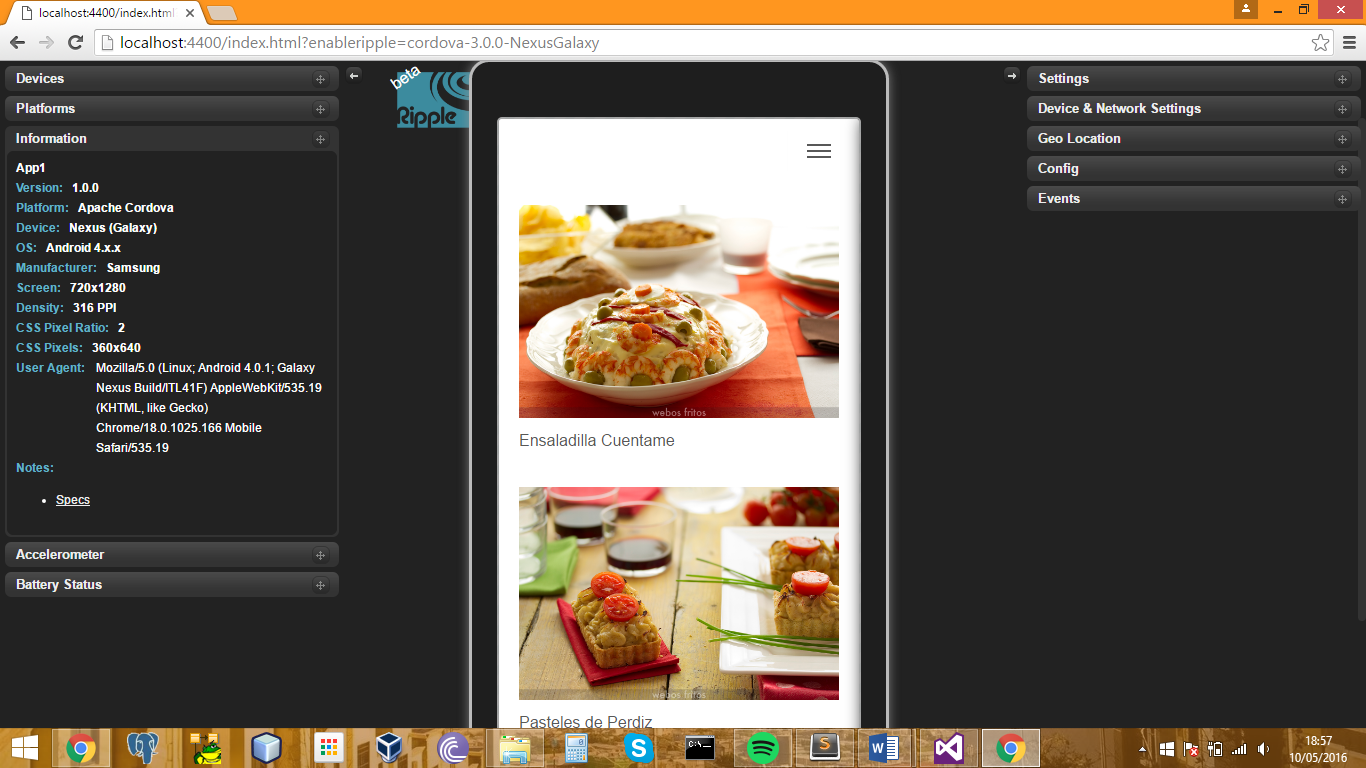
|  |  |
| --- | --- |
|  |  |

A través del menú principal podemos acceder a las demás pestañas de la aplicación:

* Entrantes:

En esta página encontraremos fotos de los entrantes con los que contamos, además de una pequeña descripción de cada uno a pie de página.

