Universidad de Valladolid

ESCUELA DE INGENIERÍA INFORMÁTICA

GRADO EN INGENIERÍA INFORMÁTICA
Mención Computación

Sistema para la visualización de series temporales obtenidas con un simulador IAM incluyendo extrapolación de valores no vistos

Autor:
Carlota Mª Prieto Martínez
Tutor:
David Escudero Mancebo

A todas las personas que han hecho posible que haya llegado hasta aquí.

RESUMEN

Las técnicas de simulación de sistemas dinámicos generan un gran volumen de datos en forma de series temporales. Disponer de sistemas que permitan la visualización de estas series temporales de forma ágil puede ser ventajoso a la hora de explorar la información almacenada en estos data sets. Las series temporales de los data sets se obtienen para un conjunto limitado de valores de entrada. Disponer de métodos que permitan extrapolar los resultados para valores de entrada no vistos puede aumentar las capacidades de análisis.

En este trabajo se presenta un sistema que permite visualizar la información de un data set de series temporales obtenido con el sistema de simulación de evolución del clima y la economía llamado MEDEAS. Se ha desarrollado un servidor que permite el acceso eficiente a las series temporales del data set y se implementa una técnica de extrapolación multivariante para generar series temporales en valores no vistos. El usuario accede mediante un interfaz web que le permite visualizar proyecciones de la evolución de temperatura y PIB en función de diversas decisiones políticas que tienen que ver con la energía, la población y la economía entre otras variables.

La posibilidad de visualizar el impacto de diversas medidas políticas en la economía y temperatura global, puede servir para concienciar sobre la magnitud del problema.

ABSTRACT

Dynamic systems simulation techniques generate a large volume of data in the form of time series. Having systems that allow the visualization of these time series in an agile way can be advantageous when exploring the information stored in these data sets. Data set time series are obtained for a limited set of input values. Having methods that allow you to extrapolate results for unseen input values can increase your analysis capabilities.

This paper presents a system that allows visualizing the information of a time series data set obtained with the climate and economy evolution simulation system called MEDEAS. A server has been developed that allows efficient access to the time series of the data set and a multivariate extrapolation technique is implemented to generate time series in unseen values. The user accesses through a web interface that allows him to view projections of the evolution of temperature and GDP based on various political decisions that have to do with energy, population and the economy, among other variables.

The possibility of visualizing the impact of various political measures on the global economy and temperature can serve to raise awareness of the magnitude of the problem.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Capítulo 1

1 INTRODUCCIÓN

1.1 CONTEXTO

Actualmente el cambio climático es uno de los grandes problemas que afecta a la sociedad. Sin embargo, pese a ser un tema recurrente en la actualidad existe un gran desconocimiento sobre este asunto, bien por exceso de información, inexactitud de la misma o falsos mitos creados sobre el cambio climático.

El término cambio climático se refiere a los procesos de cambios a largo plazo en las temperaturas y patrones climáticos [1]. Es importante diferenciar cambio climático de calentamiento global. El calentamiento global, principal causa del cambio climático, es el aumento de la temperatura causado por las emisiones de gases debidas a la actividad humana provocando variaciones en el clima de forma antinatural. Si bien es cierto que previamente la Tierra ya ha sufrido variaciones, aumentando y disminuyendo su temperatura de forma natural, estos procesos han sido mucho más lentos, necesitando millones de años.

Los expertos señalan la revolución industrial como punto de inflexión en la emisión de gases, además esta revolución dio lugar a nuevos modelos de producción y consumo. Desde este momento la población ha crecido exponencialmente, el consumo de recursos es cada vez más desmedido, el aumento en la demanda y producción de energías, en su mayoría fósiles, han provocado que la tierra entre en una nueva era geológica, el Antropoceno [2].

El principal resultado ha sido el aumento de la temperatura global 11 grados, incremento que puede ser aún mayor al finalizar el siglo. Este aumento de la temperatura trae consecuencias desastrosas que ponen en peligro la supervivencia de la flora y fauna de la Tierra. Entre los impactos del cambio climático destacan:

- La aparición de **fenómenos meteorológicos más extremos** frecuentes e intensos que conlleva pérdidas de cultivo, ganado, infraestructuras y vidas humanas.
- **Acidificación y contaminación** del agua debido a la concentración de dióxido de carbono en el aire.
- **Destrucción de ecosistemas**, debido a la alteración del ciclo del agua y al deshielo.
- **Daños sobre la salud humana.** El cambio climático provoca tanto daños directos, como el aumento de casos de asma, enfermedades pulmonares y cardiacas, como daños indirectos como el aumento de casos de ciertas enfermedades debido a la ampliación de lugares donde puedan vivir y reproducirse ciertos insectos que contagien enfermedades [3].

Los expertos señalan que en 2030 podríamos llegar a un punto de no retorno para poder revertir los daños provocados. Para esta fecha deberíamos disminuir en un 45% las emisiones netas de dióxido de carbono respecto a los niveles de 2010 para poder así mantener el calentamiento sobre los 1,5 grados centígrados. Esta reducción requeriría de cambios generalizados en energía, transporte, edificios y ciudades [4].

1.2 OBJETIVOS

El objetivo de este trabajo es crear un sistema de visualización interactivo mediante el cual se pueda observar la evolución de ciertas métricas, en este caso la temperatura y el PIB, desde el año 2000 al 2100. Este sistema permitirá además de visualizar, explorar los datos de las series temporales e implementará un algoritmo para poder extrapolar datos no vistos de estas series. Paralelo a esto, el objetivo de este trabajo también es definir y analizar los requisitos necesarios para lograr el funcionamiento de la aplicación, así como el diseño de los casos de uso y diagramas necesarios para documentar la aplicación.

1.3 REFERENTES

Actualmente existen muchos modelos de sistemas de generación de datos. Crear un alto volumen de datos conlleva muchos desafíos, desde donde almacenarlos y como tratarlos hasta como explotarlos para poder extraer información de calidad. Es importante disponer de herramientas eficientes que nos permitan extraer la mayor rentabilidad posible a los datos [5]. Un ejemplo de esto, que sirve como referente para este trabajo es En-ROADS.

En-ROADS es una herramienta de simulación desarrollada por el MIT. Esta herramienta permite a los usuarios explorar el impacto de alrededor de 30 políticas en cientos de factores como la temperatura, la energía, los precios, el nivel del mar, la calidad del aire o la temperatura. De esta forma los usuarios pueden observar los efectos a largo plazo de estas políticas [6].

En la Ilustración 1 se puede observar la aplicación del MIT, EN-ROADS, que sirve como inspiración para la interfaz gráfica el proyecto [7].

Ilustración 1. Imagen de la aplicación EN-ROADS desarrollada por el MIT.

Sin embargo, esta herramienta únicamente nos muestra la evolución de estas métricas para valores observados. En este trabajo pretendemos dar un paso más y poder mostrar y explorar la evolución de la temperatura y el PIB para valores no observados mediante una extrapolación.

1.4 ESTRUCTURA DE LA MEMORIA

A continuación, se presenta cada uno de los capítulos en los que se ha dividido este proyecto:

Capítulo 1. Introducción. En este capítulo se presenta el proyecto, el contexto, la motivación, los objetivos y la estructura de este trabajo.

Capítulo 2. Datos disponibles y método de extrapolación de valores no vistos. En este capítulo se presentan los datos, así como su origen y las operaciones que se realizan con ellos.

Capítulo 3. Análisis y requisitos. En este capítulo se exponen los requisitos de la aplicación, sus casos de uso, así como algunas definiciones necesarias para el trabajo.

Capítulo 4. Planificación y seguimiento. En este capítulo se muestra la planificación inicial del trabajo, el desarrollo seguido a lo largo de los meses y el presupuesto del proyecto.

Capítulo 5. Diseño. En este capítulo se documenta la fase de diseño de la aplicación. Se define la arquitectura, el despliegue de la arquitectura lógica sobre la arquitectura física y los mockups.

Capítulo 6. Implementación. En este capítulo se describen las tecnologías exploradas, los patrones utilizados y la API Rest. Además, se muestra la organización que sigue el código del proyecto.

Capítulo 7. Pruebas. En este capítulo se documentan las pruebas realizadas para comprobar el funcionamiento correcto de la aplicación.

Capítulo 8. Conclusiones y trabajo futuro. En este capítulo se comentan las conclusiones del proyecto, así como posibles mejoras y ampliación del trabajo.

Capítulo 9. Bibliografía. En este capítulo se muestran las referencias bibliográficas utilizadas para el desarrollo de este trabajo.

Capítulo 2

2 DATOS DISPONIBLES Y MÉTODO DE EXTRAPOLACIÓN DE VALORES NO VISTOS

Una serie temporal es un conjunto de observaciones ordenadas temporalmente. El estudio de las series temporales es diferente al del resto de las variables estadísticas ya que el interés reside en la evaluación de sus cambios a lo largo del tiempo [8]. Una serie temporal discreta está compuesta de observaciones tomadas para valores de tiempo equiespaciados [9].

Una serie temporal multivariante consta de más de una variable dependiente del tiempo y cada variable depende no solo de sus valores pasados, sino que también tiene cierta dependencia de otras variables [10]. Las series temporales que visualizamos en este proyecto se han obtenido con el sistema IAM MEDEAS que se describe a continuación.

2.1 EL SISTEMA IAM MEDEAS

MEDEAS es un IAM (Modelos de Evaluación Integrados) de simulación de políticas diseñados por el Grupo de Energía, Economía y Dinámica de Sistemas de la Universidad de Valladolid (España). Un IAM es un modelo matemático que intenta vincular las principales características de la sociedad y economía con la biosfera y la atmosfera. El objetivo principal de estos modelos es adaptarse a la formulación de políticas normalmente relativas al cambio climático, pero también en otras áreas del desarrollo humano y social [11].

El modelo MEDEAS-World ha sido diseñado aplicando Dinámica de Sistemas, lo que facilita la integración del conocimiento desde diferentes perspectivas, así como la retroalimentación de diferentes subsistemas.

Este modelo consiste en una estructura modular y flexible, donde cada módulo puede ser ampliado/simplificado/reemplazado por otra versión o submodelo. El modelo se

estructura en 7 submódulos: Economía, Energía, Infraestructuras, Materiales, Uso del suelo, Cambio climático, Indicadores de impactos sociales y ambientales. Esta estructura se puede ver en la Ilustración 2, donde se muestra estos submódulos y las uniones existentes entre ellos.

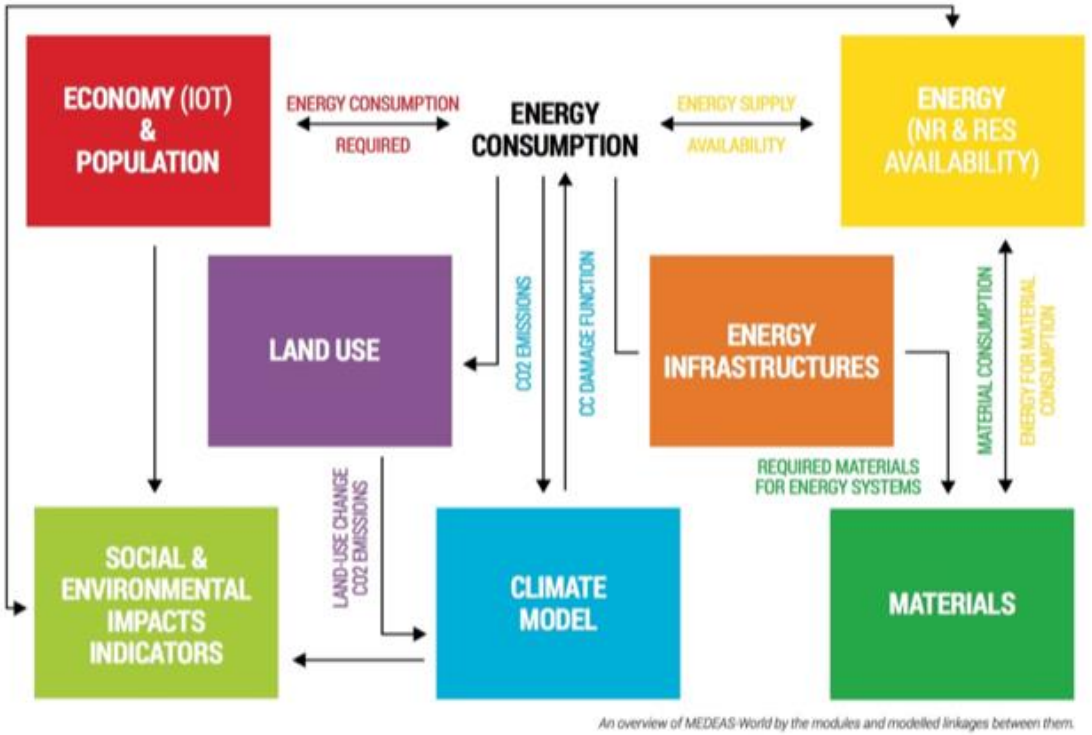

Ilustración 2. Diagrama de los módulos y sus uniones del modelo MEDEAS.

El objetivo de este modelo es explorar estrategias de transición energética con un enfoque novedoso que integra restricciones biofísicas, económicas, sociales y tecnológicas. Este modelo puede ser ampliado y modificado dependiendo de la disponibilidad de datos o información nueva [12] [13].

Este modelo se usa para realizar las simulaciones y obtener las series temporales multivariantes que predicen la evolución de la temperatura y el PIB. MEDEAS tiene 2473 variables de entrada para generar las series temporales de la evolución global de la temperatura y el PIB [9].

De estas se elige un subconjunto de variables de entrada (las más discriminantes) en un conjunto de hipótesis y medidas políticas que tomadas en el presente podrían afectar escenarios futuros. Estas medidas políticas han sido utilizadas en el contexto de los juegos serios para determinar diferentes futuros escenarios y hacerlos fáciles para que los instructores los discutan con los estudiantes. Un ejemplo de esto es Crossroads 2.0 .

Crossroads 2.0 es un juego de simulación basado en el modelo global, lo que permite al usuario explorar de forma entretenida los futuros escenarios a nivel medioambiental, social y económico y obtener una mejor comprensión de las interacciones humano-medioambiente.

Crossroads 2.0 se juega en pequeños equipos. El juego crea un escenario donde los participantes se encargan de diseñar estrategias climáticas. Basado en estas estrategias el modelo mostrara las consecuencias sociales económicas y medioambientales del escenario definido.

El objetivo del juego es estimular discusiones sobre temas cruciales para la sostenibilidad global [14].

Estos datos se han obtenido bajo Locomotion. Locomotion es un proyecto europeo que pretende diseñar un conjunto de IAM que proporcione una manera de evaluar la viabilidad, efectividad, costos y ramificaciones de diferentes políticas de sostenibilidad.

2.2 DATA SET

En esta sección se analizará el formato de los datos.

Originalmente se aportaron los datos en dos formatos distintos, uno en formato json y otro en csv. En el inicio comenzamos usando los datos en json para importarlos a una base de datos MongoDB. Finalmente, debido a cambios en el planteamiento del proyecto y para un funcionamiento más rápido del mismo se decidió cambiar al formato csv. Los datos de ambos formatos representan la misma información. Estos tienen una estructura 310x 437401.

Podemos separar tres tipos de datos:

Datos que se refieren a las medidas políticas e hipótesis o **conjunto respuesta**: Vienen dados en las primeras doce columnas, recogen la información referente a las respuestas dadas para las tres hipótesis y nueve medidas políticas que propone el juego.

Estas pueden tomar los valores “a”, “b”, “c”, “d” o “e”, por tanto, toman valores discretos. Es importante señalar que no todas las hipótesis y medidas políticas tienen el mismo número de respuestas posibles. (Véase Anexo B)

Simulation input: podemos encontrar estos datos de la fila 13 a 98. Cada conjunto de respuestas tiene asociado unos valores dependiendo de las mismas. Estos datos son unas variables internas que utiliza MEDEAS para llevar a cabo su simulación. Estas variables toman valores numéricos.

Simulation output: estos aparecen en las columnas de las 99 a la 204 para la temperatura y de la 205 a la 310 para el PIB. Estas son las variables que se representan en las gráficas de la aplicación. Estas variables toman valores numéricos.

Para finalizar este apartado es importante resaltar que de las doce primeras columnas que representan los valores de las respuestas tipo test, de las medidas políticas e hipótesis planteadas, no todas las medidas políticas e hipótesis tiene el mismo número de respuestas posibles, estas varían entre dos y cinco. En el apartado 2.3.1 se explicará el problema que supone esta diferencia.

2.3 EXTRAPOLACIÓN DE LOS DATOS

En esta sección se pretende explicar cómo son obtenidos los datos que se muestran en las gráficas. Se distinguen dos procedimientos, en primer lugar, un algoritmo de búsqueda que es usado para encontrar los valores de las gráficas dados un conjunto de valores respuesta y en segundo lugar una extrapolación que se aplica en caso de tener valores intermedios en el conjunto de valores respuesta.

2.3.1 Algoritmo de búsqueda

En esta sección se explica cómo funciona el algoritmo que busca los valores de las gráficas a partir de las combinaciones de los valores de las hipótesis y medidas políticas.

En la aplicación se tienen doce preguntas de respuesta tipo test. Estas preguntas no poseen el mismo número de opciones, se tiene que tres preguntas tienen dos posibles respuestas, siete tienen tres posibles respuestas y dos tienen cinco posibles respuestas.

Al mirar el data set se observa que las filas están ordenadas según las respuestas a las preguntas, es decir, se tiene que en la primera fila las respuestas para las preguntas son *a, a, a ,a , a ,a ,a ,a ,a ,a ,a ,a* y la última toma los valores *c , c ,e , e, b ,c ,c ,c ,b ,c ,b ,c* . De aquí se puede deducir que sigue un sistema quinario donde el valor *a* corresponde con el 0, el *b* con 1, el *c* con el 2, el *d* con el 3 y el *e* con el 4.

Sin embargo, calcular la posición que una combinación de valores ocupa en el data set no es tan sencillo como convertir los valores respuesta del quinario al decimal ya que como se ha comentado anteriormente no todas las preguntas tienen el mismo número de respuestas. Para solucionar este problema se hará lo siguiente:

1.- En primer lugar, se calculará el índice como si todas las preguntas tuviesen cinco posibles respuestas, es decir,

$$\begin{aligned} índice \; = {} & h_1 * 5^{11} + h_2 * 5^{10} + h_3 * 5^9 + m_1 * 5^8 + m_2 * 5^7 + \; m_3 * 5^6 \\ & + m_4 * 5^5 + m_5 * 5^4 + m_6 * 5^3 + m_7 * 5^2 + m_8 * 5^1 \\ & + m_9 * 5^0 \end{aligned} \qquad (1)$$

donde $h_1, h_2, h_3, m_1, m_2, m_3, m_4, m_5, m_6, m_7, m_8, m_9$ son el conjunto de valores respuesta.

2.- En segundo lugar, se deberá añadir un factor de corrección por cada pregunta que no tenga cinco respuestas, para ello se seguirá el siguiente algoritmo.

$$factor\ de\ corrección_n = \left(\frac{índice}{5^{n+1}}\right) * (5 - y_n) * \prod_{1}^{n} y_n \qquad (2)$$

Donde:

$índice$ es el valor calculado con la formula (1),

n es el valor de la posición que ocupa una pregunta,

y_n es el número de posibles respuestas que tiene la pregunta que ocupa la posición n

Una vez se han calculado todos los factores, podremos calcular el índice real que esa combinación de respuestas tiene en el data set.

$$índice\ real = índice - \sum_{0}^{11} factor\ de\ corrección_n \qquad (3)$$

2.3.2 Extrapolación

En esta sección se explica cómo se realiza la extrapolación para calcular los valores que después se representaran en las gráficas.

