PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN

IIC2513 - Sección 1 — Tecnologías y Aplicaciones Web

Tarea 2

Entrega

- **Fecha y hora**: Viernes 12 de abril del 2024, a las 22:00
- **Lugar:** Repositorio individual en la organización del curso en GitHub, main branch.

Objetivos

- **Comprender** el funcionamiento de una API a partir de documentación.
- **Crear** un programa que pueda manejar requests y consumir una API
- **Producir** documentación efectiva y clara que permita el entendimiento del proceso realizado

Descripción

En esta tarea, explorarán cómo integrar y utilizar una API (Interfaz de Programación de Aplicaciones), una habilidad muy útil en el desarrollo de software actual. Frecuentemente, esto les permitirá aprovechar recursos preexistentes en lugar de desarrollar soluciones desde cero, optimizando así el uso de su tiempo y recursos. Esta práctica no solo les ahorra esfuerzo, sino que también les introduce a la amplia gama de herramientas y servicios disponibles en la nube, preparándolos para enfrentar desafíos similares en entornos profesionales.

Para este propósito, deberán hacer uso de una **API de Platzi**, diseñada para simular la información de una tienda en línea. Esta API les suministra datos detallados sobre un e-commerce, permitiéndoles crear un prototipo funcional de una tienda virtual. Tendrán la oportunidad de interactuar con una variedad de productos disponibles, implementando funcionalidades esenciales para la navegación y gestión del catálogo. La información obtenida de la API deberá ser presentada a los usuarios a través de una interfaz desarrollada en React, asegurando una experiencia interactiva y atractiva.

Especificaciones

Para esta tarea se espera que puedan recrear un e-commerce utilizando componentes de react, a partir del código base entregado. Para lograr esto, se les entregará una **documentación en postman** de los endpoints que deben consumir para esta tarea.

Pages

Se les entregarán 3 archivos en la carpeta de pages, las cuales serán utilizadas para mostrar las vistas de la página.

- **ProductsPage**: Esta página deberá contener todas las tarjetas de productos de la API. Estas deben ser desplegadas de forma que se puedan visualizar todos los productos en la página. Además, en el header deberá existir un botón para publicar productos.

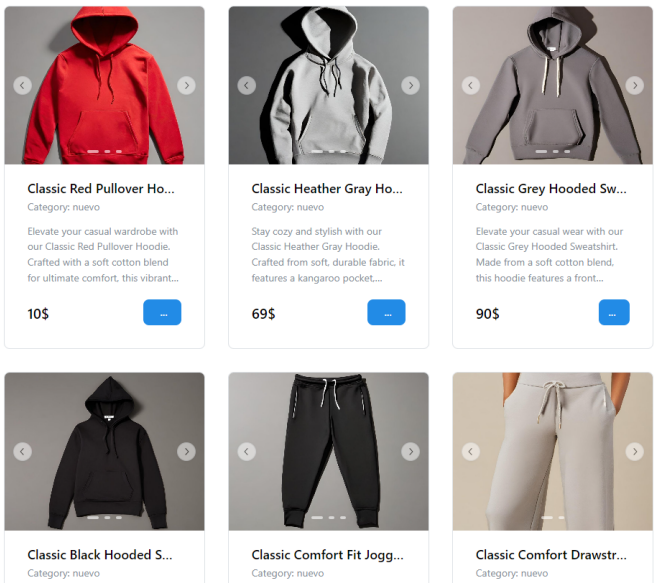

Figura 1: Ejemplo representativo de la página *ProductsPage*.

- **ProductFormPage**: Esta página será utilizada únicamente cuando quieran crear o editar un producto. Al ser un formulario, deberá contar con validación de datos y pop-ups en caso de errores, o registros exitosos.

