Installation and Configuration:
(https://next-intl-docs.vercel.app/docs/getting-started/app-router/with-i18n-routing)

1) run the following command:

```
npm install next-intl
```

2) Create folder at the root of the project:

```
/messages
```

3) Inside the /messages folder create the files:

```
es.json
en.json
```

4) Inside en.json:

```
{
  "Index": {
    "title": "Hello world!"
  }
}
```

Inside es.json:

```
{
  "Index": {
    "title": "Hello world!"
  }
}
```

5) Configuration in next.config.mjs:

Replace all existing content with this other one:

```
import createNextIntlPlugin from 'next-intl/plugin';

const withNextIntl   = createNextIntlPlugin();

/** @type {import('next').NextConfig} */
const nextConfig = {};

export default withNextIntl(nextConfig);
```

6) Create i18n.ts file inside the /src folder

7) Inside the i18n.ts file put this code:

```
import {notFound} from 'next/navigation';
import {getRequestConfig} from 'next-intl/server';

// Can be imported from a shared config
const locals = ['in', 'of'];

export default getRequestConfig(async ({locale}) => {
  // Validate that the incoming `locale` parameter is valid
  if (!locales.includes(locale as any)) notFound();

  return {
    messages: (await import(`../messages/${locale}.json`)).default
  };
});
```

IMPORTANT:

On this line: const local = ['en', 'es']; replace ed with es

8) Create middleware.ts file inside /src folder

9) Inside the middleware.ts file put this code:

```
import createMiddleware from 'next-intl/middleware';

export default createMiddleware({
  // A list of all locales that are supported
  local: ['en', 'de'],

  // Used when no locale matches
  defaultLocale: 'in'
});

export const config = {
  // Match only internationalized pathnames
  matcher: ['/', '/(from|in)/:path*']
};
```

IMPORTANT:

On this line: local: ['en', 'es'], replace de with es

On this line: defaultLocale: 'en' replace en with es

In this line: matcher: ['/', '/(es|en)/:path*'] replace de with es

10) Create [locale] folder inside /src/app folder

11) Drag the files ’layout.tsx’ and ’page.tsx’ from the /src/app folder into the /src/app/[locale] folder

12) In app/[locale]/layout.tsx:

1- Add this parameter and its type:

params: {locale}

params: {locale: string};

2- Modify this: import "./globals.css"; for this other import "@/app/globals.css";

3- Modify the lang attribute of the html document: <html lang=’en’> to <html lang={locale}>

The ’layout.tsx’ file should look like this:

```
import { ReactNode } from "react";
import type { Metadata } from "next";
import { Inter } from "next/font/google";
import "@/app/globals.css";

const inter = Inter({ subsets: ["latin"] });

export const metadata: Metadata = {
  title: "Example i18n width Next.js 14 and TS",
  description: "Example i18n width Next.js 14 and TS",
};

export default function RootLayout({
  children,
  params: {locale}
}: Readonly<{
  children: ReactNode;
  params: {locale: string};
}>) {
  return (
    <html lang={locale}>
      <body className={inter.className}>{children}</body>
    </html>
  );
}
```

13) In app/[locale]/page.tsx:

1. import: import {useTranslations} from ’next-intl’;

2. create constant inside the component body: const t = useTranslations(’Index’); // access our en.json ’Index’ and es.json ’Index’

3. we bring the ’title’ of our .json and show it this way <h1>{t(’title’)}</h1>;

The ’page.tsx’ file should look like this:

```
import {useTranslations} from ’next-intl’;

export default function Home() {

  const t = useTranslations(’Index’);

  return (
   <div className="min-h-screen flex justify-center items-center text-2xl">
     <h1>{t(’title’)}</h1>;
   </div>
  );
}
```

14) MAKE CHANGES:

If you want to modify the names of the objects and properties of the .json files you must do it this way:

For example, we change the name of the ’Index’ object to ’Home’ and the ’title’ property to ’text’.

