Descripción del diseño (SDD)

Equipo Orchestra:

Juan José Afanador Ochoa
Stiven González Olaya
John Jairo González Martínez
Emanuel Álvarez Sánchez

Piloto funcional de editor de contenido para tv híbrida en canales públicos en Colombia. caso de estudio: tv a la carta y catch-up

Departamento de Ingeniería de Sistemas Pontificia
Universidad Javeriana
Bogotá, Colombia

1 HISTORIAL DE CAMBIOS

Tabla 1 Historial de cambios

Fecha	Elemento	Encargado
10/08/2021	Resumen	Juan Afanador
15/08/2021	Introducción	Juan Afanador
22/09/2021	Arquitectura	Juan Afanador, John González, Stiven González
11/10/2021	Diseño detallado	Juan Afanador, Emanuel Álvarez
06/11/2021	Persistencia	Emanuel Álvarez. John González

2 RESUMEN

El presente documento busca proveer al lector los conocimientos necesarios para el entendimiento a nivel de componentes y arquitectura del prototipo Orchestra, un Piloto funcional de editor de contenido para tv híbrida en canales públicos en Colombia. Esto por medio de información la cual abarca conceptos teóricos orientados a la construcción del prototipo, así como especificaciones precisas y detalladas con respecto al mismo. Anexo al documento y de manera implícita en el mismo podrá encontrarse múltiples diagramas y diseños los cuales hacen referencia a las diversas fases de construcción del prototipo, su infraestructura y planeación a detalle.

3 TABLA DE CONTENIDOS

Contenido

1 HISTORIAL DE CAMBIOS 2
2 RESUMEN 3
3 Tabla de Contenidos 4
4 Lista de Figuras 5
5 Lista de Tablas 6
6 Introducción 7
7 Arquitectura 8
7.1 Vista lógica del sistema 8
7.2 Vista física del sistema 9
8 Diseño Detallado 11
8.1 Estructura del sistema 11
8.2 Comportamiento del sistema 11
8.3 Persistencia 11
8.4 Interfaz de usuario 13
9 Anexos 15
10 Referencias 16

4 LISTA DE FIGURAS

Fig. 1 Diagrama de clases - modelo de dominio 8
Fig. 2 Diagrama de despliegue Orchestra 9
Fig. 3 Diagrama de despliegue HbbTv 10
Fig. 4 Diagrama de clases – Modelo Ebc 11
Fig. 5 Modelo de persistencia general 12
Fig. 6 Modelo de persistencia de elementos 13
Fig. 7 Diagrama de navegabilidad 14

5 LISTA DE TABLAS

Tabla 1 Historial de cambios.....2
Tabla 2 Navegadores compatibles.....9

6 INTRODUCCIÓN

El presente documento busca establecer una relación de comprensión y explicación con respecto al software desarrollado y titulado Orchestra: Piloto funcional de editor de contenido para tv híbrida en canales públicos en Colombia, haciendo referencia a la necesidad de entender con un nivel alto de especificidad, los diversos componentes y flujos necesarios para el desglose del producto, esto permite a su vez, la modificación parcial o total del código por parte de cualquier lector interesado, ya que al ser código abierto no se encuentra sujeto a restricciones de licencia. La elaboración del documento: Descripción del diseño (SDD) permite visualizar el alcance del proyecto y brinda las herramientas principales a nivel de conocimiento para comprender el software a su totalidad.

Es importante comprender, por tanto, que el desarrollo del software se encuentra atado a las limitantes de este documento, y por ende debe partir de allí.

Se realiza completa cobertura en arquitectura del proyecto, su ideación, maquetación y desarrollo, restricciones físicas y componentes necesarios para su ejecución, hardware y Versionamiento requerido, así como explicación de los diversos flujos de trabajo planteados para el usuario final.

De igual manera es necesario entender la navegabilidad del proyecto, todas las posibilidades y características brindadas al usuario, el comportamiento que debe tener para llegar a cada una de ellas y su respectivo diagrama.

Encontrará también una serie de anexos correspondientes a los diferentes modelos y diagramas planteados para el desarrollo del documento, los cuales se encuentran compuestos por diagramas de clases y secuencia.

