Tecnólogo Informático

UTEC - UTU - UDELAR

San José de Mayo

23/06/2020

Documento Final

Taller de Diseño Web

Integrantes del Grupo

4.710.147-5 Juan Álvarez

4.561.580-8 Carlos Balbiani

Índice

Índice 2

Resumen 3

Palabras clave 3

Introducción 3

Marco conceptual 3

Descripción del problema 4

Solución planteada 4

Arquitectura del sistema 5

Implementación 5

- Productos y herramientas 5
- Evaluación de productos y herramientas 5

Problemas encontrados 5

Evaluación de la solución 6

Desarrollo del proyecto 6

Conclusiones y trabajo a futuro 6

Referencias 6

Resumen

En este artículo se presenta el desarrollar una aplicación del tipo SPA, utilizando el framework css Clarity y el framework javascript Angular 9. Como principal resultado se completó la tarea exitosamente y adquirimos conocimientos invaluables.

Palabras clave

Angular, SPA, API, Clarity, web, internet.

Introducción

La idea del laboratorio final es aplicar los conceptos brindados en clase más los conocimientos adquiridos de investigar los frameworks css asignados a cada grupo para desarrollar una aplicación del tipo SPA.

Dicha aplicación debe desarrollarse con el framework Angular, y deberá incorporar el framework css indicado a cada grupo. El diseño de la aplicación deberá ser full responsive, es decir que se deberá ver de forma adecuada en los distintos tamaños de pantallas.

El trabajo se plantea en el marco de la edición 2020 de la asignatura Taller de Diseño Web.

Marco conceptual

- **API:** (Application Programming Interface) Es un conjunto de subrutinas, funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.
- **Angular:** Framework de código abierto desarrollado por Google para facilitar la creación y programación de aplicaciones web SPA.
- **CSS:** (Cascading Style Sheets) es un lenguaje utilizado para estilizar elementos escritos en un lenguaje de marcado como HTML. CSS separa el contenido de la representación visual del sitio.
- **Framework:** Es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de software concretos, que puede servir de base para la organización y desarrollo de software.
- **HTML:** (HyperText Markup Language) Lenguaje de marcado que se utiliza para definir las estructuras de una página de Internet.
- **JavaScript:** Es un lenguaje de programación utilizado principalmente para aportar dinamismo a tiempo real en páginas web.

- **REST:** (Representational State Transfer) es un estilo de arquitectura para diseñar aplicaciones en red, donde las peticiones recibidas son del protocolo HTTP
- **SPA:** (Single Page Application) es una página web la cual está todo el contenido en una sola página, es decir, carga solamente un archivo HTML y todo se produce dentro de este único archivo. De esta manera se puede ofrecer una experiencia más fluida y rápida.
- **TypeScript:** Es un lenguaje de código abierto desarrollado y mantenido por Microsoft. Añade funciones a JavaScript y a la vez facilita su desarrollo. Destaca por añadir a JavaScript tipado fuerte, objetos basados en clases y por su compilador que traduce todo el código a JavaScript

Descripción del problema

El problema planteado consiste en el diseño y construcción de una aplicación para gestionar un club deportivo que brinda a sus socios la posibilidad de realizar distintas actividades.

Las principales funcionalidades que debe proveer el sistema están divididas en tres partes.

La pública, la cual permite a los visitantes del sitio obtener información sobre la club deportivo.

La privada, que a su vez se divide en las secciones de oficina y de administración. La primera permite la gestión de los socios del club, mientras que la segunda permite la administración del sistema.

Por último, el sistema debe implementarse utilizando el framework css Clarity, el framework js Angular 9, y consumir servicios de una API REST.

Solución planteada

La funcionalidad central del sistema está dada por la gestión de los contratos de los socios. Los contratos están asociados a una persona, a un convenio (que puede tener un descuento en la cuota) y a una categoría de cuota. Las categorías tienen asociadas una lista de precios con su vigencia correspondiente.

El usuario con rol “ADMIN” es quien gestiona la aplicación y tiene mayor acceso al sistema. El rol “USER” es asignado al registrarse un usuario, pero no tiene acceso a ninguna funcionalidad. El rol “SECRETARÍA” son los encargados de gestionar entre otras cosas los contratos de socios.

Pero esta aplicación en realidad es parte del propio sitio web de la empresa, es decir, que aparte de las secciones referentes a la app (que requieren estar autenticados para el acceso), se encuentra también las secciones típicas del sitio web de una empresa (que serán accesibles de forma pública).

Estas secciones son: Inicio (con noticias), quienes Somos , qué actividades brindamos, contactos, administración (parte privada con autenticación).

Dentro de la parte privada, se tiene las siguientes funcionalidades: general, login, registro de usuario.

Dentro de la parte privada, se tiene las siguientes funcionalidades: secretaria, admin, ABM (alta baja modificación) de Actividades, ABM Convenios, ABM Categorías, ABM de Precios de Categorías, ABM Prestadores de Salud, ABM Medios de Pago, ABM Contratos.

Solo para el rol ADMIN, asignación y retiro de roles a los nuevos usuarios

El ADMIN además genera las noticias que se muestran en la portada del sitio. Cada noticia tiene: Título, Descripción, Foto Asociada, Fecha de Caducidad.

Arquitectura del sistema

La aplicación posee una arquitectura en capas, donde el frontend consume recursos de una API REST de la cual se desconoce su implementación.

Implementación

Productos y herramientas

Como principal editor de código se utilizó Visual Studio Code, git y github como gestores de repositorios, Clarity y Angular como frameworks.

A modo de backend se nos proporcionó una API REST para brindar la lógica de negocio y se encarga del almacenamiento de datos.

Evaluación de productos y herramientas

Producto	Puntos Fuertes	Puntos Débiles	Evaluación General
Clarity	Código libre. Fácil de aprender. Bien documentado.	No posee una gran comunidad. Poco personalizable.	Existen mejores alternativas.
Angular	Posee una gran comunidad y documentación	Tiene una gran curva de aprendizaje.	Una poderosa herramienta par ael desarrollo web.

Problemas encontrados

Algunos elementos del framework css no fueron fáciles de ajustar a nuestro criterio.

Fue posible solucionar este inconveniente, mediante hojas de estilos propias.

Evaluación de la solución

Puede que no estemos respetando la filosofía del framework pero al no contar con una gran comunidad, aquellas ambigüedades en la documentación deben ser resueltas mediante criterios externos.

Desarrollo del proyecto

El desarrollo del proyecto ágil y constante.

Tiempos estimados y requeridos para la realización de las actividades (por integrante):

Actividad	Tiempo estimado (hs)	Tiempo requerido (hs)
Estudio de la API	4	3
Investigación de framework CSS	7	6
Diseño de páginas y componentes	7	6
Implementación	18	19
Testing	4	2

Se cumplió adecuadamente con los plazos establecidos gracias a una buena planificación y coordinación, las cuales permitieron la paralelización de las tareas entre los miembros del equipo.

Conclusiones y trabajo a futuro

En conclucion se complio con la entrega exitosamente en el plazo establecido. Se aplicaron los conceptos aprendidos en clase más los conocimientos adquiridos durante la investigación del frameworks CSS, se consiguió el diseño responsive y la estética deseada a pesar de las desventajas del framework.

Como trabajo a futuro se puede seguir mejorando la estética, la usabilidad y la implementación de iconos descriptivos para incrementar su amigabilidad.

Referencias

1. Angular 9 - https://angular.io/ - [consultado: Mayo 2020]
2. Clarity Design System - https://clarity.design/documentation/get-started - [consultado: Mayo 2020]