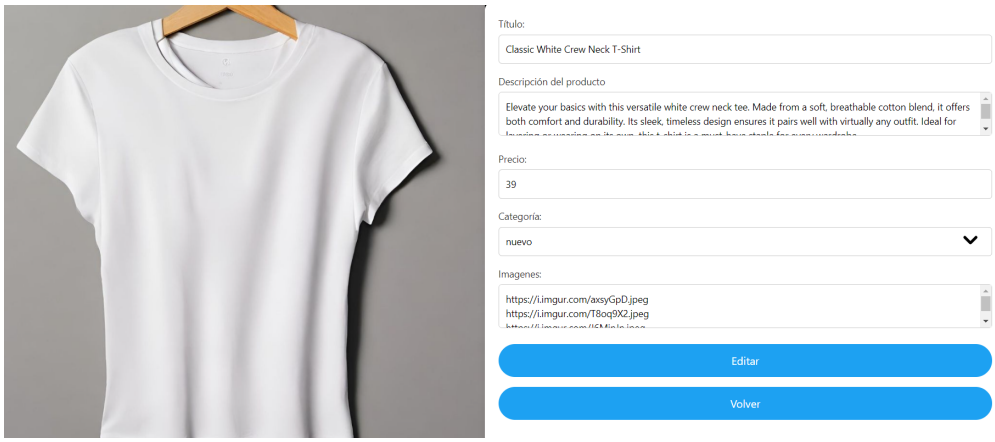

Figura 2: Ejemplo representativo de la página *ProductFormPage*.

- **ProductInfoPage**: Esta pagina está diseñada para presentar información detallada de un producto específico. Al elegir un producto desde la página principal, el usuario será redirigido a **ProductInfoPage**. Aquí, se mostrarán todas las imágenes del producto y se proporcionará una descripción completa del producto seleccionado.

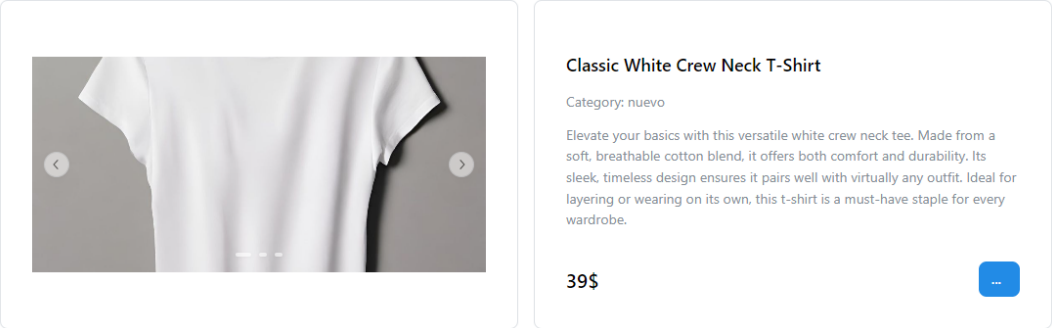

Figura 3: Ejemplo representativo de la página *ProductInfoPage*.

Components

Su página debe contar mínimo con estos componentes.

- **Buttons:** Componente utilizada para realizar una acción en especifico, además de mostrar un texto.
- **DropDown:** Componente funcional diseñada para renderizar un elemento de seleccion *<select>*
- **ProductForm:** Componente utilizada para crear y editar la información sobre un producto. Esta al ser un formulario, debe incluir validación de datos.

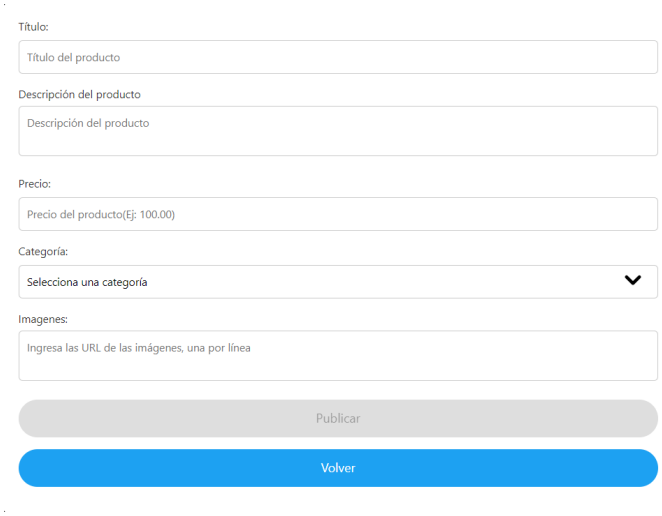

Figura 4: Ejemplo del *ProductForm*.

- **ProductCard:** Componente diseñada para mostrar información detallada de un producto y proporcionar interacciones específicas como editar y eliminar dicho producto.