7 ARQUITECTURA

7.1 VISTA LÓGICA DEL SISTEMA

La arquitectura implementada dada la complejidad del proyecto es MVC (Model-View-Controller), esto debido a las características otorgadas por el mismo y su relación con el comportamiento del proyecto dado sus componentes.[1]

Es importante, por tanto, entender que Orchestra básicamente resulta como un conjunto de elementos los cuales comprenden diversas vistas (GUI, API). Las cuales permiten las conexiones necesarias para traducir componentes materializados en HTML visual, en componentes literales los cuales pueden ser vistos como una mezcla entre código HTML, CSS y Javascript.

El proyecto a su vez puede ser visto como una aplicación de una sola página (SPA)[2], ya que resulta como un conjunto de elementos web formalizados en una única interfaz visible por un decodificador. Sin embargo, es destacable el hecho de que Orchestra, es tanto un traductor de elementos visuales a código como también una herramienta de creación de contenido interactivo basado en web. Por tanto, a nivel de editor, el aplicativo se encuentra enfocado en la definición por Modelo de dominio buscando así la especificación de los diferentes objetos, entidades y recursos disponibles en la misma.

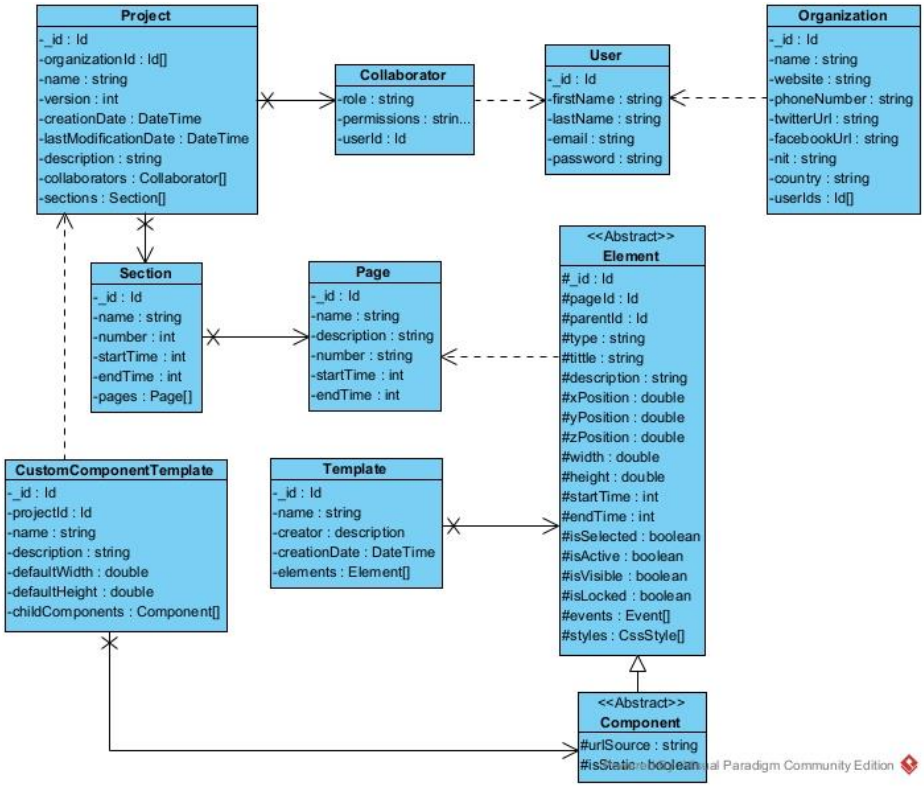

Fig. 1 Diagrama de clases - modelo de dominio

7.2 VISTA FÍSICA DEL SISTEMA

Para hacer uso y tener una experiencia optima al manipular Orchestra es importante entender los 2 conjuntos de componentes necesarios para el uso de este.

- Orchestra al ser un aplicativo web no requiere características de hardware específicas, únicamente su limitante es la versión de navegador a usar:

Tabla 2 Navegadores compatibles

Navegador	Versión estable
Google Chrome	95
Microsoft Edge	95
Safari	13
Opera	80
Firefox	94
Internet Explorer	11

Sin embargo, se recomienda el uso de un pc con mínimo 8Gb de memoria RAM y un procesador de 4 núcleos, esto haciendo referencia a los componentes promedio de los equipos usados en el desarrollo de este.

