



## ¿De qué trata?

NextHoops es un proyecto pensado para los aficionados de la NBA. En la web se podrán consultar las estadísticas tanto de equipos, como de jugadores (individual y colectivamente). También habrá un apartado de merchandising para la ropa, dedicado a todos aquellos aficionados que quieran comprar ropa de sus jugadores y equipos favoritos.



# Índice

<b>¿De qué trata?</b>	<b>1</b>
<b>Índice</b>	<b>2</b>
<b>Finalidad de la aplicación</b>	<b>3</b>
Casos de la vida real para los que se podrían usar	6
<i>Tienda (con inicio de sesión y carrito)</i>	7
Inicio de Sesión	7
Carrito de Compras	8
Experiencia de Usuario	8
<i>Premios de Temporada</i>	9
Contenido del Apartado	9
Actualización Anual	9
Experiencia de Usuario	9
<b>Objetivos principales de la aplicación</b>	<b>10</b>
Objetivos Generales a Cumplir	10
Objetivos Más Específicos	12
<b>Desarrollo de la aplicación</b>	<b>15</b>
<b>Cosas que se podrían mejorar:</b>	<b>25</b>
• A nivel general	25
• A nivel técnico	27
<b>Cuestiones técnicas</b>	<b>30</b>
<i>IA y Machine Learning:</i>	30
<i>Big Data:</i>	30
<i>DevOps:</i>	30
<i>Seguridad:</i>	31
<i>Usabilidad y Accesibilidad:</i>	31
<i>Deploy y Testing:</i>	32
<i>Git:</i>	32
<i>Lenguaje y versiones:</i>	32
<i>Uso de habilidades en el desarrollo de la aplicación:</i>	32
<b>Conclusiones</b>	<b>33</b>
<b>WebGrafía</b>	<b>34</b>



---

## Finalidad de la aplicación

La aplicación NextHoops tiene como finalidad principal ofrecer una plataforma integral y accesible para los aficionados de la NBA, brindando una experiencia enriquecida que combine información detallada, interacción social y comercio electrónico. Las finalidades específicas de NextHoops se pueden desglosar en las siguientes áreas:

### 1. Proveer Información Completa y Actualizada

NextHoops busca ser la fuente principal de estadísticas y datos sobre la NBA, proporcionando a los usuarios información actualizada y detallada sobre equipos y jugadores. Esta finalidad incluye:

- Acceso a Estadísticas Detalladas: Ofrecer un conjunto amplio de estadísticas básicas y avanzadas que permitan a los aficionados y analistas comprender mejor el rendimiento de los equipos y jugadores.
- Comparación y Análisis: Facilitar herramientas que permitan a los usuarios comparar estadísticas entre diferentes jugadores y equipos, ayudando en debates y análisis profundos.
- Actualizaciones en Tiempo Real: Mantener la base de datos actualizada con información en tiempo real durante la temporada, para que los usuarios siempre tengan acceso a los datos más recientes.



---

## 2. Fomentar la Comunidad y la Interacción

NextHoops pretende convertirse en un espacio donde los aficionados de la NBA puedan conectarse, compartir sus opiniones y participar en discusiones. Para ello, la aplicación tiene como futuros objetivos:

- Crear Foros y Espacios de Discusión: Establecer plataformas donde los usuarios puedan intercambiar ideas, discutir partidos y compartir su pasión por el baloncesto.
- Facilitar la Interacción Social: Permitir a los aficionados interactuar con otros usuarios, seguir a expertos, y participar en encuestas y debates.
- Generar Contenido Exclusivo: Proporcionar acceso a contenido multimedia, como videos destacados, entrevistas y análisis de expertos, enriqueciendo la experiencia de los usuarios y fomentando la participación activa.

## 3. Ofrecer Productos Oficiales y Personalizados

Una de las finalidades clave de NextHoops es proporcionar una tienda en línea que ofrezca productos oficiales de la NBA, permitiendo a los aficionados mostrar su apoyo a sus equipos y jugadores favoritos. Esto incluye:

- Venta de Merchandising Oficial: Ofrecer una amplia gama de productos oficiales, como camisetas, sudaderas, gorras y accesorios, de todos los equipos y jugadores de la NBA.



- Productos Personalizados: Permitir a los usuarios personalizar productos con nombres y números específicos, proporcionando una experiencia única y personalizada (en el futuro).
- Ediciones Especiales y Exclusivas: Comercializar productos limitados y exclusivos que celebren momentos icónicos y figuras legendarias de la NBA. Esto se haría a modo de drops exclusivos.

#### 4. Mejorar la Experiencia del Usuario

NextHoops está diseñada para ser intuitiva y fácil de usar, con el objetivo de ofrecer una experiencia agradable y fluida. Las finalidades en este ámbito son:

- Diseño Intuitivo y Atractivo: Crear una interfaz de usuario que sea visualmente atractiva y fácil de navegar, permitiendo a los usuarios encontrar la información y productos que buscan de manera eficiente.
- Soporte Multidispositivo: Asegurar que la aplicación sea accesible desde diferentes dispositivos, incluyendo computadoras, tabletas y teléfonos móviles, garantizando una experiencia consistente y de alta calidad en todas las plataformas.
- Retroalimentación y Mejora Continua: Recoger y analizar el feedback de los usuarios para implementar mejoras continuas en la funcionalidad y el contenido de la aplicación, adaptándose a las necesidades y preferencias de la comunidad de aficionados.



