



**BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA**

Facultad de Ciencias de la Electrónica

**“Diseño de sistema de información energético para  
la visualización de la integración de energía  
renovable al SEN”**

TESINA

para obtener el grado de:

***Licenciado en Ingeniería en Energías Renovables***

Presenta:

Sebastián Castillo Sánchez

Nombre asesor:

Prof. Jiménez Tenorio Erick Javier

Nombre coasesor:

Dr. Moreno Coria Luis Armando

*Puebla, Pue. Junio 2022*

## ÍNDICE

<b>INTRODUCCIÓN .....</b>	<b>3</b>
Objetivo general.....	3
Objetivos particulares: .....	4
Justificación .....	4
<b>PROBLEMA .....</b>	<b>6</b>
<b>CAPÍTULO 1: PANORAMA ENERGÉTICO EN MÉXICO.....</b>	<b>7</b>
1.1    Normativa Mexicana .....	7
1.2 Panorama de Generación y Capacidad en México.....	12
1.3 Conformación actual del Sistema Eléctrico Nacional.....	18
1.4 Sistemas de Información Energética.....	19
<b>CAPITULO 2: DESARROLLO DE ARQUITECTURA DEL SISTEMA .....</b>	<b>24</b>
2.1 Arquitectura de Software .....	24
2.2 Diagrama de Arquitectura.....	27
2.3 Documentación .....	30
2.4 Librerías .....	32
2.5 Desarrollo.....	34
<b>CAPITULO 3: DISEÑO DE APLICACIÓN WEB ESTATICA.....</b>	<b>38</b>
3.1    Back-end del sistema. ....	40
3.2    Costo de proyecto .....	43
<b>CAPITULO 4: RESULTADOS.....</b>	<b>45</b>

4.1 Datos .....	45
4.2 Visualización gráfica.....	47
<b>CONCLUSIONES .....</b>	<b>49</b>
<b>ANEXO 1: Encabezado index.html.....</b>	<b>51</b>
<b>ANEXO 2: Cuerpo index.html .....</b>	<b>53</b>
<b>ANEXO 3: energyvis.js .....</b>	<b>62</b>
<b>ANEXO 4: explore1vis.js .....</b>	<b>67</b>
<b>ANEXO 5: mapvis.js .....</b>	<b>72</b>
<b>ANEXO 6: rankvis.js.....</b>	<b>80</b>
<b>Referencias .....</b>	<b>4</b>
<b>Tabla de figuras .....</b>	<b>6</b>

## **INTRODUCCIÓN**

Actualmente, la industria del sistema de energía utiliza distintas aplicaciones para simular el mismo sistema de energía durante la etapa de planificación y durante la fase de operaciones en tiempo real. Por ejemplo, el algoritmo de flujo de energía de planificación es esencialmente el mismo que la función de flujo de energía en tiempo real, pero operan en modelos de datos que tienen diferentes formatos, estructuras e integración de datos.

La suposición básica es que el sistema no solo es utilizado por especialistas en gestión de energética, sino también por el consumidor general de energía. Se entrega la información tanto al usuario no profesional como al administrador de energía profesional, para mejorar la noción de consumo y respaldar decisiones de uso de energía más estratégicas y eficientes. Además, al acumular datos periódicos, el sistema también podrá proporcionar una base estadística para predecir y diseñar el plan energético futuro.

### **Objetivo general**

Diseño de un sistema de información energético en un entorno web para la visualización de la integración de energía renovable al SEN.

## **Objetivos particulares:**

- Diseñar el ambiente web para la integración de tecnologías de energías renovables al SEN.
- Desarrollar la visualización de la generación de energía bruta por región del SEN.
- Identificar la capacidad efectiva y generación bruta por entidad federativa en México.
- Analizar los tipos de sistemas de información energética para visualizar de manera simplificada la integración de sistemas de energía renovable.
- Realizar estudio de caso para verificar funcionamiento de aplicación web.

## **Justificación**

Los países desarrollados y líderes en la integración de sistemas de generación eléctrica mediante recursos renovables han creado políticas para modificar su matriz energética, permitiendo así, el aumento en la participación de las energías renovables y disminución de los sistemas convencionales de generación eléctrica.

El mercado eléctrico se considera un sistema complejo que da lugar a la generación de energía eléctrica en cada instante de tiempo para abastecer la demanda y las variaciones en la carga que son requeridas[1].

La variación del factor de disponibilidad en el mercado eléctrico es un parámetro que afecta la demanda eléctrica no atendida. La demanda eléctrica no atendida es aquella que se da cuando

parte de la población se queda sin abastecimiento energético, por lo tanto, cuando el factor de disponibilidad aumenta, la demanda eléctrica no atendida disminuye, la demanda eléctrica no atendida aumenta, en este sentido, al disponer en menor medida del sistema de generación, la generación eléctrica es insuficiente para suplir la demanda de toda la población de una determinada región.

Las plataformas para visualización de datos de generación e integración de sistemas de energía renovables tienen complejidad de interpretación si no se tiene una formación en electrónica de potencia, por eso la necesidad de que la sociedad en general conozca estos datos de integración y el porcentaje de energías renovables introducidos al SEN de manera más accesible.

Al desarrollar sistemas para el manejo y visualización de datos de generación de energía se proporciona una solución ideal para dar acceso abierto y conveniente a valiosos datos de red, lo que permite una amplia gama de aplicaciones potenciales (por ejemplo, validación de modelos, aplicación de aprendizaje automático en sistemas de potencia, etc.)[2].

Con esto en mente se tuvieron las herramientas necesarias para el desarrollo de estrategias de visualización de datos energéticos, logrando así tener instrumentos accesibles, de fácil acceso y visualmente amigables con el usuario. El análisis del manejo de la generación podrá ser analizado mediante modelos matemáticos específicos.

## **PROBLEMA**

En el siguiente apartado se observa que el apropiado análisis de sistemas energéticos se ve opacado por la poca accesibilidad en la visualización de datos de generación de energía renovable, la integración de proyectos a gran escala de generación de energía solar fotovoltaica y energía eólica en el Sistema Eléctrico Nacional (SEN) conlleva a diferentes escenarios de pronóstico de generación de energía.

La visualización para el estudio y análisis de estos formatos de texto se ve principalmente frenada por el programa o software con el que se analizarán estos datos ya que la mayoría de los softwares cuentan con licencias de uso de costo elevado.

La problemática del desarrollo de herramientas digitales para la visualización de los datos energéticos tiene que ser abordada con una implementación de manera amigable, accesible y con el impacto necesario al usuario para poder así generar una correcta comprensión del funcionamiento del SEN y así poder analizar las diferentes opciones de fuentes de energía que se tienen en nuestro país.

# **CAPÍTULO 1: PANORAMA ENERGÉTICO EN MÉXICO**

## **1.1 Normativa Mexicana**

El incremento del consumo de energía se basa en factores básicos como, por ejemplo, crecimiento poblacional, desarrollo tecnológico, aumento de actividad industrial, entre otros.

Conforme a la normatividad en el país de acuerdo con los artículos 27 y 28 de la Constitución Política de los Estados Unidos Mexicanos, así como a lo dispuesto en el artículo 33 de la Ley Orgánica de la Administración Pública Federal, a la Secretaría de Energía le corresponde establecer y conducir la política energética del país; ejercer los derechos de la Nación en materia de petróleo y todos los carburos de hidrógeno sólido, líquido y gaseoso, de minerales radioactivos, así como respecto del aprovechamiento de los bienes y recursos naturales que se requieran para generar, conducir, transformar, distribuir y abastecer energía eléctrica que tenga por objeto la prestación del servicio público[3].

En México la primera comisión interministerial de cambio climático que incluye a 14 secretarías, así como la primera conciliación de cambio climático se encargó de generar así una estrategia nacional para el cambio climático en la cual uno de los principales ejes estratégicos y líneas de acción fue acelerar la transición energética hacia fuentes de energía limpia.

La Comisión Reguladora de Energía (CRE) ha utilizado diferentes acciones para el desarrollo eficiente de las Energías Renovables en México entre ellas la utilización de instrumentos para compensar la integración de fuentes renovables debido a su intermitencia en la generación que no son utilizados por el autoconsumo en el momento.



El Banco de Energía como instrumento regulatorio por parte de la CRE describe M. C. Francisco Salazar comisionado presidente de la CRE en 2011 tenía como objetivo entregar la energía excedente generada y no consumida por fuentes renovables a usuarios en diferentes periodos horarios analógicos, en periodos distintos o en días o meses.

Las diferentes líneas de acción buscaban una completa inmersión de las energías renovables en el Sistema Eléctrico Nacional (SEN) dando paso así a aumentar la incorporación de generación renovable y reducción de pérdidas energéticas mediante el uso de redes inteligentes y generación distribuida en el SEN, así como facilitar la interconexión de centrales de generación eléctrica con energías renovables en las regiones del país con mayor potencial y viabilidad económica[4].

La generación de electricidad a partir de fuentes renovables y la diversificación de la matriz energética en México se toma en cuenta después del análisis de el alto precio de los combustibles fósiles y la creciente demanda de energía en la industria, al cierre del primer semestre de 2013, el 84.6% de la generación de electricidad provino de combustibles fósiles.

Dentro de las líneas de acción referente a la energía eólica se encontraba fomentar la generación de energía eólica y aprovechar su potencial terrestre y marino para asegurar la compatibilidad tecnológica, social y ambiental. En lo referente a la energía solar fotovoltaica se encontraba promover la inversión en sistemas fotovoltaicos en zonas del país con alto potencial y fomentar la generación distribuida mediante el uso de sistemas fotovoltaicos en el sector industrial, residencial y de servicios.

La ley para el Aprovechamiento de Energías Renovables y el Financiamiento de la Transición Energética (LAERFTE) tenía como objetivo regular el aprovechamiento de fuentes de energía renovables y las tecnologías limpias para generar electricidad con fines distintos a la prestación del servicio público de energía eléctrica, así como establecer la estrategia nacional y los instrumentos para el financiamiento de la transición energética.