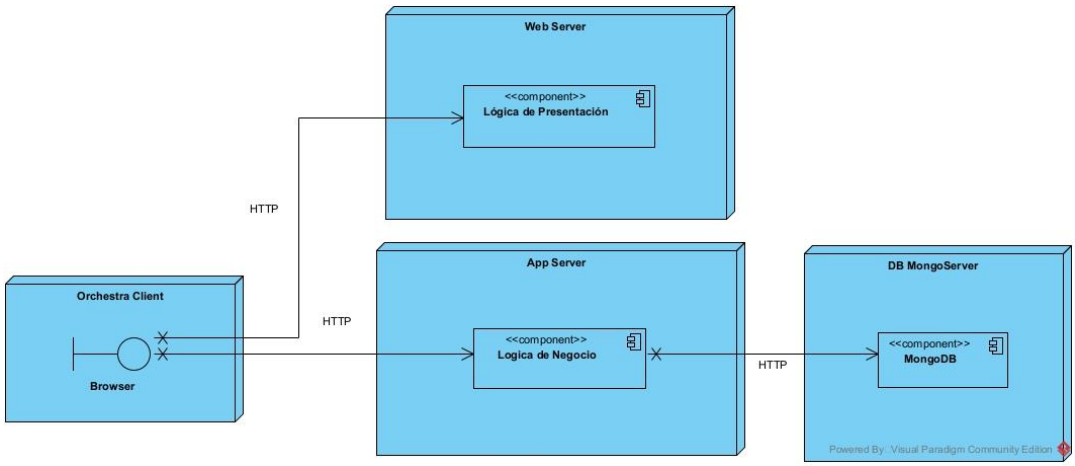

Fig. 2 Diagrama de despliegue Orchestra

- Para ejecutar el aplicativo generado por Orchestra es necesario contar con:

- Decodificador DVB-T2 compatible con HbbTv (SCHWAIGER DTR 700 HD)
- Emulador de TV digital con servicios HbbTv (BeeBeeBox)
- Conexión a internet (WIFI-LAN)
- Televisor (HDMI 2.1)

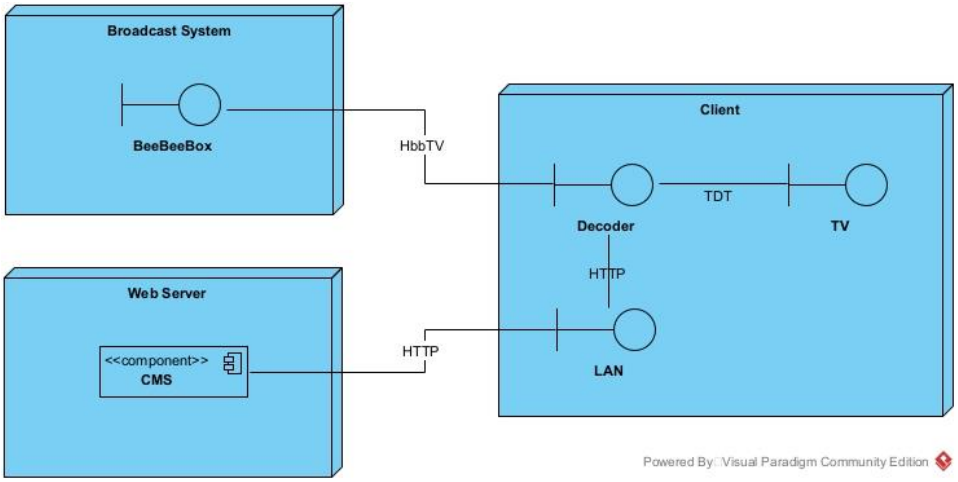

Fig. 3 Diagrama de despliegue HbbTv

8 DISEÑO DETALLADO

8.1 ESTRUCTURA DEL SISTEMA

La representación a nivel general del prototipo se encuentra presentada por el diagrama de clases, el cual evidencia las relaciones entre las diversas entidades presentes y elementos relacionados al contexto del problema.

