Documentación Técnica v1.0

Fresh Buy, Documentación Técnica.

Instituto Tecnológico de Costa Rica,

Área Académica de Ingeniería en Computadores.

Esquivel Jonathan, Ortiz Angelo, Solís Iván, Venegas Agustín.

Versión 1.0, 16 páginas.

HOJA DE CONTROL

Organismo	Instituto Tecnológico de Costa Rica	
Proyecto	FreshBuy - Tarea Corta #1, Curso de Bases de Datos	
Entregable	Documentación Técnica	
Autor	ce-itcr	
Aprobado por	ce-itcr	**Fecha de Aprobación:** 24/09/2020
		N` Total de Páginas: 16

REGISTRO DE CAMBIOS

Versión	Causa del Cambio	Responsable del Cambio	Fecha del Cambio
001	Versión Inicial	Angelo Ortiz Vega	20/10/2020

Contenidos

Presentación

Objetivos

Objetivo General

- Desarrollar una aplicación que permita gestionar la descripción del caso.

Objetivos Específicos

- Crear un WebService/REST Service (C#).
- Crear una página web.
- Usar herramientas como Angular, Bootstrap, HTML5, CSS3, Crystal Report o Reporting Services.
- Instalar localmente una aplicación Web (Front End y Back End).

Historias de Usuarios

HISTORIAS DE USUARIO - VISTA ADMINISTRACIÓN

HU001: Como administrador quiero poder agregar, eliminar y actualizar productores.	
ÉXITO: ● El usuario tipo administrador agrega, actualiza o elimina productores exitosamente de la feria digital. ● El id del productor es correcto. ● El nombre de usuario y contraseña del productor es correcto.	FALLO: Se muestra un mensaje. ● Todos los campos son obligatorios al agregar o eliminar un productor.

HU002: Como administrador quiero ver los datos principales de los productores.	
ÉXITO: ● Se muestra una tabla con todos los productores y sus datos respectivos, entre los principales: Id del Productor, Nombre del Productor, Apellido, Provincia, Cantón, Distrito, Fecha de Nacimiento, Número de Teléfono, Número SINPE y lugares de entrega.	FALLO: ● Si al ingresar a la Gestión de Productores ocurre algún fallo en conexión o error en la base de datos, la tabla de productores se muestra en blanco.

HU003: Como administrador quiero poder aceptar o negar solicitudes de afiliación.	
ÉXITO: ● Se muestra una tabla con todas las afiliaciones de productores y sus datos respectivos, entre los principales: Id del Productor, Nombre del Productor, Apellido, Provincia, Cantón, Distrito, Fecha de Nacimiento, Número de Teléfono, Número SINPE y lugares de entrega.	FALLO: ● Si al ingresar a la Gestión de Afiliaciones ocurre fallo en conexión o error en la base de datos, la tabla de productores se muestra en blanco

HU004: Como administrador quiero poder mandarle un comentario en caso de negar una solicitud de afiliación.	
ÉXITO: ● El usuario tipo administrador acepta la afiliación de un producto y este puede ingresar a FreshBuy con sus credenciales.	FALLO: ● No se logra enviar mensaje, o sólo se deniega la afiliación aparece una alerta explícita.

HU005: Como administrador quiero poder agregar, eliminar y actualizar categorías.	
ÉXITO: ● El usuario tipo administrador agrega, actualiza o elimina categorías exitosamente de la feria digital y sus productos asociados. ● El id de la categoría es correcta.	FALLO: ● Si al ingresar a la Gestión de Categorías ocurre fallo en conexión o error en la base de datos, la tabla de productores se muestra en blanco

HU006: Como administrador quiero poder ver los reportes sobre los productos, productores y clientes.	
ÉXITO: ● Se muestra una tabla con los reportes principales, entre ellos: Top 10 Productos más vendidos, Top 10 Productos que generan más ganancias, Top 10 productos más vendidos por productor, Top 10 Clientes con más compras.	FALLO: ● Si al ingresar a la Vista de Reportes ocurre fallo en conexión o error en la base de datos, la tabla de productores se muestra en blanco

HISTORIAS DE USUARIO - VISTA PRODUCTOR

HU007: Como productor quiero poder solicitar mi afiliación.	
ÉXITO: ● El usuario de tipo productor ingresa los datos correctos y completos en el registro de usuarios.	FALLO: ● EL usuario de tipo productor no completa todos los espacios para su registro, dentro de los principales: Id del Productor, Nombre del Productor, Apellido, Provincia, Cantón, Distrito, Fecha de Nacimiento, Número de Teléfono, Número SINPE y lugares de entrega, Nombre de Usuario y Contraseña.