Dentro de dicha ley en su artículo tercero describe “Energía Renovable” como cuya fuente reside en fenómenos de la naturaleza, procesos o materiales susceptibles de ser transformados en energía aprovechable por la humanidad, que se regeneran naturalmente, por lo que se encuentran disponibles de forma continua o periódica, y que se enumeran a continuación:

1. El viento;
2. La radiación solar, en todas sus formas;
3. El movimiento del agua en cauces naturales o artificiales;
4. La energía oceánica en sus distintas formas, a saber: mareomotriz, maremotérmica, de las olas, de las corrientes marinas y del gradiente de concentración de sal;
5. El calor de los yacimientos geotérmicos;
6. Los bioenergéticos, que determine la Ley de Promoción y Desarrollo de los Bioenergéticos, y
7. Aquellas otras que, en su caso, determine la Secretaría, cuya fuente cumpla con el primer párrafo de esta fracción;

En el artículo número 26 de LAERFTE se establece que cada año la Secretaría llevará a cabo la actualización de la estrategia para el desarrollo de las energías renovables y presentará una prospectiva sobre los avances logrados en la transición energética y el aprovechamiento sustentable de las energías renovables, incluyendo un diagnóstico sobre las aplicaciones de las

pág. 9

tecnologías limpias y las energías renovables, así como sobre el ahorro y uso óptimo de toda clase de energía.

La Ley de la Industria Eléctrica LIE que se promulgó el 11 de agosto de 2014 para abrir a la competencia el sector de generación de energía y el mercado mayorista. La ley establece un nuevo marco regulatorio para el sector eléctrico de México, manteniendo la planeación y el control del sistema eléctrico nacional como deberes estratégicos del Estado. La transmisión y distribución de electricidad también estará a cargo del Estado a través de la estatal CFE. Sin embargo, las entidades privadas ahora pueden firmar una variedad de contratos diferentes con el estado. Esto contribuye potencialmente con tecnología, financiamiento y experiencia a la expansión y mejora de las redes de transmisión y distribución.

La Ley de la Industria Eléctrica LIE designa al CENACE como una entidad pública descentralizada dependiente de la administración pública federal, con personalidad jurídica y patrimonio propios. El CENACE garantizará principalmente la operación del sistema eléctrico nacional en términos de eficiencia, confiabilidad, continuidad y seguridad. Tiene la responsabilidad de satisfacer la demanda de electricidad al menor costo razonable mientras garantiza la estabilidad de la red.

El CENACE tendrá la facultad de proponer a la SENER la ampliación de la red de transporte de energía. La SENER tiene la facultad de autorizar los proyectos, mientras que la CRE diseñará y emitirá las normas correspondientes de manera clara, transparente y equitativa. Tanto la expansión de la transmisión como la distribución serán planificadas por la SENER en coordinación con la CRE y el CENACE [5].

Con esto la Ley para el Uso de Energías Renovables y Financiamiento de la Transición Energética fue adoptada para cumplir con los acuerdos internacionales firmados sobre reducción de emisiones de gases de efecto invernadero y metas de generación de energía limpia, dicha ley encomendó a la SENER desarrollar un inventario nacional de energía renovable para brindar información confiable sobre los recursos de energía renovable en México.

El inventario Nacional de Energías Renovables es un sistema de servicios estadísticos y geográficos financiado por el Fondo para la Transición Energética y el Aprovechamiento Sostenible de la Energía. Recopila información sobre el potencial de las energías renovables y sobre los proyectos de generación de electricidad a partir de fuentes renovables y pone la información a disposición del público. Esta información se brinda a través del Atlas Mexicano de Recursos de Biomasa para la Generación de Energía, la primera sección del Atlas Nacional de Olas, el Atlas Nacional de Olas y Viento, el Atlas de Radiaciones Solares, el Atlas Nacional de Recursos Geotérmicos y el Atlas de recursos hidroeléctricos a pequeña escala (Secretaría de Energía, 2018)

El desarrollo de las energías renovables en el sector eléctrico generará beneficios socioeconómicos y estratégicos, dada la gran área geográfica de México y sus comunidades dispersas, las minirredes y la electrificación rural también jugarán un papel crucial. Esto ayudará a limitar los desafíos de integración de la red y la expansión de la capacidad de transmisión. En las zonas urbanas, la generación distribuida en las azoteas de los edificios también jugará un papel importante.

También hay beneficios para el sistema eléctrico, como el aumento en la seguridad general del suministro de energía que surge de la expansión de la red. A cierto nivel de integración, la generación distribuida puede tener un impacto positivo en la congestión debido a cambios en los flujos de energía.

Sin embargo, el despliegue de altas proporciones de energías renovables variables en los sistemas de energía plantea desafíos simultáneos para la integración en la combinación energética existente, la conexión a la red y la operación del sistema y del mercado. Estos factores varían según el país y no existe una solución única para integrar la electricidad renovable en el sistema eléctrico[6].

## **1.2 Panorama de Generación y Capacidad en México**

Aún no se han completado estudios cuantitativos para evaluar el impacto de la integración de la Energía Renovable variable en México. Esto hace que sea difícil estimar la generación de energía por fuentes solar fotovoltaicas y eólicas, el SEN cuenta con diferentes fuentes de energía para mantener la demanda de energética, la integración de tecnologías nuevas y eficientes se perfilan como las más competentes para su integración al SEN estas tecnologías se han ido adaptando a la regulación en el país.

Al cierre del primer semestre del primer semestre del 2015 la capacidad instalada de energías renovables alcanzó el 25.3%, y la generación por energías renovables, cogeneración eficiente y sistemas de generación distribuida, representó 18.2 % del total. Para realizar estos cálculos la SENER considera la generación reportada de los proyectos de la Empresa Productiva del Estado, CFE y de los Generadores registrados en la CRE, utilizando las cifras de capacidad en operación y generación bruta de cada generador[7].

La capacidad de generación mediante energías renovables sumó 16,953.2 MW, lo cual representa el 25.3% de la capacidad de generación total. La capacidad de generación renovable creció 9.6% respecto al mismo periodo de 2014.

La participación de las energías renovables en la capacidad instalada para el primer semestre de 2015 alcanzó el 25.3%, donde la mayor fue para las hidroeléctricas (18.6%) seguido de eólica (4.1%), geotermia (1.3%) y solar Fotovoltaica (0.2%). La mayor parte de la nueva capacidad instalada por Generadores se dedicó a centrales eólicas (732.7 MW) y cogeneración eficiente (442.6 MW), seguidas de fotovoltaica (57.8 MW) y geotermia (52.0 MW) [7].

En cuanto a la generación de energía eléctrica a partir de energías renovables alcanzó, durante el primer semestre del 2015, un total de 27,307.1 GWh, lo equivalente al 18.2% del total generado. La generación por renovables creció 2.2% comparada con el mismo periodo de 2014. La participación de los contratos de interconexión legados y/o generadores externos. Los sistemas fotovoltaicos representaron el 77.3 % de toda la capacidad instalada al primer semestre del año 2015.

Para el segundo semestre del 2015 la capacidad instalada para generar energías limpias creció el 6.63% llegando a los 19, 265 MW lo que representa el 28.31% de la capacidad total. El año 2014 fue extremadamente lluvioso, mientras que 2015, por efectos del fenómeno del “Niño” tuvo precipitaciones muy por debajo de la media. Esta situación ocasionó que la generación de electricidad limpia se redujera en 3.7% a pesar del incremento en capacidad instalada.

En cuanto a la generación por energías limpias del segundo semestre del 2015, generó un total de 47,548.76 GWh lo que representa el 15.36% de total generado. La generación mediante

energía solar Fotovoltaica fue de 190.26 GWh lo que equivale un 0.06% y la Eólica de 8,745.15 GWh.

Comparando el periodo enero-junio 2015 y enero-junio 2016, la capacidad instalada para generar energías limpias creció un 6.29%, llegando a los 20,160MW lo que representa el 28.39% de la capacidad total nacional. El primer semestre de 2016 fue un periodo con precipitaciones por debajo de la media, lo que ha contribuido a que la generación de electricidad limpia disminuya en 3,540.89 GWh lo que representa un decremento del 10.38% respecto al mismo periodo en 2015 [8].

La capacidad instalada de Energía Renovable al 30 de junio de 2016 fue de 17,810.92 MW lo que representa un 25.08% del total de la capacidad instalada. En cuanto a la generación se lograron 24,648.1 MWh por Energías Renovables lo que representa el 15.86% del total.

La tasa media anual de crecimiento de la generación con fuentes renovables ha sido del 3,2%. Al cierre del primer semestre de 2017, México terminará el 20.82 % de su energía eléctrica con fuentes limpias. Comparando el periodo enero-junio 2016 y enero-junio 2017, la capacidad instalada para generar energías limpias creció un 6,85 %, llegando a los 21.541,72 MW que representa el 29,09 % de la capacidad nacional. Al 30 de junio de 2017, la generación con energías limpias alcanzó 33,274.31 GWh, que refleja un crecimiento del 8,79% con respecto al primer semestre.

La capacidad instalada en el periodo enero-junio 2017 fue de 18,785.93MW lo que representa un 25.37% de la capacidad total, incrementando la capacidad fotovoltaica a 0.62% o 460.86MW, y la capacidad de energía eólica aumentó a 5.32% o 3942.22MW. La generación de energía renovable en el mismo periodo fue de 24,793.69GWh el cual representa un 15.51% de la

generación total aumentando la generación fotovoltaica a 0.17% o 273.45 GWh y la generación por energía eólica aumentando un 3.19% o 5,094.43 GWh[9].

El 2017 fue un año en el que la energía fotovoltaica tuvo un crecimiento sin precedentes, al alcanzar una generación de 1.149,6 GWh (934,81 GWh más que en 2016) y un crecimiento en capacidad instalada de 285,13 MW (73,73 %). El factor más importante del crecimiento en este sector fue el incremento de sistemas de generación distribuida.

Por otra parte, la generación eólica sólo incrementó en 1.5 % con respecto al 2016 (157.09 GWh), debido a los daños causados en el estado de Oaxaca por el sismo del 7 de septiembre de 2017, que utilizan paros técnicos en varios parques eólicos y en la Subestación Eléctrica de Ixtepec, la cual hasta enero de 2018 funcionó al 66 % de su capacidad.