## Casos de la vida real para los que se podrían usar

Esta web la he diseñado ya que cuando quiero mirar algo referente a la nba en mi ordenador, siempre encuentro páginas NO oficiales, desactualizadas y/o que no se encuentra la información que busco. Con **Nexthoops**, los fans pueden acceder a nuestra web a ver las *novedades*, los *partidos*, la *tabla clasificatoria*, *tienda*, e.t.c.

También cuenta con un par de apartados que se irán **renovando** año a año.

Uno es el apartado de los "**Premios temporada**", en nuestro caso, "**2023-2024**", que se **actualizará** el año que viene a los de la "**temporada 2024-2025**".

Por otro lado también se usa para el día a día. Si quieres ver cómo ha quedado tu equipo favorito en su último partido, nexthoops es la solución.

También se podrán consultar **estadísticas**, tanto del equipo, como de los jugadores, para que de esta forma, podamos estar al tanto de toda la actualidad, cifras y probabilidades.

Es una web visualmente **simple** y **atractiva**. Que no resulta difícil de usar, ya que podrían usarla perfectamente desde los fans más pequeños hasta los más mayores.

De **fácil acceso**, ya que el nombre de la marca no resulta lioso ni largo, por lo que podría buscarse sin problemas en Google, entre otros.

Cuenta con **todo aquello** que un fan **busca** en la web de su deporte favorito.





## Tienda (con inicio de sesión y carrito)

La tienda de NextHoops es el lugar perfecto para que los aficionados de la NBA puedan adquirir sus productos oficiales favoritos. Desde camisetas de los equipos hasta accesorios exclusivos, la tienda ofrece una amplia gama de artículos que cualquier fan desearía tener. Para brindar una experiencia personalizada y segura, la tienda incluye un sistema de inicio de sesión y un carrito de compras funcional.

### Inicio de Sesión

El sistema de inicio de sesión de NextHoops garantiza que cada usuario pueda acceder a sus datos de manera segura y privada. Al iniciar sesión, los usuarios pueden:




- **Guardar artículos en su carrito:** Los usuarios registrados pueden añadir artículos a su carrito y guardarlos para futuras compras.
- **Realizar un seguimiento de sus pedidos:** Pueden ver el historial de sus compras y el estado de sus pedidos en tiempo real.

El proceso de inicio de sesión es sencillo y rápido. Los usuarios existentes, solo se requiere ingresar su correo electrónico y contraseña para acceder a su cuenta.

### Inicia Sesión

Login

**Síguenos en nuestras redes**









## Carrito de Compras

El carrito de compras es una de las funcionalidades más importantes de la tienda. Permite a los usuarios:

- **Añadir y eliminar productos:** Los usuarios pueden agregar productos a su carrito con un solo clic y eliminarlos si cambian de opinión.
- **Ver el resumen del pedido:** Incluye detalles como el nombre del producto, cantidad, precio unitario y total.
- **Aplicar códigos de descuento (próximamente):** Si los usuarios tienen algún código promocional, pueden aplicarlo antes de proceder al pago.
- **Proceder al pago de manera segura (próximamente):** El sistema de pago está integrado con plataformas de pago seguras para proteger la información financiera de los usuarios.

Carrito de Compra			
Productos		Resumen	
	Camiseta Jaylen Brown Talla: L	-   +	89.99 € 
	Camiseta Jayson Tatum Talla: XL	- 2 +	179.98 € 
		Subtotal	269.97 €
		Envío	4.99 €
		Total	<b>274.96 €</b>

## Experiencia de Usuario

La tienda de NextHoops está diseñada para ser intuitiva y fácil de navegar. Los usuarios pueden buscar productos por categoría, equipo o jugador favorito. La interfaz es visualmente atractiva y optimizada para una experiencia fluida, tanto en dispositivos móviles como en ordenadores.





---

## **Premios de Temporada**

El apartado de premios de temporada de NextHoops es una sección dedicada a destacar los logros y reconocimientos de los jugadores y equipos de la NBA. Este apartado se actualiza anualmente para reflejar los resultados de la temporada en curso, permitiendo a los aficionados mantenerse al día con las últimas novedades y logros.

### **Contenido del Apartado**

En la sección de premios de temporada, los usuarios pueden encontrar:

- **Premios Individuales:** Información sobre los ganadores de premios como el Jugador Más Valioso (MVP), Novato del Año, Jugador Defensivo del Año, y más.
- **Premios de Equipo:** Reconocimientos a equipos, como el campeón de la NBA, el campeón de conferencia, y otros logros importantes. Estos serán próximamente, ya que los play-offs están actualmente en curso.
- **Estadísticas y Logros:** Datos sobre el rendimiento de los jugadores premiados y sus estadísticas destacadas durante la temporada.

### **Actualización Anual**

Cada año, al finalizar la temporada de la NBA, el apartado de premios se actualiza para reflejar los nuevos ganadores. Esto asegura que los usuarios siempre tengan acceso a la información más reciente y relevante. Por ejemplo, en la temporada 2023-2024, los usuarios pueden ver los premios y logros de dicha temporada, y al comenzar la temporada 2024-2025, el apartado se actualizará con los nuevos datos.

### **Experiencia de Usuario**

El diseño del apartado de premios de temporada es limpio y organizado. Cada premio tiene su propia sección, con fotos y descripciones detalladas de los ganadores. Además, los usuarios pueden acceder a videos y entrevistas exclusivas con los galardonados, ofreciendo una experiencia enriquecedora y completa.