HU008: Como productor quiero poder crear, actualizar y eliminar productos de mi tramo digital.	
ÉXITO: ● El usuario tipo productor agrega, actualiza o elimina productos exitosamente de la feria digital y sus productos asociados. ● El id de la categoría es correcta. ● El id del producto es correcto.	FALLO: ● Si al ingresar a la Gestión de Productos ocurre fallo en conexión o error en la base de datos, la tabla de productores se muestra en blanco

HU009: Como productor quiero poder ver un listado de las compras realizadas por cliente para cada producto.	
ÉXITO: ● Se muestra una tabla con los pedidos principales realizados por los clientes en orden descendente.	FALLO: ● Si al ingresar a la Gestión de Pedidos ocurre fallo en conexión o error en la base de datos, la tabla de productores se muestra en blanco

HISTORIAS DE USUARIO - VISTA WEB PÚBLICO GENERAL

HU010: Como cliente quiero que se me permita iniciar sesión en el sistema ingresado nombre de usuario y contraseña.	
ÉXITO: ● EL usuario ingresa correctamente su nombre de usuario y contraseña.	FALLO: ● Si el usuario de tipo cliente no ingresa correctamente el nombre de usuario o contraseña, se presenta una alerta en pantalla. ● Si al ingresar al Inicio de Sesión ocurre algún fallo en conexión o error en la base de datos,se muestra un error de conexión.

HU011: Como cliente quiero que al ingresar al sistema inmediatamente se me muestran por defecto todos los productores afiliados que entregan productos en mi distrito.	
ÉXITO: ● Se muestran cartas con la fotografía de cada productor asociado a un tramo.	FALLO: ● Se presentan en blanco las cartas por error de conexión o error de datos.

HU012: Como cliente quiero entrar virtualmente al tramo de cada productor y visualizar productores cerca de mi región.	
ÉXITO: ● Se muestran cartas con la fotografía de cada producto asociado a un productor específico, si no posee conexión se muestra una tabla con los productos asociados.	FALLO: ● Se presentan en blanco las cartas de productos por error de conexión o error de datos.

HU013: Como cliente quiere ver la disponibilidad y precio de cada artículo que se muestre en los tramos de los productores.	
ÉXITO: ● Automáticamente si se muestran los productos, también el precio y la disponibilidad asociada a ese productor.	FALLO: ● Se muestran cartas con la fotografía de cada producto asociado a un productor específico, si no posee conexión se muestra una tabla con los productos asociados.

HU014: Como cliente quiero administrar el carrito de compras para revisar los pedidos antes de realizar la compra, y eliminar productos o modificar cantidades solicitadas .	
ÉXITO: ● Se muestran correctamente todos los pedidos dentro del carrito de compras	FALLO: ● Se muestra la tabla del carrito de compra vacía.

HU015: Como cliente quiero que se me permita realizar compra y que se me envíe un mensaje con un comprobante del SINPE móvil realizado.	
ÉXITO: ● Se le envía un sms al teléfono del consumidor con el comprobante detallado de su compra incluyendo los productos, el valor, los impuestos, dirección de entrega.	FALLO: ● Si existe un fallo de conexión, el sistema no deja avanzar al consumidor a la página de feedback.

HU016: Como cliente quiero dar feedback al productor una vez que la entrega sea efectuada.	
ÉXITO: ● Se llena correctamente la entrada de comentarios con una extensión mayor a 100 caracteres.	FALLO: ● Si el consumidor no llena la entrada de comentarios o no da estrellas al sistema, no deja avanzar dentro de la aplicación.