1. en.json file:

```
{
    "Home": {
        "text": "Hello world!"
    }
}
```

2. es.json file:

```
{
    "Home": {
        "text": "Hello world!"
    }
}
```

3. pages.tsx file:

```
import {useTranslations} from ’next-intl’;

export default function Home() {

  const t = useTranslations(’Home’);

  return (
   <div className="min-h-screen flex justify-center items-center text-2xl">
     <h1>{t(’text’)}</h1>
   </div>
```

```
  );
}
```

15) ADD PAGE NOT-FOUND

1. create file not-found.tsx inside /src/app/[locale]

2. create the following code inside the not-found.tsx file:
(https://nextjs.org/docs/app/api-reference/file-conventions/not-found)

```
import Link from 'next/link'

export default function NotFound() {
  return (
    <div className="min-h-screen flex gap-4 justify-center items-center text-xl">
      <h2>Page not found:</h2>
      <Link href="/" className='underline'>Return Home</Link>
    </div>
  )
}
```

16) CREATE 'header' COMPONENT:

1. create the /components folder inside the /src folder

2, inside the /components folder create the Header.tsx file

3. inside the Header.tsx file create the code:

```
const Header = () => {
  return (
    <header>
        Header text
    </header>
  );
}

export default Header
```

4. add the 'Header' component to our 'Home' page:

```
import Header from '@/components/header';
import {useTranslations} from 'next-intl';

export default function Home() {

  const t = useTranslations('Home'); // access our en.json 'Index' and es.json 'Index' and bring the 'title' to display it like this <h1>{t('title')}</h1>;

  return (
   <div className="min-h-screen flex flex-col justify-center items-center text-2xl">
    <Header />
     <h1>{t('text')}</h1>
```

```
  </div>
 );
}
```

But the ’Header Text’ is not translated.

17) TRANSLATE ’HEADER TEXT’ (COMPONENT):

1. en.json file:

```
{
    "Home": {
        "text": "Hello world!"
    },
    "Header": {
        "text": "Header in English"
    }
}
```

2. es.json file:

```
{
    "Home": {
        "text": "Hello world!"
    },
    "Header": {
        "text": "Header in Spanish"
    }
}
```

3. header.tsx file:

```
"use client"

import {useTranslations} from ’next-intl’;

const Header = () => {

const t = useTranslations(’Header’);

  return (
    <header>
        <h1>{t(’text’)}</h1>
    </header>
  );
}

export default Header
```

IMPORTANT:

If we do this process the same as we did with ’Home’, since it is a client-side component it will give us this error:

Error: Failed to call ‘useTranslations‘ because the context from ‘NextIntlClientProvider‘ was not found.

This can happen because:
1) You intended to render this component as a Server Component, the render
   failed, and therefore React attempted to render the component on the client
   instead. If this is the case, check the console for server errors.
2) You intended to render this component on the client side, but no context was found.
   Learn more about this error here:
https://next-intl-docs.vercel.app/docs/environments/server-client-components#missing-context

It’s because ’useTranslations’ can only be used server-side.

To solve it we must do it this other way:

1. we copy from (https://next-intl-docs.vercel.app/docs/environments/server-client-components):

```
import pick from ’lodash/pick’;
import {NextIntlClientProvider, useMessages} from ’next-intl’;
```

2. we add the imports to the file where we are going to use the component, in this case on the ’Home’ page.

3. we must install lodash

```
npm i lodash
npm i --save-dev @types/lodash
```

4. we surround our component with: NextIntlClientProvider and pass it the messages: const messages = useMessages();

The ’Home’ page should look like this:

```
import pick from ’lodash/pick’;
import {useTranslations, NextIntlClientProvider, useMessages} from ’next-intl’;
import Header from ’@/components/header’;

export default function Home() {

  const t = useTranslations(’Home’); // access en.json ’Home’ and es.json ’Home’ and bring the ’text’ to display it like this <h1>{t(’text’)}</h1>;
  const messages = useMessages();

  return (
    <div className="min-h-screen flex flex-col justify-center items-center text-2xl">
      <NextIntlClientProvider messages={pick(messages, ’Header’)}>
        <Header />
      </NextIntlClientProvider>
      <h1>{t(’text’)}</h1>
    </div>
  );
}
```

To pass all messages, replace: messages={pick(messages, 'Header')} with messages={messages} and comment // import pick from 'lodash/pick';

---------- END ----------