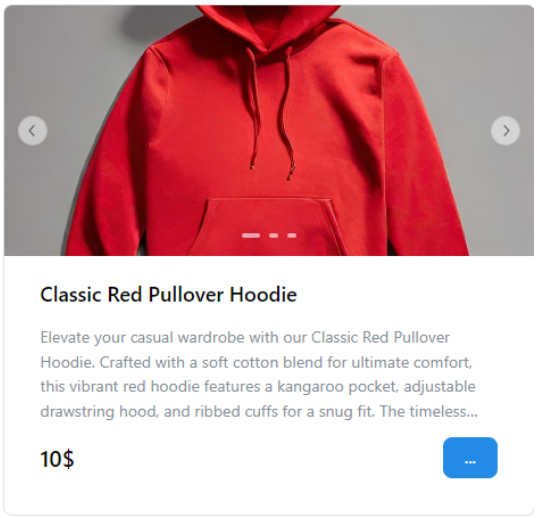

Figura 5: Ejemplo de un *ProductCard*.

Consideraciones Importantes

A continuación, se presentan algunas directrices **importantes** para el desarrollo de la tarea:

- **Restricciones de Librerías:** Está **estrictamente prohibido** el uso de cualquier librería externa o componentes preconstruidos (bootstrap, Tailwind CSS, Bulma o parecidos). El incumplimiento de esta regla resultará en una calificación automática de 1.0, sin posibilidad de re-corrección.
- **Pulls en GitHub:** No deben subir el directorio `node_modules`. Además, solo se revisará en la **rama main**. El incumplimiento de esta regla resultará en la reducción en un 50 % de su nota final, sin posibilidad de re-corrección.
- **Herramienta de petición HTTP:** Solamente se permitirá el uso de `axios` como librería externa para la realización de solicitudes HTTP a la API. Tampoco se permitirá el uso de `fetch`.
- **Manejo de URLs en JSON:** Para el endpoint POST de productos, el campo *images* debe ser una lista de URLs (tipo *string*). Al realizar un `GET` del producto creado, la respuesta incluirá este campo como una lista de URLs en formato *string*, con los corchetes (`[]`) incluidos. Es necesario manejar adecuadamente este formato, extrayendo las URLs de manera correcta. Para esto, deben completar la función `imagesFomatter`, la cual se encuentra en el archivo `imageUtils.js` para su posterior uso.

Ejemplo explicativo del manejo de URLs:

```
Ejemplo de Cuerpo de la Solicitud:
{
    "title": "Nueva Polera",
    "price": 10,
    "description": "Polera con imagenes de Disney",
    "categoryId": 1,

    # Formato esperado para el POST
    "images": [
        "https://placeimg.com/640/480/any"
    ]
}

Ejemplo de Respuesta:
{
    "title": "Nueva Polera",
    "price": 10,
    "description": "Polera con imagenes de Disney",

    # Formato que deben manejar
    "images": [
        "[\"https://placeimg.com/640/480/any\"]"
    ],
    ...
}
```

Interfaz gráfica

Para esta parte, tendrán que mostrar visualmente los resultados de las *requests* anteriormente descritas. Para ello, les proveeremos un código base en React que podrán encontrar en sus respectivos repositorios.

Documentación del proceso

Este ítem tiene como propósito:

- Explicar qué logra hacer su programa y también aquello que no lograron implementar.
- Indicar apropiadamente el proceso para levantar tu aplicación, incluyendo que comandos se deben correr.
- Dar a entender cómo funciona cada parte de su programa, es decir, cómo funciona cada *endpoint* implementado y qué consideraciones tuvieron que tener para que estos funcionaran correctamente.
- Indicar consideraciones particulares de su trabajo (en caso de haber).

Además, en sus respectivos repositorios encontrarán una serie de preguntas teóricas acerca de su tarea y del consumo de APIs en general. Deberán responder estas preguntas de forma más completa posible.

Para un mejor entendimiento sobre cómo escribir la documentación, se les hará entrega de una plantilla con la estructura que debe seguir, su descripción y las preguntas que deben responder.

Entregables

Cada entrega deberá incluir los siguientes archivos:

- `README.md` con la documentación de tu programa. Debes mencionar brevemente aquello que pudiste implementar y también lo que no. Debes indicar cómo levantar tu aplicación y responder las preguntas indicadas en el *template* que te facilitamos. Para finalizar, debes explicar claramente cómo funciona cada implementación de tu programa.
- Aplicacion en React desarrollada en base al material entregado. Se deben considerar todos los directorios necesarios para poder levantar y correr el programa.