Especificación de Requerimientos de Software

TIPOS DE USUARIO

Tipo de Usuario	Descripción
Usuario Administrador	Dentro de sus principales funciones en la aplicación se encuentran: 1. Gestión de Productores 2. Administración de Afiliaciones 3. Gestión de Categorías 4. Vista de Reportes
Usuario Productor	Dentro de sus principales funciones en la aplicación se encuentran: 1. Gestión de Productos 2. Gestión de Pedidos
Usuario Consumidor	Dentro de sus principales funciones en la aplicación se encuentran: 1. Visualización de Tramos 2. Visualización de Productos 3. Administración de Carrito de Compras

REQUERIMIENTOS FUNCIONALES DE APLICACIÓN WEB

1. Módulo Login / Logout y Registro de Usuarios
    a. RF001 - Iniciar Sesión
        i. Descripción: La pantalla de inicio de sesión permite a los usuarios registrados iniciar sesión en la aplicación acceder a todas las funcionalidades a las que su cuenta les dé acceso.
        ii. Validaciones:
            1. El campo Contraseña permite máximo 8 caracteres.
            2. El campo contraseña se captura con una máscara de asteriscos.
    b. RF002 - Seleccionar tipo de Perfil para registrarse
        i. Descripción: En esta pantalla se le solicita al usuario que defina si su tipo de cuenta va a ser del tipo productor o consumidor.
    c. RF003 - Solicitar Afiliación como productor
        i. Descripción: La pantalla de Afiliación como productor es un formulario donde el productor ingresa sus datos con el fin de que le incluya en el sistema.
        ii. Validaciones:

1. El campo Número de Cédula debe poseer mínimo 7 caracteres y máximo 10
2. El campo Residencia debe tener el formato: Provincia, Cantón, Distrito.
3. El campo fecha de nacimiento debe poseer formato DD/MM/YY
4. El campo Número Telefónico debe poseer 8 caracteres.
5. El campo Número de SINPE Móvil debe poseer 8 caracteres.

d. RF004 - Registrarse como Consumidor
   i. Descripción: La pantalla de Registro como Consumidor es un formulario donde el cliente ingresa sus datos para que le de ingreso al sistema de compras.
   ii. Validaciones:
      1. El campo Residencia debe tener el formato: Provincia, Cantón, Distrito.
      2. El campo Contraseña permite máximo 8 caracteres.
      3. El campo contraseña se captura con una máscara de asteriscos.

e. RF005 - Cerrar Sesión
   i. Descripción: El usuario presiona un botón para salir de la aplicación.

2. Módulo Vista Administración
   a. RF006 - Gestión de Productores
      i. Descripción: En esta pantalla el administrador debe poder actualizar, crear, y eliminar productores.
      ii. Validaciones: Los datos que debe poseer esta tabla estrictamente son: Número de Cédula, Nombre, Apellidos, Dirección en formato: Provincia, Cantón, Distrito, Fecha de Nacimiento, Número Telefónico, Número de SINPE Móvil y lugares de entrega de los pedidos.
   b. RF007 - Administración de Afiliaciones
      i. Descripción: En esta pantalla el administrador acepta o declina una solicitud de afiliación
      ii. Validaciones:
         1. Si se deniega afiliación se debe enviar un mensaje con las razones por las cuáles se denegó.
   c. RF008 - Gestión de Categorías
      i. Descripción: En esta pantalla el administrador puede crear, actualizar y eliminar categorías.
   d. RF009 - Vista de Reportes
      i. Descripción: En esta pantalla se le permite al administrador visualizar:
         1. Top 10 Productos más vendidos
         2. Top 10 Productos que generan más ganancias
         3. Top 10 Productos más vendidos por productor
         4. Top 10 Clientes con más compras

3. Módulo Vista Productor
   a. RF010 - Gestión de Productos

i. Descripción: Se le permite al productor crear, actualizar y eliminar productos.
ii. Validaciones: Los datos que debe poseer esta tabla estrictamente son: Nombre, Categoría, Foto, Precio, Modo de Pago y Disponibilidad.

b. RF011 - Gestión de Pedidos
i. Descripción: En esta pantalla se le muestra al productor el listado de pedidos que le han llegado a su tramo.
ii. Validaciones:
1. Cada pedido debe poseer la lista de productos; nombre de consumidor, dirección y comprobante de pago.

c. RF012 - Configuración de Usuario
i. Descripción: Se le permite al Productor cambiar contraseña de su usuario

4. Módulo Vista Web Público General
a. RF013 - Visualización de Tramos
i. Descripción: Se le debe permitir a los consumidores visualizar tramos cerca de la provincia.

b. RF014 - Ingreso a un tramo específico
i. Descripción: Al presionar en un tramo, se deben visualizar todos los productos que este tramo posee.

