Proyecto EnFeriaTec

Área de Ingeniería en Computadores

Instituto Tecnológico de Costa Rica

CE 3101 - Bases de datos

Estudiantes

Allan Josué Calderón Quirós - 2018114634
Ronny Josué Santamaria Vargas - 2018109283
Antony Fabián Fallas Elizondo - 2018178906
Edgar Alberto Gonzalez Chacon - 2018156158

Profesor

Marco Rivera Meneses

Manual de Usuario

Esta es una aplicación que llega para brindar a las personas, tanto clientes como productores una manera muy fácil y sencilla de comprar o vender sus productos orgánicos y cuenta con una gran facilidad de manejo, al momento de su publicación la podrás descargar desde la play store o desde nuestra página web sin ningún costo adicional.

Al iniciar la aplicación mostrará una ventana la cual le brinda a la persona tres opciones muy sencillas según el interés de dicha persona, las cuales son: Administración, productor y público en general.

- Administracion.
  Se desplegará una ventana con 4 cuatro opciones, la primera es "**gestión de productores"** si la persona selecciona este botón podrá observar los productores que se encuentran registrados hasta el momento en dicha aplicación. La segunda opción se llama "**Administración de afilaciones**", en esta opción la persona podrá admitir o eliminar las solicitudes que se encuentren en la cola hasta el momento. La tercera opción corresponde a "**Gestion de categorias**" en ella puedes clasificar tus productos en las categorías existentes o si la persona lo desea puede agregar nuevas categorías o eliminar alguna que exista y no corresponda, la cuarta opción se llama "**Vista de reporte**", esta es una opcion muy util ya que permite mostrar un reporte los productos mas,con mas ganancia e incluso el productor que más compras realizo.
- Productor.
  Para la persona que seleccione este botón la aplicación le mostrará tres botones: el primero corresponde a "**Solicitud de afiliación**" esta opción está creada para las personas que aún no están afiliadas a la aplicación y desean hacerlo, al seleccionarla les brindara un formulario en el cual les solicita todos los datos correspondientes para la afiliación, luego de llenarlo la persona debe esperar ser aceptada, la segunda opción que les muestra es "**Gestión de producto**", es una opción muy importante ya que en ella el productor registra los productos que desea poner a la venta. Además de que los puede clasificar e incluso indicar la forma de venta y la tercera y última opción que se muestra en el botón de productor es "**Gestión de pedidos**", en esta el productor puede ver las personas que compraron el producto, la cantidad, el lugar donde se debe entregar e incluso el comprobante de pago.
- Público en general .
  Corresponde al botón donde las personas se dispondrán a realizar su compras, existen distintas opciones al seleccionar este botón, la primera que se muestra en pantalla es "**Gestion de cliente**", esta opción permitirá a cada persona formar parte de los clientes de la aplicación para una mayor comodidad a la hora de

pedir o comprar productos, si la persona la selecciona se le brindara un formulario el cual llenará con los datos solicitados para la posterior afiliación, la segunda opción es es para las personas que ya están afiliados, la cual se llama "**Login**", en ella las personas ingresan su nombre de usuario y contraseña y les permitirá incluso observar los productos que se venden en su distrito, "**entrar al tramo del productor**" será el tercer botón que esta ventana muestra, es de gran utilidad porque le permite al consumidor registrado observar las rutas que realiza un determinado vendedor, sus productos en existencia y el precio de los mismos, para poder agregarlos al carrito posteriormente, "**administrar carrito de compras**" serial siguiente botón, se recomienda siempre observar antes de realizar el pago de sus productos ya que ahí se muestra tu lista de pedidos a realizar, después de revisarlo y observar si todo se encuentra correcto deberás continuar al siguiente botón el cual lleva por nombre **"realizar compra**"; buscamos adaptarnos lo más posible a la comodidad de la persona, por lo que el medio de pago se realiza por medio de SINPE móvil, luego de realizar este paso solo le queda esperar el dia indicado de la entrega según la ruta del proveedor.El último botón que esta pantalla te muestra corresponde a "**Feedback**" el cual nos ayuda que el cliente deje su comentario sobre la compra y el producto que recibió, para de esta forma llevar un control de calidad sobre el servicio brindado.

1.Documentacion

1.1Estructuras:

-Arrays:

Se utilizaron arrays para la comunicación entre Front End y Back End. Para el funcionamiento de los mismos, por ejemplo, se envían los datos de un HTML form a un método javascript que los recibe y los parsea a formato JSON, para posteriormente enviarlos al Back End, el cual se encarga del manejo lógico de los datos y de la base de datos.