La extrapolación se calcula en el caso de obtener valores intermedios en el conjunto de datos respuesta.

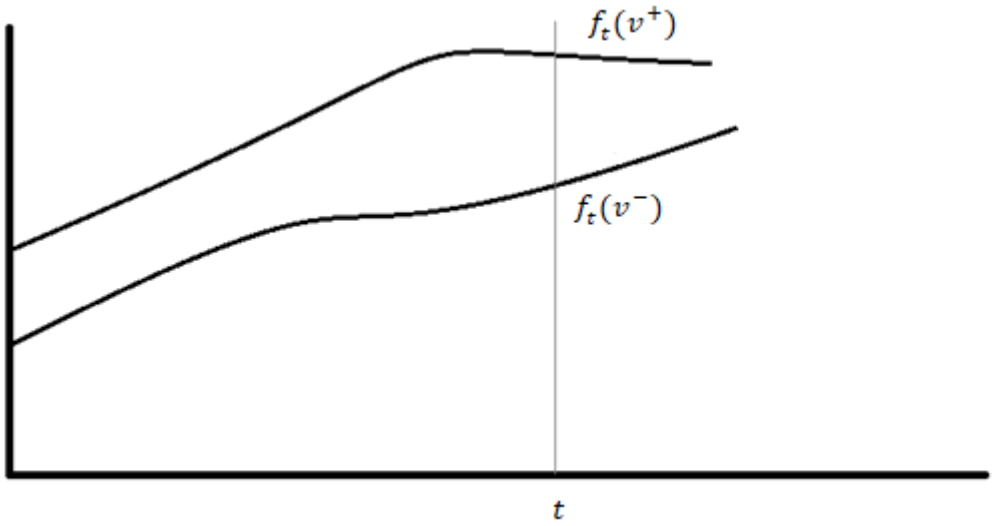

Ilustración 3. Ilustración del problema de extrapolación.

En la Ilustración 3 se observa una gráfica con dos curvas $f_t(v^+)$ y $f_t(v^-)$ que representan la estimación de cierta métrica (en nuestra aplicación podrían ser el PIB o la temperatura) para un conjunto de hipótesis y medidas políticas v^+ y v^- respectivamente. En nuestro problema, v^+ y v^- son un conjunto de 12 valores numéricos discretos.

Supongamos que ahora se quiere estimar esta curva para cierto conjunto v que contiene una hipótesis o medida política no discreta y que se encuentra entre $v^+ \, y \, v^-$.

$$v \in [\, v^-, v^+ \,] \tag{4}$$

Esta curva que llamaremos $\hat{f}_t(v)$, tomara valores intermedios entre $f_t(v^+)$ y $f_t(v^-)$.

$$\hat{f}_t(v) = (f_t(v^+) - f_t(v^-)) * \lambda + f_t(v^-) \tag{5}$$

Donde $\lambda \epsilon [0,1]$, es un factor que indica la cercanía de $\hat{f}_t(v)$ y $f_t(v^+)$.

$$\hat{f}_t(v) = \begin{cases} f_t(v^+), & \gamma = 1 \\ f_t(v^-), & \gamma = 0 \\ f_t(v^-) < \hat{f}_t(v) < f_t(v^+), & 0 < \gamma < 1 \end{cases}$$

λ es una función de v que nos indicará la proximidad de la curva que estamos estimando a $f_t(v^+)$ y $f_t(v^-)$.

$$\lambda = g(v) \quad (6)$$

Cuando únicamente tenemos una respuesta con valor no discreto

$$\lambda = \frac{v - v^-}{v^+ - v^-} \quad (7)$$

Para múltiples respuestas con valores intermedios extrapolamos la fórmula (7)

$$\lambda = \sum_{1}^{n} \alpha_n \frac{v_n - v_n^-}{v_n^+ - v_n^-} \quad (8)$$

$$\alpha = \frac{1}{n} \quad (9)$$

Donde

n es el número de preguntas cuyas respuestas tienen valores no discretos.

Dentro del código de la aplicación estas fórmulas se aplican mediante los siguientes algoritmos.

```
DATAEXTRAPOLATION (v, f_t(v^+), f_t(v^-), λ )

INPUT   v       // Es un vector que contiene los valores elegidos
                // por el usuario para cada una de las hipótesis
                // y medidas políticas de la aplicación
        f_t(v^+)  // Serie de valores correspondientes a la curva superior
        f_t(v^-)  // Serie de valores correspondientes a la curva inferior
        λ       // Valor que indica la proximidad a la curva superior

OUTPUT f̂_t(v)   // Serie de valores correspondientes a la curva extrapolada

BEGIN
      n ← longitud (f_t(v^+) )
      FOR i=0 TO n
                  z ← |f_t(v^+)_i - f_t(v^-)_i| * λ + min(f_t(v^+)_i , f_t(v^-)_i )
                  IF z < 0 THEN
                        z ← 0
                  END IF
                  f̂_t(v)_i ← z
      END FOR
      RETURN f̂_t(v)
END
```

Algoritmo 1: Extrapolación de curvas

```
CALCULATELAMBDA(v, v+ ,v- , n)

INPUT   v      // Es un vector que contiene los valores elegidos
               // por el usuario para cada una de las hipótesis
               // y medidas políticas de la aplicación
        v+     // es un vector que contiene los valores para cada
               // una de las hipótesis y medidas políticas de la
               // aplicación de la curva superior
        v-     // es un vector que contiene los valores para cada
               // una de las hipótesis y medidas políticas de la
               // aplicación de la curva inferior
        n      // es un valor que indica el número de v con
               // valores no discretos

OUTPUT  λ      // un valor numérico entre 0 y 1 necesario para hacer
               // la extrapolación
BEGIN
      λ ← 0
      vH ← 0
      m  ← longitud(v)
      FOR j=1 TO m
            IF v_j^+ ! = v_j^- THEN
                  vH ← vH + (v_j − v_j^-)/(v_j^+ − v_j^-)
            END IF
      END FOR
      α ← 1/n
      λ ← α * vH
      RETURN λ
END
```

Algoritmo 2: Cálculo de λ

DATOS DISPONIBLES Y MÉTODO DE EXTRAPOLACIÓN DE VALORES NO VISTOS

Capítulo 3

3 ANÁLISIS Y REQUISITOS

En esta sección se determinarán las necesidades, objetivos y requisitos de nuestra aplicación.

3.1 DEFINICIONES NECESARIAS

En esta subsección definiremos algunos conceptos necesarios para la comprensión del proyecto.

Dashboard. Un dashboard o cuadro de mando es una herramienta que representa, de manera visual, las principales métricas que intervienen en la consecución de unos objetivos [15]. En este trabajo, las métricas que se representan son la evolución de la temperatura y el PIB a lo largo de 100 años.

Data set. Un data set o conjunto de datos es una colección de datos normalmente tabulada. Los datos tabulados son conjuntos de datos que contiene valores para cada una de las variables organizadas en columnas [16].

Hipótesis y medidas políticas. En este trabajo las hipótesis y medidas políticas son aquellas políticas que los usuarios podrán modificar para comprobar que efectos tiene su cambio a largo plazo y para ver como interactúan unas con otras. Se pueden ver estas políticas más detalladamente en el Anexo B**.**

Conjunto respuesta. Conjunto de valores respuesta a las hipótesis y medidas políticas. Estos son los valores con los que interactúa el usuario en la aplicación.

Simulation input. Cada conjunto respuesta tiene asociados unos valores dependiendo de los valores de las respuestas. Estos datos hacen referencia a valores internos que MEDEAS utiliza para realizar las simulaciones.

Simulation output. Son los datos que se representan en las gráficas, varían según la elección de los valores del conjunto respuesta.

3.2 REQUISITOS

En este apartado, se van a describir los requisitos del sistema: requisitos funcionales, no funcionales, de información y de usabilidad. Los requisitos para un sistema son descripciones de lo que el sistema debe hacer: el servicio que ofrece y las restricciones en su operación. Tales requisitos reflejan las necesidades de los clientes por un sistema que atienda a cierto propósito [17].

3.2.1 Requisitos funcionales

Los requisitos funcionales definen funcionalidades del sistema software o sus componentes. Estas funcionalidades pueden ser descritas como un conjunto de entradas, comportamientos y salidas. Los requisitos funcionales pueden ser: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que se supone, un sistema debe cumplir. En algunos casos los requisitos funcionales explican lo que no debe hacer el sistema [17].

ID	DESCRIPCIÓN
RQF-01	El sistema deberá permitir al usuario visualizar la evolución de la temperatura resultante de tomar una medida.
RQF-02	El sistema deberá permitir al usuario visualizar la evolución del PIB resultante de tomar una medida.
RQF-03	El sistema deberá permitir al usuario hacer cambios discretos en los parámetros.
RQF-04	El sistema deberá permitir al usuario hacer cambios continuos en los parámetros.
RQF-05	El sistema deberá permitir al usuario resetear las medidas realizadas.
RQF-06	El sistema deberá permitir al usuario cambiar la continuidad de los parámetros. (Parámetros discretos y continuos)
RQF-07	El sistema deberá permitir cambiar el idioma.
RQF-08	El sistema deberá ser capaz de acceder a los datos guardados.
RQF-09	El sistema deberá tener las unidades en medidas europeas.

RQF-10	El sistema deberá permitir al usuario obtener los valores de cada punto de las gráficas al pasar el ratón por encima.

Tabla 1. Tabla que describe los requisitos funcionales.

3.2.2 Requisitos no funcionales

Los requisitos no funcionales también llamados atributos de calidad, son aquellos requisitos que indican criterios para valorar la operación de un sistema. Por tanto, se refieren a todos los requisitos que no describen información a guardar, ni funciones a realizar, sino características de funcionamiento. Por esto, suelen denominarse atributos de calidad de un sistema. Los requisitos no funcionales, son las restricciones o condiciones que impone el cliente al programa que necesita, por ejemplo, el tiempo de entrega del programa, el lenguaje o la cantidad de usuarios [17].

ID	**DESCRIPCIÓN**
RQNF-01	El sistema sólo permitirá interactuar al usuario si dispone de acceso a la red.
RQNF-02	El sistema deberá funcionar en los principales navegadores.
RQNF-03	El sistema deberá funcionar en un ordenador con unos requisitos base bajos.

Tabla 2. Tabla que describe los requisitos no funcionales.

3.2.3 Requisitos de usabilidad

Los requisitos de usabilidad determinan que se va a entender por un nivel aceptable de utilización y aceptación del producto final por parte del usuario.

ID	**DESCRIPCIÓN**
RQU-01	El sistema deberá permitir al usuario cambiar los valores de los parámetros mediante scrolls.
RQU-02	El sistema deberá tener una interfaz fácil e intuitiva.
RQU-03	El sistema deberá mostrar los cambios de manera instantánea.
RQU-04	El sistema deberá permitir al usuario obtener los valores de cada punto de las gráficas al pasar el ratón por encima.

Tabla 3 Tabla que describe los requisitos de usabilidad.

3.3 ANÁLISIS

3.3.1 Casos de uso

En este apartado se especificarán los casos de uso del sistema. Los casos de uso indican como debería interactuar el sistema con el usuario o con otro sistema para lograr un objetivo específico. El conjunto de casos de uso representa todas las posibles interacciones que se describen en los requisitos [17].

3.3.1.1 CU-01. Cambiar valor de una hipótesis o de una medida política.

Nombre e ID del CU	CU-01. Cambiar valor de una hipótesis o de una medida política.
Actor	Usuario.
Descripción	El usuario cambia el valor que toma una hipótesis o una medida política.
Precondiciones	1. La aplicación está en proceso de ejecución y escuchando peticiones. 2. El usuario ha accedido a la dirección web donde está desplegada la aplicación.
Postcondiciones	1. El sistema muestra la evolución de los datos de los nuevos valores de la hipótesis y de las medidas políticas.
Flujo normal	1. El usuario modifica el valor de una hipótesis o de una medida política. 2. El sistema recoge el cambio realizado. 3. El sistema realiza una extrapolación para calcular los valores de la evolución. 4. El sistema devuelve visualmente la evolución de los valores.
Flujo alternativo	3.a Los valores de las hipótesis y las medidas políticas son continuos. El sistema debe buscar los valores en los datos origen.

Tabla 4. CU-01. Cambiar valor de una hipótesis o de una medida política.

3.3.1.2 CU-02. Cambiar idioma.

Nombre e ID del CU	CU-02. Cambiar idioma.
Actor	Usuario.
Descripción	El usuario cambia de idioma.
Precondiciones	1. La aplicación está en proceso de ejecución y escuchando peticiones. 2. El usuario ha accedido a la dirección web donde está desplegada la aplicación.
Postcondiciones	1. La aplicación cambia el idioma.
Flujo normal	1. El usuario solicita cambiar de idioma. 2. El sistema obtiene los literales del idioma seleccionado.
Flujo alternativo	

Tabla 5. CU-02. Cambiar idioma.

3.3.1.3 CU-03. Cambiar entre continuo y discreto.

Nombre e ID del CU	CU-03. Cambiar entre continuo y discreto.
Actor	Usuario.
Descripción	El usuario cambia la continuidad los valores que toman las respuestas de las hipótesis y medidas políticas.
Precondiciones	1. La aplicación está en proceso de ejecución y escuchando peticiones. 2. El usuario ha accedido a la dirección web donde está desplegada la aplicación.
Postcondiciones	1. La aplicación cambia el valor de las respuestas de continuo a discreto o viceversa.
Flujo normal	1. El usuario solicita cambiar la continuidad de los valores de las respuestas de las hipótesis y las medidas políticas. 2. El sistema cambia la continuidad de las respuestas.

	3. El sistema recalcula los valores gráficos a partir de las nuevas respuestas. 4. El sistema muestra las gráficas actualizadas. 5. El sistema muestra los valores de las respuestas actualizados.
Flujo alternativo	

Tabla 6. CU-03. Cambiar entre continuo y discreto.

3.3.1.4 CU-04. Resetear la aplicación.

Nombre e ID del CU	CU-04. Resetear la aplicación.
Actor	Usuario.
Descripción	El usuario devuelve la aplicación a su estado inicial.
Precondiciones	1. La aplicación está en proceso de ejecución y escuchando peticiones. 2. El usuario ha accedido a la dirección web donde está desplegada la aplicación.
Postcondiciones	1. La aplicación recupera sus valores iniciales.
Flujo normal	1. El usuario solicita resetear la aplicación. 2. El sistema vuelve a los valores iniciales de las respuestas. 3. El sistema vuelve a la configuración de continuidad de las repuestas inicial. 4. El sistema recalcula las gráficas. 5. El sistema muestra las gráficas actualizadas. 6. El sistema muestra los valores de las respuestas actualizados.
Flujo alternativo	

Tabla 7. CU-04. Resetear la aplicación.

3.3.1.5 CU-05. Visualización de valores concretos.

Nombre e ID del CU	CU-04. Visualización de valores concretos.
Actor	Usuario.
Descripción	El usuario quiere saber qué valor toma la gráfica en un punto concreto.
Precondiciones	1. La aplicación está en proceso de ejecución y escuchando peticiones. 2. El usuario ha accedido a la dirección web donde está desplegada la aplicación.
Postcondiciones	1. La aplicación muestra el valor para un punto concreto de la gráfica.
Flujo normal	1. El usuario sitúa el cursor encima del punto cuyo valor desea conocer. 2. El sistema devuelve el valor para el punto elegido.
Flujo alternativo	

Tabla 8. CU-05. Visualización de valores concretos.

Capítulo 4

4 PLANIFICACIÓN Y SEGUIMIENTO

En esta sección tratamos de planificar y organizar el tiempo de trabajo necesario para completar el proyecto. Podremos ver las distintas fases por las que ha pasado el desarrollo de la aplicación y el tiempo que se ha destinado a las mismas mediante diagramas de Gantt realizados con la aplicación Cacoo [18].

4.1 PLANIFICACIÓN INICIAL

La primera tarea de este proyecto fue realizar una planificación temporal del mismo. Para ello se desgloso el proyecto en diferentes fases y se les asignó una duración estimada. El proyecto se compone de las siguientes fases:

Análisis y requisitos. Esta primera etapa es esencial, ya que se establecen las bases del proyecto que vamos a desarrollar. En esta etapa se determinan necesidades y objetivos de la aplicación, así como los requisitos que esta tiene que cumplir.

Diseño del sistema. En esta etapa del proyecto se define la estructura de los elementos del proyecto. Además, se determinan las relaciones entre estos elementos. En esta fase también se realizan los bocetos de la interfaz de la aplicación.

Formación. Esta etapa sirve para estudiar el funcionamiento de las nuevas tecnologías utilizadas para desarrollar el proyecto.

Implementación. En la etapa de implementación vamos a distinguir dos fases:

> **Implementación de la aplicación.** En esta primera fase se desarrolla el código de la aplicación cliente-servidor, realizando las conexiones entre ambos, los datos y la interfaz gráfica.
>
> **Extrapolación**. En la segunda fase de la implementación se añade a la aplicación ya desarrollada el código para realizar la estimación de los valores respuesta para valores de las hipótesis y medidas políticas continuos.

Pruebas de implementación. En esta fase una vez implementado el código, se prueba y ejecuta para verificar su funcionamiento y comparar los resultados obtenidos con los objetivos iniciales.

Documentación. Esta fase se extiende a través de todo el proyecto, consiste en ir documentando los avances tanto en el propio código como en la memoria entregable.

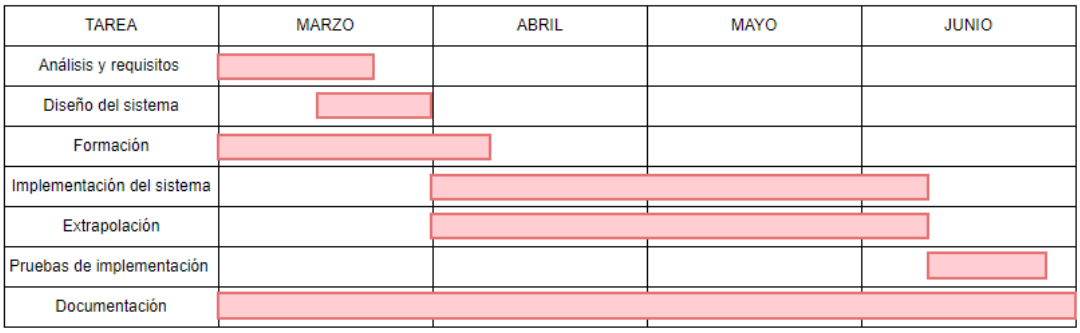

Ilustración 4. Diagrama de Gantt planificado.

En la Figura 4 podemos observar el diagrama de Gantt que describe la planificación inicial propuesta. Como se puede ver la fase que más tiempo abarca es la documentación, ya que se pensó su desarrollo de forma continuada, a lo largo del proyecto. Las siguientes fases que más tiempo se estimaban en durar son la implementación del sistema, así como la extrapolación que tiene lugar en el mismo, cuyo tiempo estimado supera los dos meses. Para el resto de las fases la estimación de tiempo oscila entre dos y cuatro semanas.

4.2 MODIFICACIÓN DE LA PLANIFICACIÓN INICIAL

Pese a haber realizado una estimación inicial lo más realista posible, se dieron una serie de situaciones que hicieron variar el horario inicial. Las principales situaciones que han retrasado la planificación son la realización de prácticas en empresa a jornada completa, la realización del TFG de Estadística y problemas con las tecnologías previstas inicialmente.