---

## Objetivos principales de la aplicación

### Objetivos Generales a Cumplir

- **Centralizar la Información de la NBA:**
  - **Descripción:** NextHoops tiene como meta ser la fuente principal (si se pudiera) de información sobre la NBA, proporcionando un punto único de acceso a todas las noticias, resultados de partidos, estadísticas y novedades importantes.
  - **Acciones Específicas:**
    - Crear una red de colaboradores y fuentes confiables que aporten información actualizada y verificada.
    - Desarrollar secciones específicas para noticias, resultados en tiempo real, estadísticas de equipos y jugadores, y una base de datos histórica de la NBA.
    - Implementar alertas y notificaciones para mantener a los usuarios informados sobre eventos importantes, como el inicio de partidos, resultados finales y noticias de última hora.
  
- **Ofrecer una Experiencia de Usuario Excepcional:**
  - **Descripción:** Garantizar que la aplicación sea intuitiva, fácil de usar y visualmente atractiva para una amplia gama de usuarios, desde jóvenes hasta mayores.
  - **Acciones Específicas:**
    - Diseñar una interfaz de usuario (UI) que sea limpia, moderna y fácil de navegar.
    - Realizar pruebas de usabilidad con diferentes grupos de usuarios para identificar y corregir posibles problemas de navegación.



- **Proveer Productos Oficiales y Exclusivos:**

- **Descripción:** A través de la tienda integrada, ofrecer productos oficiales de la NBA, incluyendo camisetas, accesorios y artículos exclusivos que no se encuentran en otros lugares.
- **Acciones Específicas:**
  - Establecer acuerdos de distribución con proveedores oficiales de la NBA.
  - Actualizar regularmente el inventario de la tienda con nuevos productos y ediciones limitadas.
  - Crear una sección de productos exclusivos que solo estén disponibles a través de NextHoops.

- **Actualizar Contenido de Manera Continua y Relevante:**

- **Descripción:** Mantener a los usuarios informados con contenido actualizado y relevante, incluyendo noticias diarias, resultados de partidos y estadísticas.
- **Acciones Específicas:**
  - Contratar un equipo editorial dedicado a la actualización diaria del contenido.
  - Implementar un sistema automatizado para la actualización de resultados y estadísticas en tiempo real.
  - Desarrollar una estrategia de contenido que incluya artículos, vídeos y análisis profundos sobre los eventos actuales de la NBA.



---

## Objetivos Más Específicos

- **Mejorar la Precisión y Rapidez de la Información:**
  - **Descripción:** Asegurar que la información proporcionada sea precisa y se actualice rápidamente, especialmente durante los partidos en vivo, para que de esta forma, se pueda estar al tanto jugada a jugada.
  - **Acciones Específicas:**
    - Integrar una API de datos deportivos para obtener resultados y estadísticas en tiempo real.
    - Establecer algunos procedimientos de verificación para todas las noticias y actualizaciones antes de publicarlas.
    - Ofrecer un sistema de alertas en tiempo real para mantener a los usuarios informados de los eventos importantes.
  
- **Optimizar el Sistema de Inicio de Sesión y Gestión de Usuarios:**
  - **Descripción:** Simplificar y asegurar el proceso de registro e inicio de sesión, permitiendo a los usuarios personalizar su experiencia en la aplicación.
  - **Acciones Específicas:**
    - Implementar autenticación segura utilizando métodos modernos como OAuth y autenticación de dos factores (2FA).
    - Facilitar el registro a través de redes sociales y cuentas de correo electrónico, etc.
    - Desarrollar un sistema de perfil de usuario donde puedan guardar preferencias, historial de compras y equipos favoritos, para que de esta forma se puedan recomendar noticias personalizadas.
  
- **Desarrollar una Funcionalidad de Carrito de Compras Eficiente:**
  - **Descripción:** Crear un carrito de compras fácil de usar que facilite el proceso de compra y gestione eficientemente los pedidos de los usuarios.
  - **Acciones Específicas:**



- 
- Diseñar una interfaz intuitiva para agregar y eliminar productos del carrito.
  - Integrar múltiples métodos de pago seguros y confiables, incluyendo tarjetas de crédito, PayPal, GPay y opciones locales.
  - Implementar funciones de seguimiento de pedidos y notificaciones sobre el estado del envío. De esta forma, el cliente podrá estar al día del estado de su pedido.
- 
- **Actualizar y Ampliar el Apartado de Premios de Temporada:**
    - **Descripción:** Mantener la sección de premios actualizada y detallada, proporcionando una experiencia enriquecida con contenido multimedia y estadísticas.
    - **Acciones Específicas:**
      - Actualizar anualmente la información de los premios con los ganadores y estadísticas relevantes.
      - Incluir fotos, videos y entrevistas exclusivas con los galardonados.
      - Crear una base de datos histórica accesible donde los usuarios puedan ver los ganadores de temporadas anteriores.
- 
- **Fomentar la Comunidad y el Compromiso de los Usuarios:**
    - **Descripción:** Desarrollar características que promuevan la interacción y el compromiso entre los usuarios, creando una comunidad activa de aficionados a la NBA.
    - **Acciones Específicas:**
      - Integrar foros de discusión y secciones de comentarios en artículos y noticias.
      - Facilitar la compartición de contenido en redes sociales y otras plataformas.
      - Organizar eventos especiales, concursos y promociones para mantener a los usuarios comprometidos.