Además se utilizaron los arrays para almacenar los datos de los archivos Json, se toma la información del json y se genera una instancia con los datos de cada objeto para posteriormente ser almacenados en el array.

1.2 descripción detallada de los métodos utilizados

Algunos métodos fueron reutilizados y su funcionamiento es similar, así que se mencionan los métodos más importantes y su funcionamiento

-Registro(): Un método javascript que se encarga de recibir los datos que le son inyectados desde un form HTML. Este método debe obtener los datos de cada uno de los campos del form, convertirlos a variables y luego darles formato. Para esto, se

unen todos los datos que fueron ingresados y se les da un formato JSON. Este JSON es enviado al backend mediante ajax (de la biblioteca de Javascript Jquery), donde será procesado con la base de datos

-Constructor(): El Constructor es un método genérico de una clase en ES6 (en este caso TypeScript) que no guarda relación con angular. Es un método de JavaScript que llama al constructor antes que cualquier otra cosa, como en cualquier clase.

“OnInit() : Es un método relacionado con angular que da una señal de que Angular ha terminado de inicializar el componente.

Para el Rest Service los métodos implementados son:

getSize(): El método se encarga de cargar los datos del archivo Json y retorna la cantidad de entidades que contiene el archivo.

FarmersInfo(): Lee el archivo Json en el que está almacenada la información de los productores, genera un arreglo con las entidades que contiene el archivo Json y carga dicho archivo en una dirección url local.

editFarmers(): Se encarga de editar la información específica del archivo Json, por ejemplo si se quiere editar el nombre de una entidad, busca el nombre que quiere cambiar y coloca el nuevo nombre que se desea.

addDataJsonFarmers(): Este método se encarga de recibir una instancia de Farmers con los datos pertinentes a dicha clase, carga el archivo Json, agrega los datos del Json a un array, agrega la instancia nueva y sobreescribe el archivo Json cargado.

ListAllClients(): Lee el archivo Json en el que está almacenada la información de los clientes que solicitan unirse como productores, genera un arreglo con las entidades que contiene el archivo Json y carga dicho archivo en una dirección url local.

ListAllProducts(): Lee el archivo Json en el que está almacenada la información de los productos, genera un arreglo con las entidades que contiene el archivo Json y carga dicho archivo en una dirección url local.

1.3 Problemas sin solución:

Comunicación Front End- Back End:

La comunicación entre Front End y Back End es bastante limitada, ya que las diferentes formas de envío y recepción de datos que se intentaron no funcionaron o solo lo hicieron parcialmente. Primero se intentaron enviar los datos por medio de Urls, sin embargo, angular prohíbe este método, por lo que debió ser descartado.
Posteriormente se intentó utilizar el método Ajax() de la biblioteca Jquery, sin embargo, existieron muchas limitaciones. Primero, no se encontró documentación referente al respecto para el Back End exacto que teníamos. En PHP, hacer un post de esta manera es relativamente sencillo mediante este método, sin embargo para realizarlo con el Back End basado en C#, se complicó mucho más.
El front end puede recibir los datos, lo que permite acceder a los valores de la base de datos y mostrarlos. Sin embargo, no se logró trabajar sobre los datos, por lo que no se pueden modificar o crear. Los métodos en Typescript lograron parsear los datos, pero al tratar de establecer conexión siempre se obtienen errores. Se intentaron métodos alternativos de envío de datos, como postear los datos directamente en la página, sin embargo, dichos métodos debieron ser descartados.

1.5 Problemas encontrados:

Problemas con los urls y las rutas del programa:

Para la redirección hacia nuevas páginas, primero se realizó un llamado normal a la carga de una página con la etiqueta “href” de algunos componentes HTML. Sin embargo, este método estaba directamente prohibido por angular, por lo que al intentar cargar una página desde un href, el programa se caía o no reaccionaba de ninguna manera. Posteriormente, consultando la documentación de angular se halló que para cargar páginas (lo que en angular se llaman views) se debe utilizar la etiqueta [routerLink] indicando el nombre de la view. Así que se procedió a modificar las etiquetas para poder cargar las vistas adecuadamente.
Sin embargo, el Back End recibía sus datos desde un POST en una url, por lo que al modificar la forma en la que se cargan las vistas, no se pueden pasar datos por medio de la url (ya que en angular las variables se trabajan de forma diferente), por lo que se procedió a cambiar la forma en la que se envían datos al Back End.