Es por todo ello que la planificación seguida se puede observar en el siguiente diagrama de Gantt.

Como se puede ver el inicio de la realización del TFG se retrasó hasta Enero de 2022 y se han aumentado los tiempos tanto de formación como de implementación del sistema. Además, se retrasó el inicio de la etapa de extrapolación hasta casi la

finalización de la etapa de implementación. En la siguiente Figura 5 podemos observar el diagrama de Gantt final seguido.

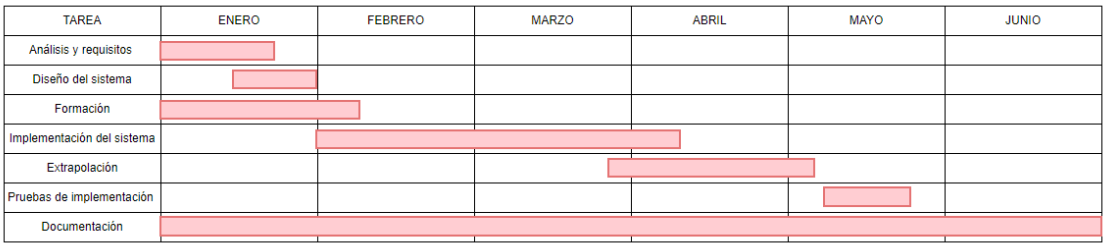

Ilustración 5. Diagrama de Gantt seguido.

4.3 PRESUPUESTO

En esta sección se realiza un análisis del presupuesto del proyecto. El coste de todo el proyecto se desglosa en la amortización de las máquinas de trabajo empleadas y las horas de trabajo de desarrollo. En este caso, no ha sido necesario adquirir licencias de software de pago. Así, el presupuesto se desglosa en:

Personal : se estima el sueldo de un ingeniero informático junior en unos 20.000€ brutos anuales. Trabajando a jornada completa (8 horas diarias) y suponiendo unos 250 días laborales al año aproximadamente, se obtiene un coste del desarrollador de 10€ la hora.

Personal	**Coste/hora ()**	**Horas**	**Total**
Alumno	10	300	3000
Total			3000

Tabla 9. Tabla del presupuesto del personal.

Hardware : el equipo usado para este proyecto es un portátil HP de 16 GB de RAM, procesador Intel Core i5. Tuvo un coste de 700€. Suponemos una vida útil de 3 años por cada componente y 6 meses de uso dedicados para este proyecto.

Hardware	Precio	Total
Ordenador HP	700/6	117
Ratón HP	20/6	3,4
Total		120,4

Tabla 10. Tabla del presupuesto del Hardware.

Para el desarrollo de este TFG se han usado programas gratuitos, por lo que el costo en software es de 0€.

Actividad	Coste
Horas de trabajo personal	3000
Hardware	120,4
Software	0
Total	3120,4

Tabla 11. Tabla del presupuesto total.

En total se ha estimado un presupuesto de 3120,4€ para el desarrollo de este TFG.

Capítulo 5

5 DISEÑO

Una vez finalizado el proceso de análisis de la aplicación, el siguiente paso es realizar su diseño. En esta etapa se definen los diagramas de secuencia, la arquitectura del sistema y el diagrama de despliegue que muestra como quedan distribuidos los diferentes subsistemas en la aplicación. Además, se elaboran los bocetos de la interfaz gráfica.
El diseño de software es la actividad donde se identifican los componentes del software y sus relaciones, en base a los requisitos de un cliente [17].

5.1 DIAGRAMA DE CASOS DE USO

La primera etapa en cualquier proceso de diseño de software es desarrollar la comprensión de las relaciones entre el software que se diseñará y su ambiente externo. Esto es esencial para decidir cómo proporcionar la funcionalidad requerida del sistema y cómo estructurar el sistema para que se comunique con su entorno [17]. Los diagramas de casos de uso muestran la relación entre un actor y sus requisitos o expectativas del sistema, sin representar las acciones que tienen lugar o ponerlas en un orden lógico [19].

En la Figura 6 se muestra el diagrama de casos de uso del sistema.

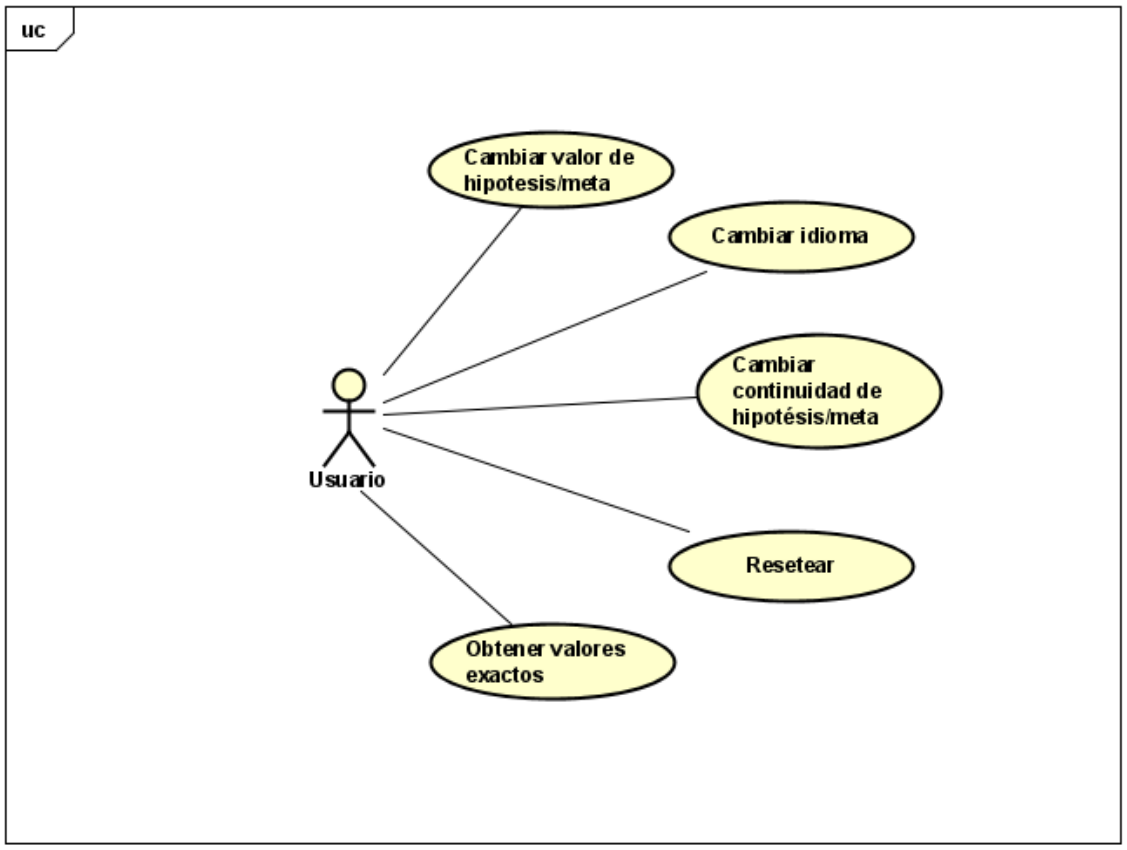

Ilustración 6. Diagrama de casos de uso del sistema.

5.2 DIAGRAMAS DE SECUENCIA

En la siguiente subsección se presentan los diagramas de secuencia correspondientes a los casos de uso de la sección 3.3.1.

Un diagrama de secuencia es un tipo de diagrama de interacción porque describe cómo y en qué orden un grupo de objetos funcionan en conjunto [20]. Estos diagramas son el puente entre los requisitos y la implementación de un sistema[17] .

5.2.1 Cambiar valor de una hipótesis o de una medida política.

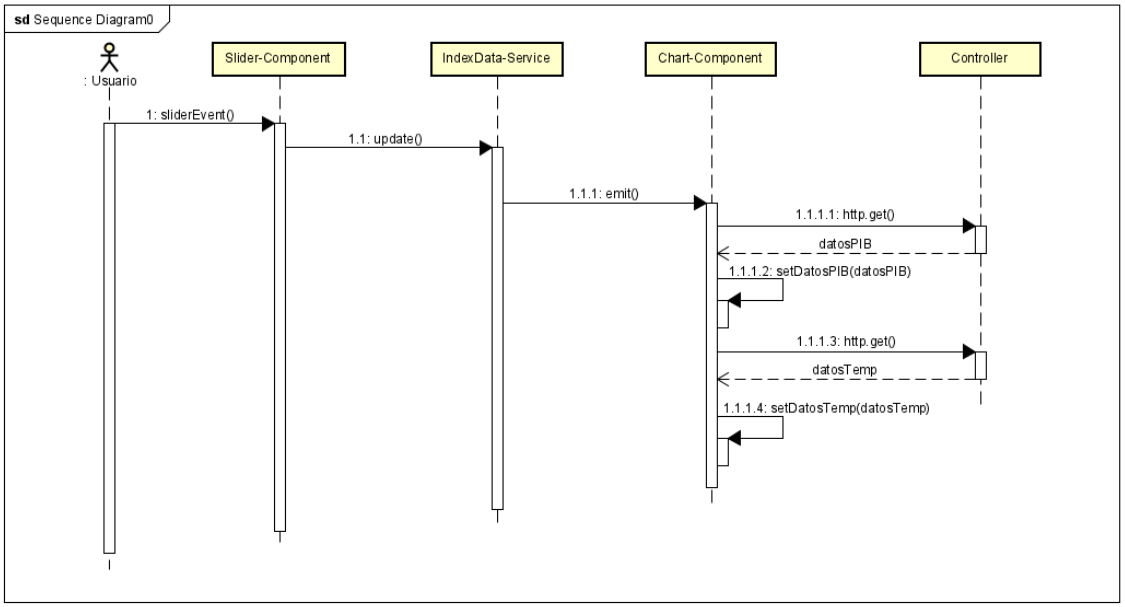

Ilustración 7. Diagrama de secuencia del caso de uso Cambiar el valor de una hipótesis o medida politica.

5.2.2 Cambiar idioma

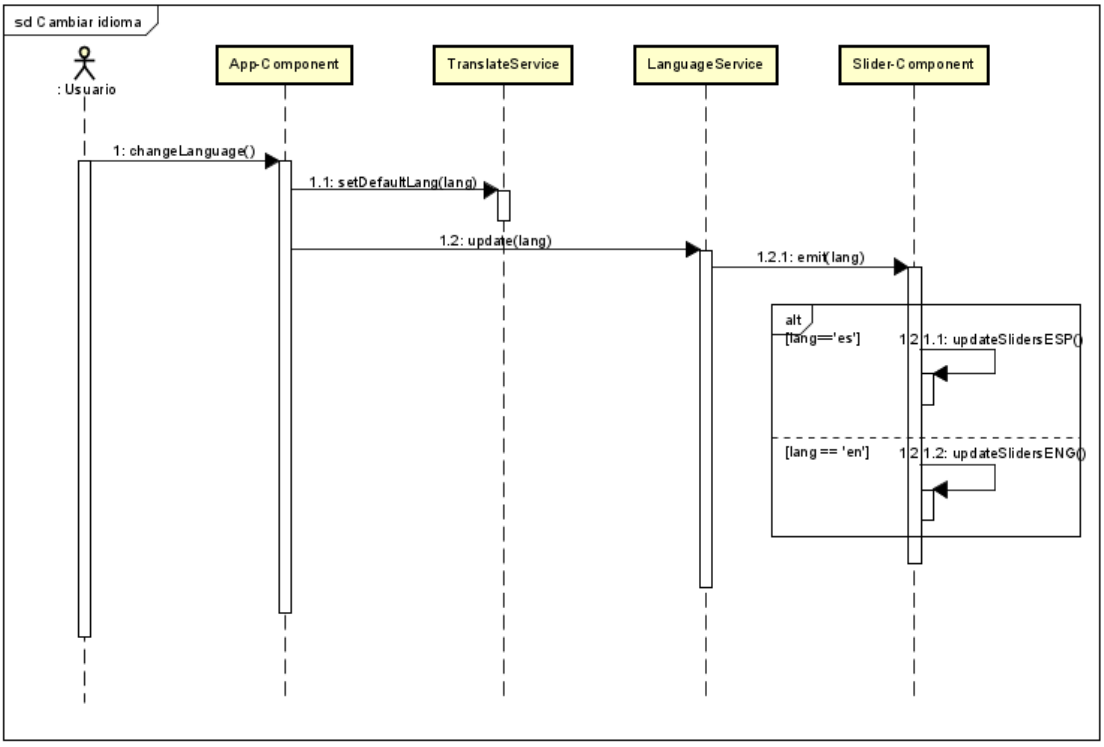

Ilustración 8. Diagrama de secuencia del caso de uso Cambiar idioma.

5.2.3 Resetear

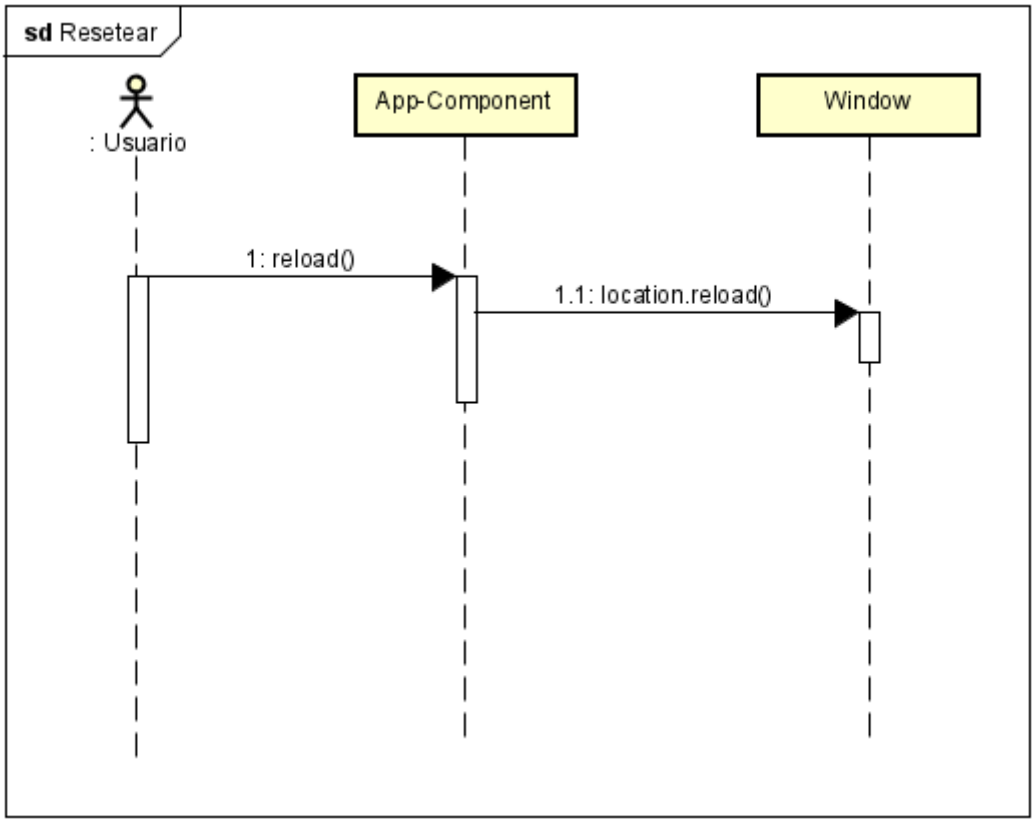

Ilustración 9. Diagrama de secuencia del caso de uso Resetar.

5.2.4 Cambiar la continuidad de hipótesis/medida política

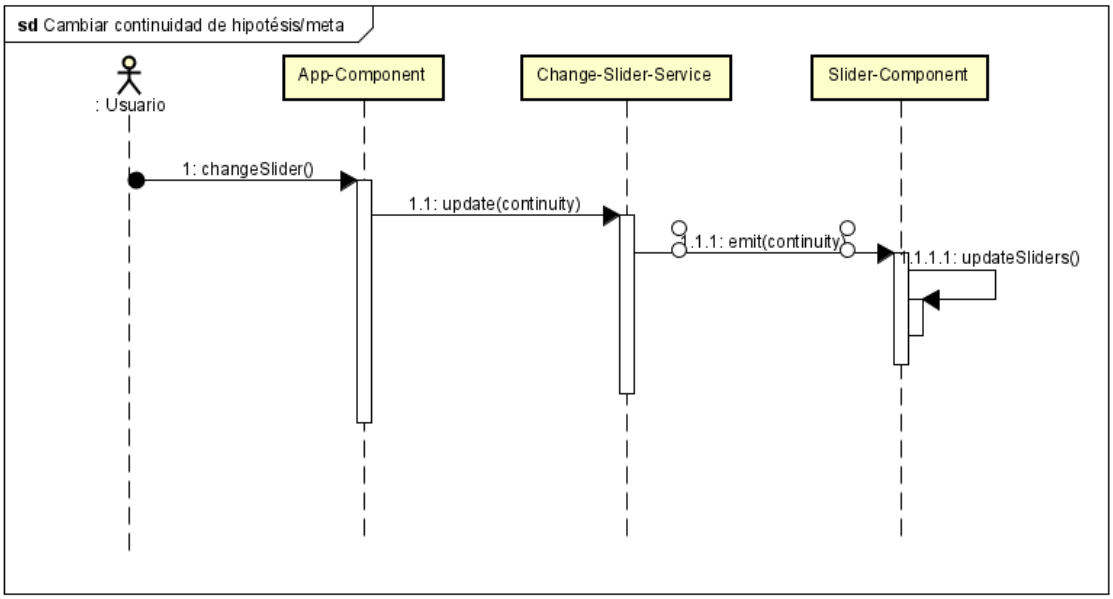

Ilustración 10. Diagrama de secuencia del caso de uso Cambiar la continuidad de una hipotesis/medida política.

5.2.5 Visualización de valores concretos

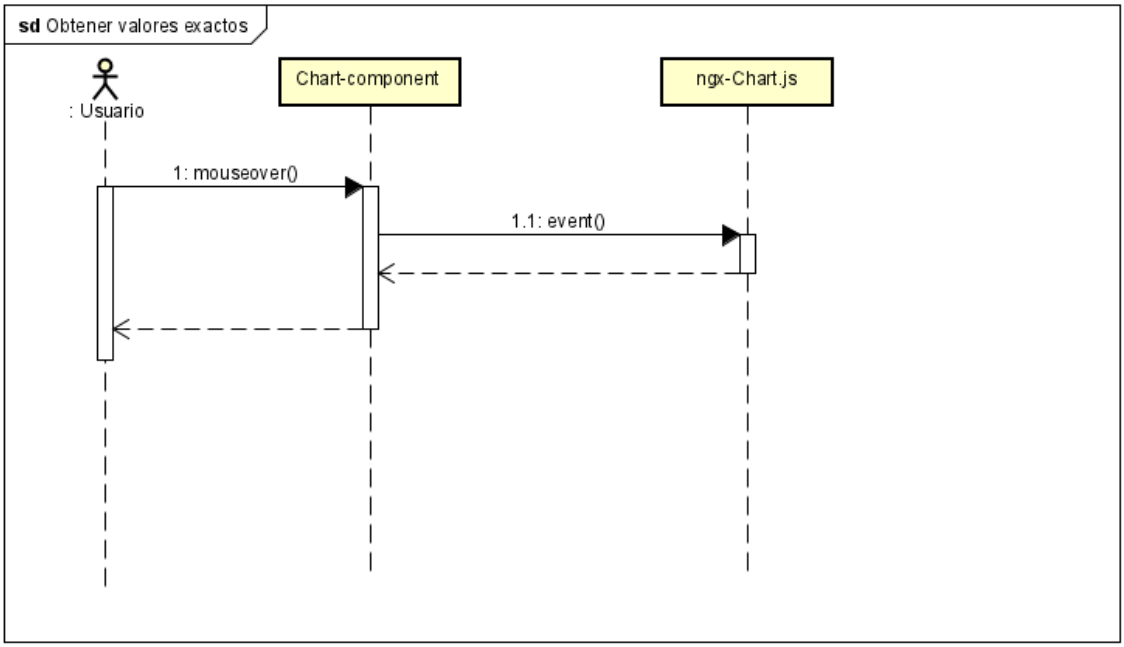

Ilustración 11. Diagrama de secuencia del caso de uso Visualización de valores concretos.