- 
- **Analizar y Adaptar el Comportamiento de los Usuarios:**
    - **Descripción:** Utilizar herramientas de análisis para comprender mejor el comportamiento de los usuarios y adaptar el contenido y las ofertas en consecuencia.
    - **Acciones Específicas:**
      - Implementar herramientas de analítica como Google Analytics para rastrear el comportamiento del usuario.
      - Analizar los datos de uso para identificar tendencias y áreas de mejora.
      - Adaptar el contenido y las recomendaciones de productos en función de las preferencias y el comportamiento de los usuarios.



## Desarrollo de la aplicación

Para poder desarrollar nuestra aplicación, debemos tener en cuenta el **qué vamos a usar** y el **por qué lo vamos a usar**.

En mi caso, estoy usando **Angular 17.2**, ya que esta versión supone algunos cambios que simplifican los procesos. Se eliminan los **modules** por lo que importaríamos el “módulo” que queramos directamente en nuestro componente. A partir de Angular 16 se introduce el concepto “**standalone**”. Esto indica que al ponerlo como *true*, haremos que el componente no dependa necesariamente de un módulo y que pueda funcionar por sí solo.

También hay algunos **cambios** a nivel **de lógica**, ya que cambia la forma en la que se hacen los bucles **if** y **for**, por ejemplo.

```
1 <div class="example">
2   <h1 *ngIf="touristFrom === 'england'">Hello Angular 17!</h1>
3   <h1 *ngIf="touristFrom === 'france'">Bonjour Angular 17!</h1>
4 </div>
```

ngIf en Angular v14

```
1 <div class="example">
2   @if (touristFrom === 'england') {
3     <h1>Hello Angular 17!</h1>
4   } @else {
5     <h1>Bonjour Angular 17!</h1>
6   }
7 </div>
8
```

if en Angular v17

Por otro lado, también hago uso de **librerías** como **Angular Material** y **Angular Icons**.



Angular Material lo uso para usar **elementos compatibles y optimizados** para Angular. De esta forma, me ahorro muchos errores a la hora de escribir código, ya que todos los elementos vienen adaptados a Angular.

```
"@angular/material": "^17.3.8",
```

Dependencia de Angular Material

Por otro lado, una librería que, personalmente, me parece increíblemente útil, **Angular Icons**.

Esta librería no es oficial de Angular, si no que alguien ha realizado un proyecto creándola, y la ha publicado usando un servicio de hosting gratuito. A continuación dejo el enlace a dicha web pinchando [AQUÍ](#).

En dicha librería, como su nombre indica, nos sirve para iconizar la web y que de esta forma, sea más atractiva visualmente. Mediante dos sencillas líneas de código (explicadas en su web), podremos importar esta librería en el componente que queramos para usarla sin problemas.

```
"@ng-icons/bootstrap-icons": "^27.4.0",  
"@ng-icons/core": "^27.4.0",  
"@ng-icons/feather-icons": "^27.4.0",  
"@ng-icons/heroicons": "^27.4.0",
```

Dependencia de ng-icons con algunos ejemplos

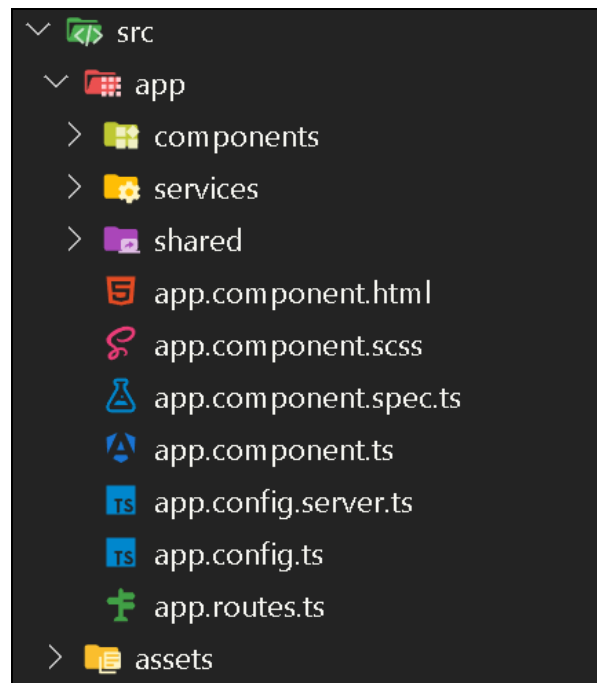




Por otro lado, una cosa muy importante es entender la estructura de carpetas de Angular.

En mi caso, lo he dividido en diferentes “secciones”.

- La carpeta **assets**: para recursos (img, json...)
- La carpeta **shared**: para componentes y archivos compartidos dentro de la aplicación
- La carpeta **services**: para los servicios de la app (consumos)
- La carpeta **components**: donde irán los componentes independientes.



Estructura de carpetas del proyecto

Dentro de mi aplicación tengo **consumos** a **JSON local**, **API** (*mockapi*) y **base de datos**.