c. RF015 - Agregar Productos a carrito desde tramo
i. Descripción: en la pantalla de tramo se deben añadir productos al carrito de compras.

d. RF016 - Administración de Carrito
i. Descripción: El usuario puede visualizar, editar y eliminar los productos que ha agregado al carrito de compras. Además debe presionar el botón de Finalizar Compra, para terminar la compra en Fresh Buy.
ii. Validaciones:
1. Debe existir productos en el carrito para finalizar la compra.
2. Debe poder aumentar o disminuir la cantidad sobre un producto que desea comprar.

e. RF017 - Enviar comentarios al productor
i. Descripción: Al terminar la compra, el consumidor debe enviar un comentario de feedback obligatorio y debe dar calificación al productor y calificación a la aplicación.

f. RF018 - Configuración de Usuario
i. Descripción: Se le permite al Consumidor cambiar contraseña de su usuario

Descripción de los métodos implementados

MÉTODOS PRINCIPALES FRONT-END:

Logic.ts:
Este archivo almacena todas las funciones exportadas, con el fin de mantenerlas almacenadas y ordenadas en un archivo.

- update_components: Este método utiliza una instancia de CS (Comunication Sevice), la cual le permite comunicarse con el API y obtener un json de los datos necesarios para actualizar la tabla de cada uno de los componentes, posteriormente mediante un bucle se asignan a una variable global los los key-values necesarios para construir o actualizar la tabla de productores.
- back-disable: Este método fue implementado para restringir a los usuarios la posibilidad de retroceder (mediante el buscador) entre componentes, la misma fue implementada de manera simple mediante una instancia de “LocationStrategy”.

Comunication.Service.ts:
Mediante este archivo se realizará toda la comunicación entre FrontEnd-API, tanto los métodos “get” como “post”.

- sendData: Se encarga de hacer un “post” desde el FrontEnd hacia el API. En la mayoría de los casos se utiliza para actualizar el .json de productores, clientes, productos, etc. O también a la hora de solicitar un inicio de sesión.
- getData: Se encarga de hacer un “get” de un json, con el fin de actualizar elementos gráficos como tablas. Comúnmente se ligaron estos métodos a logic.ts

MÉTODOS PRINCIPALES BACK-END:

DataRequest.cs:

- INSERT: La misma recibe un “path” el cual indicará la dirección de almacenamiento, y un json con datos a probar (deben pertenecer al apartado respectivo relacionado con el “path” indicado: productor, producto, categoría, etc), una vez verificados los datos se añade el mismo a la base de datos.
- DELETE: Recibe un “path”, el cual indicará a la funcionalidad el tipo de dato y el destino, recibe un json con el “id” del objeto a eliminar, una vez encontrada la coincidencia, se remueve el objeto de la base de datos respectiva.
- SELECT: Es el método utilizado por DELETE para encontrar el objeto a eliminar, específicamente, “SELECT” se encarga de buscar en un “path” un objeto, mediante la utilización de un json que contiene el “id” del objeto que se busca.

Descripción de las estructuras de datos implementadas

En la implementación de este proyecto se vió útil utilizar las siguientes estructuras de datos: JSON, Hashmap y Arrays. Se realizó toda la comunicación entre el API y el frontend por medio de JSON, esto fue útil para el proyecto ya que los JSON son una de las estructuras que más se han utilizado en los proyectos anteriores y el equipo estaba familiarizado con esta misma. Su estructura de hashmaps hace que su implementación sea relativamente fácil. Como se mencionó anteriormente todos los mensajes enviados del API al frontend y del frontend al API tienen el formato JSON, con un par de excepciones. JSON es un formato de texto para el intercambio de datos, este se basa en que cuando se vaya a enviar un mensaje se pase de un hash a un string y hacerle pasarlo de vuelta cuando llega al lugar destinado por medio de un Parse.

Aunque los hashmaps se utilizaron mayormente por la implementación de JSON este no fue el único uso que se le dio a los hashmaps ya que por medio de la función LocalStorage de angular 9 también se hizo uso de este. Los hashmaps son estructuras de datos en la que se guardan datos asociados primero se le asigna un valor a una llave y luego para llamar a este se utiliza la llave.