5.3 DISEÑO DE LA ARQUITECTURA LÓGICA

El diseño arquitectónico se interesa por entender cómo debe organizarse un sistema y cómo tiene que diseñarse la estructura global de ese sistema. Es el enlace crucial entre el diseño y los requisitos, ya que identifica los principales componentes estructurales en un sistema y la relación entre ellos. La salida del proceso de diseño arquitectónico consiste en un modelo arquitectónico que describe la forma en que se organiza el sistema como un conjunto de componentes en comunicación [17]. Los diagramas de arquitectura lógica nos muestran los componentes del software y las relaciones existentes entre ellos.

En la Ilustración 12 se muestra la arquitectura lógica definida para la aplicación. Se componen de un cliente Angular y un servidor Springboot. La comunicación se realiza a través de peticiones HTTP a una API REST.

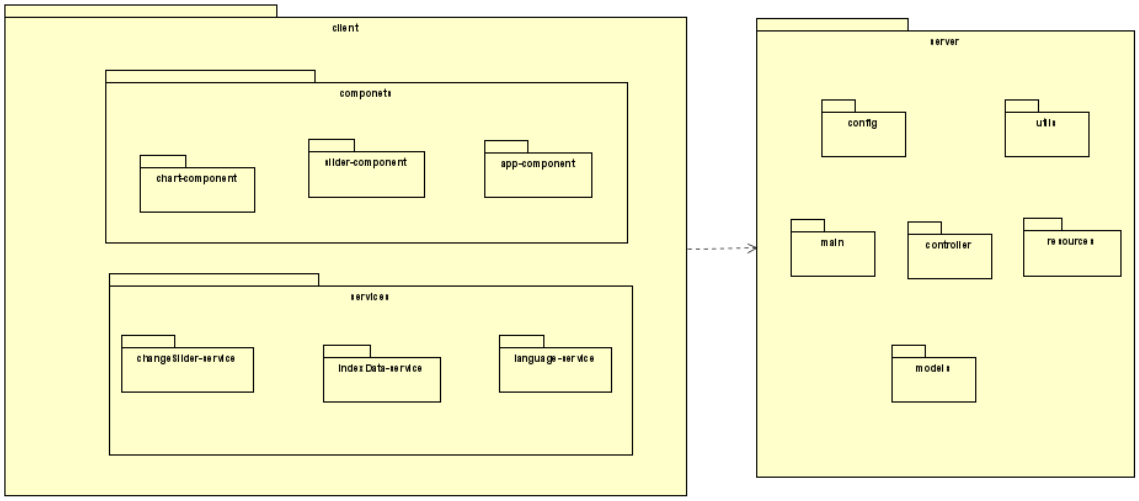

Ilustración 12. Diagrama arquitectura lógica de la aplicación.

En la Ilustración 13 se muestra la arquitectura lógica definida para el servidor.

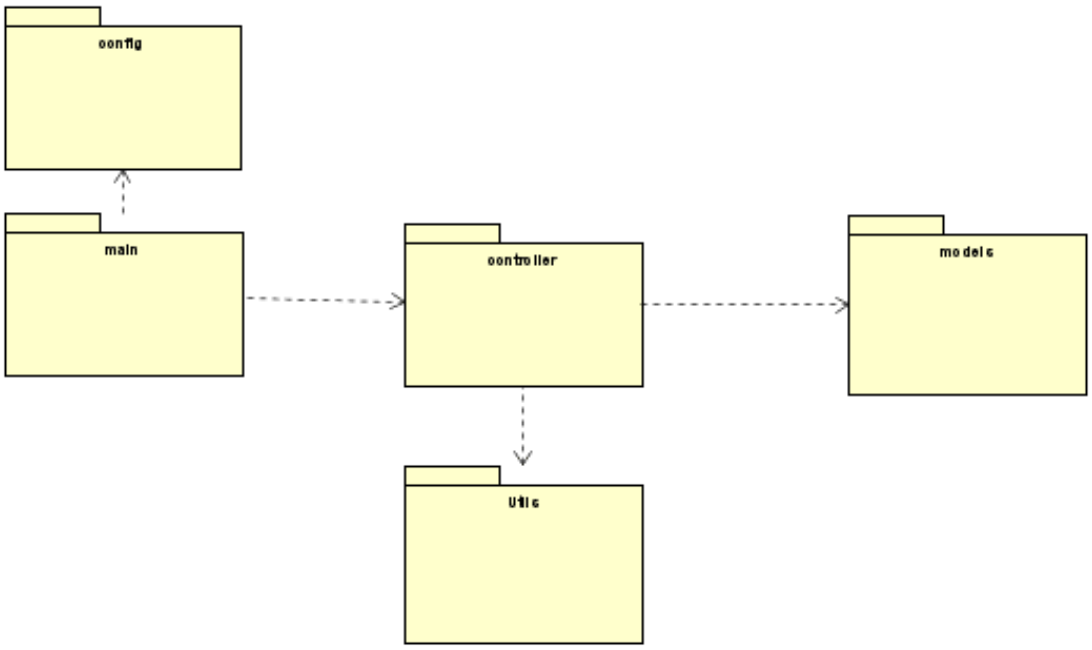

Ilustración 13. Diagrama de arquitectura del servidor.

En la Ilustración 14, se muestra el diagrama de arquitectura lógica detallado.

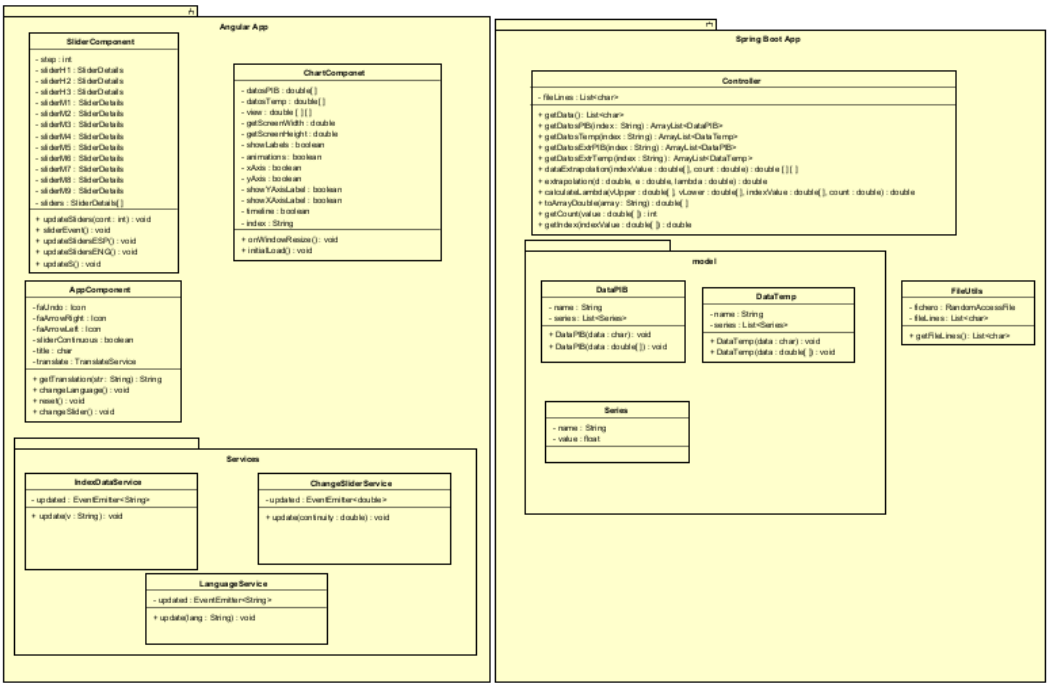

Ilustración 14. Diagrama de arquitectura lógica detallado.

5.4 DISEÑO DE DESPLIEGUE

Los diagramas de despliegue describen la arquitectura de ejecución de un sistema incluyendo los nodos como entornos de ejecución de hardware software y el middleware que los conecta. Sirven para entender cómo el sistema se desplegará físicamente en el hardware [21].

En la Figura 15 se muestra el diagrama de despliegue de la aplicación. La arquitectura de esta es cliente-servidor.

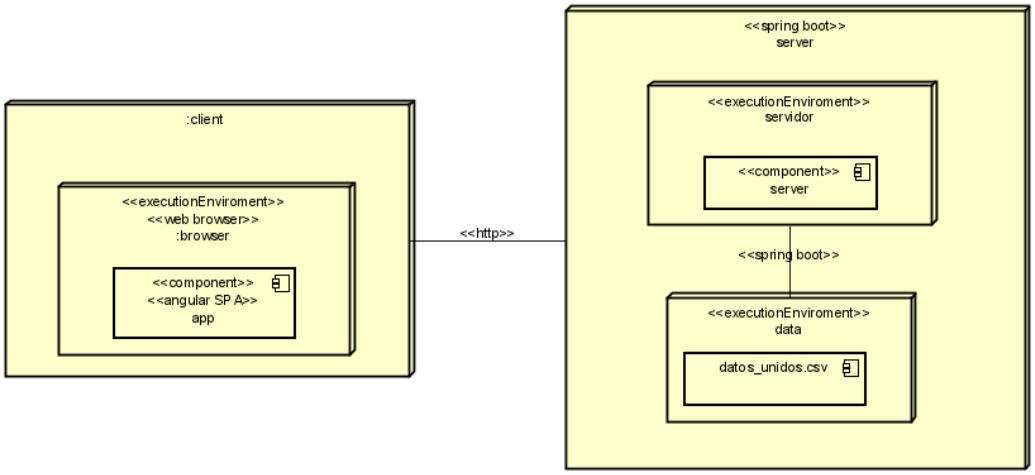

Ilustración 15. Diagrama de despligue.

5.5 BOCETOS DE LA INTERFAZ

En esta sección se presentan los bocetos de la interfaz de usuario. En el primer boceto propuesto, los datos del PIB y de la temperatura se mostraban en una misma gráfica, como se puede ver en la imagen 16.

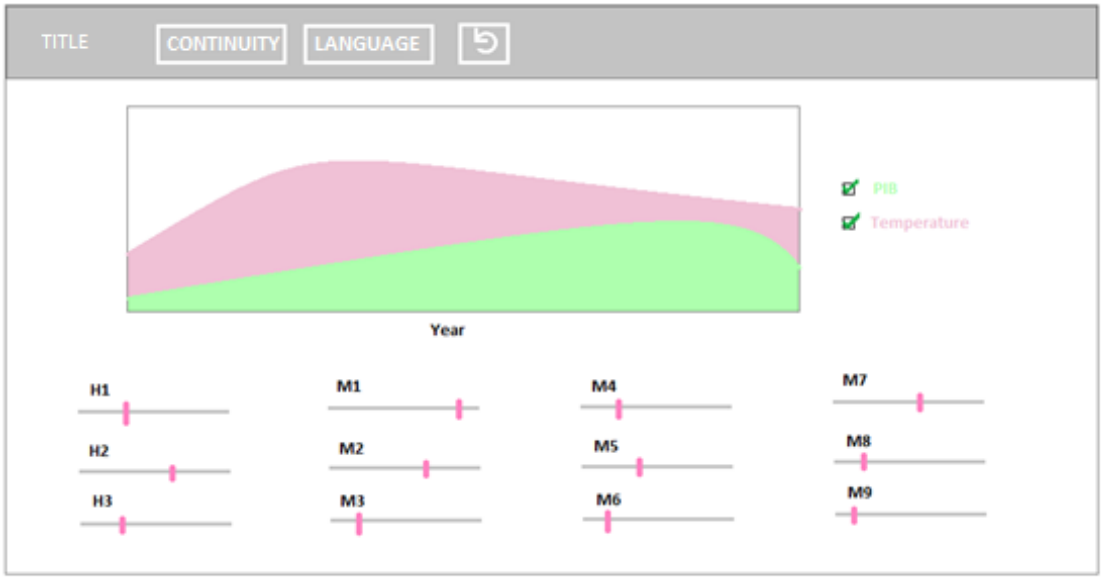

Ilustración 16. Ilustración del primer boceto de la interfaz.

Sin embargo, debido a las diferentes magnitudes de los datos, el boceto final los muestra en gráficas separadas, como se puede ver en la imagen 17. Al representar una característica por gráfica se decidió cambiar las gráficas a gráficos de líneas en vez de gráfico de área apilada. En ambos bocetos los sliders se mantienen iguales, separando los que se refieren a hipótesis los tres sliders de la izquierda, de las medidas políticas, los nueve sliders restantes. En cuanto a la barra de menú superior desde el inicio se planteó poder cambiar el idioma de la aplicación entre inglés y español , poder resetear las gráficas a los valores iniciales y poder cambiar entre valores continuos y discretos para las hipótesis y medidas políticas.

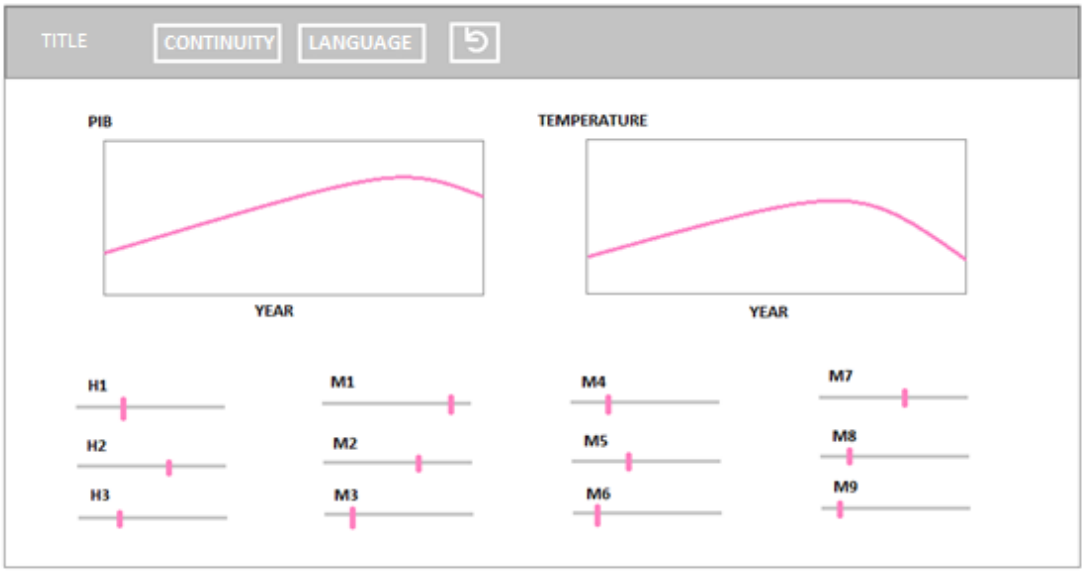

Ilustración 17. Ilustración del segundo boceto de la interfaz.

Capítulo 6

6 IMPLEMENTACIÓN

La etapa de implementación de desarrollo del software corresponde al proceso de convertir una especificación del sistema en un sistema ejecutable [17]. En esta sección se describen las tecnologías exploradas, los patrones utilizados y la API Rest. Además, se muestra la organización que sigue el código del proyecto.

6.1 TECNOLOGÍAS EXPLORADAS

En este apartado introduciremos las tecnologías que han sido consideradas para este proyecto. En principio, la aplicación se iba a desarrollar con tecnología Stack Mean. Sin embargo, creímos conveniente usar Spring Boot para una mayor agilidad en el manejo de los datos.

Stack Mean es un conjunto de tecnologías basadas en JavaScript para desarrollar el proyecto completo, tanto el servidor como el cliente. Está compuesto por cuatro de las más importantes tecnologías de la industria: MongoDB, Express, Angular y Node.js [22].

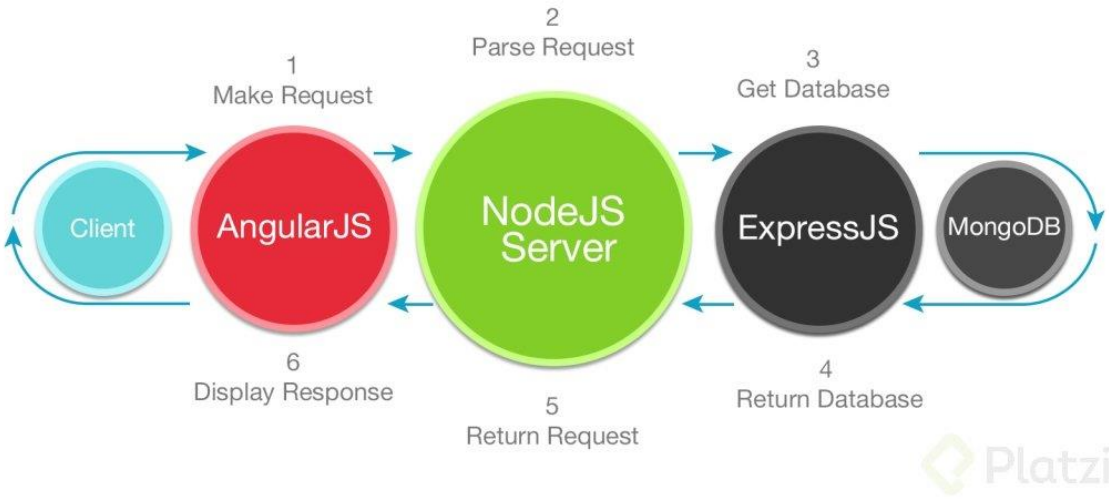

Ilustración 18: Imagen que representa la relación entre las diferentes tecnologías que forman el stack mean.

6.1.1 MongoDB

MongoDB es una base de datos NoSQL. Esta base de datos plantea modelos de datos más específicos y esquemas más flexibles que se adapten a los requisitos propuestos. Es una base de datos orientada a documentos en los que cada registro o conjunto de datos se almacena en documentos que funcionan como unidad autónoma de información. A su vez, los documentos se pueden agrupar en colecciones.

Los documentos están almacenados en BSON (Binary JSON) una versión de JSON que permite hacer búsquedas más rápidas [23].

En esta base no es necesario recorrer todas las columnas para realizar una consulta. Por el contrario, se asigna un identificador único a cada documento para que al hacer la consulta se compruebe este mismo documento. El identificador puede ser de distintos tipos [24].

6.1.2 Node

Para la implementación inicial del servidor de esta aplicación se decidió usar Node. Node es un entorno de ejecución para JavaScript construido sobre el motor de JavaScript V8 de Google Chrome.

Como tal, el entorno omite las APIs de JavaScript específicas del explorador web y añade soporte para APIs de sistema operativo más tradicionales que incluyen HTTP y bibliotecas de sistemas de ficheros [25].