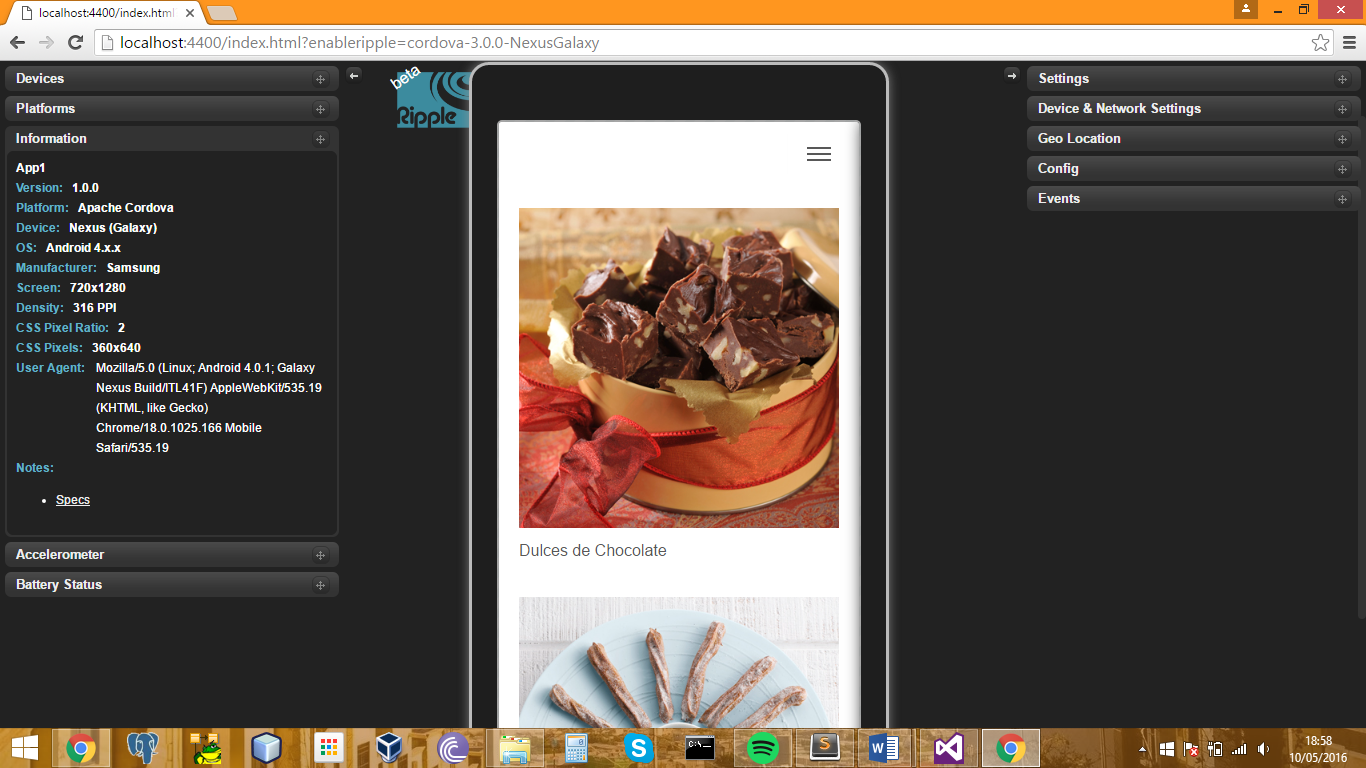
Como podemos apreciar, en la parte superior derecha seguimos teniendo el botón del menú principal. Este siempre se encontrará visible en esa posición.

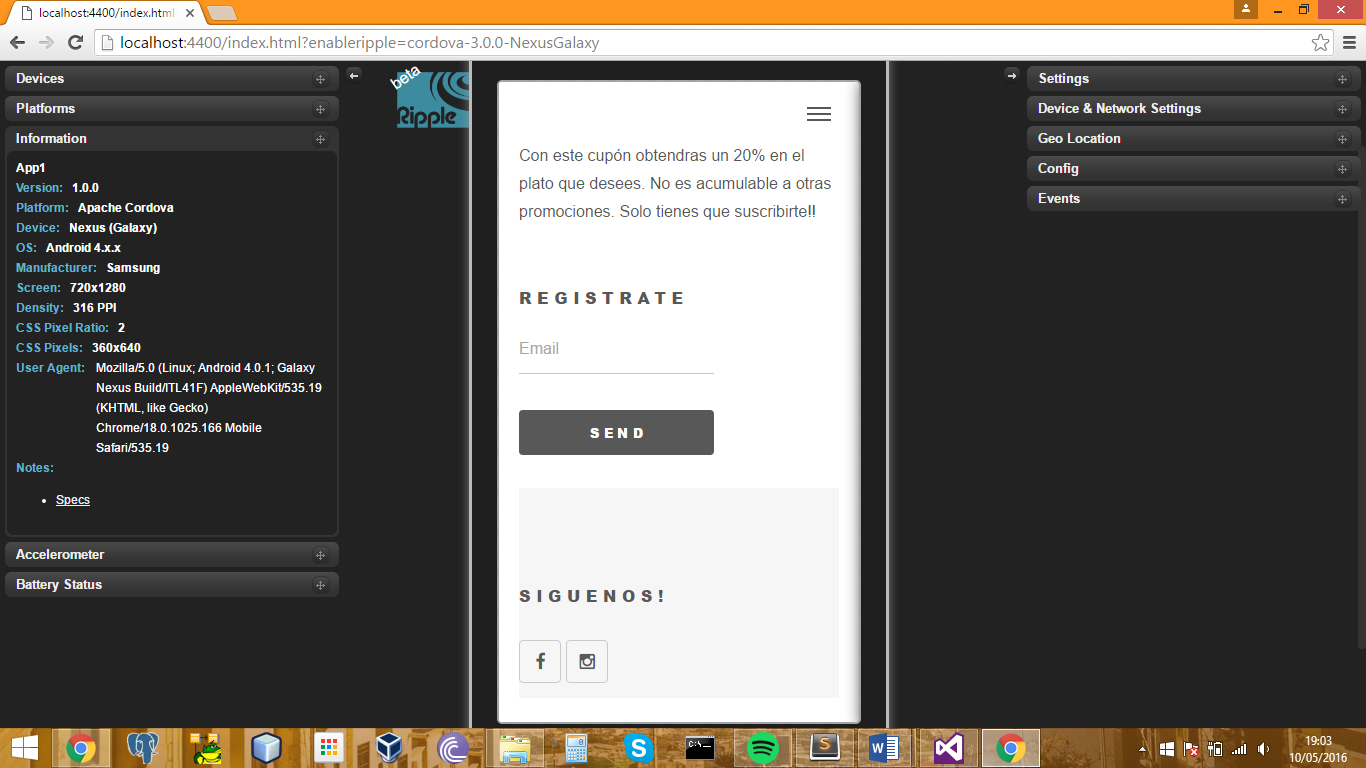
* Principales:

En esta página encontraremos fotos de los platos principales con los que contamos, además de una pequeña descripción de cada uno a pie de página.

Como podemos apreciar, en la parte superior derecha seguimos teniendo el botón del menú principal. Este siempre se encontrará visible en esa posición.

* Postres:

En esta página encontraremos fotos de los postres con los que contamos, además de una pequeña descripción de cada uno a pie de página.

* Promociones:

En esta página ofrecemos información sobre descuentos, además de incluir una sección para registrarse en la aplicación y así conseguir descuentos o información de forma individual.

En la parte inferior incluimos acceso a las redes sociales.

# Comparación de las dos implementaciones

## Evaluación de los criterios en la implementación usando Ionic

| **CRITERIO** | **EVALUACIÓN** |
| --- | --- |
| Criterio 1: Horas empleadas en el desarrollo del sistema |  |
| Criterio 2: Curva de aprendizaje |  |
| Criterio 3: Herramientas externas |  |
| Criterio 4: Estructura del código |  |
| Criterio 5: Capacidad de adaptación |  |
| Criterio 6: Rendimiento |  |
| Criterio 7: Tiempo de arranque |  |
| Criterio 8: Recursos existentes en la red |  |

## Evaluación de los criterios en la implementación usando Apache Cordova

| **CRITERIO** | **EVALUACIÓN** |
| --- | --- |
| Criterio 1: Horas empleadas en el desarrollo del sistema | 30 |
| Criterio 2: Curva de aprendizaje | 5 |
| Criterio 3: Herramientas externas | Si |
| Criterio 4: Estructura del código | 6 |
| Criterio 5: Capacidad de adaptación | 3 |
| Criterio 6: Rendimiento | 0.5 |
| Criterio 7: Tiempo de arranque | 23 |
| Criterio 8: Recursos existentes en la red | Abundante |

# Comparación de la implementación de las tecnologías

| **CRITERIOS** | **Ionic** | **Apache Cordova** | **COMENTARIOS** |
| --- | --- | --- | --- |
| Criterio 1: Horas empleadas en el desarrollo del sistema | 20 | 30 |  |
| Criterio 2: Curva de aprendizaje | 5 | 5 |  |
| Criterio 3: Herramientas externas | Sí | Si |  |
| Criterio 4: Estructura del código | 15 | 6 |  |
| Criterio 5: Capacidad de adaptación | 2 | 3 |  |
| Criterio 6: Rendimiento | 0.05 | 0.5 |  |
| Criterio 7: Tiempo de arranque | 3 | 23 |  |
| Criterio 8: Recursos existentes en la red | Abundante | Abundante |  |

# Conclusiones

El uso de aplicaciones multiplataforma es una buena opción porque permite simplificar los lenguajes de programación y no hace falta saber programar en una determinada plataforma de manera concreta.

Finalmente nos hemos decantado por Ionic ya que, a pesar de ser una tecnología similar a la de Apache cordova, tiene la ventaja de ofrecer un framework con una hoja de estilos y diseños predeterminados que nos facilita en gran medida el desarrollo de la construcción de la aplicación, esto hace que las horas empleadas en el desarrollo del código sean menores que en Apache.

Además, la capacidad de adaptación en Ionic es un tanto más sencilla que la de Cordova, por lo que al usuario no le cuesta tanto poder usar esta interfaz.

Por último, el tiempo de respuesta de Ionic es mucho menor, por lo que lo hace una tecnología ganadora