Problemas con el envío de datos desde HTML hacia Javascript:
En primera instancia, al tratar de enviar datos desde un form HTML hacia Javascript, se procedió a crear un método que recibirá los datos luego de presionar el botón submit. Sin embargo, angular bloquea esta forma de envío de datos, por lo que , incluso insertando el script directamente en el HTML, no se pudieron enviar datos de esta manera. Para solucionar este problema, se procedió a utilizar el archivo typescript que

se genera al crear un componente. Sin embargo, el problema persistía. Posteriormente, se encontró que hay que hacer un llamado en una clase generada por angular, de otra manera, angular ignora el método. Aun así, luego de crear el método dentro de la clase, no se logró que el método fuera llamado correctamente. Esto debido a que la sintaxis para la creación e invocación del método debía ser diferente a la de javascript. Esto no fue evidente ya que se supone que Typescript es un super-set de Javascript, por lo que todo código Javascript debería ser compatible directamente con Typescript. Por lo que se desconoce si es una limitación de angular o del propio Javascript

Problemas con el acceso a los datos generados en la dirección url local:
Cuando se inició con la implementación de las conexiones entre el rest service y el front end, se buscó una manera de cargar los datos en el front end desde el url local, sin embargo, el rest service no estaba permitiendo al front end acceder a los mismos ya que no contaba con el permiso para acceso. Para solucionar este problema se tuvo que permitir manualmente el acceso al rest service desde cualquier localización.

1.6 Plan de actividades realizadas por estudiante

Allan Josué Calderón Quirós

Fecha	**Tiempo**	**Actividad**
25/09	1 h	Análisis y división del trabajo
26/09	2 h	Instalación de angular, nodejs, bootstrap y otras herramientas necesarias. Propuesta de la organización de la página
30/09	5.5 h	Estudio del funcionamiento de bootstrap, Inicia la creación de la página web. Reunión para verificar el progreso y reafirmar la división de tareas
01/10	3 h	Debugging, solución de problemas con la instalación de Bootstrap, reinstalación de proyecto debido a esto. Instalación de .NET para acceso remoto, para facilitar el trabajo en equipo.
02/10	3.5 h	Reinstalación de nuevo del proyecto debido a problemas en las dependencias (se sospecha que hay problemas con las rutas).Inicio de la estructura básica de la página principal
03/10	2.5 h	Identificación y solución del error en tiempo de ejecución de angular, creación con bootstrap de la página de entrada, inicio de creación de routes.

04/10	2 h	Desarrollo de la GUI
05/10	2 h	Solución de aspectos con Bootstrap, Creación de views de muestra de datos, adaptación para implementación de aplicación
08/10	0.75 h	Reunión para verificación del progreso, adaptación a la aplicación y unión del backend con el front-end. Se acuerda cómo se realizará el desarrollo en Typescript
09/10	2h	Desarrollo de la GUI, desarrollo de vistas y componentes para administrador, productor y clientes
10/10	2.5 h	Continúa el desarrollo del Front End. Reunión para comenzar a unir el Front End con el Back End. Agregar funcionalidades necesarias para su funcionamiento en la app
11/11	1 h	Debug, se trabaja en el envío de datos desde form HTML a Typescript y posteriormente al back end
13/10	5.5h	Debug, unión de Back End con Front End. Desarrollo de las vistas.
14/10	4h	Documentación, unión de Back End con Front End, adición de funcionalidades visuales y aceptación de errores de comunicación entre Back End y Front End
16/10	2h	Intento de comunicación entre back end y front end
20/10	3 h	Intento de comunicación entre el back end y el front end
Total	**42.25**	

Ronny Santamaria Vargas

Fecha	**Tiempo**	**Actividad**
23/09	1 h	Identificación de requerimientos de la especificación
24/09	2 h 30 min	Investigación de las herramientas para realizar el trabajo del backend. Creación de un Rest Service de prueba.
25/09	4 h	Distribución del trabajo. Lectura e implementación de archivos Json en el rest service.

27/09	2 h	Implementación de la información leída del archivo Json en el Rest Service.
02/10	2 h	Implementación de los módulos necesarios para la ejecución de aplicaciones basadas en el framework angular. Ejecución de aplicaciones con pocos componentes.
03/10	3 h	Lectura de la documentación referente al framework de angular. Corrección de errores con las pruebas realizadas.
06/10	2 h	Modificación, creación y eliminación de elementos en los archivos json desde el rest service.
08/10	1 h	Reunión para verificar el progreso del proyecto.
09/10	3.5 h	Obtención de datos en angula desde el rest service (solo se logró mostrar los datos en la consola). Corrección de los protocolos de comunicación entre el rest service y el framework utilizado para el frontend.
10/10	2 h	Construcción de una tabla en el frontend que muestre la información extraída del rest service.
12/10	1 h	Creación de una función que obtenga los datos de un url.
13/10	5 h	Pruebas de ejecución para validar el funcionamiento correcto de los componentes. Unión de Back End con Front End.
14/10	2 h	Finalización de la documentación requerida para la entrega del proyecto
20/10	6 h	Se hicieron múltiples pruebas, con el fin de lograr la comunicación del frontend con el rest service y poder modificar el estado de los archivos que almacenan los datos, las pruebas realizadas no fueron exitosas.
Total	**37 h**	

