Tareas de proyecto “Banco Santander”

Series de tareas para Hernán, fecha de entrega aún por definir
Cualquier duda preguntarle al profesor Antonio Labraña

OBLIGATORIO Utilizar angular para la elaboración de estas tareas (Entregar un HTML y CSS NO serán tomados como evaluación)

Tareas de proyecto:

1. Estructurar el proyecto

Crear la estructura inicial del proyecto en Angular.

- Inicializar un nuevo proyecto Angular usando Angular CLI.
- Configurar el enrutamiento básico de la aplicación.
- Crear componentes iniciales para las diferentes vistas (Login, Registro, Dashboard).

2. Crear componentes y sus rutas

Crear los componentes necesarios y configurar las rutas.

- Crear los componentes: LoginComponent, RegisterComponent, DashboardComponent.
- Configurar las rutas en app-routing.module.ts para que apunten a estos componentes.

3. Formularios al registrarse

Implementar un formulario de registro para nuevos usuarios.

- Crear un formulario en RegisterComponent con los siguientes campos: Nombre, contraseña, DNI, Correo, Número de Teléfono.
- Validar de la siguiente manera:
    - nombre y contraseña obligatorios
    - DNI con 8 dígitos
    - correo electrónico válido
    - número de teléfono de 8 dígitos
- Al enviar el formulario, simular el almacenamiento de los datos del usuario en un array de clientes.

4. Formulario al accedes a la cuenta

Implementar un formulario de login para usuarios existentes.

- Crear un formulario en LoginComponent con campos para Correo y Contraseña.
- Validar que ambos campos sean obligatorios.
- Al enviar el formulario, verificar los datos ingresados contra el array de clientes.
- Si las credenciales son correctas, redirigir al usuario a DashboardComponent.

5. Persistencia Simulada de Usuarios

Gestionar el almacenamiento simulado de usuarios.

- Crear un servicio (UserService) que gestione un array de usuarios.
- Incluir métodos para agregar un usuario (registro) y verificar un usuario (login).
- Utilizar el servicio en los componentes de registro y login.

6. Dashboard y Solicitud de Tarjeta de Crédito

Crear la vista del Dashboard con funcionalidad para solicitar una tarjeta de crédito.

- En DashboardComponent, mostrar un mensaje de bienvenida con el nombre del usuario.
- Incluir un botón "Solicitar Tarjeta de Crédito".
- Al presionar el botón, generar una tarjeta de crédito con número aleatorio y ccv aleatorio y fecha de vencimiento aleatorio y agregarla al perfil del usuario.

7. Visualización de la Tarjeta de Crédito

(IMPORTANTE: Implementar las tareas realizadas anteriormente en la primera serie de tareas dadas)

Mostrar la tarjeta de crédito generada en el Dashboard.

- Si el usuario ha solicitado una tarjeta de crédito, mostrar los detalles de la tarjeta (número y fecha devencimineto, y ccv).

8. Validaciones y Mensajes de Error

Implementar validaciones en los formularios y mostrar mensajes de error apropiados.

- Mostrar mensajes de error bajo los campos de los formularios cuando no se cumplan las validaciones.
- Asegurarse de que los mensajes sean claros y útiles para el usuario.

Consideraciones

Simulación de Datos: Al no usar backend, todos los datos (usuarios y tarjetas) deben almacenarse en arrays en el frontend. Se pueden usar servicios de Angular para gestionar estos arrays.

Evaluación: El equipo evaluará todas estas tareas (Y además la entrega dada anteriormente) como nota para la entrega final (Que equivale a un 50% de la nota del proyecto), y se evaluará con los requisitos que se piden en la pauta de front-end (Solamente apartado front-end).

La fecha de entrega de este trabajo tiene que ser 1 día antes de la presentación para poder revisarlas.

Recordar que la pauta de front-end está subida desde el principio del semestre en intranet

CONTEXTO	CRITERIO	DESCRIPCIÓN	1 (bajo)	2 (medio)	3 (alto)
Front-End	Uso de directivas y properties	Se usaron directivas y propiedades como: ngIf, ngFor, property binding, event binding, etc, para gestionar eventos y cambios de forma dinámica del proyecto.			
	Comunicación entre componentes	Existe comunicación entre componentes. Estas comunicaciones tienen un propósito claro y se siguieron estándares o patrones en su uso.			
	Servicios, Inyeccion y Observables	Se controlan los cambios a través de servicios, inyecciones y observables. El uso de estos tiene un propósito específico y coherente.			
	Buenas prácticas de desarrollo	Se siguieron convenciones y estándares al momento de desarrollar la aplicación frontend. Se siguieron convenciones como el estilo del nombramiento de los diferentes elementos, como CamelCase, y estos son significativos en su uso. Se usó consistentemente el uso significativo de comentarios, indentación y legibilidad del código, además de una estructuración del proyecto estandarizada.			
	Consumo de APIs	Se realiza de forma correcta y estandarizada el consumo de endpoints y se tiene una estructuración definida para los request y response.			
	Control de errores, validación de datos	Se controlan errores sin que la aplicación quede inactiva y se realizan validaciones en los datos para los diferentes procesos de la aplicación.			
	Implementación de JWT	Se implementa y uso de forma correcta JWT y se manejan a través del mismo restricciones, tiempos de sesión y control de acceso a los datos o funcionalidades del proyecto.			
	Seguridad en rutas	Las rutas proveídas por el frontend tienen protección para evitar el acceso no autorizado y controlan correctamente la respuesta para que los usuarios continúen su navegación en el sistema.			
	Uso de Bootstrap	Se utilizó correcta y consistentemente el uso de Bootstrap en el proyecto.			
	Testing	Se establecieron pruebas unitarias, integración y de sistema de forma automatizada para probar la confiabilidad, robustez y flujos del proyecto.			
	Control de accesos	Se definieron roles para los diferentes usos del sistema. Estos roles son controlados correctamente a fin de separar lógica y funcionalidades de los usuarios.			
Total					