Actividad evaluable UT2A1 (AE1.3) – Estructura del proyecto y Guía de estilos

RA1, RA3, RA4

La actividad evaluable UT2A1 es el comienzo del proyecto que realizarán durante UT2, UT3 y UT4. En esta actividad se centrarán en preparar la parte frontend del proyecto y en elaborar una guía básica de estilos con MUI para la aplicación que van a realizar.

¿Qué tengo que entregar? (Requisito)

- Enlace al github del proyecto.
- Memoria en formato PDF con capturas de pantalla y breve explicación de lo especificado en verde en este documento. La memoria debe tener portada e índice, el texto debe tener alineación justificada, la expresión escrita debe ser correcta.

¿Cómo identifico mi trabajo? (Requisito)

- Debes poner tu nombre y apellidos en el fichero README.md
- Debes nombrar el PDF como sigue:

PrimerApellido_SegundoApellido_Nombre_UT2A1

Desarrollo de la actividad

El presente proyecto va a tener frontend y backend, es decir, creamos la interfaz en el frontend y luego accederemos a una base de datos local mediante un backend (en realidad es una API que funciona como backend). Por lo tanto, el proyecto tendrá dos carpetas, la carpeta frontend y la carpeta backend. Es en la carpeta frontend donde usaremos React+Vite+Typescript.

1. Comienzo del proyecto y creación del frontend

Dentro de la carpeta donde tienes todos tus proyectos React abre el Visual Studio Code. Abre el terminal y crea la carpeta proyecto*tusiniciales* (ejemplo: proyectophr)

mkdir proyecto*tuiniciales*

cd proyecto*tusiniciales*

npm create vite@latest

→ Llama a tu proyecto **frontend**

→ Usaremos React

→ Usaremos TypeScript + SWC

```
PS C:\Users\Patricia\Documents\PATRICIA\SISTEMAS 24-25\DAD 24-25\ProyectosReact\proyectopatricia> npm create vite@latest

> npx
> create-vite

√ Project name: ... frontend
√ Select a framework: » React
√ Select a variant: » TypeScript + SWC
```

El backend lo crearemos más adelante.

2. Instalación de la librería MUI:

Trabajaremos con la librería MUI, así que instala lo necesario dentro de la carpeta frontend. No te olvides de los iconos.

3. Estructura del frontend:

En el proyecto habrá varias páginas:

→ Página de Login: donde el usuario se logueará al sistema.

→ Página de Inicio: una vez accedamos a la aplicación con el Login, iremos directamente a la página de inicio.

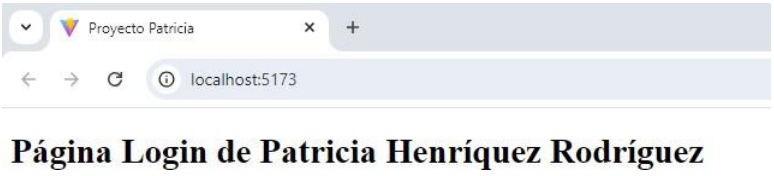
→ Otras páginas que iremos creando para generar informes y documentación entre otras cosas.

Normalmente, cada página es un fichero .tsx que se guarda en la carpeta **src/pages**. Así que vas a crear la carpeta src/pages. Cada página es también un componente, así que los ficheros que crees deben empezar por mayúscula.

Además, también tendremos componentes que se usarán en las páginas. Muchos de ellos se repetirán en todas las páginas, como el menú, por ejemplo. Estos componentes los crearemos en ficheros .tsx que estarán alojados dentro de la carpeta **src/components**. Vamos a crear dicha carpeta.

Ahora crearemos el fichero que contendrá la página de Login: **src/pages/Login.tsx**

Dentro de dicho componente devuelve una Typography (variante h1) con el texto Página de Login de *Nombre y Apellidos* Realiza los cambios necesarios en tu proyecto para que se muestre la página de Login al iniciarse tu aplicación.



4. Componente CSSBaseline: normalización CSS

CssBaseline es un componente que no muestra nada en pantalla. Lo que hace es la normalización css, que es que diferentes elementos se vean igual en diferentes navegadores. Es decir, corrige incoherencias entre navegadores. Los diferentes navegadores web tienen diferentes settings por defecto para aspectos como el margen, así que la aplicación web se verá ligeramente diferente entre

navegadores. El CssBaseline soluciona eso puesto que lo unifica estableciendo un css de base (por ejemplo, el margen lo pone a 0, se aplica de fondo el color por defecto de Material UI, …).
El componente CssBaseline debe envolver a nuestro componente raíz, es decir, a toda la aplicación.
En el fichero main.jsx importamos el componente CssBaseline

```
import CssBaseline from '@mui/material/CssBaseline'
```

Después lo usamos envolviendo al componente raíz <App />:

src > main.tsx

```tsx
import { StrictMode } from 'react'
import { createRoot } from 'react-dom/client'
import App from './App.tsx'
import CssBaseline from '@mui/material/CssBaseline'
//import './index.css'

createRoot(document.getElementById('root')!).render(
  <StrictMode>
    <CssBaseline />
    <App />
  </StrictMode>,
)
```

De aquí en adelante, en todos nuestros proyectos usaremos CssBaseline.