6.1.3 Express

Express es el framework web más popular de *Node*. Aunque es bastante minimalista, Express dispone de paquetes de middleware compatibles para abordar casi cualquier problema de desarrollo web y múltiples librerías que permiten trabajar con cookies, parámetros URL, etc … [26].

6.1.4 Spring Boot

Para la implementación del servidor de la aplicación finalmente se utilizó Spring Boot. Spring Boot es una herramienta que nos permite crear un proyecto Spring de forma más sencilla, eliminando ciertas configuraciones repetitivas requeridas.

Spring Framework es un Framework que permite crear aplicaciones Java. Su principal característica es la inyección de dependencias, Spring está dividido en diversos módulos como se puede observar en la Figura 19. Estos son opcionales pudiendo utilizar solo aquellos que sean necesarios. Esto implica además menos código acoplado.

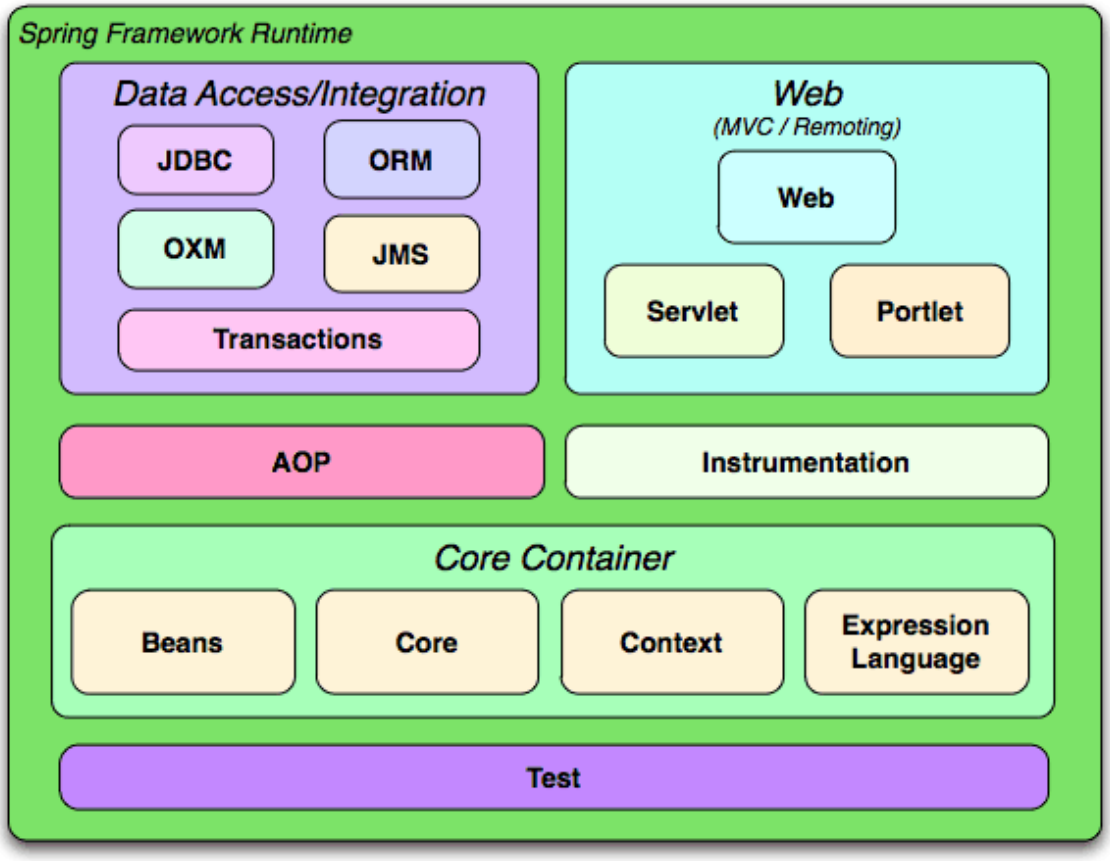

Ilustración 19. Ilustración de los diferentes módulos de Spring.

Spring Boot simplifica Spring Framework gracias a:

Contenedor de aplicaciones integrado: permite compilar las aplicaciones Web como un .jar que puede ser ejecutado como una aplicación Java.

Starters: nos proporciona una serie de dependencias, starters, que pueden ser añadidos al proyecto dependiendo de lo que necesitemos. Estos starters están configurados por defecto minimizando así la necesidad de configuración para desarrollar [27] [28] [29].

6.1.5 Maven

Maven es una herramienta que permite construir, administrar cualquier proyecto basado en Java, generar información de compilación e iniciar la aplicación Spring Boot antes de ejecutar las pruebas de integración [30].

6.1.6 Angular 12

Angular es un Framework de JavaScript de código abierto escrito en TypeScript. Su objetivo principal es desarrollar aplicaciones SPA (Single Page Application). Las aplicaciones SPA son un tipo de aplicación web donde todas las pantallas se muestran en la misma página, sin tener que recargar el navegador. En estas aplicaciones únicamente se tiene un punto de entrada, generalmente un index.html.

Angular está basado en componentes. Un componente es una pieza de código preelaborado que encapsula alguna funcionalidad. La idea es que un componente debe ser independiente para ser reutilizable, sin importar los componentes que le referencian. Un componente en Angular está formado por tres partes: un template, una clase y una función decoradora.

Angular además, usa una colección de bibliotecas bien integradas que cubren una amplia variedad de características, que incluyen enrutamiento, administración de formularios, comunicación cliente-servidor y más. Un conjunto de herramientas para desarrolladores que permiten desarrollar, compilar, probar y actualizar el código fuente de la aplicación.

6.1.6.1 ngx-chart.js

Ngx-Charts es la librería que se utiliza para la representación de gráficos. Utiliza Angular para renderizar y animar los elementos SVG. Además, utiliza d3 para las funciones matemáticas, ejes, escalas, generadores de forma, … [31] [32].

6.1.6.2 ngx-sliders.js

Ngx-Slider es la librería que se utiliza para la representación de los sliders en este proyecto. Nos permite crear los sliders así como cambiar entre sliders discretos y continuos. Es una librería sin dependencias basada angularjs-slider [33].

6.1.6.3 ngx-translate.js

Es una librería de internacionalización para angular. Nos permite definir traducciones para nuestro contenido en diferentes idiomas y cambiar entre ellos de forma sencilla. Esta librería es modular, permite cambiar fácilmente cualquier parte con una implementación personalizada en caso de que no exista ninguna que sirva [34].

6.1.7 Postman

Postman es una aplicación que nos permite realizar pruebas API. Es un cliente HTTP que nos da la posibilidad de testear 'HTTP requests' a través de una interfaz gráfica de usuario, por medio de la cual obtendremos diferentes tipos de respuesta que posteriormente deberán ser validados [35].

6.1.8 Visual Studio

Visual Studio es un conjunto de herramientas y otras tecnologías de desarrollo que proporciona servicios integrales para facilitar la creación de software, permitiéndonos desarrollar aplicaciones, sitios y aplicaciones web, así como servicios web en cualquier entorno que soporte la plataforma .NET, algunos lenguajes que podemos encontrar son: Visual Basic, Visual C# y Visual C++ [36].

6.1.9 Astah

Astah es una herramienta de modelado y diseño de sistemas que soporta UML, Diagrama de Relación de Entidades, Diagramas de flujo, CRUD, Diagrama de flujo de datos, Tabla de Requisiciones y Mapas Mentales [37].

6.2 DECISIONES FINALES

Inicialmente la tecnología planteada para desarrollar el proyecto fue Stack Mean. Stack Mean nos permite desarrollar una aplicación SPA, cuyo cliente y servidor usen el mismo lenguaje. Además, MongoDB nos permitiría hacer búsquedas más rápidas que otras bases de datos SQL. Sin embargo, la velocidad de respuesta de MongoDB era insuficiente. Fue por ello que decidimos cambiar de tecnologías y dejar de acceder a una basa de datos MongoDB para mejorar los tiempos. Finalmente se decidió mantener el cliente de la aplicación que ya estaba bastante desarrollado y cambiar el servidor para poder acceder de forma más eficiente al fichero de datos. Con estos cambios hemos conseguido uno de los principales objetivos de la aplicación que era la velocidad de respuesta al cambiar algún valor de alguna hipótesis o medida política.

6.3 PATRONES

En este apartado se definen los patrones aplicados para la implementación de la aplicación. Un patrón de diseño describe un problema que ocurre una y otra vez en nuestro entorno, así como la solución a ese problema, de tal modo que se pueda aplicar esta solución un millón de veces sin hacer lo mismo dos veces [38], es decir, es una solución general, reutilizable y aplicable a diferentes problemas de diseño software [39].

6.3.1 MVVM

Este patrón permite separar la vista de la lógica de negocio, incluyendo un componente intermedio para la comunicación entre la vista y el modelo.

Hay tres componentes principales en el patrón MVVM: el modelo, la vista y el modelo de vista. Cada uno tiene un propósito distinto. En la imagen 20 se muestran las relaciones entre los tres componentes.

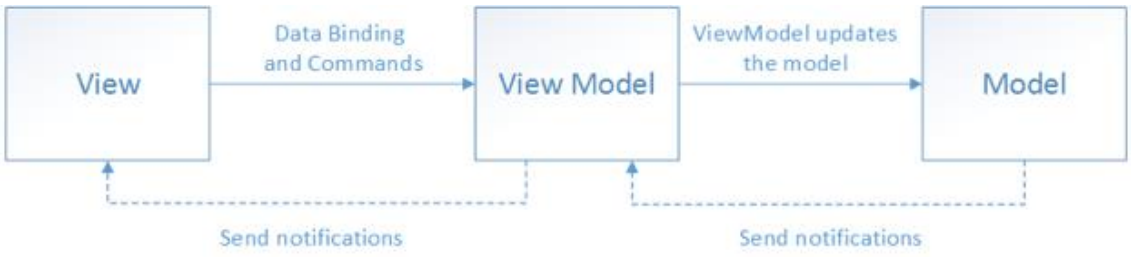

Ilustración 20. Ilustración del patrón MVVM.

Modelo

Representa la capa de datos y/o la lógica de negocio. El modelo contiene la información, pero nunca las acciones o servicios que la manipulan. No tiene dependencia con la vista.

En Angular el modelo está representado por clases de dominio, interfaces o incluso hasta los servicios inyectables mediante inyección de dependencias.

Vista

Se encarga de mostrar la información al usuario. Las vistas en MVVM son activas, contienen comportamientos, eventos y enlaces a datos que, en cierta manera, necesitan tener conocimiento del modelo subyacente.

En Angular, este elemento se corresponde con el fichero plantilla (Ficheros .html y .css) de los componentes.

Modelo de vista

El ViewModel (modelo de vista) es el intermediario entre el modelo y la vista, contiene toda la lógica de presentación y se comporta como una abstracción de la interfaz. La comunicación entre la vista y el viewmodel se realiza por medio los enlaces de datos (binders).
En Angular el ViewModel corresponde con la parte TypeScript de los componentes [40].

6.3.2 Publish-subscribe

El patrón publish-subscribe es un patrón que permite crear módulos que se puedan comunicar entre ellos pero sin depender directamente el uno del otro. [41]

En este patrón un conjunto de objetos se suscribe a un evento y otro tipo de objetos publican datos bajo un evento [42].

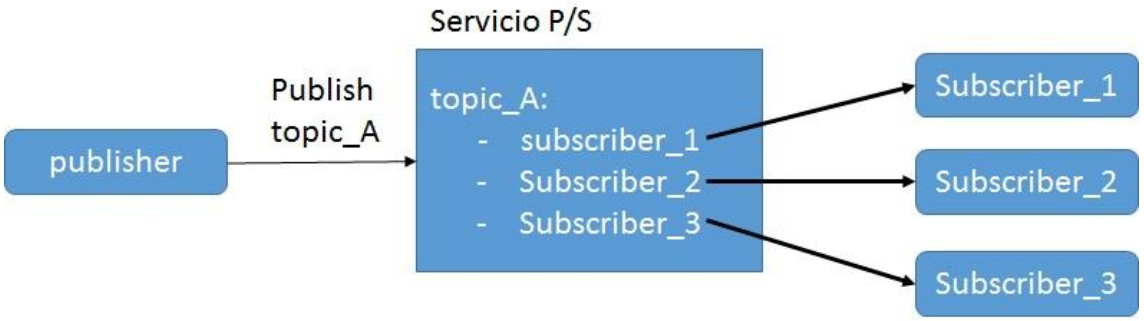

Ilustración 21. Ilustración del patrón Public-Subscribe.

En esta aplicación este patrón se utiliza para trasladar el cambio de los sliders a las gráficas, para trasladar el cambio de valor continuo-discreto de los sliders y para cambiar el idioma de los sliders.

Al realizar un cambio en una medida mediante un slider el componente traslada al servicio los valores de todos los sliders actualizados.

```
sliderEvent() {
    let slidersValue = [this.sliderM9.value, this.sliderM8.value,
                        this.sliderM7.value, this.sliderM6.value,
                        this.sliderM5.value, this.sliderM4.value,
                        this.sliderM3.value, this.sliderM2.value,
                        this.sliderM1.value, this.sliderH3.value,
                        this.sliderH2.value, this.sliderH1.value]
    this.serv.update(slidersValue)
  }
```

En el servicio se emiten estos valores para que lleguen a quien este suscrito.

```
 update(v: number[]) {

    this.updated.emit(v);

  }
```

El componente suscrito es chart-component, el cual una vez tiene los valores de los sliders hace una petición http para obtener los valores que representará en las gráficas.

```
this.serv.updated.subscribe((index: String) => {

      this.http.get("http://localhost:8080/datosExtrPIB/"+index,
               { responseType: 'json' }).subscribe((resp: any) =>
             {
          this.datosPIB = resp;
        },
      (error: any) => {
```

```
      console.log(error)
    }
  )
    this.http.get("http://localhost:8080/datosExtrTemp/"+index,
    { responseType: 'json' }).subscribe((resp: any) =>
    {
       this.datosTemp = resp;
    },
    (error: any) => {
      console.log(error)
    }
 )
});
```

También se utiliza este patrón para cambiar entre continuo y discreto los valores de las hipótesis y medidas políticas. Cuando un usuario cambia entre continuo y discreto, la clase app.components.ts emite al servicio el nuevo valor del step de los sliders.

```
changeSlider(){
    this.sliderContinuous= !this.sliderContinuous;
    if(this.sliderContinuous){
      this.serv.update(0.01);
    }else {
    this.serv.update(1);
    }
  }
```

La clase del servicio emite el valor para las componentes suscritas.

```
 update(v: number[]) {

    this.updated.emit(v);

  }
```

El componente suscrito es slider-component el cual cambia el step de sus sliders, actualiza los sliders y en caso de pasar de continuo a discreto recalcula el valor de las gráficas.

```
this.updateSliders(this.step);
    this.servChange.updated.subscribe((cont: number) => {
      this.step = cont;
      this.updateSliders(cont);
      if (cont == 1) {
        this.sliderEvent();
      }
```

```
      this.ngOnInit();
    })
```

Por último, este patrón se utiliza para realizar el cambio de idioma en los sliders. Cuando un usuario cambia el idioma, la clase app.components.ts emite al servicio LanguageService el nuevo valor del idioma, siendo 'es' si es español y 'en' para cambiar a inglés.

```
changeLanguage( ){
   if (this.translate.getDefaultLang()=='en'){
     this.translate.use('es');
     this.translate.setDefaultLang('es');
     this.lanServ.update('es');

   }
   else{
     this.translate.use('en');
     this.translate.setDefaultLang('en');
     this.lanServ.update('en');

   }
 }
```

La clase del servicio emite el valor para las componentes suscritas.

```
 update(v:String) {
    this.updated.emit(v);
 }
```

El componente suscrito es slider.component.ts el cual cambia el idioma de sus sliders y los actualiza.

```
this.lanServ.updated.subscribe((lan: String) => {
   if(lan=='es'){
     this.updateSlidersESP(this.step);
   }else {
     this.updateSlidersENG(this.step);
}
```

6.4 ESTRUCTURA API REST

Una API de REST, o API de RESTful, es una interfaz de programación de aplicaciones (API o API web) que se ajusta a los límites de la arquitectura REST y permite la interacción con los servicios web de RESTful [43].

Las API son definiciones, funciones y protocolos informáticos para diseñar e integrar el software de las aplicaciones.

Estas permiten interactuar con una computadora o un sistema para obtener datos o ejecutar una función, de manera que el sistema comprenda la solicitud y la cumpla.

REST no es un protocolo ni un estándar, sino más bien un conjunto de límites de arquitectura. Para que una API se considere REST debe cumplir ciertos criterios como:

Arquitectura cliente-servidor con la gestión de solicitudes con HTTP.

Comunicación sin estado. Cada petición al servidor deberá ser independiente.

Cacheable. Los datos pueden almacenarse en caché para optimizar las interacciones cliente servidor.

Interfaz uniforme. Define una interfaz genérica para que la comunicación se manera uniforme.

Sistema de capas. El servidor puede disponer de varias capas de implementación [43].

En este trabajo se realizan llamadas http a una API Rest desde chart.component.ts para obtener los datos que se representarán en este componente. Estas llamadas le pasan al servidor el valor de los sliders, el conjunto respuesta.

```
this.serv.updated.subscribe((index: String) => {

 this.http.get("http://localhost:8080/datosExtrPIB/"+index, {
 responseType: 'json' }).subscribe((resp: any) =>
 {
      this.datosPIB = resp;
  },
  (error: any) => {
      console.log(error)
  }
)
 this.http.get("http://localhost:8080/datosExtrTemp/"+index, {
   responseType: 'json' }).subscribe((resp: any) =>
```

```
{
      this.datosTemp = resp;
},
(error: any) => {
      console.log(error)
 }
)
});
```

El servidor SpringBoot recibirá los valores de los sliders con los que podrá devolver los datos a representar o bien extrayéndolos directamente del data set o mediante una extrapolación.

```
@GetMapping("/datosExtrPIB/{index}")
   public ArrayList<DataPIB> getdatosRegPIB(@PathVariable("index")
    String index) throws IOException {
       double[] indexValue = new double[12];
       for (int i = 0; i < 12; i++) {
           indexValue[i] = Double.parseDouble(index.split("-")[i]);
       }
       if (getCount(indexValue) == 0) {
           int indexPIB = (int) getIndex(indexValue);
           ArrayList<DataPIB> dataPIB = new ArrayList<DataPIB>();
           DataPIB dataPIB1 = new DataPIB(fileLines.get(indexPIB));
           dataPIB.add(dataPIB1);
           return dataPIB;
       } else {
           double[] datPIB = dataExtrapolation(indexValue,
            getCount(indexValue))[0];
           ArrayList<DataPIB> dataPIB = new ArrayList<DataPIB>();
           DataPIB dataPIB1 = new DataPIB(datPIB);
           dataPIB.add(dataPIB1);
           return dataPIB;
       }
   }
```

6.5 ORGANIZACIÓN DEL CÓDIGO

El proyecto se encuentra distribuido en diferentes directorios, donde se alojan los ficheros de la aplicación. Por un lado, tenemos el servidor:

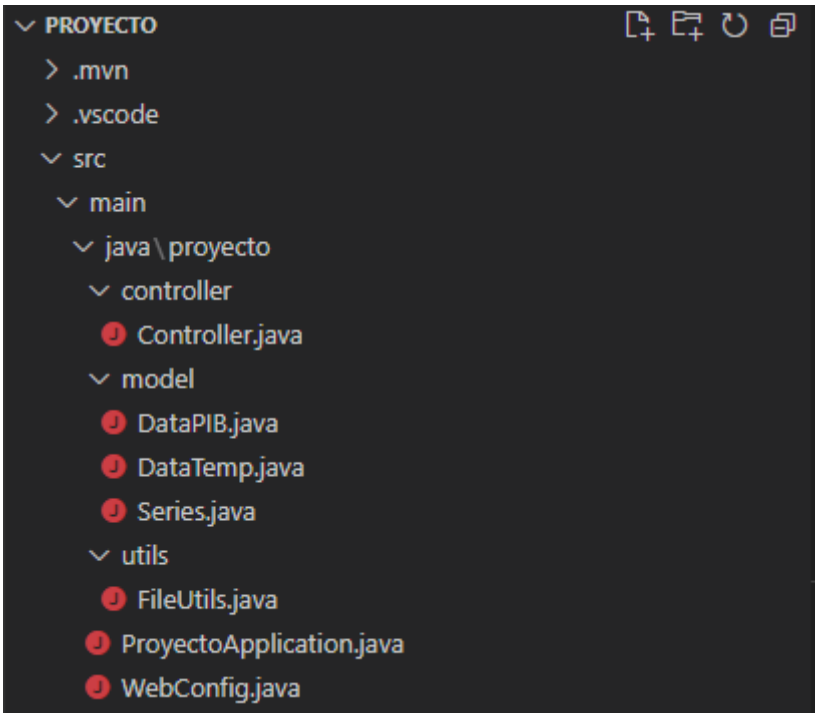

Ilustración 22. Ilustración de la organización del código del servidor.