Al 31 de diciembre de 2017 la capacidad instalada en energías limpias fue de 25.68% o 19.436.69 MW logrando una capacidad de energía fotovoltaica de 673.74 MW que representa un 0.89% de la generación y la energía eólica representando un 5.55% de la generación o 4,198.98 MW. La generación llegó a 51,542.18GWh lo que representa el 15.66% representando la energía fotovoltaica el 0.35% o 1,149.6 GWh y la generación eólica el 3.23% representando el 10,619.66 GWh [9].

Durante 2017, el 36% de las nuevas inversiones realizadas en Latinoamérica para el desarrollo de proyectos de energías renovables se realizó en México, colocando a nuestro país en el lugar 12 del Índice Atractivo-País para Energías Renovables de Ernst & Young Global Limited y en el lugar 10 entre los Países Líderes en Nuevas Inversiones del New Energy Finance de Bloomberg.



Al cierre del primer semestre de 2018 la generación por fuentes limpias alcanzó 24.12 % (40,499.01 GWh), menos de un punto porcentual para cumplir la meta del 25 % de generación de energía limpia establecida por México en la Ley de Transición Energética. Las tecnologías con mayor crecimiento fueron la fotovoltaica, la eólica y la cogeneración eficiente contribuyendo a que la capacidad instalada por fuentes limpias se incrementara 11.84 % (2,550.41 MW) y la generación en 21.71 % con respecto al primer semestre del 2017 (Secretaría de Energía, 2018).

La capacidad instalada en el periodo enero-junio 2018 fue de 20,620.20MW lo que representa un 27.09% de la capacidad total, incrementando la capacidad fotovoltaica a 2.16% o 1,646.55MW, y la capacidad de energía eólica aumentó a 5.74% o 4367.34MW. La generación de energía renovable en el mismo periodo fue de 29,026.54GWh el cual representa un 17.29% de la generación total aumentando la generación fotovoltaica a 0.72% o 1,204.54 GWh y la generación por energía eólica aumentando un 3.63% o 6,093.02 GWh (Secretaría de Energía, 2019).

La generación neta de energías limpias en 2019 fue de 21.80% o 69,772.23 GWh teniendo un aumento en la generación por energía eólica en 2019, representó el 5.2% con 16,726.91GWh en cuanto a la generación fotovoltaica total fue de 8,339.47 GWh lo que representó el 2.6%, cabe señalar que para los años antes mencionados se incluyó dentro de la generación fotovoltaica la generación neta de los Sistemas Fotovoltaicos Interconectados financiados por el Fideicomiso de Riesgo Compartido (FIRCO), así como la generación de los distintos permisionarios incluyendo CFE y lo generado por autoabasto aislado (SENER, 2020).

La expansión de la capacidad de transmisión será esencial para entregar generación a escala de servicios públicos desde áreas remotas del norte ricas en recursos hasta centros de demanda en el centro y sur de México.

En 2020, el consumo bruto nacional del SEN fue de 315,968 GWh lo que significa un decremento de 2.76% respecto al consumo de 2019. La contracción en el crecimiento se debe en parte a las restricciones de conexión a la red impuestas por el regulador y el operador del sistema en respuesta a las preocupaciones de confiabilidad causadas por los cambios en el patrón de demanda durante la crisis de Covid-19.

La generación por energías limpias en 2020 fue de 66,879 GWh lo que representa el 19.47% de la generación total de energía, en cuanto a la generación por energía solar fue de 3.94% o 13,528 GWh y la generación por energía eólica fue de 19,701 GWh o el 5.74% según datos de la Agencia Internacional de Energía IEA.

En 2021, el consumo neto nacional del SEN ascendió a 322.541 GWh, lo que significa un incremento de 3,5% respecto al consumo de 2020. Este incremento es reflejo de la recuperación en ascenso de la económica del país, luego de los estragos ocasionados por la contingencia sanitaria, la cual ocurre la suspensión de algunas actividades productivas en todo el país (SENER, 2021).

La infraestructura relacionada con la integración en la red de energías renovables variables en México no ha sido estudiada en detalle utilizando modelos de sistemas de potencia. Esto deberá corregirse en los próximos años a medida que crezca la proporción de energías renovables variables. Es necesario comprender mejor las soluciones tecnológicas, las medidas adecuadas y el apoyo normativo necesario y adecuado al carácter específico de México.

### **1.3 Conformación actual del Sistema Eléctrico Nacional**

Actualmente el sistema eléctrico en México consta de 9 centros de control regional ubicados en las ciudades de México, Puebla, Guadalajara, Mérida, Hermosillo, Gómez Palacio, Monterrey, Mexicali, La Paz y un pequeño centro de control en Santa Rosalía Baja California Sur, para el Sistema Interconectado Mulegé (SIM). El Centro Nacional en la Ciudad de México en conjunto con el Centro Nacional Alterno, ubicado en la Ciudad de Puebla coordinan el MEM y la operación segura y confiable del SEN.

El Sistema Interconectado Nacional (SIN) está integrado por las siete regiones: Central, Oriental, Occidental, Noroeste, Norte, Noreste y Peninsular. En ellas se comparten los recursos y reservas de capacidad ante la diversidad de demandas y situaciones operativas; esto permite el intercambio de energía para lograr un funcionamiento más económico y confiable en su conjunto. El Sistema Interconectado Baja California (SIBC), opera interconectado a la Red Eléctrica de la región Oeste de EE UU. Western Electricity Coordinating Council (WECC) por medio de dos líneas de transmisión conectadas a un nivel de tensión de 230 kV en corriente alterna. Mientras que los Sistemas Interconectados Baja California Sur (SIBCS) y SIM están eléctricamente aislados entre sí, así como del SIN y SIBC.

En el Programa de Desarrollo del Sistema Eléctrico Nacional 2022-2036 se menciona que en el año 2020 la capacidad instalada de las Centrales Eléctricas de Energía Limpia Renovable tales como hidráulica, geotérmica, eólica, fotovoltaico y de biocombustibles, fue de 25,594MW y al cierre de diciembre de 2021 se tienen 26,899MW lo que representa un incremento del 5.1% con respecto al 2020 siendo las centrales eléctricas eólicas y fotovoltaicas las sobresalientes en ese año.

## **1.4 Sistemas de Información Energética**

La creciente complejidad e interconexión de las infraestructuras de energía, telecomunicaciones, transporte y finanzas plantean nuevos desafíos para una gestión y operación seguras y confiables. Ninguna entidad individual tiene el control completo de estas redes multiescala, distribuidas y altamente interactivas, o la capacidad de evaluar, monitorear y administrar en tiempo real. Además, las metodologías matemáticas convencionales que sustentan el modelado, la simulación, el paradigma de control son incapaces de manejar su complejidad e interconexión.

Las interacciones entre redes como estas aumentan la complejidad de las operaciones y el control. La naturaleza interconectada de las redes las hace vulnerables a fallas en cascada con consecuencias generalizadas[10].

En los últimos años la incorporación de energía renovable en los diferentes sistemas eléctricos mundiales ha llevado a cada país a modelar y desarrollar diferentes estrategias para la incorporación correcta de este tipo de energía, mucha de esta generación de energía utiliza sistemas de electrónica de potencia lo cual se diferencia de la generación convencional energía síncrona.

La tendencia de crecimiento de las energías renovables en México llevó a incorporar cada vez más este tipo de tecnología debido a su bajo costo de mantenimiento y efectividad sin embargo el Sistema Eléctrico Nacional (SEN) se vuelve dependiente de estas variaciones del recurso renovable principalmente de flujo de viento en cuanto a la energía eólica y radiación solar en cuanto a paneles fotovoltaicos.

La complejidad en la planificación de los sistemas de gestión energética implica que las herramientas deben ser altamente interactivas, informatizadas, visuales y fáciles de usar. Tales

herramientas pueden ayudar a los tomadores de decisiones a analizar las ventajas y desventajas entre objetivos energéticos, ambientales y socioeconómicos, identificando alternativas de decisión óptimas bajo escenarios de políticas ambientales y energéticas y planes regionales, así como sus combinaciones.

Además, en la planificación de los sistemas de gestión de energía, existen incertidumbres entre los sectores/procesos y factores del sistema, como la eficiencia energética de las instalaciones, los factores de emisión de gases de efecto invernadero y los impactos de la contaminación, que pueden expresarse como intervalos en lugar de valores deterministas. Estas incertidumbres inevitablemente complicarían el proceso de toma de decisiones en la asignación de actividades y servicios energéticos, que también son importantes en la planificación de los sistemas de gestión energética[11]

Mediante herramienta de modelado se puede obtener una mejor predicción de este recurso renovable ya que se puede obtener información valiosa para la correcta integración de este tipo de energía al SEN.

Las tecnologías que mejoran el ahorro de energía se pueden dividir en las dos áreas. La primera área solo informa sobre los datos de consumo de energía relacionados con el uso patrones para motivar a los ciudadanos a ahorrar energía.

La segunda área no solo monitorea el consumo de energía, sino que también controla la fuente de alimentación de una manera muy interactiva. Los Sistemas de Información Energética (EIS por sus siglas en inglés) se refieren al software, el hardware de adquisición de datos y los sistemas de comunicación administrados por una empresa, sociedad o colectivo para proporcionar

información energética a los administradores de energía de edificios comerciales, administradores de instalaciones, administradores financieros y empresas de servicios eléctricos.

Existen varios tipos de Sistemas de Información Energética pero los más comunes son: "basados en la web" y "no basados en la web". Los productos EIS basados en la web acceden y manipulan los datos de las instalaciones del edificio o estación a través de Internet, mientras que los EIS no basados en la web manipulan los datos almacenados en el sitio [12].

Aunque los EIS han estado en desarrollo desde mediados de la década de 1990 (Levy, et al., 2001), sus capacidades se han mejorado y ampliado para incluir una amplia variedad de operatividad y conectividad. Para hacer frente a los problemas de confiabilidad de la electricidad en los últimos años, las principales empresas de servicios públicos han creado programas de respuesta a la demanda (DR por sus siglas en inglés), que ofrecen a los clientes incentivos en efectivo para reducir las cargas máximas.

La actualización diaria o en tiempo real de los datos de consumo de energía por hora permite a los usuarios evaluar los problemas de rendimiento que han sido difíciles de observar. También mejora el proceso de puesta en retro-puesta en marcha[13]. Dado que la mayoría de los productos EIS brindan actualizaciones diarias o en tiempo real de los datos de tendencias por hora, los operadores de las instalaciones pueden verificar el impacto de una estrategia operativa inmediatamente después o dentro de un día de la operación. En ausencia de un EIS, una evaluación de impacto tendría que posponerse hasta que llegara la factura mensual de servicios públicos.

