«Talento Tech»

React JS

Clase 15

Buenos Aires Ciudad

Clase N° 15| Despliegue de la Aplicación

Índice:

- Despliegue de la aplicación en plataformas como Vercel o Netlify.
- Pruebas de funcionamiento en el entorno de producción.
- Revisión de los pasos de despliegue y configuración.

Objetivos de la Clase:

- Publicar la aplicación en una plataforma en la nube (Vercel o Netlify).
- Asegurar que la aplicación funcione correctamente en un entorno de producción.
- Revisar los pasos de configuración para un despliegue exitoso.

Despliegue de la Aplicación en Vercel o Netlify

Una vez que la aplicación está lista y optimizada, el siguiente paso es publicarla en un servidor para que cualquier usuario pueda acceder a ella.

Configuración del Proyecto para Despliegue

Antes de desplegar, es importante asegurarse de que el código está limpio y sin errores.

📌 **Pasos previos:**

✔ Revisar que el código funcione correctamente en el entorno local.

✔ Verificar que todas las dependencias están instaladas (`npm install`).

✔ Configurar correctamente las variables de entorno (si es necesario).

✔ Hacer un último commit en GitHub.

Desplegar en Vercel

Vercel es una de las plataformas más usadas para desplegar aplicaciones React con facilidad.

Pasos para el despliegue en Vercel:

1️⃣ **Instalar la CLI de Vercel** (si no la tienes):

```
npm install -g vercel
```

2️⃣ **Autenticarse en Vercel**:

```
vercel login
```

3️⃣ **Desplegar la aplicación**:

```
vercel
```

4️⃣ **Seguir las instrucciones en consola** para completar el proceso.

Una vez finalizado, Vercel proporcionará una URL donde la aplicación estará en línea.

Desplegar en Netlify

Netlify es otra excelente opción para desplegar aplicaciones estáticas como las de React.

Pasos para el despliegue en Netlify:

1. **Crear una cuenta en Netlify** y conectar con GitHub.
2. **Subir el proyecto a GitHub** (si no está subido aún).
3. **En Netlify, seleccionar "New Site from Git"** y conectar el repositorio.

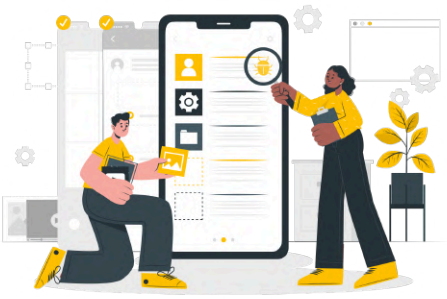
Configurar los parámetros de build:

- **Build Command:** `npm run build`
- **Public Directory:** `dist` (si usas Vite) o `build` (si usas Create React App).

5️⃣ **Hacer clic en "Deploy" y esperar a que finalice.**

Una vez terminado, Netlify generará una URL donde la aplicación estará disponible.

Pruebas de Funcionamiento en Producción



Una vez desplegada la aplicación, es fundamental probar su funcionamiento en el entorno de producción.

Verificaciones clave:

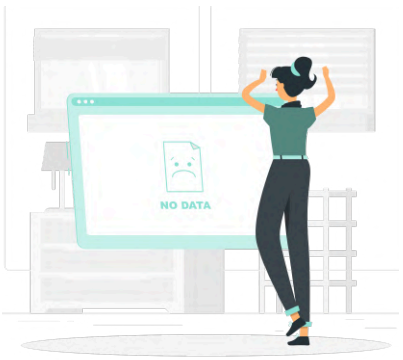
- ✔ Comprobar que todas las páginas y rutas cargan correctamente.
- ✔ Probar la responsividad en diferentes dispositivos y navegadores.
- ✔ Validar el funcionamiento de formularios, botones e interacciones.
- ✔ Verificar la carga de imágenes y estilos.
- ✔ Analizar el rendimiento y tiempos de carga con DevTools.

💡 **Herramientas útiles:**

- Google Lighthouse para evaluar el rendimiento y accesibilidad.
- PageSpeed Insights para analizar la velocidad de carga.
- Modo responsive en DevTools (F12 > Ctrl + Shift + M).

Revisión Final del Despliegue y Configuración

🔴Errores comunes y cómo solucionarlos:



La aplicación no carga después del despliegue:

- Revisar que el build (`npm run build`) se haya generado correctamente.
- Asegurar que la carpeta de salida (`dist` o `build`) es la correcta en la configuración del hosting.

Problemas con las rutas en React Router DOM:

- En Vercel y Netlify, asegurarse de usar `react-router-dom` en modo BrowserRouter con una configuración adecuada.
- Para Netlify, agregar un archivo `_redirects` en `public/` con el siguiente contenido:

```
/* /index.html 200
```

Esto redirige todas las rutas al `index.html`, evitando errores 404.

🔴 Errores con variables de entorno (.env):
✔ En Vercel o Netlify, configurar manualmente las variables en la sección de "Environment Variables".

