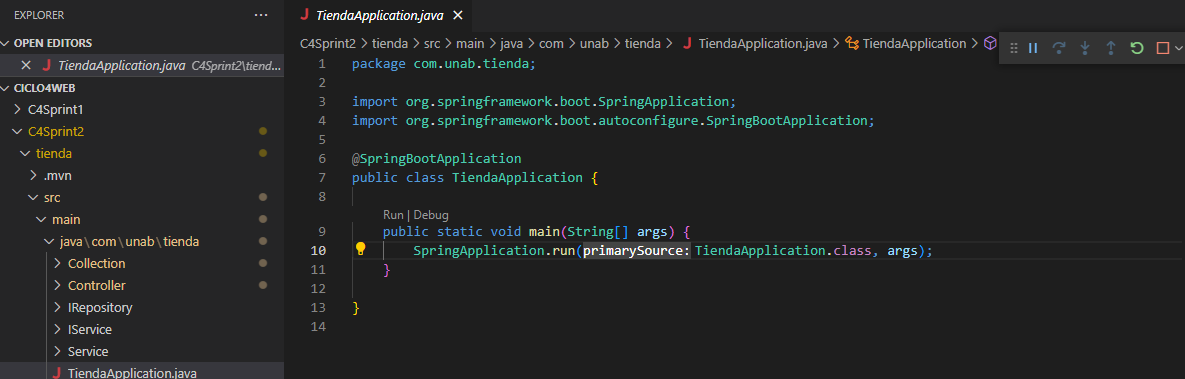
|  |
| --- |
| **SPRINT 3: Desarrollo del Frontend** |

|  |  |
| --- | --- |
| **Identificación Proyecto** | |
| Nombre Proyecto: | Software para la gestión de una tienda de barrio |
| Número Equipo: | N.A |
| **Integrantes del equipo** | |
| Rol  (Líder-Desarrollador – Cliente) | Nombre |
| Líder | Robinson Daniel Cruz Castro |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| Stakeholder | Rodolfo Ortega |

|  |
| --- |
| **Evidencia construcción del Frontend** |

El frontend se construyó mediante React SpringBoot y se revisó su funcionalidad con la información proveniente del backend del anterior split (API ejecutándose en el localhost:8080 mediante archivo TiendaApplication,java disponible en la carpeta C4Sprint2), la ejecución del fronend se realizó en el localhost:3000 con el comando “npm start” ejecutado desde la carpeta sprint3)



A screenshot of a computer

Description automatically generated with medium confidence

Como se ve en la anterior imagen, con el documento App.js se utilizó tres páginas tipo single page application (SPA), siendo Crear Productos la página encargada de la creación/eliminación de las estructuras definidas (marcas, categorías, unidades de medida y productos) consumiendo de esta forma los servicios de Post y Delete respectivos mediante los componentes Crear/Eliminar+”estructura”+Formulario.js

Text

Description automatically generated

Para consumir los servicios de eliminación, se generó un formulario que captura los datos recopilados en una lista desplegable, al seleccionarlo toma el id respectivo y lo envía al componente Boton Eliminar, utilizado de manera general para eliminar cualquier tipo de estructura, por lo que requiere de los parámetros id y endpoint.

Text

Description automatically generated

Text

Description automatically generated

La pagina Inventarios Creados se utilizó para mostrar la información resumida de la estructura Inventario que recopila los datos de las demás estructuras y la pagina permite también la creación de nuevos inventarios; para mostrarlos llama al componente ProductoCard. Se anexa de igual forma el esquema de datos de la estructura en formato JSON para mejor entendimiento.(por practicidad y falta de tiempo, se dejo a un lado la variable bool estado, debido a que nunca se utilizó en la elaboración del frontend).