Un EIS también permite a los operadores de las instalaciones ver el detalle del impacto por hora, mientras que una factura mensual de servicios públicos mostraría solo el total mensual. Con

capacidades de Internet, un EIS puede ayudar a administrar cientos de sitios distribuidos geográficamente.

Aunque los administradores de energía pueden reunir y organizar datos de facturas de servicios públicos mensuales para cientos de sitios, este trabajo consume mucho tiempo. Los productos EIS pueden facilitar tales tareas de administración de energía en múltiples instalaciones.

Las interacciones entre redes como estas aumentan la complejidad de las operaciones y el control. La naturaleza interconectada de las redes las hace vulnerables a fallas en cascada con consecuencias generalizadas.

Comprender las funciones y la facilidad de uso de los diferentes productos EIS. El conocimiento de las tecnologías de medición y conectividad también es importante para obtener más información sobre las tecnologías subyacentes que permiten la aplicación.

La mayoría de los productos EIS tienen capacidades de visualización de datos, algunos más complejos que otros. Las funciones de visualización de datos se pueden dividir aproximadamente en dos categorías, visualización de series temporales y visualización analítica.

Las empresas de servicios energéticos (ESCOs por sus siglas en inglés) normalmente brindan dos servicios principales a sus clientes: diagnósticos de edificios y contratos de desempeño. Un EIS puede ayudar con ambas tareas. Mediante el uso de un EIS, las ESCO pueden diagnosticar los edificios de sus clientes de forma remota y en tiempo real, en lugar de visitar los sitios repetidamente como se requiere en ausencia de un EIS. Esto ahorra tiempo y recursos humanos.

Los sistemas de información de energía basados en la web han evolucionado a partir de la industria de servicios eléctricos para administrar datos de consumo eléctrico de series temporales.

Sin embargo, las otras tecnologías de gestión energética también han ampliado sus funcionalidades y en parte han llegado a fusionarse con la tecnología EIS. Dado que los productos EIS son tecnologías relativamente nuevas, están cambiando rápidamente a medida que se desarrolla el mercado.

Existen diferentes tipos de sistemas de información de energía EIS:

- Sistemas básicos de información energética (Basic-EIS): recopila, archiva, resume y muestra datos de electricidad de todo un edificio.
- Sistemas de respuesta a la demanda (DRS): comunicación entre las empresas de servicios públicos y los clientes para facilitar los programas de respuesta a la demanda.
- Administración de energía empresarial (EEM): administra los costos generales de energía al facilitar la evaluación comparativa de energía y la optimización de adquisiciones en una empresa comercial.
- Sistemas de control y gestión de energía basados en la web (Web-EMCS): integra múltiples sistemas de construcción (por ejemplo, HVAC, iluminación, generación) y/o monitorea y controla los sistemas de construcción a nivel de componente comunicándose con un EMCS a través de Internet.

Los EIS son una combinación de tecnologías que se extienden entre la medición y la interfaz de usuario. Cada EIA tiene su propio énfasis y características únicas. Por ejemplo, un EEM tiende a centrarse en un software de visualización fácil de usar, mientras que un Web-EMCS tiende a centrarse más en la integración del sistema.



## **CAPITULO 2: DESARROLLO DE ARQUITECTURA DEL SISTEMA**

Es muy complicado poder comprender el funcionamiento de un sistema solo con leer su código, por lo que un diagrama de arquitectura es una ayuda visual y comprensible del sistema con el cual puede ayudar no solo a los desarrolladores a agregar (o no) nuevas funciones, también facilita la localización de problemas en la estructura para su correcto mantenimiento.

### **2.1 Arquitectura de Software**

La arquitectura de software es el esqueleto con todos los componentes de alto nivel de un sistema y cómo interactúan entre sí. Responde a las preguntas "¿Dónde?" y "¿Cómo?".

Con una arquitectura de software, es más fácil ver el panorama general. El objetivo central es identificar los requisitos funcionales y de calidad y manejarlos para mejorar la calidad general de una aplicación. En general, con la arquitectura de software, se puede monitorear el rendimiento, la escalabilidad y la confiabilidad del sistema.

La descripción de cada servicio debe ir seguida de un diagrama de arquitectura. Este diagrama debe detallar la arquitectura del servicio, incluyendo sus componentes, sus puntos finales, el flujo de solicitudes, sus dependencias (tanto ascendentes como descendentes) e información sobre cualquier base de datos o caché[14].

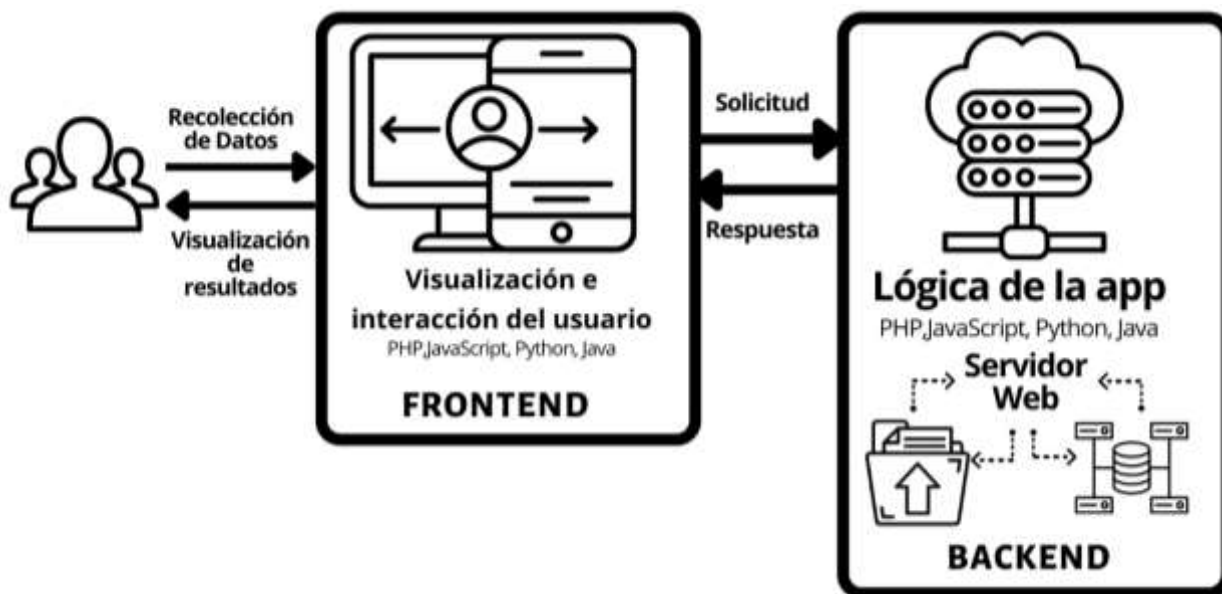
El Front-end o punto inicial suele ser la parte visible de cualquier aplicación mediante la cual el usuario interactúa y visualiza los datos. Generalmente, es esta interfaz la responsable de la correcta interacción y adaptación del usuario, es decir, este punto inicial en la aplicación presenta al usuario la visualización del sistema sin interactuar con su código.

Cada punto inicial cuenta con una interfaz que podría ser única en términos de la infraestructura que utiliza para visualizar sus datos. Algunos de los softwares de uso popular, como las aplicaciones de escritorio, tienen sus interfaces frontales escritas en lenguajes específico de programación, estos son diferentes a los que se usan en el desarrollo web y móvil.

Front-end también se refiere al lado del cliente de la aplicación. Entonces, cualquier cosa que se esté desarrollando en la interfaz es principalmente el desarrollo del lado del cliente. El desarrollo del lado del cliente puede estar basado en la web (basado en un navegador) o puede usar algunos marcos móviles para presentar la interfaz a los usuarios.

En el desarrollo de una aplicación Web como se ve en la **Figura 1** el Front- end es la primera etapa y la única con la que el usuario entra en contacto, el usuario envía, recibe e interactúa con los datos que requiere. Al interactuar solo con la parte frontal o punto inicial de la aplicación Web, si utiliza un navegador para mostrar la aplicación, entonces las tecnologías o la pila que se usa en el desarrollo son principalmente las siguientes:

- HTML.
- CSS.
- JavaScript.
- Marcos CSS como Bootstrap.
- Bibliotecas de JavaScript como jQuery.



**Figura 1: Arquitectura Aplicación Web (imagen de autoría propia)**

El Back-end o punto final se refiere al lado del servidor de la aplicación. Cualquier código que se esté desarrollando o implementando en el extremo del servidor es principalmente la infraestructura del sistema, esta infraestructura utiliza varios servicios y componentes para mostrar los resultados al cliente.

Dependiendo de la infraestructura del sistema se crea una solicitud con datos específicos que los clientes envían al servidor, el servidor los procesa utilizando varios servicios y programas, que se ejecutan en el extremo del servidor, y estos están ocultos en la parte interna de la aplicación.

En la **Figura 1** se esquematiza el constante traslado de datos entre el punto inicial(Front-end) y el punto final (Back-end) de la aplicación a desarrollar, dependiendo de la solicitud del usuario, el punto final se encarga de enviar la respuesta a la que esta programada, una arquitectura de software permite analizar si el programa responde a lo que el usuario requiere.