En el servidor podemos encontrar las siguientes carpetas:

Controller. Es responsable de procesar las solicitudes de API REST entrantes, preparar los datos y devolverlos como respuesta [44].

Models. Donde se define la estructura de datos de una carga.

Utils. Donde el FileUtils carga los datos al inicio de la ejecución de la aplicación.

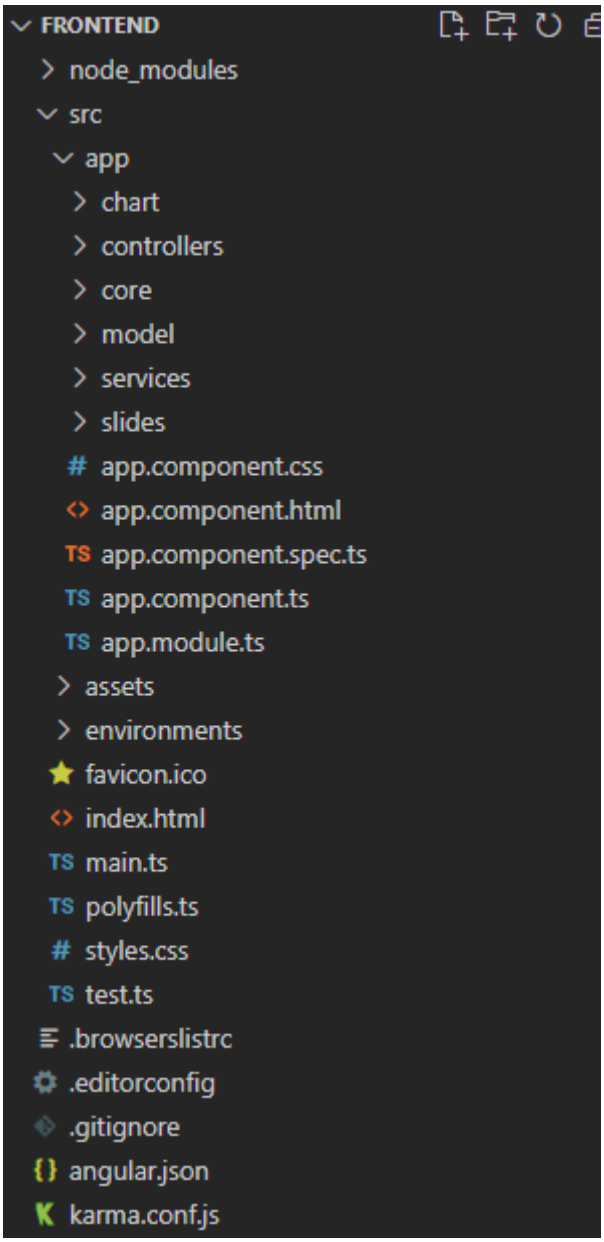

Ilustración 23. Ilustración de la organización del código del cliente.

En la aplicación Angular encontramos las siguientes carpetas:

Chart. Donde se encuentra el componente con la que dibujamos gráficos. Esta recibe del servidor los datos que tiene que representar.

Services. En esta carpeta se encuentran los servicios.

IndexService. Servicio que se usa para calcular los valores a dibujar. Recibe de la componente slides los valores de los sliders y los publica para que el componente Chart llamando al servidor le devuelva los valores a graficar.

SliderService. Servicio que nos permite cambiar las hipótesis y medidas políticas entre valores continuos y discretos. Recibe los datos de app-

component y los publica para que el componente slides cambie los sliders de discretos a continuos o viceversa.

LanguageService. Servicio que nos permite cambiar el idioma de los sliders. Recibe los datos de app-component y los publica para que el slides-component cambie el idioma.

AppComponent, la componente principal donde se inicia la aplicación y se llama al resto de componentes.

Slides. Esta carpeta es para el componente de los sliders. Se encarga de representar los sliders en el Dashboard así como transmitir el cambio de cualquier valor de ellos a través de un servicio.

Assets. En esta carpeta se encuentra i18n, que se encarga de parte de la traducción de la aplicación cuando cambiamos de idioma. En esta carpeta tenemos dos archivos una para los literales en español y otra en inglés.

Capítulo 7

7 PRUEBAS

En este apartado se realizan las pruebas para comprobar el funcionamiento de la aplicación. Las pruebas intentan demostrar que un programa hace lo que se intenta que haga, así como descubrir defectos en el programa antes de usarlo. Hay que verificar los resultados de la prueba que se opera para buscar errores, anomalías o información de atributos no funcionales del programa [17].

7.1 PRUEBAS API REST CON POSTMAN

7.1.1 Prueba 1

Requisito. RQF-08 El sistema deberá ser capaz de acceder a los datos guardados.

Descripción. El sistema debe ser capaz de devolver los datos de PIB para valores discretos.

Resultado esperado. Se hace una consulta a la API cuyo resultado esperado es un data set de longitud 100 con los datos correspondientes a la fila indicada.

Resultado obtenido.

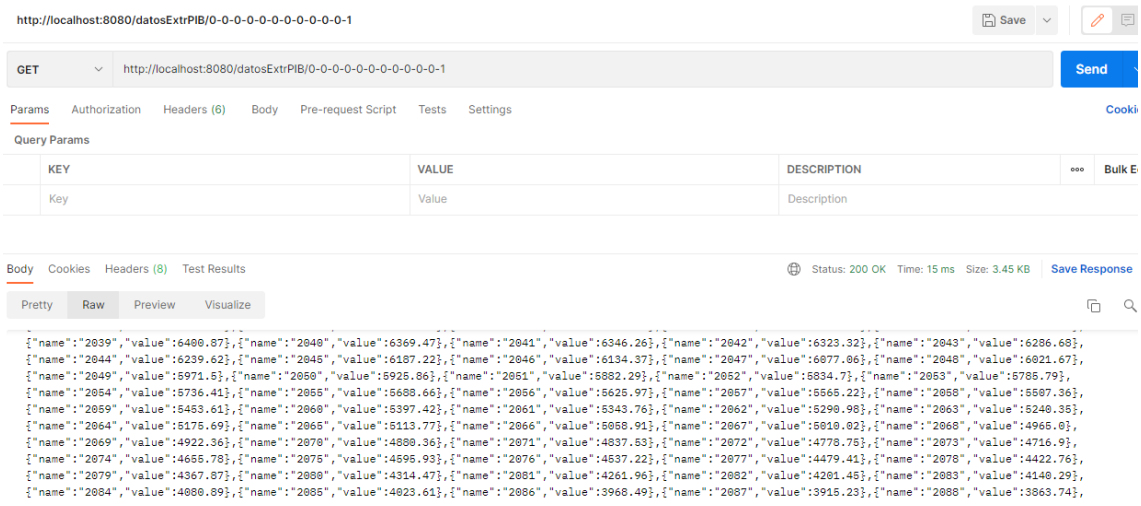

Ilustración 24. Resultado obtenido de la prueba 1.

Resultado: SUPERADO.

7.1.2 Prueba 2

Requisito. RQF-08 El sistema deberá ser capaz de acceder a los datos guardados.

Descripción. El sistema debe ser capaz de devolver los datos de PIB para valores continuos.

Resultado esperado. Se hace una consulta a la API cuyo resultado esperado es un data set de longitud 100 con los datos correspondientes a la fila indicada.

Resultado obtenido.

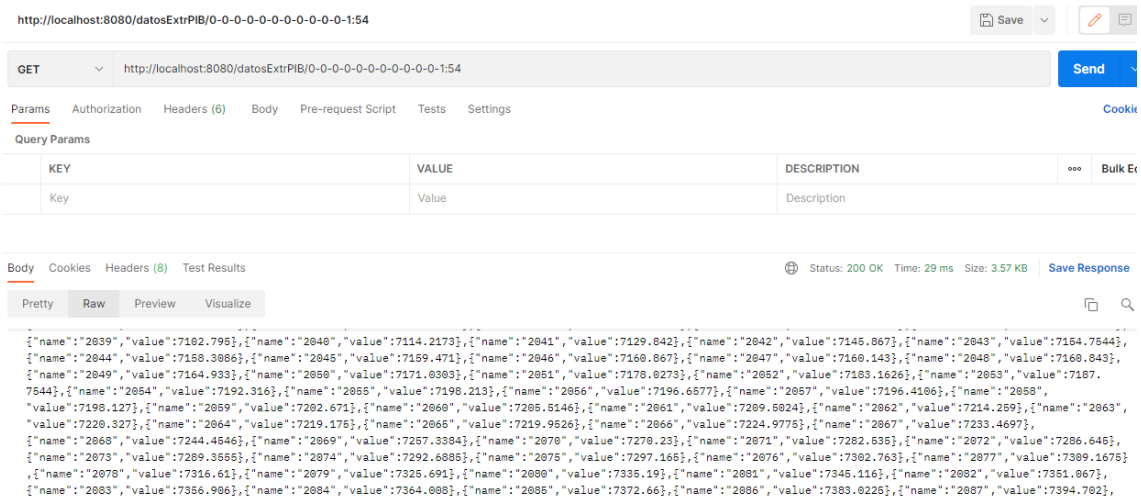

Ilustración 25. Resultado obtenido de la prueba 2.

Resultado: SUPERADO.

7.1.3 Prueba 3

Requisito. RQF-08 El sistema deberá ser capaz de acceder a los datos guardados.

Descripción. El sistema debe ser capaz de devolver los datos de temperatura para valores discretos.

Resultado esperado. Se hace una consulta a la API cuyo resultado esperado es un data set de longitud 100 con los datos correspondientes a la fila indicada.

Resultado obtenido

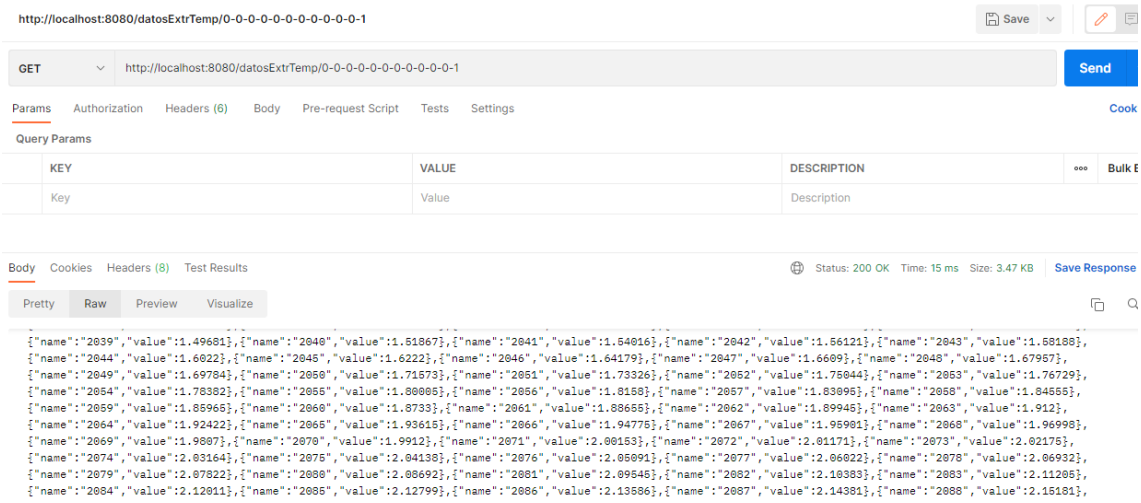

Ilustración 26. Resultado obtenido de la prueba 3.

Resultado: SUPERADO.

7.1.4 Prueba 4

Requisito. RQF-08 El sistema deberá ser capaz de acceder a los datos guardados.

Descripción. El sistema debe ser capaz de devolver los datos de temperatura para valores continuos.

Resultado esperado. Se hace una consulta a la API cuyo resultado esperado es un data set de longitud 100 con los datos correspondientes a la fila indicada.

Resultado obtenido

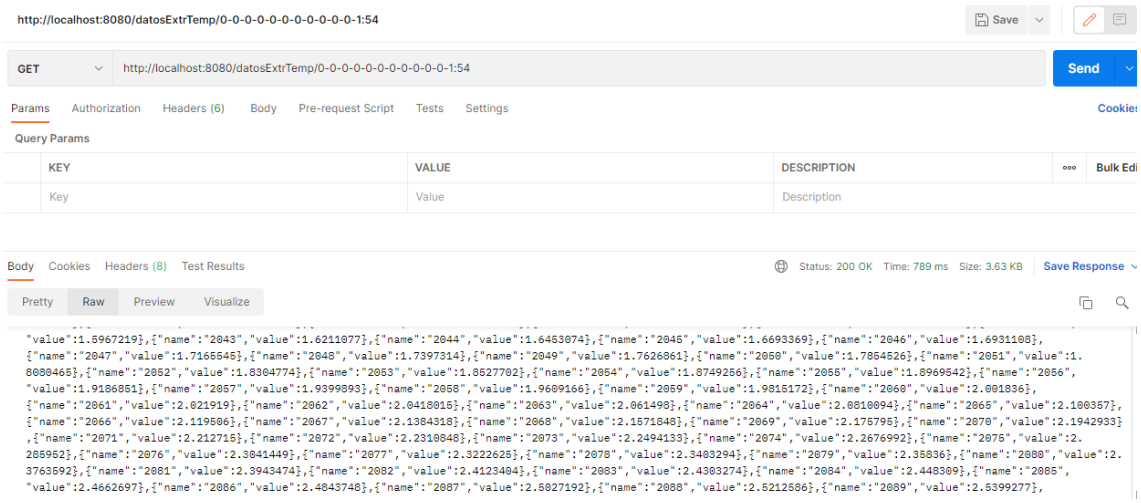

Ilustración 27. Resultado obtenido de la prueba 4.

Resultado: SUPERADO.

7.2 PRUEBAS DE INTERFAZ

7.2.1 Prueba 5

Requisito. RQF-01 El sistema deberá permitir al usuario visualizar la evolución de la temperatura resultante de tomar una medida.

Descripción. El sistema debe ser capaz de mostrar cambios en la gráfica de la temperatura derivados de modificar alguna hipótesis o medida política.

Resultado esperado. Se espera que la gráfica de la temperatura muestre cambios al cambiar alguna medida.

Resultado obtenido

Ilustración 28. Ilustración inicial de la aplicación.

Ilustración 29. Resultado obtenido de la prueba 5.

Resultado: SUPERADO.

7.2.2 Prueba 6

Requisito. RQF-02 El sistema deberá permitir al usuario visualizar la evolución del PIB resultante de tomar una medida.

Descripción. El sistema debe ser capaz de mostrar cambios en la gráfica de la temperatura derivados de modificar alguna hipótesis o medida política.

Resultado esperado. Se espera que la gráfica de la temperatura muestre cambios al cambiar alguna medida.

Resultado obtenido. En la Ilustración 28 se puede ver el estado inicial de la gráfica.

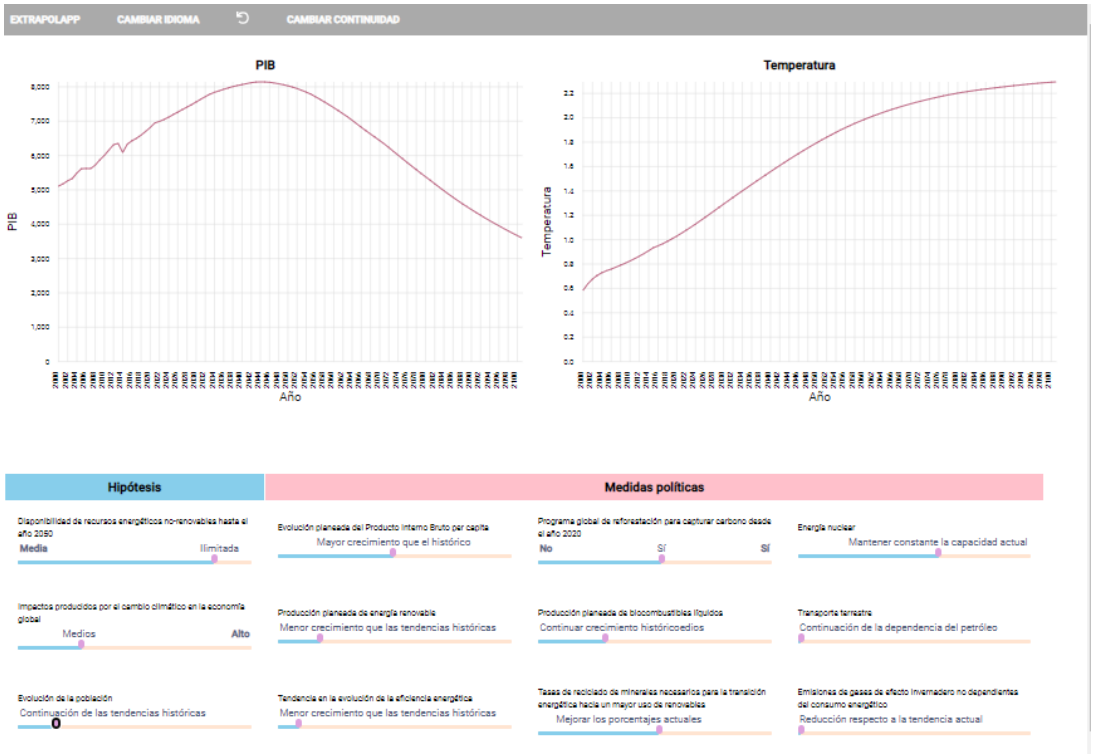

Ilustración 30. Resultado obtenido de la prueba 6.

Resultado: SUPERADO.

7.2.3 Prueba 7

Requisito. RQF-07 El sistema deberá permitir cambiar de idioma.

Descripción. El sistema debe ser capaz de cambiar de idioma entre español e inglés.

Resultado esperado. Se espera pasar de una aplicación completamente en español a otra completamente en inglés.