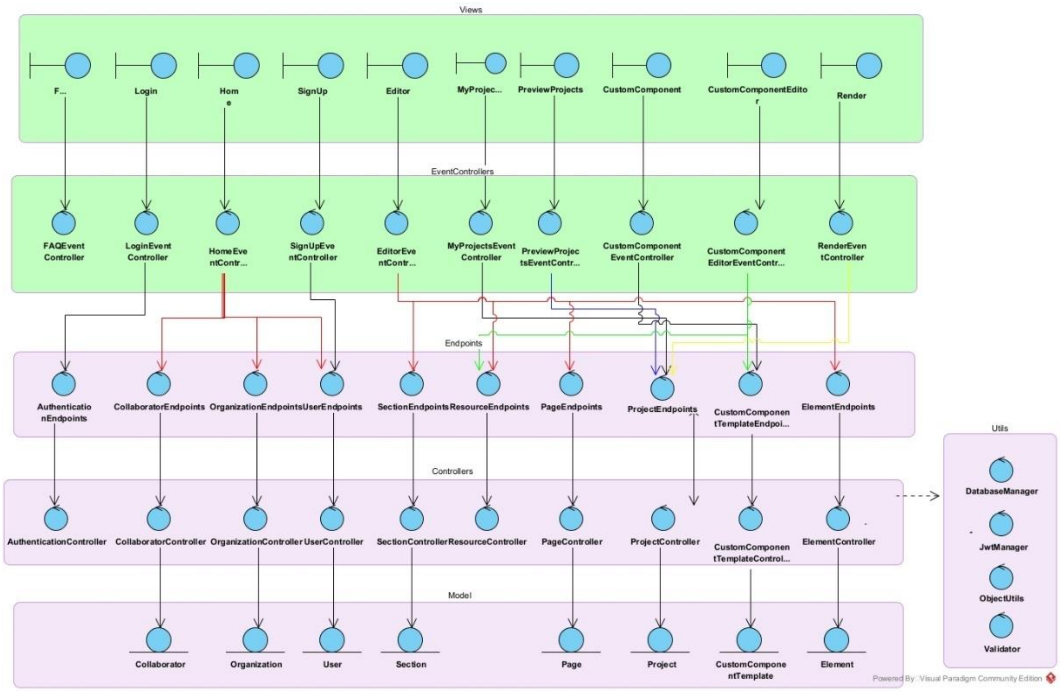

Fig. 4 Diagrama de clases – Modelo Ebc

8.2 COMPORTAMIENTO DEL SISTEMA

En esta sección se presenta el flujo de trabajo entre las diversas entidades relacionadas al usuario y el sistema para cada uno de los casos de uso, representa la respuesta por parte del sistema una vez el usuario inicia una interacción con alguno de los eventos indicados en el diagrama de navegabilidad (Fig. 7). Cada uno de ellos puede encontrarse como anexo del presente documento.

8.3 PERSISTENCIA

Los datos son almacenados por medio de una base de datos no relacional, en este caso MongoDB, la cual usa se basa en una estructura documental [3]. Acorde a esto se diseña un esquema de datos teniendo en cuenta los principios para su acceso (lecturas/escrituras), esto puede ser visto en los diagramas consecuentes.