## 2.2 Diagrama de Arquitectura.

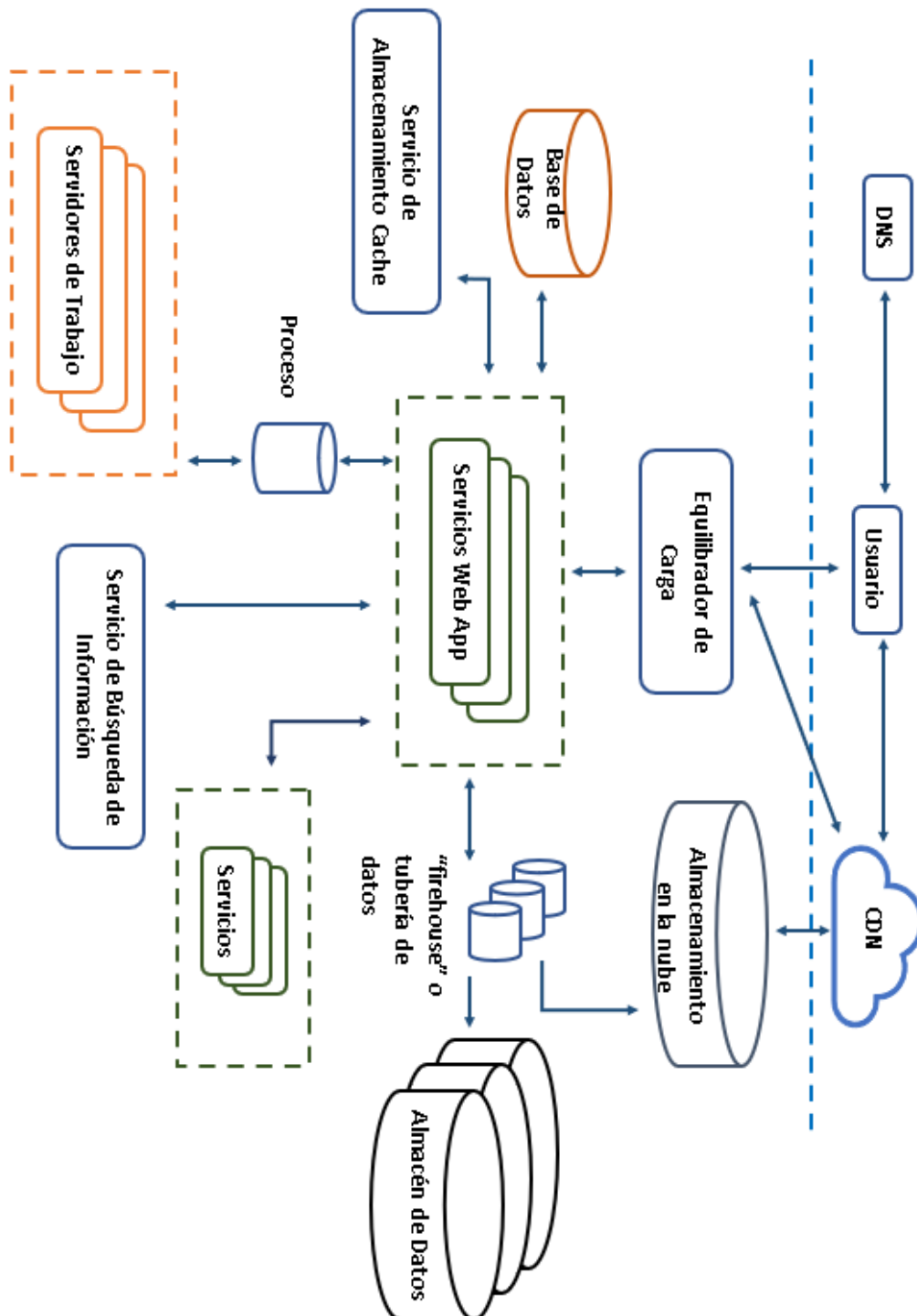


Figura 2: Diagrama de Arquitectura propuesto.

Como se muestra en la **Figura 2** el primer nivel del diagrama de arquitectura comienza con el Sistema de Nombres de Dominio o DNS por sus siglas en inglés, el DNS es un sistema fundamental que ayuda a que la aplicación pueda ser buscada con un nombre de dominio y una dirección IP específica, de esta manera, un servidor en particular recibe una solicitud enviada por un usuario.

El equilibrador de carga se ocupa principalmente del escalado horizontal. Al dirigir las solicitudes entrantes a uno de los múltiples servidores, el balanceador de carga envía la respuesta específica al usuario sin escalar los servicios a su mayor nivel, es decir, solo escala la solicitud a el nivel que el usuario necesita. los servidores de aplicaciones web existen en forma de copias múltiples que se reflejan entre sí, cualquier servidor procesa las solicitudes de la misma manera y el equilibrador de carga distribuye las tareas entre ellos para evitar que se sobrecarguen.

En el componente de Servicios de aplicaciones web o Web App se procesa la solicitud y se envían los documentos (JSON, SMK, etc.) a un navegador. Para realizar esta tarea, generalmente se refiere a las infraestructuras de back-end, como la base de datos, el servidor de caché, la cola de trabajos y otros. Al menos dos servidores, conectados al balanceador de carga, logran procesar las solicitudes de usuarios.

En los servicios de aplicaciones web las bases de datos brindan instrumentos para organizar, agregar, buscar, actualizar, eliminar y realizar cálculos. En la mayoría de los casos, los servidores de aplicaciones web interactúan directamente con los servidores de trabajo. El servicio de almacenamiento en caché proporciona un almacenamiento temporal de datos, lo que permite almacenar y buscar datos, es decir, cada vez que un usuario obtiene alguna información del servidor, los resultados de esta operación van a la memoria caché, por lo tanto, las solicitudes futuras regresan más rápido, el almacenamiento caché es más efectivo cuando:

- El cálculo computacional es lento.
- Cuando los resultados son los mismos para una solicitud en particular.

El componente denominado “procesos” es opcional ya que consta del propio flujo de trabajos y los servidores. Estos servidores procesan trabajos en la cola. La mayoría de los servidores web necesitan operar una gran cantidad de trabajos que no son de importancia primordial, por lo tanto, cuando se debe completar un trabajo, va a la cola de trabajos y se opera según un cronograma.

El componente “servicios” se refiere a cuando una aplicación web alcanza un nivel específico, los servicios se crean en forma de aplicaciones separadas. No son tan visibles entre otros componentes de la aplicación web, pero la aplicación web y otros servicios interactúan con ellos.

La mayoría de las aplicaciones modernas implican el manejo de datos como la recopilación, el almacenamiento y el análisis. Estos procesos requieren tres etapas:

1. Los datos se envían por la “tubería de datos” o “firehouse”, que proporciona una interfaz de transmisión para la absorción y el procesamiento de datos.
2. Los datos sin procesar, procesados y adicionales se envían al almacenamiento en la nube, este almacenamiento es un modelo particular en línea donde se genera un intercambio de datos a través de Internet.
3. El almacén de datos se puede usar para almacenar una variedad de archivos de diferentes tipos, como videos, fotos, etc.

El Sistema de Entrega de Datos o CDN por sus siglas en inglés se ocupa del envío de archivos HTML, archivos CSS, archivos JavaScript e imágenes. Entrega el contenido del servidor final en todo el mundo, por lo que los usuarios pueden cargar varios datos.

## 2.3 Documentación

El paso a la nube presenta un modelo de responsabilidad compartida. En este modelo, el proveedor de nube administra determinados aspectos de la aplicación y deja al usuario la responsabilidad restante.

En un entorno local, el usuario es responsable de todo. A medida que pasa a una infraestructura como servicio (IaaS), luego a plataforma como servicio (PaaS) y a software como servicio (SaaS), el proveedor de nube asume más parte de esta responsabilidad.

Esta responsabilidad compartida juega un papel en las decisiones de arquitectura, ya que estas pueden tener implicaciones en los costos, la seguridad, las capacidades técnicas y operativas de la aplicación. Al trasladar estas responsabilidades al proveedor, se puede centrar en aportar valor a la empresa y desentenderse de las actividades que no sean una función empresarial esencial.

El uso de un enfoque multicapa para proteger el entorno aumentará su seguridad. Normalmente se conoce como *defensa en profundidad* y las capas se desglosan de la siguiente manera:

1. Datos
2. Aplicaciones
3. Máquina virtual/proceso
4. Redes
5. Perímetro
6. Directivas y acceso
7. Seguridad física

Cada capa se centra en un área distinta en la que se pueden producir ataques y crea una protección profunda en caso de que se produzca un error en una capa o de que un atacante la traspase.

El enfoque de seguridad por capas aumenta el trabajo que un atacante debe hacer para obtener acceso a los sistemas y los datos. En cada capa se aplican diferentes controles de seguridad, tecnologías y capacidades.

La integración continua o CI por sus siglas en inglés, es el proceso de compilar y probar código automáticamente cada vez que un miembro del equipo confirma los cambios de código en el control de versiones. Una confirmación de código en la rama principal o troncal de un repositorio compartido desencadena el sistema de compilación automatizado para compilar, probar y validar la rama completa.

CI mantiene actualizada la rama principal. Los desarrolladores pueden usar sistemas de control de versiones modernos como Git para aislar su trabajo en ramas de características de corta duración. Una vez completada la característica, el desarrollador envía una solicitud de incorporación de cambios de la rama de características a la rama principal. Al aprobar la solicitud de incorporación de cambios, los cambios se combinan en la rama principal y se puede eliminar la rama de características.[15]

*Acciones de GitHub* son scripts empaquetados para automatizar tareas en un flujo de trabajo de desarrollo de software en GitHub. Estas acciones se pueden configurar para desencadenar flujos de trabajo complejos que satisfagan las necesidades de la organización, cada vez que los desarrolladores confirmen código fuente nuevo en una rama específica o intervalos de tiempo manualmente. El resultado es un flujo de trabajo automatizado confiable y sostenible, que supone una disminución significativa del tiempo de desarrollo.[16]



## 2.4 Librerías

En programación, las librerías son colecciones de recursos no volátiles utilizados por los programas. Estos recursos también pueden consistir en datos de configuración, documentación, código escrito previamente, subrutinas y clases. La razón principal para usar librerías es la reutilización. Todo lo que necesitamos hacer es usar una llamada de función y se realizará una tarea completa que de otro modo requeriría código que va de varias a varias docenas de líneas.

La mayoría de las librerías contienen funciones que el usuario normal no podría volver a crear. Sin el uso de estas librerías, cada usuario tendría que entender el funcionamiento interno de cada aspecto de su respectivo lenguaje de programación.

Se debe tener en cuenta que las librerías que se utilizan en el siguiente trabajo fueron desarrolladas por profesionales con años de experiencia, en continua actualización y mejora.