A continuación muestro los códigos en ese orden.



```
import { HttpClient } from '@angular/common/http';

import { Injectable } from '@angular/core';

@Injectable({
  providedIn: 'root',
})
export class PlayersService {

  players: any = {};

  loaded = false;

  constructor(private http: HttpClient) {

    console.log('PlayersService initialized');

    this.loadData();

  }

  loadData() {

    this.http.get('../assets/data/players.json').subscribe((data) => {

      this.players = data;

      this.loaded = true;

      console.log('Data de PLAYERS');

      console.log(this.players);

    });

  }

}
```

Consumo a **JSON local**



```
import { Injectable } from '@angular/core';

import { HttpClient } from '@angular/common/http';

@Injectable({
  providedIn: 'root',
})
export class WestConferenceService {

  constructor(private http: HttpClient) {}

  getWestConference() {

    return
    this.http.get('https://apimocha.com/west-conference/west');

  }
}
```

Consumo a **API** (*mockup / mockapi*)



```
import { Injectable } from '@angular/core';

import { HttpClient } from '@angular/common/http';

import { Observable } from 'rxjs';

@Injectable({
  providedIn: 'root',
})

export class ConsumoTiendaService {

  private baseUrl = 'http://localhost:8095/tienda'; // Ajusta la URL de tu
backend

  constructor(private http: HttpClient) {}

  getTienda(): Observable<any[]> {

    return this.http.get<any[]>(`${this.baseUrl}/todos`);

  }

  getShirtById(id: string): Observable<any> {

    return this.http.get<any>(`${this.baseUrl}/${id}`);

  }

}
```

Consumo a mi **base de datos**



Del código de mi aplicación quiero destacar varias cosas. La primera es la gestión de la sesión mediante session storage y auth.

Para esta autenticación y autorización, hay que hacer uso de un **guard**. En Angular se usan los guards para la gestión de accesos.

Mi ruta se establecerá como “bajo autorización” con la siguiente línea:

```
{
  path: 'tienda/:id',
  loadComponent: () =>
    import('./components/ver-mas/ver-mas.component').then(
      (m) => m.VerMasComponent
    ),
  canActivate: [authGuard],
},
```

De esta forma, establecemos la ruta como “*protegida*”.

Mi **auth guard** lo que hace es **verificar si el usuario está logueado**:

```
export const authGuard: CanActivateFn = (route, state) => {
  const router = inject(Router);
  const authService = inject(AuthService);

  if (authService.isLoggedIn()) {
    return true;
  } else {
    router.navigate(['/login']);
    return false;
  }
};
```

Como podemos leer en el código, llama a mi servicio invocando la función `isLoggedIn()`.

En mi AuthService, **verificará si existe el token** de inicio de sesión. **Si no existe, no podré acceder** a la ruta solicitada, pero **si existe, podré acceder** sin ningún problema.



Aquí se puede ver mi AuthService:

```
@Injectable({
  providedIn: 'root',
})
export class AuthService {
  private get token(): string | null {
    return localStorage.getItem('authToken');
  }

  private set token(value: string | null) {
    if (value === null) {
      localStorage.removeItem('authToken');
    } else {
      localStorage.setItem('authToken', value);
    }
  }

  isLoggedIn(): boolean {
    return !!this.token;
  }

  login(username: string, password: string): boolean {
    if (username === 'user' && password === '123') {
      this.token = 'sesion-iniciada';
      return true;
    }
    return false;
  }

  logout(): void {
    this.token = null;
  }
}
```

Un código muy interesante gracias a la lógica que contiene.



Por último, **si no se detecta el token** de inicio de sesión, me **redirige al login**, el cual, al poner bien ambos campos y le demos a iniciar sesión, hará uso de la **creación del token** usando la **función** de auth service `login(username: string, password: string)`.

```
export class LoginComponent {
  loginForm: FormGroup;
  errorMessage: string = '';

  constructor(
    private fb: FormBuilder,
    private authService: AuthService,
    private router: Router
  ) {
    this.loginForm = this.fb.group({
      username: ['', Validators.required],
      password: ['', Validators.required],
    });
  }

  onSubmit(): void {
    if (this.loginForm.valid) {
      const { username, password } = this.loginForm.value;
      const success = this.authService.login(username, password);
      if (success) {
        this.router.navigate(['/tienda']);
      } else {
        this.errorMessage = 'Error en el login. Revisa tus credenciales.';
      }
    }
  }
}
```

Si por lo que sea se ponen **mal** las **credenciales** que tengo por defecto, salta una **validación** diciendo que **están mal**, que se revisen. Eso sería la validación.



Por otro lado, quiero destacar el **back** de mi aplicación, ya que está realizado en **Java Spring Boot**. Lo que hago es un consumo a mi base de datos, **MySQL**. Hago uso de lo básico, un **select**, pero se podrían implementar funciones de administrador, crear un panel de administrador en el front, y hacer un CRUD completo.

En mi caso voy a mostrar mi select:

```
@RestController  @ eloy6lega +1
@RequestMapping("/tienda")
@CrossOrigin(origins = "*")
public class ShirtsController {
    @Autowired
    ShirtsRepository shirts;

    @CrossOrigin(origins = "*")  @ eloy6lega
    @GetMapping("/todos")
    public List<Shirt> todos() { return shirts.findAllShirts(); }
}
```

Foto del **controlador**

Con eso creo mi **endpoint** para **consumirlo** desde mi **servicio** en el front.

```
server.port=8095
spring.datasource.url=jdbc:mysql://localhost:3306/nexthoops
spring.datasource.username=root
spring.datasource.password=root
spring.datasource.driver=com.mysql.cj.jdbc.Driver
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL8Dialect
spring.jpa.show-sql=true
```

Foto del **application.properties**

En esta foto lo que vemos es la **conexión a mi base de datos**, gracias a esa configuración y algún archivo más (clase de java, repositorio y controlador) puedo **consumir** mi **base de datos** en **MySQL**.





---

## Cosas que se podrían mejorar:

Como ya sabemos, siempre se puede mejorar todo. Es por ello que mi aplicación no es menos. Hay algunas cosas que se podrían mejorar dentro de mi aplicación.