Resultado obtenido

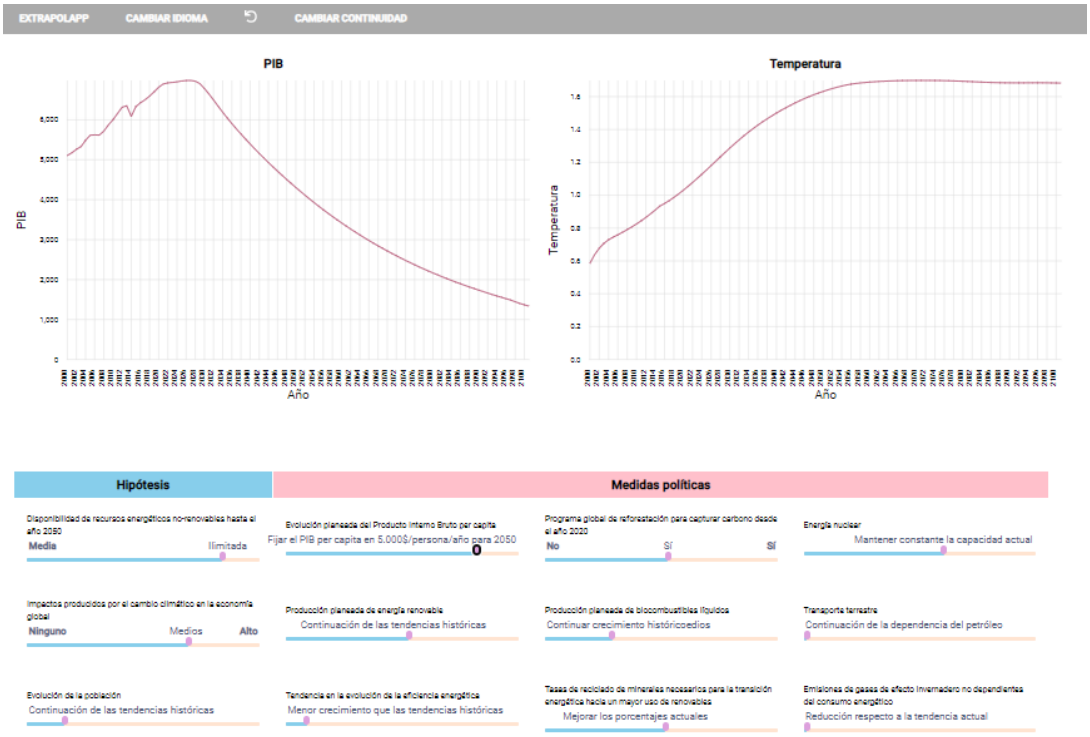

Ilustración 31. Ilustración que refleja la aplicación en español.

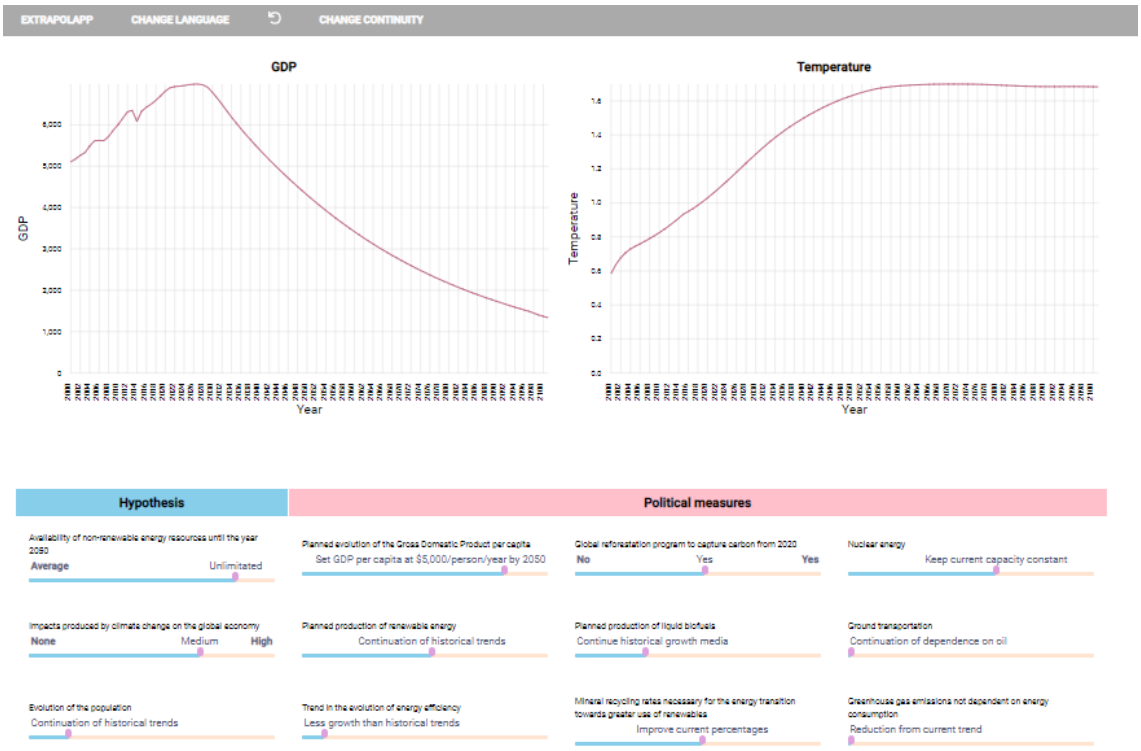

Ilustración 32. Resultado obtenido de la prueba 7.

Resultado: SUPERADO.

7.2.4 Prueba 8

Requisito. RQF-10 El sistema deberá permitir al usuario obtener los valores de cada punto de las gráficas al pasar el ratón por encima.

Descripción. El sistema debe ser capaz de mostrar el valor para un punto concreto de la gráfica.

Resultado esperado. Se espera que al pasar el ratón por encima de la gráfica se nos muestre el valor de ese punto.

Resultado obtenido

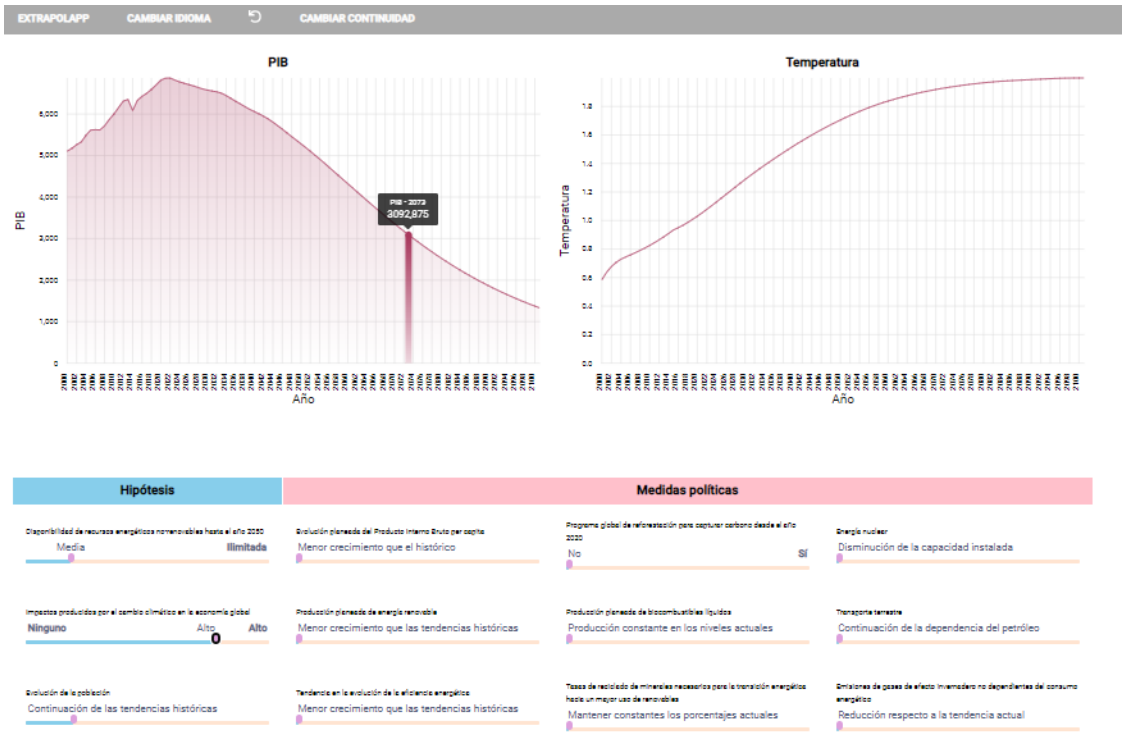

Ilustración 33. Resultado obtenido de la prueba 8.

Resultado: SUPERADO.

7.3 PRUEBA DE EXTRAPOLACIÓN

7.3.1 Prueba 9

Requisito. RQF-06 El sistema deberá permitir al usuario cambiar la continuidad de los parámetros.

Descripción. El sistema debe ser capaz de cambiar entre valores continuos y discretos los parámetros.

Resultado esperado. Los valores cambien de continuos a discretos.

Resultado obtenido

Ilustración 34. Ilustración con los valores de las medidas continuos.

Ilustración 35. Resultado obtenido de la prueba 9.

Resultado: SUPERADO.

7.3.2 Prueba 10

Requisito. RQF-04 El sistema deberá permitir al usuario hacer cambios continuos en los parámetros.

Descripción. El sistema debe ser capaz de realizar una estimación para los cambios continuos en los parámetros.

Resultado esperado. Se espera que el valor de estos cambios se encuentre entre los valores de las curvas discretas.

Resultado obtenido.

Ilustración 36. Curva inferior.

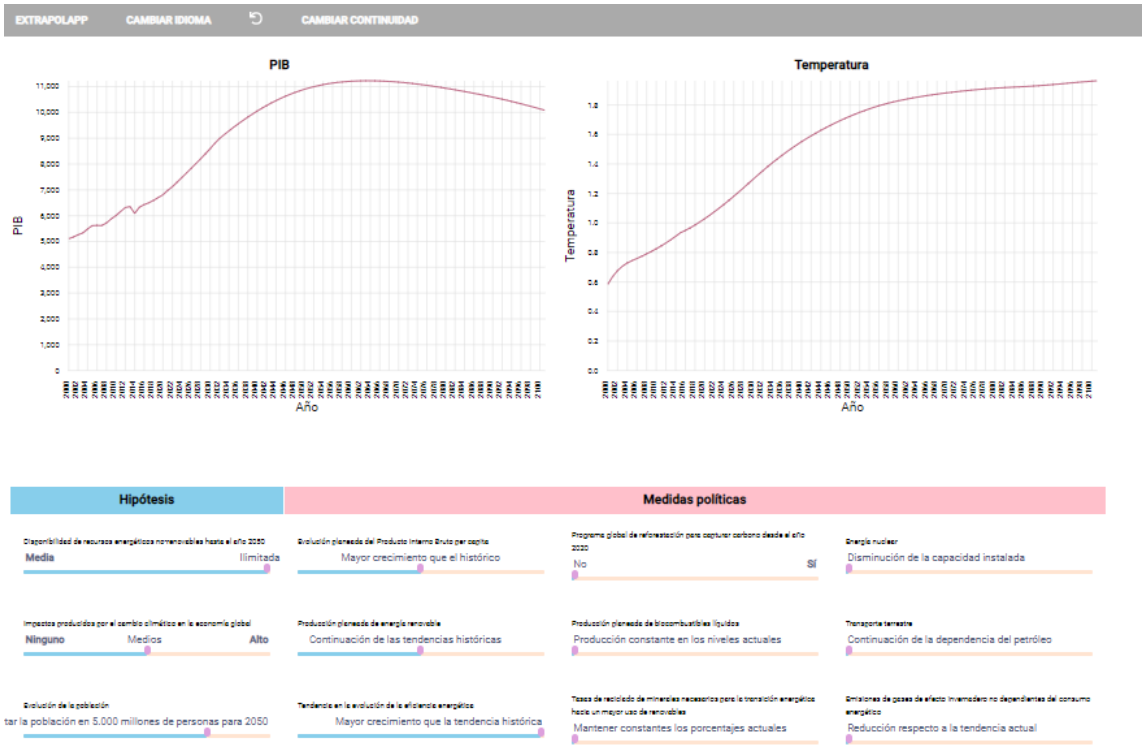

Ilustración 37. Curva superior.

Ilustración 38. Resultado obtenido de la prueba 10. Curva estimada.

Resultado: SUPERADO.

7.4 PRUEBAS DE COMPATIBILIDAD CON NAVEGADORES

7.4.1 Prueba 11

Requisito. RQNF-02 El sistema deberá funcionar en los principales navegadores.

Descripción. El sistema debe funcionar en el navegador Chrome.

Resultado esperado. Se espera que la aplicación funcione en este navegador.

Resultado obtenido.

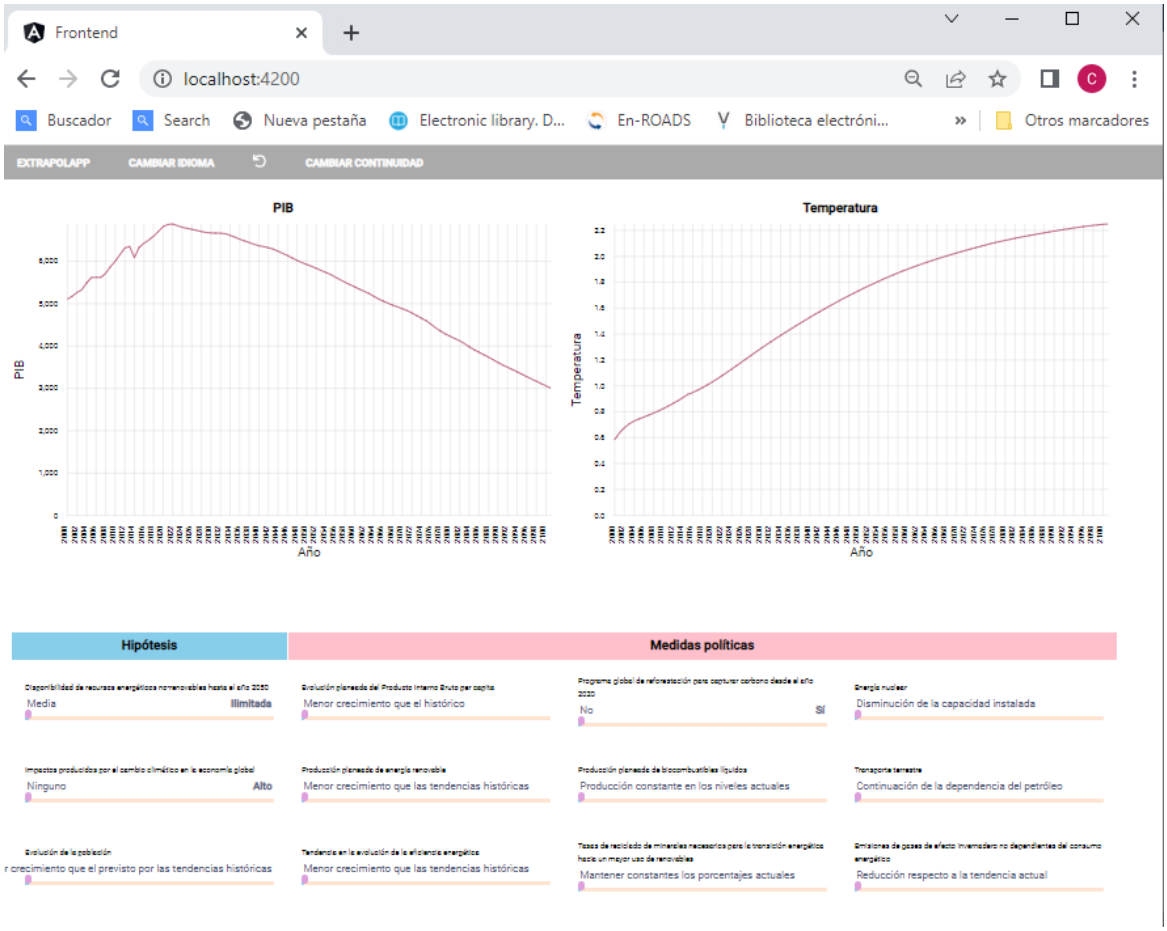

Ilustración 39. Resultado obtenido de la prueba 11.

Resultado: SUPERADO.

7.4.2 Prueba 12

Requisito. RQNF-02 El sistema deberá funcionar en los principales navegadores.

Descripción. El sistema debe funcionar en el navegador Edge.

Resultado esperado. Se espera que la aplicación funcione en este navegador.

Resultado obtenido.

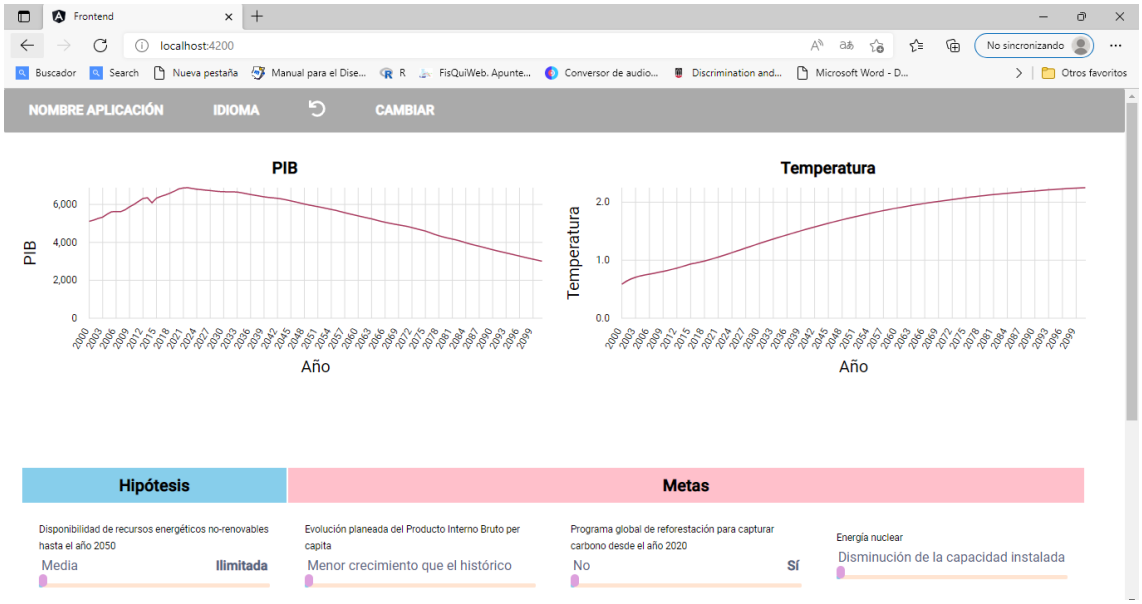

Ilustración 40. Resultado obtenido de la prueba 12.

Resultado: SUPERADO.

7.4.3 Prueba 13

Requisito. RQNF-02 El sistema deberá funcionar en los principales navegadores.

Descripción. El sistema debe funcionar en el navegador Mozilla Firefox.

Resultado esperado. Se espera que la aplicación funcione en este navegador.

Resultado obtenido.