La primera librería utilizada es Bootstrap, Bootstrap fue creado en Twitter a mediados de 2010 por Mark Otto y Jacob Thornton. Antes de ser un marco de código abierto, Bootstrap era conocido como Twitter Blueprint. Unos meses después del desarrollo, Twitter celebró su primera Hack Week y el proyecto explotó cuando los desarrolladores de todos los niveles de habilidad saltaron sin ninguna orientación externa. Sirvió como guía de estilo para el desarrollo interno de herramientas en la compañía durante más de un año antes de su lanzamiento público, y continúa haciéndolo hoy [17].

Al ser una librería de CSS, significa que está escrito en CSS, por lo que todo lo que nos da Bootstrap, se puede hacer utilizando CSS básico, con Bootstrap podemos crear páginas web que se adapten completamente a cualquier tipo de pantalla y que estén adaptadas a los estándares estéticos de las páginas web de hoy en día.[18]

**D3.js** es una biblioteca JavaScript para manipular documentos basados en datos. **D3** le ayuda a dar vida a los datos mediante HTML, SVG y CSS. El énfasis de D3 en los estándares web le brinda todas las capacidades de los navegadores modernos sin atarse a un marco propietario, combinando potentes componentes de visualización y un enfoque basado en datos para la manipulación de DOM.

Con una sobrecarga mínima, D3 es extremadamente rápido, admite grandes conjuntos de datos y comportamientos dinámicos para interacción y animación. El estilo funcional de D3 permite la reutilización del código a través de una colección diversa de módulos oficiales y desarrollados por la comunidad.[19].

jQuery es una biblioteca JavaScript rápida, pequeña y rica en funciones. Hace cosas como recorrido y manipulación de documentos HTML, manejo de eventos, animación, y Ajax mucho más simple con una API fácil de usar que funciona en una multitud de navegadores. Con una combinación de versatilidad y extensibilidad, jQuery ha cambiado la forma en que millones de personas escriben JavaScript.[20]

Queue es una colección diseñada para contener elementos antes del procesamiento, queue proporciona inserción, extracción e inspección adicionales de operaciones. Cada uno de estos métodos existe en dos formas: uno lanza una excepción si se produce un error en la operación, la otra devuelve un valor (o , dependiendo de la operación). La última forma de la operación de inserción está diseñada específicamente para su uso con implementaciones de capacidad restringida.[21]

Normalmente, pero no necesariamente, las queue ordenan elementos en un Manera FIFO (first-in-first-out). Entre las excepciones se encuentran: queue de prioridad, que ordenan los elementos

según un comparador, o el orden natural de los elementos, y queue LIFO (o pilas) que ordenan los elementos LIFO (last-in-first-out). Cualquiera que sea el orden utilizado, la *cabeza* de la queue es que elemento que se eliminaría mediante una llamada a [remove\(\)](#)[22] o [poll\(\)](#)[23].

## 2.5 Desarrollo de código

La parte frontal del proyecto se enfoca en un archivo HTML (HyperText Markup Language) es el bloque de construcción más básico de la Web. Define el significado y la estructura del contenido web. Otras tecnologías además de HTML se utilizan generalmente para describir la apariencia/presentación (CSS) o funcionalidad/comportamiento (JavaScript) de una página web. "Hipertexto" se refiere a los enlaces que conectan páginas web entre sí, ya sea dentro de un solo sitio web o entre sitios web. Los enlaces son un aspecto fundamental de la Web.

HTML utiliza "comandos" para anotar texto, imágenes y otro contenido para su visualización en un navegador Web. El marcado HTML incluye "elementos" especiales como `<head>`, `<title>`, `<body>`, `<header>`, `<footer>`, `<article>`, `<section>`, `<p>`, `<div>`, `<span>`, `<img>`, `<aside>`, `<audio>`, `<canvas>`, `<datalist>`, `<details>`, `<embed>`, `<nav>`, `<output>`, `<progress>`, `<video>`, `<ul>`, `<ol>`, `<li>` y muchos otros[24].

En el principio del archivo se escribe el código para llamar a las librerías antes mencionadas en el subcapítulo 2.4, se agrega el código `"script src="` junto con la biblioteca que se quiere llamar en este caso se llaman las 4 bibliotecas y se termina la línea cerrando con el código `"</script>"` como se puede ver en las líneas de código 9 a 11 del anexo 1, también se agregó código para incluir un ícono y un título principal usando `<link>` y `<title>`, importante mencionar que este código se encuentra dentro del mando `<head></head>`, el elemento `<head>` en HTML contiene información

legible por máquina (metadatos) sobre el documento, como su título, scripts y hojas de estilo. haciendo referencia a que es la parte frontal superior de la página[25]

De manera consecuente se agregó el comando `<script>` de tipo texto JavaScript del api key que se utiliza para la visualización del mapa utilizando la herramienta de Google Maps, en el subcapítulo 3.2 se explica más sobre esta herramienta, se continua agregando código para estilo de tabla y fuente utilizando librería de Bootstrap y fuentes de Google Apis.

La línea de código 19 del **ANEXO 1** hace referencia a el funcionamiento correcto de las Apis de Google permitiendo las métricas de Google Analytics Google Analytics es un servicio de análisis web ofrecido por Google que rastrea e informa el tráfico del sitio web, actualmente como una plataforma dentro de la marca Google Marketing Platform.[26]

Continuando con la parte de diseño se agregan variables para definir los colores que se utilizaran en la parte visual del usuario. Como se puede ver en las líneas 24 a 30 del **ANEXO 1** se llama a las diferentes clases que se utilizaran para el manejo y visualización de los datos, más adelante se explicará el funcionamiento de estas clases que se pueden encontrar en los ANEXOS.

De manera consecuente se encuentran las funciones locales del código estas funciones locales, las funciones son uno de los bloques de construcción fundamentales en JavaScript. Una función en JavaScript es similar a un procedimiento: un conjunto de instrucciones que realiza una tarea o calcula un valor, pero para que un procedimiento califique como una función, debe tomar alguna entrada y devolver una salida donde haya alguna relación obvia entre la entrada y la salida. Para utilizar una función, debe definirla en algún lugar del ámbito desde el que desea llamarla.[27]

El cuerpo del archivo HTML comienza con el mando `<body></body>` se agrega un contenedor para incluir el título que se visualizará en la pestaña de la página web y la explicación principal del proyecto, se incluye los diferentes enlaces utilizando el mando `<a href=></a>`. En las líneas 30 a 33 del **ANEXO 2** se observa el código para visualizar la filtración de los datos a utilizar en las opciones que se necesita, en este caso la información se divide en generación y capacidad por entidad federativa, también se incluye un filtro para separar los estados por su región específica del SEN como se explica en el **subcapítulo 1.3**.

Para facilitar la interacción del usuario visualmente con los datos se agrega otro filtro ubicado en las líneas 37 a 41 del ANEXO 2 usando el id "Sort\_ranking\_by" para filtrar por orden alfabético y uso de energía.

En la parte inferior de la página se encuentra la separación de los datos en su manera "Básica" o "Explorativa", como estos datos se separan mediante botones se agrega línea de código con el método `onClick="SelectTab(#)"` dentro del comando `<nav></nav>`. Con el botón "Básica" solo visualizará la generación bruta por entidad federativa seleccionada, mediante una gráfica de barras. El botón "Explorativa" guía a la visualización de los datos referentes a la generación de energía por fuentes renovable y convencional, como se puede ver en las líneas de código 77 a 85 del **ANEXO 2** se agrega las opciones de cada tipo de energía que se trata de analizar y comparar.

Importante mencionar que en esta opción se agrega una pequeña explicación del coeficiente de determinación R2 el cual se utiliza para tener un dato de la exactitud de la relación de los puntos en la regresión lineal.

En la línea 101 del **ANEXO 2** se llama a la función "placeTooltip () ", las herramientas In-place se utilizan para mostrar cadenas de texto para objetos que se han recortado. La diferencia entre la información sobre herramientas ordinaria y la In-place es el posicionamiento. De forma predeterminada, cuando el puntero del mouse se desplaza sobre una región que tiene una información sobre herramientas asociada, la información sobre herramientas se muestra junto a la región. Sin embargo, la información sobre herramientas son ventanas y se pueden colocar en cualquier lugar que elija llamando a SetWindowPos.[28]

Crear una información sobre herramientas en In-place es una cuestión de colocar la ventana de información sobre herramientas para que se superponga a la cadena de texto, para incluir las variables que se utilizarán en este caso se seleccionó el estado de Puebla como visualización principal esto se refiere a que este estado será el primero que se cargará cuando se carga la página

Después de cargar, reformar y vincular los datos se llama a una función que incluye a la variable "initVis" como se puede ver en las líneas 111 a 117 del ANEXO 2 para leer estos valores se utiliza el comando de la librería d3, ver capítulo anterior, también se agrega un comando "EventHandler".

La clase EventHandler proporciona soporte para la generación dinámica de detectores de eventos cuyos métodos ejecutan una declaración simple que involucra un objeto de evento entrante y un objeto de destino. EventHandler está pensada para que la utilicen herramientas interactivas, como los creadores de aplicaciones, que permiten a los desarrolladores establecer conexiones entre beans. Normalmente, las conexiones se realizan desde un bean de interfaz de usuario (el origen del evento) a un bean lógico de aplicación (el destino). Las conexiones más efectivas de este tipo aíslan la lógica de la aplicación de la interfaz de usuario.[29]

Comenzando en la línea 121 y hasta la línea 126 se crean las instancias de los objetos Vis que se utilizarán, se anexa un comando para ocultar el bloque utilizando ".style.display"

También se agrega un comando de onSelectionchange a la variable map\_vis, como se puede ver en la línea 132, el evento selectionchange de la API de selección se desencadena cuando se cambia la selección actual de un documento. Este evento no es cancelable. El evento se puede controlar agregando un detector de eventos para selectionchange o usando el controlador de eventos onselectionchange.

Para evitar que la opción seleccionada no tenga el tipo de energía seleccionado se crea un "if" en todas las instancias de objetos vis con un controlador de eventos, la animación del Scroll o barra de movimiento de los estados se crea con una variable de posición y un comando "if", para construir las selecciones de cada estado u opción se le da un valor verdadero a la opción seleccionada y falso a las demás opciones para que no se muestren.