- **A nivel general**

A un nivel general, se podrían mejorar distintas cosas.

La primera es **mejorar** el **sistema** de **autenticación**. En vez de verificar que mi usuario sea el mismo que "user" y el pass "123", lo suyo sería **leer** mi **tabla** de **usuarios** en mi **base de datos**, y si mis credenciales coinciden, entonces me da **acceso**.

Para esto habría que hacer **algunos** cambios y añadir alguna cosa en el **back** y modificar alguna cosa del **front** (*guard, service, .ts*). Al final sería una **custom query param** para verificar que coincidan los usuarios.

Por otro lado, la aplicación también mejoraría teniendo en cuenta el cambio anterior y **además añadiendo** la **posibilidad** de **registrarse** si es el caso de que el usuario **NO tiene una cuenta**. Para ello sería hacer **validaciones** para asegurarse que todos los campos se rellenen y se vayan a **insertar** en la base de datos **como** nosotros **queremos**. Una vez las validaciones se cumplan, el usuario haría clic en "**registrarse**" y sus credenciales se **insertarán** directamente en la base de datos. Después el usuario podría **iniciar sesión**, ya que tiene una **cuenta creada**.

Otra mejora sería hacer una "*pasarela*" en el proceso de registro. Es decir, primero tener que **introducir un correo electrónico** (con el que se vaya a registrar el usuario) y que al dar al botón de continuar, se **verifique que el correo no esté registrado** en nuestra base de datos. De esta forma **evitamos** que haya **varias cuentas con el mismo correo**.



---

Por último, se podría mejorar la aplicación **añadiendo una pasarela de pago** y que la **tienda** sea **funcional**.

Hay varias formas de hacerlo, **PayPal**, **Google Pay**, entre las más conocidas. Lo único malo de esto, es que si no eres una empresa, es complicado hacerlo, ya que necesitas en todos los casos ser **membership** para poder hacerlo. Por esto, he optado a no hacer 100% funcional mi tienda.

En el caso de que se haga funcional, se podría añadir también la **información del seguimiento del pedido** para que de esta forma, cuando el cliente finalice la compra, se le envíe un correo con la información de su pedido (*Tiempo estimado, N° pedido, Productos comprados, Precio total...*).



---

- **A nivel técnico**

Poner cosas como cdn/webp las img, optimizar códigos, reducir scripts...

A nivel técnico se podrían hacer también mejoras.

En primer lugar, las **imágenes** se podrían poner como **CDN** o como **WEBP**. Cada una tiene sus propias ventajas.

Usar **CDN** me permite tener una **mayor velocidad de carga**, tiene una **menor carga en el servidor** y además es **escalable**.

Usar **WEBP** me permite un **menor tamaño de archivo**, **mejor calidad** y un **soporte amplio**.

Se debería **estudiar** bien a fondo la web y sus **propósitos** para ver qué opción sería mejor.

Por otro lado, entramos en la **optimización de código scss**, por ejemplo.

Sería bueno que todos los **estilos comunes** se encuentren en el archivo **styles.scss** junto con las **clases que vaya a compartir la aplicación** (por ejemplo `class="icons"`). Esto nos permite **trabajar más rápido y eficiente**, a la vez que **reducimos el tamaño** que ocupan los **archivos**.

De igual manera, deberíamos hacer **lo mismo con los estilos.scss** de cada componente, **optimizando las clases que se vayan a compartir dentro del componente** para **no** tener que **sobreescribir** los mismos estilos, haciendo que **pese más la aplicación**.

Al igual que con los estilos.scss, tendríamos que hacer **lo mismo con los archivos.ts**. Estos archivos también han de ser optimizados para reducir el tamaño de mi aplicación web.

Una buena forma de optimizarlo es **importar los módulos** necesarios de cada librería. Poniendo un ejemplo se entiende mejor.



---

Es mejor y más correcto importar *“FormsModule”* que importar *“MatInput, MatLabel, MatSelect...”*.

Se debería hacer **lo mismo con las Pipes**. Lo correcto es **crear un módulo personalizado** donde se **metan las pipes** de mi aplicación, y una vez todas dentro del módulo, **se importaría dicho módulo**. Por ejemplo:

*“import { MisPipesModule } from ‘../ruta’”*.

También se debería realizar un **mantenimiento** de la aplicación para tenerla **actualizada**. De esta manera se añadirán **nuevas funcionalidades, mejorarán las existentes** y se **corregirán errores**.

Se deberían hacer también **mejoras en la seguridad**. Ya se han mencionado anteriormente en aspectos generales, pero se pueden hacer **mejoras** en la seguridad **como** por ejemplo **LogIn / Register** con **Google, LinkedIn, GitHub...** Esto hace que además de dar acceso, se tenga **verificación**.

Por último, se deberían implementar mejoras de caché.

**Existen diferentes tipos de técnicas de caché:**

- **Caché en memoria:** La caché en memoria es la más rápida, pero también la más volátil, ya que los datos se borran cuando se apaga la computadora o cuando la aplicación se cierra.
- **Caché en disco:** La caché en disco es menos rápida que la caché en memoria, pero es más persistente, lo que significa que los datos se conservan incluso después de que se apaga la computadora o se cierra la aplicación.
- **Caché distribuida:** La caché distribuida se utiliza para almacenar datos en varios servidores, lo que puede mejorar aún más el rendimiento y la escalabilidad de las aplicaciones con mucho tráfico.