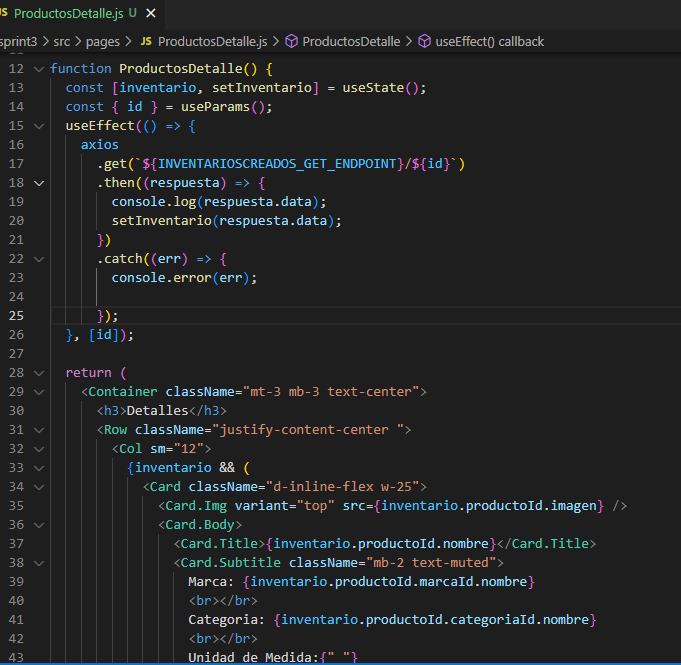
Text

Description automatically generated

Text

Description automatically generated

Finalmente la pagina Productos Detalle muestra toda la información almacenada dentro de la estructura más completa (inventario) para el id especificado en la URL y también permite realizar la eliminación de este elemento.

Text

Description automatically generated

|  |
| --- |
| **Evidencias del Frontend en funcionamiento** |

Creación de Inventario {stock:30 , Cantidad máxima: 200 , Cantidad mínima:50 , Costo:1000 , precio 1500} producto: Chocolatina Kinder , Categoría:dulce, Unidad de medida:Unidad, Imagen: https://http2.mlstatic.com/D\_NQ\_NP\_611124-MCO46393001069\_062021-O.webp

\*condición inicial frontend:

Graphical user interface, application

Description automatically generated

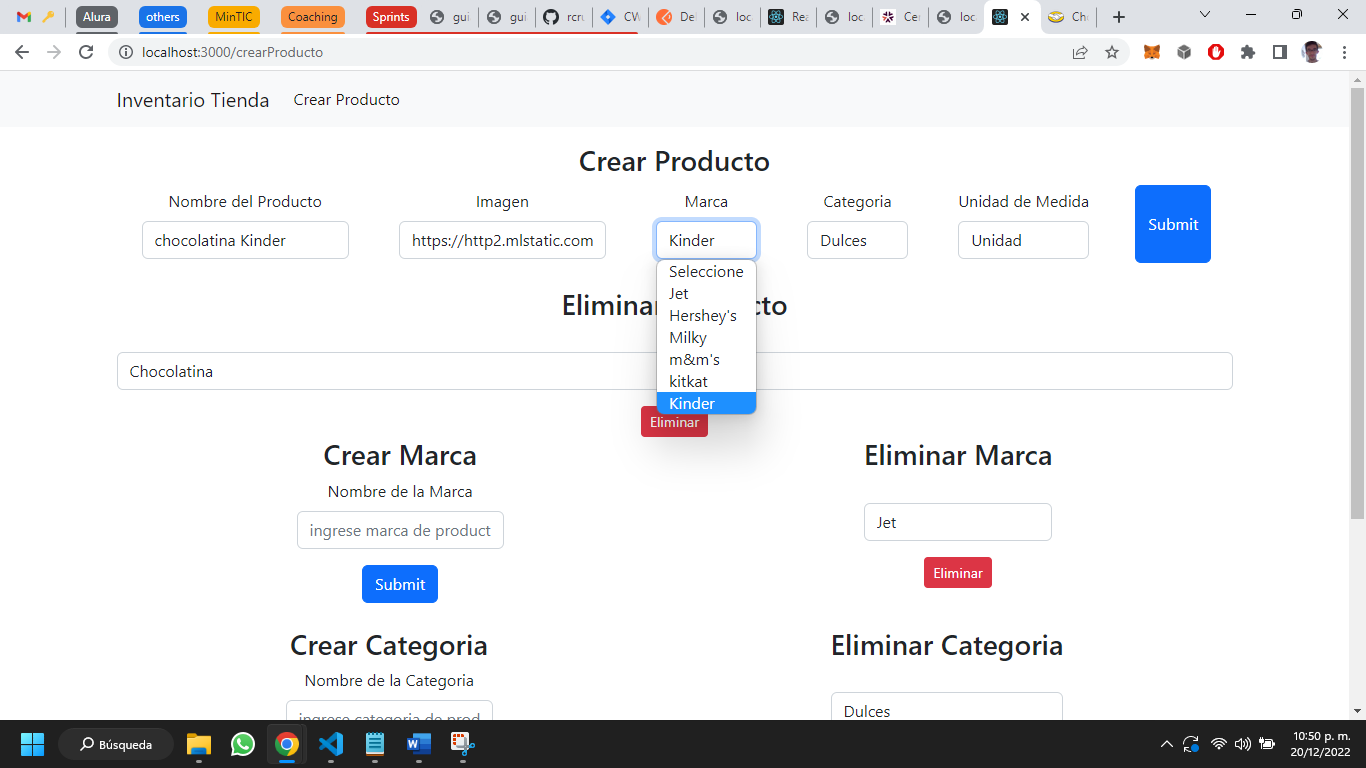
Graphical user interface, application

Description automatically generated

\*debido a que los otros productos ya ingresados tienen la categoría dulce y la unidad de medida “unidad”, no se realizará su creación si no que se partirá de la creación de la marca “kínder”

Graphical user interface, application

Description automatically generated



Después se crea al producto asignándole el inventario:

Graphical user interface, application

Description automatically generated

Después de crear el inventario, se debe recargar la pagina para actualizar la visualización de inventario (el tiempo no me alcanzó).