## **CAPITULO 3: DISEÑO DE APLICACIÓN WEB ESTÁTICA**

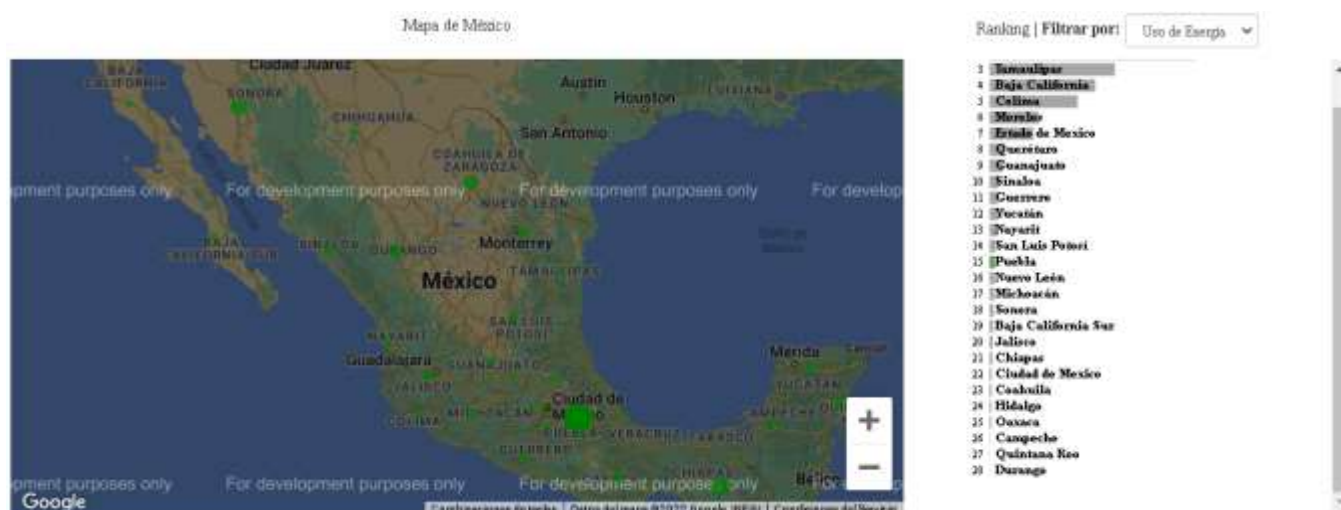
El diseño de la aplicación web estática se enfocó en la visualización de datos de generación bruta y capacidad efectiva por entidad federativa en México esta información se obtuvo directamente de la base de datos abiertos de la Secretaría de Energía SENER.

La aplicación web denominada “Sistema de Información Energética” divide en tres partes, la parte superior, central e inferior, en la parte superior se cuenta con tres opciones para visualizar datos, estas opciones se dividen en “Tipo de Información”, “Selección de tipo de energía” y “Filtración por regiones del SEN” estas opciones fueron evaluadas para el manejo y filtración de datos.



**Figura 3: Opciones principales en aplicación web.**

En la segunda parte de la aplicación o parte central se muestra una interfaz gráfica del mapa de México, del lado derecho del mapa se encuentran los estados de México, los podemos filtrar por su demanda o capacidad de mayor a menor o también por orden alfabético, como se observa en la **Figura 4** los puntos en el mapa representan el estado seleccionado, al ubicar el puntero en alguno de estos puntos se despliega la información del estado.

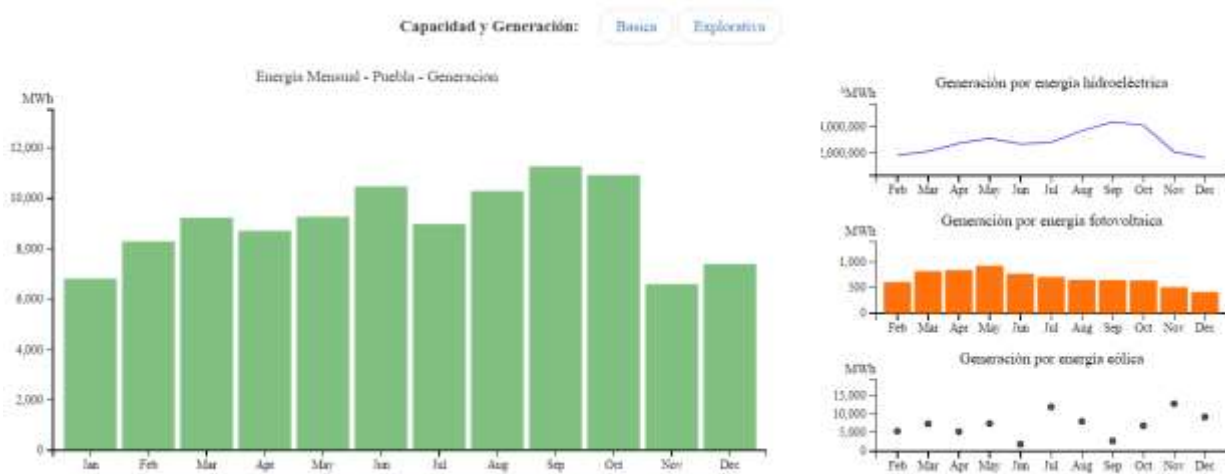


**Figura 4: Parte central visualización de mapa de México**

En la parte inferior se muestra una gráfica con los datos mensuales del último año, así como tres gráficos con datos meteorológicos, esta parte es la más importantes ya que cuenta con dos botones los cuales ofrecen diferentes combinaciones de datos para que se pueda visualizar los datos de una manera “Básica” o “Explicativa”. Como se muestra en la **Figura 5** en la opción de gráfica básica se visualizan los datos de generación o capacidad de cada mes del año en curso, en la opción



explorativa se muestra una regresión lineal entre los datos de generación o capacidad y datos específicos de generación por energía hidráulica, fotovoltaica y energía eólica.



**Figura 5: Parte inferior, visualización de datos en gráficas.**

### 3.1 Back-end del sistema.

El código de desarrollo de la visualización de cada uno de los elementos en la aplicación web se puede encontrar en la documentación del proyecto localizado en el siguiente repositorio: <https://github.com/SCastillo14/Energy-Visualization>.

Hay cuatro elementos en una visualización web típica:

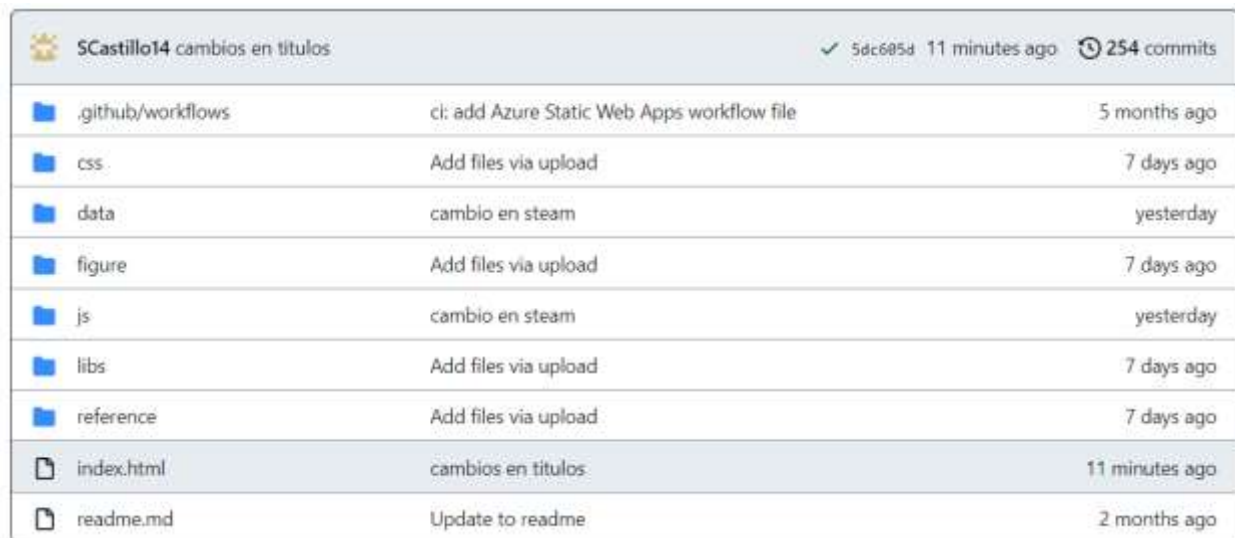
- Un esqueleto HTML, con marcadores de posición para nuestra programación de visualización.
- Hojas de estilo en cascada (CSS) que definen la apariencia (por ejemplo, anchos de borde, colores, tamaños de fuente, ubicación de bloques de contenido).
- JavaScript para construir la visualización.
- Datos para transformar.

Como menciona Kyran Dale en el libro “data visualization with python and JavaScript Crafting a Data-visualisation Toolchain for the Web” la entrega de los archivos HTML, CSS y JS

que se utilizan para crear una página web en particular (y cualquier archivo de datos, video multimedia, audio) se negocia entre un servidor y un navegador mediante el Protocolo de transferencia de hipertexto. HTTP proporciona una serie de métodos, el más utilizado es GET, que solicita un recurso web, recuperando datos del servidor si todo va bien o arrojando un error si no es así.

Una visualización web típica utiliza un esqueleto HTML, sobre el cual se construye la visualización con JavaScript. Por lo tanto, solo se declaró el documento HTML, el conjunto de caracteres Unicode de ocho bits y una etiqueta de cuerpo debajo de la cual agregar el contenido de nuestras páginas.

Como se ve en la **Figura 6** la estructura del repositorio se dividió en 7 carpetas el código principal del sistema se programó en un archivo “.html” con nombre index.html, este archivo contiene las principales fuentes y declaraciones de variables de texto para la visualización de los títulos, tablas y formas.