---

**La elección de la técnica de caché adecuada depende de varios factores, como:**

- **El tipo de datos que se va a almacenar en caché:** Algunos tipos de datos, como los objetos grandes o los datos que cambian con frecuencia, no son adecuados para la caché en memoria.
- **La cantidad de tráfico que recibe la aplicación:** Las aplicaciones con mucho tráfico pueden necesitar una caché distribuida para poder manejar la carga.
- **Los recursos disponibles:** La caché en memoria requiere más memoria que la caché en disco, por lo que es importante tener en cuenta los recursos disponibles al elegir una técnica de caché.

**Existen varias herramientas y bibliotecas disponibles para ayudar a implementar técnicas de caché.** Algunas opciones populares incluyen:

- **Memcached:** Memcached es una caché en memoria de alto rendimiento que se utiliza ampliamente en aplicaciones web y móviles.
- **Redis:** Redis es una base de datos en memoria de código abierto que se puede utilizar como caché en memoria o como almacenamiento de datos persistente.
- **Varnish:** Varnish es un servidor web de caché que se puede utilizar para mejorar el rendimiento de los sitios web estáticos y dinámicos.



---

## Cuestiones técnicas

### IA y Machine Learning:

- **Predicción de resultados:** Implementar un modelo de Machine Learning para predecir los resultados de los partidos en base a estadísticas históricas, tendencias de los equipos y datos de los jugadores. Esto mejoraría la experiencia del usuario al ofrecer información valiosa y personalizada sobre los partidos.
- **Recomendaciones personalizadas:** Utilizar un sistema de recomendación basado en IA para sugerir a los usuarios contenido relevante, como noticias, equipos favoritos, jugadores o productos de la tienda, en función de sus preferencias e historial de navegación. Esto aumentaría el engagement y la satisfacción del usuario.
- **Análisis de datos de usuarios:** Implementar herramientas de análisis de datos para comprender mejor el comportamiento de los usuarios, identificar patrones y tendencias, y tomar decisiones informadas sobre la mejora de la aplicación.

### Big Data:

- **Almacenamiento de datos masivos:** Implementar una base de datos distribuida o un data lake para almacenar y procesar grandes conjuntos de datos, como estadísticas de partidos, datos de jugadores, datos de usuarios y datos de la tienda. Esto permitiría un análisis más profundo y la toma de decisiones más informadas.
- **Análisis en tiempo real:** Implementar herramientas de análisis de streaming para procesar datos en tiempo real, como estadísticas de partidos en vivo, comentarios de usuarios y actividad en la tienda. Esto permitiría una experiencia más dinámica y personalizada para los usuarios.



---

## DevOps:

- **Automatización de pruebas:** Implementar herramientas de integración continua y entrega continua (CI/CD) para automatizar las pruebas, la implementación y la entrega de nuevas versiones de la aplicación. Esto reduciría el tiempo de desarrollo y lanzamiento, y aumentaría la calidad del software.
- **Monitorización del rendimiento:** Implementar herramientas de monitorización para supervisar el rendimiento de la aplicación, identificar cuellos de botella y solucionar problemas de forma rápida y eficiente. Esto garantizaría una experiencia fluida y sin problemas para los usuarios.
- **Infraestructura como código:** Implementar la infraestructura como código (IaC) para definir y gestionar la infraestructura de la aplicación de forma automatizada. Esto facilitaría la escalabilidad y la gestión de la aplicación.

## Seguridad:

- **Autenticación:** Implementar un sistema de autenticación robusto para verificar la identidad de los usuarios y proteger sus datos. Esto podría incluir autenticación de dos factores, biometría o autenticación basada en tokens.
- **Autorización:** Implementar un sistema de autorización para controlar el acceso a los recursos de la aplicación en función de los roles y permisos de los usuarios. Esto garantizaría que solo los usuarios autorizados puedan acceder a la información y funcionalidades relevantes.
- **Cifrado de datos:** Implementar cifrado de datos para proteger los datos confidenciales de los usuarios, como contraseñas, información de pago y datos personales. Esto evitaría el acceso no autorizado y las filtraciones de datos.



---

## Usabilidad y Accesibilidad:

- **Diseño intuitivo:** Diseñar una interfaz de usuario intuitiva y fácil de usar, con una navegación clara y un diseño atractivo. Esto facilitaría la navegación y el uso de la aplicación para todos los usuarios.
- **Accesibilidad:** Implementar características de accesibilidad para que la aplicación sea usable por personas con discapacidades, como lectores de pantalla, subtítulos y ajustes de contraste. Esto ampliaría el alcance de la aplicación y la haría más inclusiva.

## Deploy y Testing:

- **Despliegue en la nube:** Implementar un proceso de despliegue en la nube para facilitar la escalabilidad y la disponibilidad de la aplicación. Esto permitiría manejar un mayor número de usuarios y tráfico sin problemas.
- **Pruebas exhaustivas:** Realizar pruebas exhaustivas de la aplicación para identificar y corregir errores antes de su lanzamiento. Esto garantizaría una experiencia libre de errores y satisfactoria para los usuarios.

## Git:

- **Control de versiones:** Utilizar Git para gestionar el control de versiones del código de la aplicación. Esto facilitaría la colaboración entre desarrolladores, el seguimiento de cambios y la reversión a versiones anteriores si es necesario.
- **Historial de cambios:** Mantener un historial de cambios detallado en Git para facilitar la comprensión de los cambios realizados en el código y la resolución de problemas.