Rúbrica

A continuación, se describe el criterio de cada uno de los ítems de la rúbrica con la que se evaluará esta entrega. La nota se calcula al 50 % considerando un total de **22** puntos.

- **Página de productos [3 puntos]**: Se espera que la información obtenida a través del uso de la API dada, muestre todos los productos disponibles de manera satisfactoria. Además, contar con un botón para publicar productos. Los productos deben estar paginados.
- **Página formulario [5 puntos]**: Se espera que se haga uso del *endpoint* para publicar nuevos productos y actualizar de forma efectiva la información de un producto existente. Para esto deben cumplir los siguientes requisitos:
  - Se muestre la(s) imagen(es) del producto, en caso de que se este editando.
  - No se puedan dejar campos vacíos al enviar el formulario. Se debe contar con titulo, descripción, precio, categoría e imágenes como mínimo.
  - Mostrar mensaje o pop-up de éxito o error al enviar el formulario. En caso de error, especificar el *error response*. En caso de éxito, redirigir a la página principal.
  - El formulario debe contar con una validación simple. (i.e, El precio debe ser un numero positivo, deben ser categorías existentes, etc)
  - Tener un botón para volver a la página principal.
- **Página producto [3 puntos]**: Se espera que esta pagina muestre la información de un producto seleccionado de la pagina de productos. Para cumplir con este punto, deben cumplir con los siguientes requisitos:
  - Tener carrusel de imágenes, en caso de que tenga varias imágenes.
  - Mostrar todas la información correctamente del producto.
  - Tener un botón para volver al menú principal.
- **Tarjetas de productos [5 puntos]**: Se espera que se utilicen los *endpoints* necesarios para conseguir la información de los productos y los muestren correctamente. La tarjeta debe mostrar lo siguiente:
  - Carrusel de imágenes, en caso de que tenga varias imágenes.
  - Nombre del producto
  - Descripción abreviada del producto
  - Precio del producto
  - Botón para editar / eliminar. Al eliminar o editar un producto, la página debe actualizarse con la información con los cambios más recientes.
- **Escritura de documentación [6 puntos]**: Se espera que se explique de manera clara, ordenada y suficiente cada parte del programa. Se debe notar un claro entendimiento acerca de cómo se consume una API y de cómo se realizó (y cómo se corre) el código entregado. Además, se esperan respuestas claras y completas explicando las preguntas teóricas asociadas a la tarea.

Bonus

Se entregarán 2 desafíos, los cuales deben estar completos para poder recibir la bonificación total.

- **Página Login [2 décimas]**: Deberán crear una página extra, la cual debe contener un formulario de autenticación. Para lograr el bonus, deberán:
  - Crear un botón en el header que dirija a la página.
  - Usar el *endpoint* de login.
  - Validar correctamente los datos del formulario (i.e Debe tener campos completos, formato correcto de email)
  - Mostrar las respuesta de error
  - En caso de un login autorizado, deberán mostrar la token entregada. Y luego redirigir automáticamente a la página de inicio.
- **Filtro categorías [1 décima]**: Deberán crear un filtro, el cual debe ser colocado en la página principal. Para lograr el bonus, deberán:
  - Contener todas las categorías disponibles
  - Obtener las categorías del *endpoint* correspondiente
  - Al seleccionar una de las categorías, se deben mostrar únicamente los artículos que pertenezcan a esa categoría.

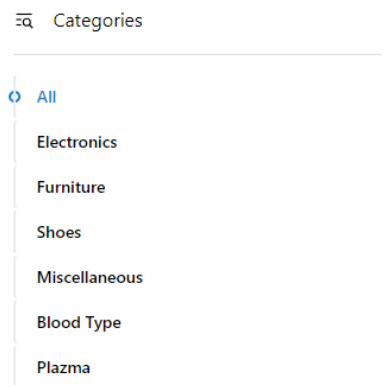

Figura 6: Ejemplo de un filtro por categoría.

Dudas

Cualquier duda que se presente acerca del enunciado debes consultarla en las ***issues*** del repositorio del curso. Recuerda que como equipo docente estaremos atentos para poder ayudarte :)

Notas

No se responderán dudas por correo.

La política de atraso podrán encontrarla en el programa del curso