Instrucciones para Ejecutar y Corregir la Aplicación Cambridge Prep

Este documento proporciona instrucciones detalladas para ejecutar la aplicación Cambridge Prep y corregir los errores encontrados durante la migración de React/Next.js a Ionic/Angular.

Requisitos Previos

- Node.js (v18 o superior)
- npm (v9 o superior)
- Ionic CLI (`npm install -g @ionic/cli`)
- Angular CLI (`npm install -g @angular/cli`)

Instalación

1. Descomprimir el archivo `cambridge-prep-ionic-src.tar.gz` :
   `bash`
   `tar -xzf cambridge-prep-ionic-src.tar.gz`
2. Navegar al directorio del proyecto:
   `bash`
   `cd cambridge-prep`
3. Instalar las dependencias:
   `bash`
   `npm install`

Ejecución del Proyecto

Para ejecutar el proyecto en modo desarrollo:

```
ionic serve
```

Corrección de Errores

A continuación se detallan los errores encontrados y cómo corregirlos:

1. Errores de Importación de Módulos

Algunos componentes están marcados como standalone pero se declaran en módulos. Para corregir este problema:

1. Abrir el archivo `src/app/pages/auth/auth.module.ts` y modificar:
   ```typescript
   // Cambiar
   declarations: [AuthPage]

// Por
imports: [
// otros imports
AuthPage
]
```

1. Repetir el mismo proceso para otros módulos con componentes standalone.

2. Errores de TypeScript

Para corregir los errores de propiedades sin inicializar:

1. Abrir el archivo `src/app/pages/auth/auth.page.ts` y modificar:
   ```typescript
   // Cambiar
   loginForm: FormGroup;
   registerForm: FormGroup;

// Por
loginForm!: FormGroup;
registerForm!: FormGroup;
```

1. Para los errores de índice en objetos, añadir una firma de índice:
   ```typescript
   // Cambiar
   mockQuizPerformance = {
   grammar: 75,
   ```

```
    vocabulary: 80,
    reading: 65,
    listening: 70
    };

// Por
mockQuizPerformance: { [key: string]: number } = {
grammar: 75,
vocabulary: 80,
reading: 65,
listening: 70
};
```

3. Errores de Directivas

Para corregir los errores de directivas como *ngIf y* ngFor:

1. Importar CommonModule en los módulos correspondientes:
   ```typescript
   import { CommonModule } from '@angular/common';

@NgModule({
imports: [
CommonModule,
// otros imports
],
// …
})
```

1. Para componentes standalone, añadir las directivas a los imports:
   ```typescript
   import { Component } from '@angular/core';
   import { NgIf, NgFor } from '@angular/common';

@Component({
selector: 'app-component',
templateUrl: './component.html',
styleUrls: ['./component.scss'],
standalone: true,
imports: [NgIf, NgFor]
})
```

export class Component {
// …
}
```

### 4. Errores de Bibliotecas

1. Para jwt-decode, actualizar la importación:
   ```typescript
   // Cambiar
   import jwt_decode from 'jwt-decode';

// Por
import { jwtDecode } from 'jwt-decode';

// Y cambiar todas las instancias de jwt_decode por jwtDecode
```

1. Para ng2-charts, actualizar la importación:
   ```typescript
   // Cambiar
   import { NgChartsModule } from 'ng2-charts';

// Por
import { provideCharts, withDefaultRegisterables } from 'ng2-charts';

// Y actualizar la configuración en el módulo
providers: [
provideCharts(withDefaultRegisterables())
]
```

Compilación para Producción

Una vez corregidos los errores, para compilar la aplicación para producción:

```
ionic build --prod
```

Despliegue en Dispositivos Móviles

Para añadir soporte para iOS:
```

```
ionic capacitor add ios
ionic capacitor build ios
```

Para añadir soporte para Android:

```
ionic capacitor add android
ionic capacitor build android
```

Contacto

Si tienes alguna pregunta o problema, no dudes en contactar al equipo de desarrollo.