---

## Lenguaje y versiones:

- **Lenguaje de programación:** Elegir un lenguaje de programación moderno y escalable para el desarrollo de la aplicación, como Python, JavaScript o Java. Esto garantizaría la capacidad de mantenimiento y la escalabilidad a largo plazo de la aplicación.
- **Gestión de versiones:** Implementar una estrategia de gestión de versiones para mantener diferentes versiones de la aplicación en paralelo y facilitar la transición entre versiones.

## Uso de habilidades en el desarrollo de la aplicación:

- **Desarrollo web:** Utilizar habilidades de desarrollo web front-end y back-end para crear una aplicación, gracias al aprendizaje del curso DAW y a las prácticas realizadas en la empresa.



---

## **Conclusiones**

Este proyecto ha supuesto un **gran reto**, no sólo a **nivel técnico**, sino también a **nivel mental**. Empezar un proyecto como **NextHoops** implica **identificar problemas reales** y **desarrollar soluciones** efectivas, lo cual requiere una combinación de creatividad, análisis y perseverancia. En mi caso, elegí centrarme en la **NBA** debido a mi frustración con las fuentes de estadísticas y noticias disponibles, que a menudo eran páginas de terceros no oficiales y desactualizadas. Este fue el principal impulso para crear una **plataforma integral y confiable** que cubriera todas las necesidades de los aficionados a la NBA.

A nivel técnico, el desarrollo de **NextHoops** ha sido una experiencia enriquecedora y desafiante. Aprender **Angular**, especialmente en su versión **17.2**, ha sido una parte crucial del proceso. Esta **versión relativamente nueva** trajo consigo una serie de **retos**, desde comprender sus nuevas características hasta solucionar errores inesperados. Sin embargo, estos **desafíos han sido superados** con éxito, lo que ha **resultado en un aprendizaje** significativo y una **mejora en mis habilidades de desarrollo web**.

Además de los aspectos técnicos, el proyecto ha exigido una considerable **planificación** y reflexión estratégica. Esto incluye **definir** claramente los **objetivos** tanto generales como específicos, **diseñar** una **interfaz** de usuario **atractiva y funcional**, y asegurar que la plataforma sea **fácil de usar** para una amplia audiencia. A través de este proceso, he aprendido la importancia de la **experiencia del usuario** y cómo las decisiones de diseño pueden influir en la satisfacción y el compromiso de los usuarios.

Otro aspecto clave del proyecto ha sido la **integración** de **funciones avanzadas** como el **inicio de sesión**, el **carrito** de compras, y la **actualización continua de contenidos**. Estas características no solo **añaden valor** a la plataforma, sino que también **garantizan** que los usuarios tengan una **experiencia rica y personalizada**. Implementar estas funcionalidades ha requerido una cuidadosa consideración de la seguridad, la eficiencia y la usabilidad.

Por último, **NextHoops** no sólo es una solución a un problema personal, sino que también busca mejorar la experiencia de muchos otros aficionados a la NBA. Al centralizar la



---

información y ofrecer una plataforma confiable y actualizada, **NextHoops** contribuye a una comunidad más informada y conectada. Este aspecto comunitario es fundamental, ya que fomenta la interacción y el compromiso entre los usuarios, creando un espacio donde los aficionados pueden compartir su pasión por el baloncesto.

En resumen, el desarrollo de **NextHoops** ha sido un viaje lleno de aprendizajes y satisfacciones. Ha requerido superar numerosos obstáculos técnicos y estratégicos, pero el resultado es una plataforma robusta y útil que cumple con las necesidades de su audiencia.

Gracias por la atención del lector de este documento, y desde aquí te animo a que emprendas tu propio proyecto. Un gran saludo, Eloy Pérez Gómez.



---

## WebGrafía

1. Sporting News. (s.f.). **NBA**. Recuperado de <https://www.sportingnews.com/es/nba?gr=www>
2. NBA Store Europe. (s.f.). **NBA Store Europe**. Recuperado de <https://www.nbastore.eu/en/?CMP=FUS-QYHJ9CFB&portal=QYHJ9CFB>
3. Bootstrap Icons. (s.f.). **Icons**. Recuperado de <https://icons.getbootstrap.com/?q=minus>
4. NG Icons. (s.f.). **NG Icons**. Recuperado de <https://ng-icons.github.io/ng-icons/#/>
5. Flaticon. (s.f.). **Flaticon**. Recuperado de <https://www.flaticon.es/>
6. Clerk. (s.f.). **Clerk**. Recuperado de <https://clerk.com/>
7. Stack Overflow. (s.f.). **Stack Overflow**. Recuperado de <https://stackoverflow.com/>
8. Modern CSS. (s.f.). **Custom Select Styles with Pure CSS**. Recuperado de <https://moderncss.dev/custom-select-styles-with-pure-css/>
9. I Love IMG. (s.f.). **I Love IMG**. Recuperado de <https://www.iloveimg.com/es>
10. W3Schools. (s.f.). **W3Schools**. Recuperado de <https://www.w3schools.com/>
11. Tailwind CSS. (s.f.). **Tailwind CSS**. Recuperado de <https://tailwindcss.com/>
12. Material Angular. (s.f.). **Material Angular**. Recuperado de <https://material.angular.io/>
13. Vercel. (s.f.). **Vercel**. Recuperado de <https://vercel.com/>

### *Inteligencias Artificiales:*

14. **ChatGPT** (OpenAI) - <https://chatgpt.com/>
15. **Gemini** (Google) - <https://gemini.google.com/>
16. **V0** - <https://v0.dev/>