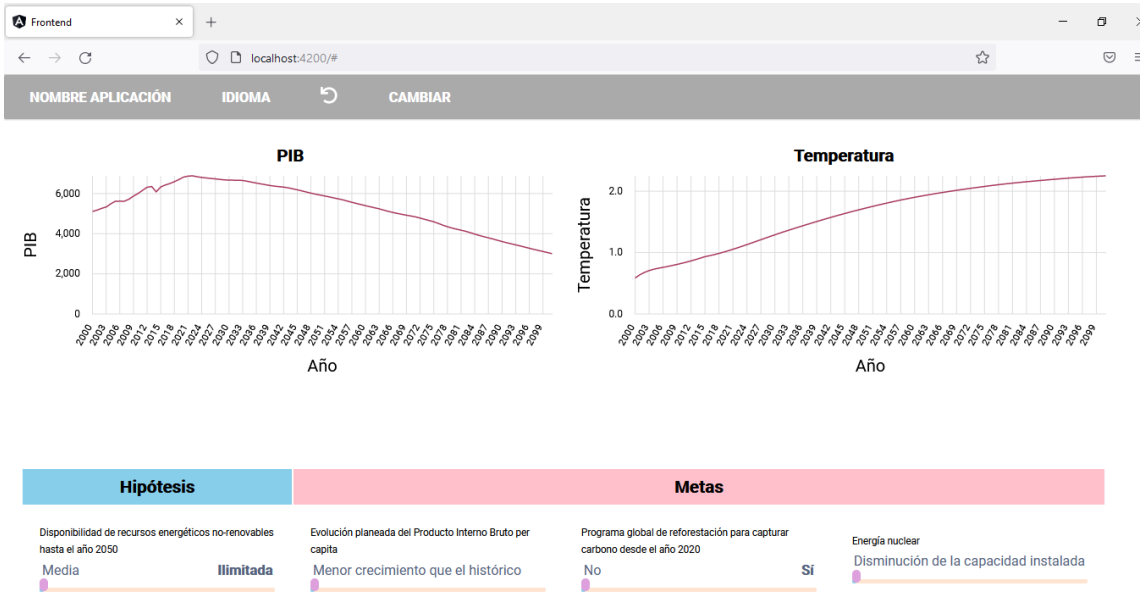

Ilustración 41. Resultado obtenido de la prueba 13.

Resultado: SUPERADO.

7.4.4 Prueba 14

Requisito. RQNF-02 El sistema deberá funcionar en los principales navegadores.

Descripción. El sistema debe funcionar en el navegador Explorer.

Resultado esperado. Se espera que la aplicación funcione en este navegador.

Resultado obtenido.

Ilustración 42. Resultado obtenido de la prueba 14.

Resultado: FALLIDO.

Capítulo 8

8 CONCLUSIONES Y TRABAJO FUTURO

Este proyecto presenta un sistema de visualización interactivo cuyo fin es explorar futuros escenarios dependiendo de una serie de valores de entrada, que podrá ir modificando dinámicamente el usuario.

La colección de requisitos recogidos garantiza el funcionamiento correcto de la aplicación, así como su rapidez al realizar cambios y una interfaz intuitiva para los usuarios. Se han cumplido todos los requisitos requeridos, destacando la velocidad de respuesta de la aplicación, realizando cambios casi instantáneos en las gráficas al modificar cualquier parámetro, así como poder modificar el valor de los parámetros no solo de manera discreta sino continua mediante una extrapolación.

El principal objetivo del trabajo es poder visualizar y explorar datos de series temporales de forma ágil. Para ello se ha logrado implementar un sistema que no solo permite un acceso eficiente a este data set sino que también implementa un algoritmo de extrapolación para generar series temporales a partir de valores no observados de las hipótesis y medidas políticas.

Otro de los objetivos más importantes del trabajo es crear de inicio a fin una aplicación full stack. Para ello he tenido que aprender tanto como funciona una aplicación full stack como las tecnologías necesarias para ello. En mi caso, dado que Desarrollo de Software no es mi mención, tuve que realizar cursos para aprender Angular, Spring Boot y Node que una vez terminados, complementaron los conocimientos adquiridos en las diferentes asignaturas para poder desarrollar la totalidad del proyecto.

Personalmente con este proyecto no solo he adquirido conocimientos técnicos, como el aprendizaje de nuevas tecnologías o la aplicación de patrones estudiados en la carrera, sino que me he tenido que adaptar a los imprevistos y aprender a buscar otras alternativas y soluciones a problemas que han ido surgiendo. Considero que este proyecto ha sido un reto en el que he podido aplicar conceptos estudiados en la carrera junto con nuevas tecnologías que he aprendido durante el desarrollo de este.

Una posible ampliación del trabajo es mejorar la extrapolación realizada mediante redes neuronales. Las redes aprenden examinando registros individuales, generando una predicción para cada registro y realizando los ajustes adecuados en las ponderaciones cuando la predicción es errónea [45].Para este problema se puede plantear resolver la extrapolación mediante redes neuronales teniendo como entradas los valores de las gráficas, los valores de las hipótesis y medidas políticas y los input simulation y obteniendo como salida los valores que se han de representar en las gráficas de temperatura y PIB.

Otra posible mejora sería incluir más graficas que aporten más información sobre la evolución de ciertos factores a lo largo de estos cien años. Esto implicaría en primer lugar añadir más fuentes de datos que dependiendo de su estructura podrían implicar también cambiar el algoritmo de búsqueda y la extrapolación que actualmente se utiliza. Además, la interfaz gráfica necesitaría ciertos cambios para o bien añadir más graficas o mostrar una opción para cambiar las gráficas que se visualizan.

Capítulo 9

9 BIBLIOGRAFÍA

[1] «Naciones Unidas,» [En línea]. Available: https://www.un.org/es/climatechange/what-is-climate-change. [Último acceso: 10 04 2022].

[2] «Iberdrola,» [En línea]. Available: https://www.iberdrola.com/sostenibilidad/que-es-el-antropoceno. [Último acceso: 15 06 2022].

[3] «Agencia europea del medioambiente,» [En línea]. Available: https://www.eea.europa.eu/es/senales/senales-2015/entrevista/el-cambio-climatico-y-la#:~:text=La%20contaminaci%C3%B3n%20atmosf%C3%A9rica%20puede%20provocar,afectan%20a%20la%20salud%20p%C3%BAblica.. [Último acceso: 18 06 2022].

[4] «Cnn España,» [En línea]. Available: https://cnnespanol.cnn.com/2018/10/08/el-planeta-solo-tiene-hasta-2030-para-detener-un-cambio-climatico-catastrofico-advierten-los-expertos/. [Último acceso: 18 06 2022].

[5] «Xenonstack,» [En línea]. Available: https://www.xenonstack.com/insights/big-data-challenges. [Último acceso: 19 06 2022].

[6] «Climateinteractive,» [En línea]. Available: https://www.climateinteractive.org/en-roads/. [Último acceso: 18 06 2022].

[7] MIT, «En-roads,» [En línea]. Available: https://en-roads.climateinteractive.org/scenario.html?v=22.1.1&p1=43&p7=68. [Último acceso: 19 06 2022].

[8] «Universidad de Valladolid,» [En línea]. Available: http://www5.uva.es/estadmed/datos/series/series.htm#:~:text=Una%20variable%20estad%C3%ADstica%20cuyos%20valores,a%20lo%20largo%20del%20tiempo.. [Último acceso: 14 06 2022].

[9] A. Manzano-Santos, D. Escudero-Mancebo y L.-J. , «A multivariate time series clustering algorithm for the,» p. 35, 2022.

[10] «SAP Community,» [En línea]. Available: https://blogs.sap.com/2021/05/06/a-multivariate-time-series-modeling-and-forecasting-guide-with-python-machine-learning-client-for-sap-hana/. [Último acceso: 14 06 2022].

[11] «Hmong,» [En línea]. Available: https://hmong.es/wiki/Integrated_assessment_modelling. [Último acceso: 18 06 2022].

[12] I. Capellán-Pérez, I. de Blas, J. Nieto , C. de Castro , L. J. Miguel , O. Carpintero , M. Mediavilla, L. F. Lobejón, P. Rodrigo , F. Frechoso y D. Álvarez-Antelo, «MEDEAS: a new modeling framework integrating global biophysical and socioeconomic constrains,» *Energy & Enviromental Science.*

[13] «Medeas,» [En línea]. Available: https://www.medeas.eu/model/medeas-model. [Último acceso: 17 06 2022].

[14] «Locomotion,» [En línea]. Available: https://www.locomotion-h2020.eu/locomotion-models/global-sustainability-crossroads-ii/. [Último acceso: 15 06 2022].

[15] «Kyoceradocumentsolutions,» [En línea]. Available: https://www.kyoceradocumentsolutions.es/es/smarter-workspaces/business-challenges/procesos/dashboard-y-su-significado-estrategico.html. [Último acceso: 16 06 2022].

[16] «Wikipedia,» [En línea]. Available: https://es.wikipedia.org/wiki/Conjunto_de_datos#:~:text=Un%20conjunto%20de%20datos%20(conocido,colecci%C3%B3n%20de%20datos%20habitualmente%20tabulada.. [Último acceso: 11 04 2022].

[17] I. Sommerville, Ingeniería de software, Pearson, 2011.

[18] «Cacoo,» [En línea]. Available: https://cacoo.com/. [Último acceso: 17 06 2022].

[19] «Ionos,» [En línea]. Available: https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/diagrama-de-casos-de-uso/#:~:text=El%20diagrama%20de%20casos%20de%20uso%20es%20una%20forma%20de,de%20programaci%C3%B3n%20orientada%20a%20objetos.. [Último acceso: 10 06 2022].

[20] «Lucidchart,» [En línea]. Available: https://www.lucidchart.com/pages/es/diagrama-de-secuencia/#section_0. [Último acceso: 15 06 2022].

[21] «Creately,» [En línea]. Available: https://creately.com/blog/es/diagramas/tutorial-de-diagrama-de-despliegue/. [Último acceso: 18 06 2022].

[22] «MongoDB,» [En línea]. Available: https://www.mongodb.com/mean-stack. [Último acceso: 17 06 2022].

[23] «Genbeta,» [En línea]. Available: https://www.genbeta.com/desarrollo/una-introduccion-a-mongodb. [Último acceso: 16 06 2022].

[24] «Ayuda para la ley de proteccion datos,» [En línea]. Available: https://ayudaleyprotecciondatos.es/bases-de-datos/documentales/. [Último acceso: 18 06 2022].

[25] «Developer mozilla,» [En línea]. Available: https://developer.mozilla.org/es/docs/Learn/Server-side/Express_Nodejs/Introduction. [Último acceso: 18 06 2022].

[26] «Developer mozilla,» [En línea]. Available: https://developer.mozilla.org/es/docs/Learn/Server-side/Express_Nodejs/Introduction. [Último acceso: 17 06 2022].

[27] «Campusmvp,» [En línea]. Available: https://www.campusmvp.es/recursos/post/que-son-spring-framework-y-spring-boot-tu-primer-programa-java-con-este-framework.aspx#:~:text=Spring%20Boot%20permite%20compilar%20nuestras,de%20aplicaciones%20en%20el%20propio%20.. [Último acceso: 18 06 2022].

[28] «Java desde 0,» [En línea]. Available: https://javadesde0.com/diferencias-entre-spring-y-spring-boot/. [Último acceso: 18 06 2022].

[29] «Platzi,» [En línea]. Available: https://platzi.com/blog/que-es-spring-boot/?utm_source=google&utm_medium=paid&utm_campaign=14603491644&utm_adgroup=&utm_content=&gclid=CjwKCAiA6Y2QBhAtEiwAGHybPQGTUQhzXEvuAi6zaaOCjXKHdUJR7Fx5Fftkk49HBMgBxRjBN4MIJhoCtW4QAvD_BwE&gclsrc=aw.ds. [Último acceso: 17 06 2022].

[30] «Documentación spring,» [En línea]. Available: https://docs.spring.io/spring-boot/docs/current/maven-plugin/reference/htmlsingle/. [Último acceso: 18 06 2022].

[31] «Gráficos en Angular,» [En línea]. Available: https://swimlane.gitbook.io/ngx-charts/. [Último acceso: 18 06 2022].

[32] «Angular charts,» [En línea]. Available: https://www.ngdevelop.tech/how-to-use-ngx-charts-in-angular/. [Último acceso: 16 06 2022].

[33] «Github,» [En línea]. Available: https://github.com/angular-slider/ngx-slider/blob/master/README.md. [Último acceso: 18 06 2022].

[34] «Github,» [En línea]. Available: https://github.com/ngx-translate/core. [Último acceso: 19 06 2022].

[35] «Encora,» [En línea]. Available: https://www.encora.com/es/blog/como-realizar-pruebas-automatizadas-con-postman. [Último acceso: 17 06 2022].

[36] «Espacio Honduras,» [En línea]. Available: https://www.espaciohonduras.net/microsoft-visual-studio-concepto-y-que-es-y-para-que-sirve-microsoft-visual-studio. [Último acceso: 17 06 2022].

[37] «Software,» [En línea]. Available: https://software.com.co/p/astah-professional. [Último acceso: 18 06 2022].

[38] E. Gamma, R. Helm, R. Johnson y J. Vlissides, «Patrones de diseño,» Addison Wesley, 2003.

[39] «Profile,» [En línea]. Available: https://profile.es/blog/patrones-de-diseno-de-software/. [Último acceso: 17 06 2022].

[40] «Malcoded,» [En línea]. Available: https://malcoded.com/posts/angular-2-components-and-mvvm/. [Último acceso: 18 06 2022].

[41] «Medium,» [En línea]. Available: https://medium.com/@thebabscraig/javascript-design-patterns-part-2-the-publisher-subscriber-pattern-8fe07e157213. [Último acceso: 18 06 2022].

[42] «Natapuntes,» [En línea]. Available: https://www.natapuntes.es/publish-subscribe-en-php/. [Último acceso: 18 06 2022].

[43] «Redhat,» [En línea]. Available: https://www.redhat.com/es/topics/api/what-is-a-rest-api#:~:text=Una%20API%20de%20REST%2C%20o,de%20estado%20representacional%20(REST).. [Último acceso: 17 06 2022].

[44] «pharo,» [En línea]. Available: https://pharos.sh/controller-y-restcontroller-anotaciones-en-spring-boot/#:~:text=En%20Spring%20Boot%2C%20la%20clase,Controller%20o%20la%20%40RestController%20anotaci%C3%B3n.. [Último acceso: 17 06 2022].

[45] F. Chollet, Deep Learning with Python, Manning, 2018.

[46] «Creately,» [En línea]. Available: https://creately.com/blog/es/diagramas/tutorial-diagrama-caso-de-uso/#importance. [Último acceso: 18 06 2022].

[47] «El pais,» [En línea]. Available: https://cincodias.elpais.com/cincodias/2021/11/12/economia/1636736144_856933.html. [Último acceso: 11 04 2022].

[48] «Europapress,» [En línea]. Available: https://www.europapress.es/economia/finanzas-00340/noticia-bce-calcula-caida-pib-ue-10-si-no-mitiga-cambio-climatico-20210922172529.html. [Último acceso: 15 06 2022].

[49] «Energia Murcia,» [En línea]. Available: https://www.energiamurcia.es/eurozona-2030/. [Último acceso: 10 04 2022].

[50] «Platzi,» [En línea]. Available: https://platzi.com/blog/que-es-mean-full-stack-javascript/#:~:text=MEAN%20es%20desarrollo%20full%2Dstack,%2C%20Express%2C%20Angular%20y%20Node.. [Último acceso: 18 06 2022].

[51] «Qualitydevs,» [En línea]. Available: https://www.qualitydevs.com/2019/09/16/que-es-angular-y-para-que-sirve/. [Último acceso: 15 06 2022].

Anexo A

MANUAL DE DESPLIEGUE E INSTALACIÓN

En esta sección se detallarán los requisitos necesarios para poner en marcha la aplicación.

INSTALACIÓN

Requisitos back- end

Para desplegar el código del back-end hay que instalar las siguientes dependencias:

Java. Instalar la versión. 17.0.1

Maven. Instalar la versión 3.8.3

Spring Boot. Instalar la versión v2.5.6

Requisitos front-end

Para desplegar el código del front-end hay que instalar las siguientes dependencias:

Node.js. Instalar la versión v14.18.0.

NPM. Instalar la versión 7.24.0.

Angular. Instalar la versión 12.2.7.

DESPLIEGUE

En primer lugar realizamos el **despliegue del servidor**. Para ello tendremos que descomprimir el archivo y asegurarnos de que los datos se encuentran en el mismo directorio que el servidor. Una vez comprobado esto, ejecutamos el .jar :

java -jar proyecto-0.0.1-SNAPSHOT.jar

Con esto ya temenos nuestro servidor desplegado.

Para **desplegar el cliente**, tenemos que descomprimir el archivo y ejecutar el comando:

npm install

Esto descargará todos los módulos de los que depende el front-end. Después habrá que ejecutar el comando:

ng serve

Con esto ya temenos nuestra aplicación completa desplegada.

El código del cliente se encuentra en el siguiente repositorio
https://github.com/carprie/TFG-client

El código del servidor se encuentra en el siguiente repositorio

https://github.com/carprie/TFG-server

Anexo B

HIPÓTESIS Y MEDIDAS POLÍTICAS

Hipótesis 1. Disponibilidad de recursos energéticos no-renovables (petróleo, carbón, gas natural y uranio) hasta el año 2050

A.- Media.

B.- Alta.

C.- Ilimitada.

Hipótesis 2. Impactos producidos por el cambio climático en la economía global

A.- Ninguno.

B.- Medios.

C.- Altos.

Hipótesis 3. Evolución de la población

A.- Menor crecimiento que el previsto por las tendencias históricas (8.500 millones de personas en 2050).

B.- Continuación de las tendencias históricas (9.200 millones de personas en 2050).

C.- Mayor crecimiento que lo previsto en las tendencias históricas (10.000 millones de personas en 2050).

D.- Limitar la población mundial en 5.000 millones de personas para 2050.

E.- Limitar la población mundial en 7.000 millones de personas para 2050.

Medida política 1. Evolución planeada del producto interno bruto (PIB) per capita

A.- Menor crecimiento que el histórico (+0.75%/persona/año).

B.- Continuación de las tendencias históricas(+1,5%/persona/año).

C.- Mayor crecimiento que el histórico(2,5%/persona/año).

D.- Fijar el PIB per capita en 5.000$/persona/año para 2050 y mantenerlo constante.

E.- Fijar el PIB per capita en 7.000$/persona/año para 2050 y mantenerlo constante

Medida política 2. Programa global de reforestación para capturar carbono desde el año 2020

A.- No.

B.- Sí.

Medida política 3. ENERGÍA NUCLEAR

A.- Disminución de la capacidad instalada.

B.- Mantener constante la capacidad actual.

C.- Aumento de la capacidad instalada.

Medida política 4. Producción planeada de energía renovable (electricidad y calor)

A.- Menor crecimiento que las tendencias históricas.

B.- Continuación de las tendencias históricas.

C.- Mayor crecimiento que las tendencias históricas.

Medida política 5. Producción planeada de biocombustibles líquidos (bioetanol, biodiesel, etc.)

A.- Producción constante en los niveles actuales.

B.- Continuar crecimiento histórico.

C.- Mayor crecimiento que la tendencia histórica.

Medida política 6. Transporte terrestre (coches, motos, autobuses, trenes, camiones, etc.)

A.- Continuación de la dependencia del petróleo.

B.- Transición a vehículos a gas y eléctricos

Medida política 7. Tendencia en la evolución de la eficiencia energética

A.- Menor crecimiento que las tendencias históricas.

B.- Continuar la tendencia de mejora histórica.

C.- Mayor crecimiento que la tendencia histórica.

Medida política 8. Tasas de reciclado de minerales necesarios para la transición energética hacia un mayor uso de renovables
A.- Mantener constantes los porcentajes actuales.

B.- Mejorar los porcentajes actuales.

Medida política 9. Emisiones de gases de efecto invernadero no dependientes del consumo energético
A.- Reducción respecto a la tendencia actual.

B.- Continuación de la tendencia actual.

C.- Aumento respecto a la tendencia actual.