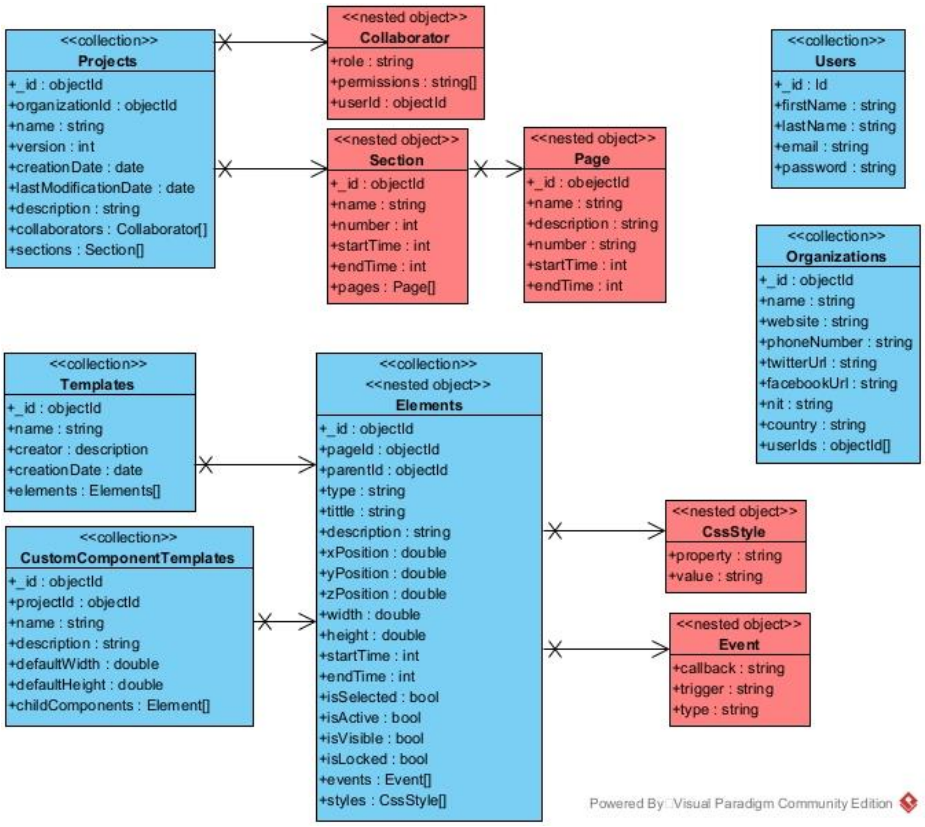

Fig. 5 Modelo de persistencia general

En el diagrama modelo de persistencia general se especifican los elementos que serán persistidos, donde las entidades de color azul representan las colecciones de documentos independientes, mientras que las de color rojo representan objetos anidados dentro de las colecciones principales. De este punto cabe resaltar que los elementos (*Elements*) presentan ambos comportamientos, pues pueden ser trabajados como documentos o como objetos anidados de acuerdo con el problema.

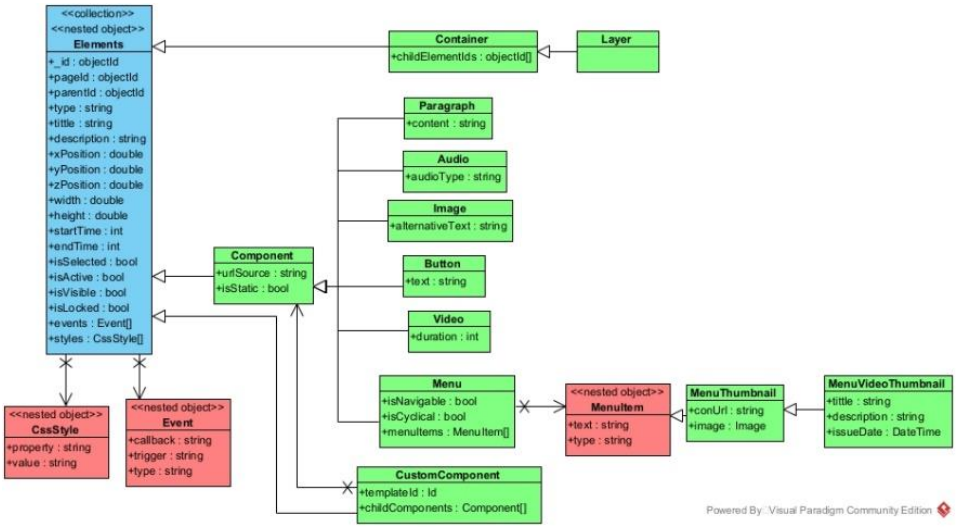

Fig. 6 Modelo de persistencia de elementos

Por otra parte, es pertinente especificar la colección de elementos como se muestra en el modelo de persistencia de elementos, donde la colección principal es *Elements* de color azul, mientras que todas sus posibles variaciones (tipos de elementos disponibles en el editor) se muestran en color verde. De nuevo las entidades de color rojo representan objetos anidados dentro de un documento.
Tanto en M como en N se especifican los tipos de datos que debe llevar cada campo de cada documento de acuerdo con los tipos de datos utilizados por MongoDB por medio de los esquemas BSON[4].

8.4 INTERFAZ DE USUARIO

El siguiente diagrama pretende mostrar los diferentes comportamientos que puede tomar un usuario al usar Orchestra, todos los flujos de interacción posibles con respecto a las diferentes divisiones del proyecto, así como la referencia a los pasos necesarios para llevar a cabo una determinada acción dentro del proyecto.
En el diagrama (Fig. 7) se puede ver de manera macro la navegabilidad del usuario dentro del ciclo de vida del aplicativo.