Antony Fabián Fallas Elizondo

Fecha	**Tiempo**	**Actividad**
23/09	0.5 h	Analisis del proyecto

24/	3.5 h	Investigación de teoría y ejemplos del Rest Services
26/09	1 h	Descarga e instalación de Visual Studio 2019, herramienta recomendada para realizar este proyecto
28/09	3 h	Investigacion e implementacion de archivos Json en el Rest service
03/10	2 h	Busqueda de informacion para solucionar los errores presentados a la hora de realizar el Rest service
5/10	3 h	Prueba de creacion y eliminacion de elementos desde el Rest services con json
12/10	2h	Creación del documento entregable de la Tarea
13/10	1h	Modificación del documento entregable de la tarea
17/10	2h	Realización de pruebas poder modificar el estado de los archivos que almacenan los datos
Total	**18h**	

Edgar Alberto Gonzalez Chacon

Fecha	**Tiempo**	**Actividad**
25/09	1 h	Análisis y división del trabajo.
30/09	4 h	Instalación de Android Studio, angular, nodejs, y otras herramientas necesarias.
02/10	5 h	Creación del primer proyecto para digitalizar la página de enFeriaTec.
04/10	2 h	Creación de un usuario y un correo para poder entrar a la aplicación como cliente.
09/10	4 h	Creación de una app en Android Studio para poder abrir páginas web con cualquier link.
13/10	3 h	Localización y comunicación entre el servidor de angular y el ip para loguear la aplicación con el localhost.
Total	**19 h**	

1.7 Conclusiones:

-El framework angular restringe lo que pueden hacer los elementos que conforman un componente, de manera que se sigan las buenas prácticas de programación para SPA's. Por ejemplo, se deben utilizar las routes de angular en vez de URLs, y no se deben inyectar scripts directamente en el html de un componente, ya que angular invalida su funcionamiento

- Para realizar el rest service se utilizó el editor de código visual studio code, el cual presentó una serie de complicaciones para la carga y obtención de datos al url local debido a esto y a la poca documentación que hay disponible para dicho editor los contratiempos aumentaron, lo que provocó que no se pudiera alcanzar una mayor completitud del proyecto.

-Existen diferentes formas de comunicación entre el Back end y Front end, tales como métodos proporcionados por bibliotecas (por ejemplo el método ajax de la biblioteca Jquery), REST APIs ,mediante sockets o mediante métodos propios de los lenguajes como los métodos POST y GET

1.7.1 Recomendaciones

- Seguir las buenas prácticas de programación recomendadas en la documentación de angular para SPAs, para no tener inconvenientes con las restricciones de angular.

- Utilizar el editor de código dedicado para la creación del rest service que posee microsoft (visual studio), ya que la documentación que que existe para dicho editor es muy abundante, lo cual puede facilitar la creación, comunicación y carga de datos con el frontend.

-Investigar sobre los métodos de comunicación más eficiente entre back end y front end dependiendo de los lenguajes que se están utilizando en ambas partes

1.8 Bibliografía consultada para la tarea corta

Get Started with C#. W3schools.com. (2020). Retrieved 14 October 2020, from https://www.w3schools.com/cs/cs_getstarted.asp.

Enabling CORS on Web API project in Visual Studio Code. Scott Stoecker vs. .NET. (2020). Retrieved 14 October 2020, from

https://scottstoecker.wordpress.com/2018/08/22/enabling-cors-on-web-api-project-in-visual-studio-code/.
Enable Cross-Origin Requests (CORS) in ASP.NET Core. Docs.microsoft.com. (2020). Retrieved 14 October 2020, from https://docs.microsoft.com/en-us/aspnet/core/security/cors?view=aspnetcore-2.1.
Code, E., & Howley, N. (2020). *Enable CORS at Visual Studio Code*. Stack Overflow. Retrieved 14 October 2020, from https://stackoverflow.com/questions/50276120/enable-cors-at-visual-studio-code.
C# Tutorial (C Sharp). W3schools.com. (2020). Retrieved 14 October 2020, from https://www.w3schools.com/cs/default.asp.