Reflexión final

Aprendimos a desplegar nuestra aplicación de React en plataformas como Vercel y Netlify, asegurándonos de que funcione correctamente en un entorno de producción. Realizamos pruebas para verificar la responsividad, la accesibilidad y el rendimiento, asegurando una experiencia fluida para el usuario final.

El despliegue es un paso fundamental en el desarrollo, ya que permite que el proyecto pase de una fase local a estar accesible en Internet, listo para su uso real.

Materiales y Recursos Adicionales:

- ★ ✅ Documentación de Vercel
- ★ ✅ Documentación de Netlify
- ★ ✅ React Router DOM y configuración de rutas
- ★ ✅ Google Lighthouse para auditoría de sitios web

Preguntas para Reflexionar:

- ¿Qué ventajas tiene desplegar en Vercel frente a Netlify y viceversa?
- ¿Cómo podrías solucionar problemas de rutas en una aplicación con React Router DOM?
- ¿Por qué es importante realizar pruebas de rendimiento antes de publicar la aplicación?
- ¿Qué medidas tomarías para optimizar la velocidad de carga de la aplicación en producción?

Próximos Pasos:

- Despliegue de la aplicación final.
- Revisión de la integración de todas las funcionalidades desarrolladas (CRUD, autenticación, carrito, rutas protegidas).

Consignas - Entrega Final del Proyecto

¡Hemos llegado a la entrega final del proyecto! Es el momento de presentar la versión completa y optimizada del eCommerce que hemos desarrollado. Como en cualquier entorno profesional, esta entrega simula la presentación del producto final a nuestro cliente de Talento Lab. Se espera que el sitio cumpla con todos los requerimientos establecidos y brinde una experiencia de usuario fluida, accesible y visualmente atractiva.

A continuación, se detallan los puntos que deben estar implementados para la entrega final:

Requerimiento #1: Gestión del Carrito y Autenticación de Usuarios

Objetivo: Implementar un carrito de compras funcional y restringir el acceso a secciones privadas mediante autenticación de usuarios.

✅ Carrito de Compras con Context API

- Implementar un **CarritoContext** que gestione los productos agregados.
- Permitir **agregar, eliminar y vaciar el carrito**.
- Mantener el estado global con Context API.

✅ Autenticación de Usuarios

- Crear un **AuthContext** para manejar el estado de autenticación.
- Implementar un **login simulado con localStorage**.
- Restringir el acceso al carrito y otras secciones a usuarios autenticados con **rutas protegidas**.

Requerimiento #2: CRUD de Productos con MockAPI

Objetivo: Permitir la administración completa del catálogo de productos mediante operaciones de creación, lectura, actualización y eliminación.

✅ Formulario para Agregar Productos

- Implementar un **formulario controlado con useState**.
- Validar que los campos sean correctos:
    - **Nombre obligatorio**.
    - **Precio mayor a 0**.
    - **Descripción mínima de 10 caracteres**.
- Enviar los datos a **MockAPI mediante una solicitud POST**.

✅ Edición y Eliminación de Productos

- Permitir la edición de productos utilizando **MockAPI y Context API**.
- Mostrar **mensajes de error y confirmaciones** al usuario.
- Implementar un **modal de confirmación** antes de eliminar un producto.

✅ Manejo de Errores

- Mostrar **mensajes de error** en pantalla si hay problemas con la API.
- Manejar estados de **carga y error** al obtener los productos.

Requerimiento #3: Optimización de Diseño y Responsividad

Objetivo: Mejorar la apariencia y la accesibilidad del sitio utilizando herramientas modernas de diseño y estilización.

✅ Diseño Responsivo con Bootstrap y Styled-components

- Implementar el **sistema de grillas de Bootstrap** para adaptar el contenido a distintos dispositivos.
- Usar **styled-components** para personalizar los estilos y hacer el código más modular.

✅ Interactividad Mejorada con React Icons y React Toastify

- Agregar **iconos** en botones y elementos interactivos con **React Icons**.
- Implementar **React Toastify** para mostrar **notificaciones de éxito y error**.

✅ **SEO y Accesibilidad con React Helmet**

- Modificar el **<title> y <meta>** con React Helmet para mejorar el SEO.
- Asegurar que los elementos interactivos tengan etiquetas **ARIA** para accesibilidad.

Requerimiento #4: Preparación para el Despliegue

Objetivo: Ajustar los últimos detalles para que la aplicación esté lista para ser publicada en un servidor.

✅ **Pruebas de Compatibilidad**

- Verificar el funcionamiento en **móviles, tablets y escritorios**.
- Revisar **tiempos de carga y experiencia de usuario**.

✅ **Optimización del Código**

- Revisar el código y eliminar elementos innecesarios.
- Asegurar que el **estado global esté bien gestionado**.

✅ **Documentación Básica**

- Incluir instrucciones en el **README.md** sobre instalación y uso de la aplicación.

🛑 **IMPORTANTE**
Esta entrega representa el cierre del proyecto y debe reflejar un **producto funcional, bien estructurado y con una experiencia de usuario optimizada**. ¡Éxitos en la presentación final! 🚀

Buenos Aires aprende

Agencia de Habilidades para el Futuro

Buenos Aires Ciudad