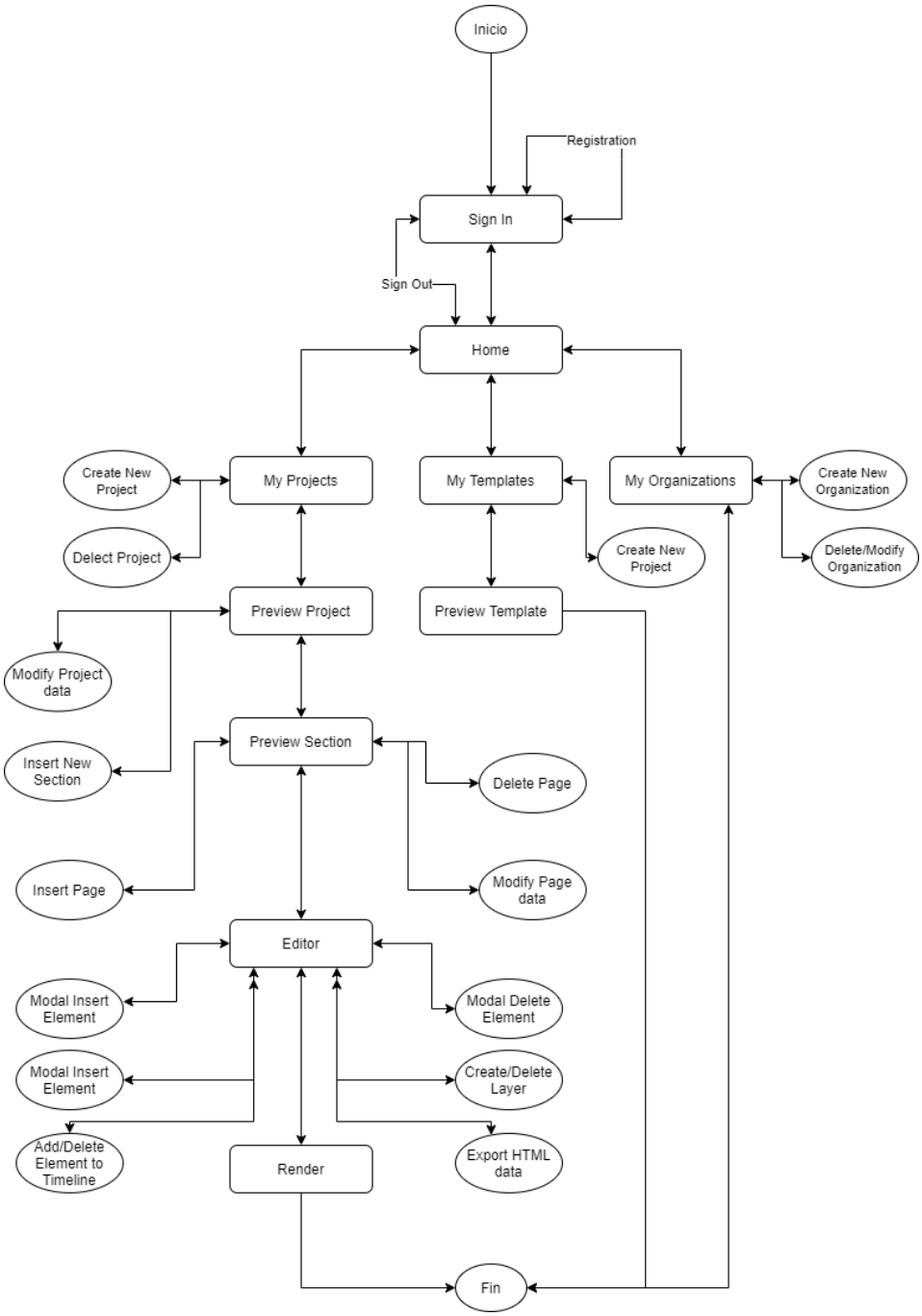

Fig. 7 Diagrama de navegabilidad

9 ANEXOS

- Diagramas de secuencia
- Diagrama de clases
- Diagramas despliegue

10 REFERENCIAS

[1] J. Deacon, “Model-View-Controller (MVC) Architecture”, p. 7, 2009.

[2] E. A. S. Jr, *SPA Design and Architecture: Understanding single-page web applications*. Simon and Schuster, 2015.

[3] Z. Parker, S. Poe, y S. V. Vrbsky, “Comparing NoSQL MongoDB to an SQL DB”, en *Proceedings of the 51st ACM Southeast Conference*, New York, NY, USA, abr. 2013, pp. 1–6. doi: 10.1145/2498328.2500047.

[4] “BSON Types — MongoDB Manual”, *https://github.com/mongodb/docs-bi-connector/blob/DOCSP-3279/source/index.txt*. https://docs.mongodb.com/manual/reference/bson-types/ (consultado nov. 15, 2021).