Graphical user interface, application, website

Description automatically generated

Al hacer click sobre el nombre, se dirige a la pagina de detalles, en este caso se eliminará el inventario del producto kitkat.

Graphical user interface, application, website

Description automatically generated

Graphical user interface, application

Description automatically generated

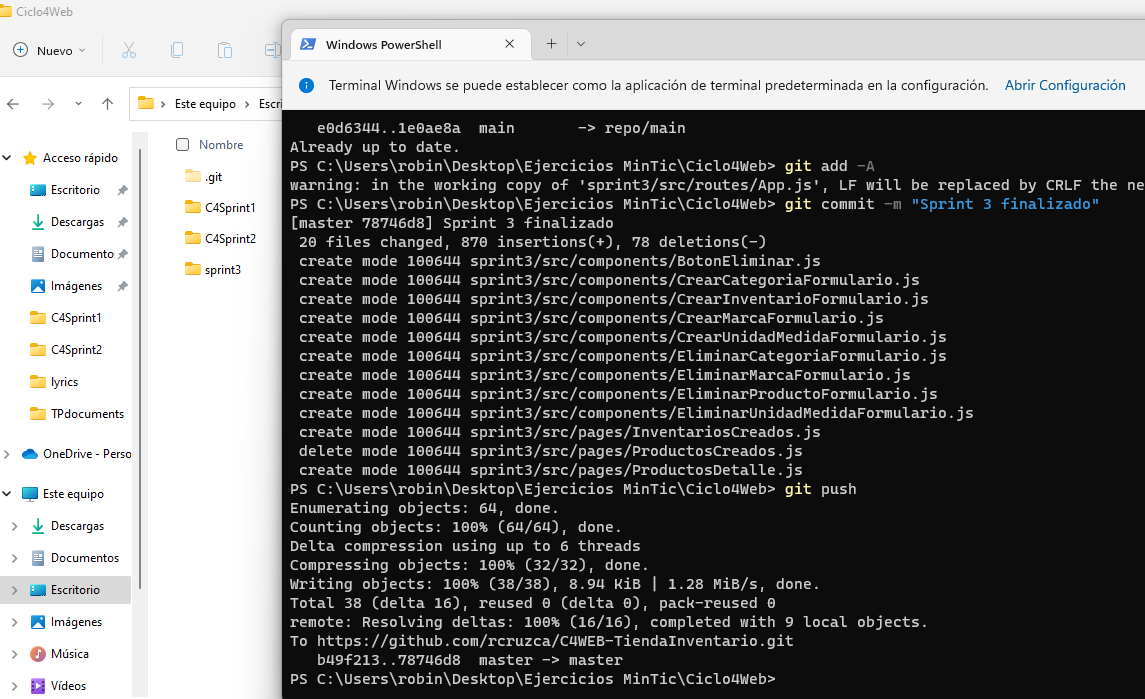
Graphical user interface, application

Description automatically generated

|  |
| --- |
| **Evidencia GitLab o GitHub** |

Repositorio: <https://github.com/rcruzca/C4WEB-TiendaInventario.git>

Push del sprint 3 finalizado



Recibimiento de la petición para el merge

Graphical user interface, text, application, email

Description automatically generated

Aceptación de cambios

Graphical user interface, text, application, email

Description automatically generated

Estado Actual – Subiré todo esto al servidor, pero no alcanzaré a subir el Sprint 4 antes de media noche, espero pueda probar mi frondend exitosamente, por lo que en el comprimido adjunto irá la carpeta completa que incluye la API del sprint 2 usado como base para este frondend. Solo eliminare la carpeta node\_modules que pesa demasiado y no se requiere ya que de igual forma la tiene de manera local.

Graphical user interface, text, application, email

Description automatically generated

|  |
| --- |
| **Evidencia JIRA (Seguimiento del proyecto)** |

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application, Teams

Description automatically generated

|  |
| --- |
| **Evidencias de las Reuniones de Equipo** |

Debido a falta de tiempo, se decidió desde el Sprint 2 realizar los entregables de manera individual.