Los arrays fueron una de las estructuras que más utilizamos por su utilidad con elementos que tienen “n” cantidad de elementos y su fácil implementación. Estas son zonas de almacenamiento continuo que contiene una serie de elementos del mismo tipo. Los arrays fueron utilizados muchas veces cuando se ocupaba guardar información que llegaba del servidor y cuando se ocupaba leer los JSON guardados en el servidor.

Descripción detallada de los algoritmos desarrollados

Agregar datos: Como Administrador y productor, existe la opción de agregar productores, categorías, productos, etc. Esto se hace mediante el ingreso de los datos necesarios para crear cada uno de ellos, de este modo se construye un json, el cual se envía al API, y mediante el método “INSERT”, se añade al apartado de la base de datos respectivo.

Eliminar datos: Diversos componentes le permiten al usuario eliminar datos (entiéndase como productores, productos, categorías, etc), de modo que mediante el ingreso del id respectivo, se envía un json con el mismo al API, el cual se encarga mediante el método “DELETE” de eliminarlo de la base de datos.

Modificar datos: Se debe ingresar el id de un objeto ya almacenado (mediante el SELECT se verifica que exista), posteriormente se comprueban los datos ingresados (que pertenezcan a el tipo de dato almacenado en el “path” especificado), una vez hecho, se elimina el objeto encontrado con el SELECT (mediante la utilización del DELETE), por último, se añade el objeto modificado mediante el INSERT.

Verificar Datos: Cuando se realiza el login o se ocupa registrarse se tiene que validar si los datos que se ingresan ya existen en el servidor. En el caso del login se ocupa revisar si existe el usuario ingresado y si existe se ocupa revisar que la contraseña del usuario sea la correcta, y en registar se ocupa validar que el id no se repita. Esto se realiza con la función FILTER, la que va a devolver una lista con la información que contiene el usuario con ese nombre de usuario y luego se revisa si la contraseña coincide con la que se ingresó.

Manejo de Afiliaciones de Productores: Cuando un usuario se registra como productor, inicialmente su solicitud se almacena en un json de productores, de este modo, cuando un administrador ingresaba a este apartado puede aceptar o denegar las solicitudes, en ambos casos, mediante el ingreso del id de la afiliación a aceptar o denegar, el server elimina la afiliación con dicho id, y lo agrega a productores, o, si lo deniega, únicamente lo elimina de la misma.

Manipular JSONs: Los json fueron utilizados en toda comunicación realizada, debido a lo útiles que resultan a la hora de almacenar datos. De modo que, en todo momento, los datos en front end, o en backend, eran almacenados mediante dicha estructura. Fueron utilizados para enviar y recibir datos de usuarios, así también como productos y afiliaciones.

Problemas Conocidos

- Hicieron falta algunas verificaciones de datos, por ejemplo, el ingreso de datos no esperados en algunos apartados del servidor, hace que el mismo se detenga y entregue un error no previsto.

Problemas Encontrados

- Comunicación API-frontend: Este fue un error cuando se comenzó a trabajar en la parte de comunicación en el que por más que las funciones estaban bien escritas estas no funcionaban, se utilizó postman para ver si las funcionaban en el API y sí respondía lo que debía pero no funcionaba con el frontend. Este error se arregló implementando los Cores en el API.
- Reseteo de información cuando se recarga: Este error salió casi en toda la etapa de implementación y se arregló casi al final por lo que muchos de los métodos fueron implementados alrededor de este y muchas de las funcionalidades que tenemos se pudieron implementar de una mejor forma. Este error era que cuando se pasaba de páginas las variables globales borraban su valor. Se arregló con el uso de LocalStorage que es una función de angular que guarda valores de la misma forma que lo hace un hash.
- En ciertos componentes se impedía la comunicación: Este error sale cuando se estaba implementando la eliminación de productores en las afiliaciones. El error era que cuando se enviaba la señal del frontend al API el mensaje era bloqueado por el cliente . Esto se arregló con la desactivación de programas externos.

Bibliografía

Juntadeandalucia.es. 2020. *Plantilla Manual De Instalación | Marco De Desarrollo De La Junta De Andalucía*. [enlinea] Disponible en: <http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/466>

Jackson, L. (2020). *.NET Core 3.1 MVC REST API - Full Course* [Video]. Retrieved from https://youtu.be/fmvcAzHpsk8

Angular University. (2020). Retrieved 20 October 2020, from https://angular-university.io/