5. Creación de una guía de estilos de mi aplicación

Antes de avanzar tenemos que definir cómo será el tema o el estilo de la aplicación, es decir, paleta de colores, tipo de letra, contraste entre el fondo y la letra, etc. Definir un tema nos permite tener una consistencia de estilo en toda nuestra aplicación. Esto nos servirá para crear una interfaz usable y accesible. Con la librería MUI podemos personalizar el tema de forma bastante fácil. Tienes una guía de cómo hacerlo en el Campus.
Crea un tema personalizado usando la herramienta de creación de temas de MUI:

https://zenoo.github.io/mui-theme-creator/

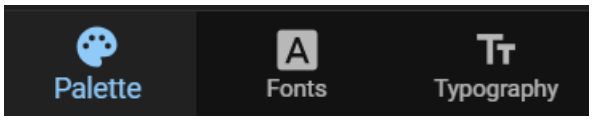
Además, puedes ayudarte del generador de paleta de colores de MUI:

https://m2.material.io/inline-tools/color/

o de cualquier otro generador de colores.

A la hora de elegir tus preferencias de estilo, ten en cuenta que deben cumplir los estándares de usabilidad y accesibilidad: usa colores con un contraste adecuado, una fuente (tipo de letra) apropiada, etc. Puedes chequear el contraste con la web: https://contrastchecker.com/ .

Debes elegir tu **Paleta de colores personalizada**:



Una vez elegidas tus preferencias saca **captura de pantalla** de:

a) la paleta elegida: esto lo tienes en la parte de Palette donde se ve si el tema es claro u oscuro, el color de *background, Text, Primary*, etc (debe verse toda la paleta).

b) de cómo quedarían los botones: esto lo tienes en la parte izquierda de la herramienta, en *Buttons*.

c) Además, **copia y pega en tu memoria el código generado por la herramienta de creación de temas de MUI.**

Debes explicar en la memoria cómo has elegido la paleta de colores. Decir qué herramientas has usado a parte de la herramienta de creación de temas MUI.

6. Uso de la guía de estilos generada

En tu aplicación debes usar la guía de estilos que has generado. En el fichero **main.tsx** crea el código para poder usar tu tema personalizado.

En el fichero **./pages/Login.tsx** crea:

a) Un componente Container

b) Dentro del Container crea:

i) varios componentes Typoghaphy con diferentes variantes (h1, h2, h3, subtitle1, body1, caption) y con diferentes colores del tema.

ii) varios componentes Button de diferente tipo (text, contained, outlined) a los que cambies el color (primary, secondary, error, success, alert, ...).

Captura de pantalla de cómo se ve la aplicación en el navegador

7. Medida de usabilidad y accesibilidad de tu aplicación

En clase hemos visto algunas herramientas para medir la usabilidad y la accesibilidad de una aplicación. Aquí usaremos la Wave Evaluation Tool, que debes instalar como una extensión de tu navegador Google Chrome.

a) Ejecuta tu aplicación y aplica dicha extensión. **Saca captura de pantalla del resultado. Comenta en la memoria qué errores o avisos te muestra la herramienta.**

b) Investiga la alerta de *No page regions.* **¿De qué se trata? ¿Cómo podrías solucionarlo? (contesta estas preguntas en la memoria). Realiza captura de pantalla de la solución aportada.**

8. Cómo subo mi proyecto al gitHub

Como cualquier otro proyecto, pero teniendo en cuenta que te debes situar en la carpeta proyecto*Tusiniciales* para hacer el git init y todo el procedimiento posterior.

PUNTUACIÓN

Entrega	Nota MÁXIMA
1) El ejercicio se entrega en plazo y resuelto correctamente.	10
2) Igual que el 1) pero fuera de plazo. Máximo 24 horas	6
3) No se admiten entregas fuera de plazo cuando se han pasado más de 24 horas de la fecha límite de entrega.	0
4) Si no se cumplen las especificaciones de entrega la tarea irá directamente a recuperación.	0

En la tabla se indican las notas máximas en cada situación.