SCastillo14 cambios en títulos			✓ 5dc605d 11 minutes ago	🕒 254 commits
📁 .github/workflows	ci: add Azure Static Web Apps workflow file			5 months ago
📁 css	Add files via upload			7 days ago
📁 data	cambio en steam			yesterday
📁 figure	Add files via upload			7 days ago
📁 js	cambio en steam			yesterday
📁 libs	Add files via upload			7 days ago
📁 reference	Add files via upload			7 days ago
📄 index.html	cambios en títulos			11 minutes ago
📄 readme.md	Update to readme			2 months ago

**Figura 6: estructura de repositorio**



El repositorio también cuenta con archivo `readme.md` con instrucciones básicas del funcionamiento de la aplicación web, las carpetas “reference” como “libs” contienen librerías necesarias para el funcionamiento de la aplicación, como el desarrollo de la aplicación se enfocó en el lenguaje Javascript se incluyeron 6 archivos Javascript con los siguientes nombres:

- `Energyvis.js`
- `Explore1vis.js`
- `Mapvis.js`
- `Rankvis.js`
- `Toastr.js`
- `CleanEnergyvis.js`

Estos archivos localizados en la carpeta de nombre “js” están hechos para que las herramientas de la aplicación funcionen correctamente, estos archivos están conectados entre si en todo el código cada uno con un propósito específico, la carpeta con nombre “data” contiene todos los datos que se visualizan en la aplicación, los datos se encuentran en formato “.json”, este formato que significa Notación de objetos de JavaScript y es un formato de transferencia de datos. Es liviano y conveniente para trabajar en muchos idiomas, especialmente en JavaScript. Es solo una combinación de la matriz y la notación literal del objeto. La única diferencia de sintaxis entre JSON y el objeto literal es que los nombres de las propiedades deben estar entre comillas para que sean JSON válidos. En los objetos literales, las comillas solo se requieren cuando los nombres de las propiedades no son identificadores válidos.[30]

### 3.2 Costo de proyecto

Al tratarse de una aplicación web estática desarrollada en un entorno de nube de Azure se tiene que seleccionar un plan de hospedaje, como se observa en la **Figura 7**, para fines de demostrativos o proyectos personales en desarrollo el plan gratis resulta el más adecuado, conforme el proyecto crece se puede elevar el plan de hospedaje al siguiente nivel.[31]

Plan y características	 <b>Gratis</b> Para aficiones o proyectos personales	 <b>Estándar</b> Para aplicaciones de producción de uso general
Precio	Gratis	9.00 USD por aplicación al mes
Ancho de banda incluido	100 GB por suscripción	100 GB por suscripción
Uso de ancho de banda por encima del límite	Gratis	0.20 USD por GB por suscripción
Dominios personalizados	2 por aplicación	5 por aplicación
Certificados SSL	Gratis	Gratis
Autenticación personalizada	-	✓
Puntos de conexión privados	-	✓
Tamaño máximo de la aplicación	250 MB	500 MB
Entornos de ensayo	3	10
Azure Functions	Administrado	Administrado o incorpore el suyo
Borde de grado empresarial	-	17.52 USD por aplicación al mes

**Figura 7: Planes de Hospedaje**

Una de las principales características de la aplicación web desarrollada se enfoca en la visualización de la energía dentro del Sistema Eléctrico Nacional, como se menciona en el comienzo del Capítulo 3 en la parte central de la aplicación web se visualiza un mapa de México con diferentes herramientas que muestran la información deseada, esto es posible gracias a la API de Maps para JavaScript esta API cuenta con cuatro tipos de mapas básicos (mapa de rutas, satélite, pág. 43

híbrido y terreno), los cuales puedes modificar mediante capas y diseños, controles y eventos, y varios servicios y bibliotecas. La API de Maps JavaScript se carga con una etiqueta script, que se puede agregar de forma intercalada en tu archivo HTML o de forma dinámica con un archivo JavaScript independiente.[32]

La API de JavaScript de Google Maps utiliza un modelo de precios de pago por uso. Las API y los SDK de Google Maps Platform se facturan por SKU.

Se realiza un seguimiento del uso de cada SKU, y cualquier API o SDK puede tener más de un SKU de producto. El costo se calcula por

$$\text{Uso de SKU} \times \text{Precio por cada uso}$$

Una aplicación que muestra un mapa cargado con o sin una ID de mapa usando la API de JavaScript de Maps, o un mapa cargado con una ID de mapa y usando Maps SDK para Android o Maps SDK para iOS.

#### RANGO DE VOLUMEN MENSUAL

(Precio por CARGA DE MAPA)

0–100,000	100,001–500,000	500,000+
0.007 USD por cada uno (7.00 USD por 1000)	0.0056 USD por cada uno (5,60 USD por 1000)	Para precios en volumen se necesita contactar más información

## CAPITULO 4: RESULTADOS

Es muy importante mencionar que durante la realización de este proyecto se tuvieron en la mira los objetivos propuestos, el ambiente web desarrollado cumple con las necesidades de ser una herramienta de fácil acceso e interpretación, para explicar los resultados obtenidos se toma en cuenta primordialmente los datos manipulados por la aplicación web.

### 4.1 Datos

Como se menciona en el CAPITULO 1 y en el CAPITULO 3 la Secretaría de Energía (SENER) conduce la política energética del país para garantizar el suministro competitivo, suficiente, de alta calidad y ambientalmente sustentable de energía, también, es la principal fuente de información energética en el país.

La SENER en su portal de datos abiertos administra y actualiza las bases de datos de generación de energía, estos datos con nombre “Generación bruta de electricidad por entidad federativa” se pueden descargar de este portal en formato .csv al descargarlos como se observa en la Tabla 1 la base de datos cuenta con un orden en formato tabla definido por fecha y entidad federativa.

REALES-MENSUAL			
	ene-05	feb-05	mar-05
TOTAL	16498369.8	15577533.1	17328905.9
Baja California	820646.25	795704.87	879358.08
Baja California Sur	99113.58	89393.96	108156.58
Campeche	218720.02	211890.54	199603.7
Chiapas	456931.58	528953.66	474266.07
Chihuahua	836891.92	756232.14	848479.46
Coahuila	1440510.89	1388450.67	1782210.91
Colima	585835.82	531433.52	783958

Tabla 1: Datos descargados de página oficial SENER

El manejo de estos datos comienza con una filtración para obtener la información necesaria, en este caso los datos que se utilizaron están definidos por el periodo de tiempo de los años entre 2005 a 2021 y tienen una unidad de medida en Megawatts-hora ya que se trata de datos de generación de energía en un tiempo determinado el Megawatt es el múltiplo de la potencia activa, que equivale a un millón de watts; se abrevia MWh.

Al tratarse de una aplicación web desarrollada en el lenguaje JavaScript se utiliza el formato de texto .json o JavaScript Object Notation para hacer un mejor uso de estos datos, como se ve en la Figura 8 el acomodo de los datos está definido por los datos de generación en MWh y diferentes funciones que ayudarán a conectar estos datos con la visualización respectiva en la plataforma.

```
"Querétaro": {  
  "function": "region",  
  "generacion": [  
    {  
      "consumption": 345601.83,  
      "month": "2016-01-01"  
    },  
    {  
      "consumption": 266964.78,  
      "month": "2016-02-01"  
    }  
  ],  
  "area": 11699,  
  "longitud": -100.391,  
  "latitud": 20.591,  
  "capacidad": [  
    {  
      "month": "2016-01-01",  
      "consumption": 591  
    }  
  ],  
}
```

Figura 8: Formato de texto tipo .json

## 4.2 Visualización gráfica

El algoritmo de regresión lineal muestra una relación lineal entre una variable dependiente (x) y una o más variables independientes (y), dado que la regresión lineal muestra la relación lineal entre las dos variables esto da como resultado una solución al problema de encontrar la mejor línea recta a través de un conjunto de puntos.

El método de regresión lineal busca encontrar la mejor solución posible y aun cuando no se logra alcanzar el resultado exacto este es suficientemente cercano como para tomarlo como solución, básicamente la línea recta pasa lo más cercano posible de los puntos en este caso los datos proporcionados para poder encontrar su relación.

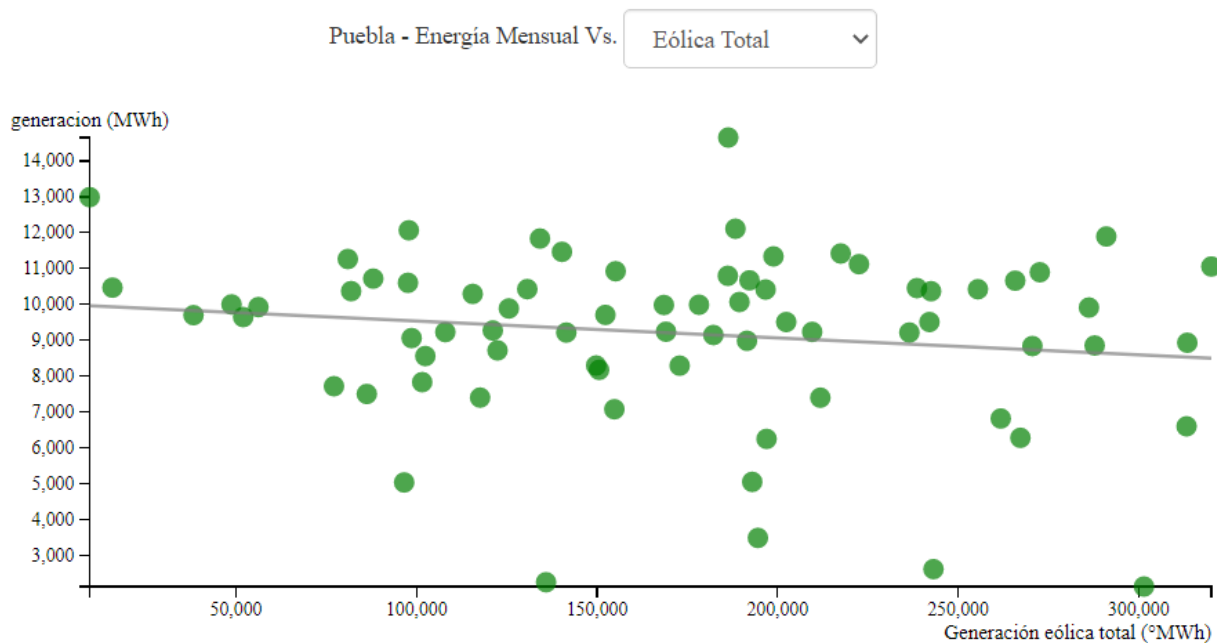


Figura 9: Resultado de regresión lineal.

Como se observa en la **Figura 9** la gráfica muestra la relación entre la generación bruta del estado de Puebla y la generación eólica total en México, para poder saber la medida estadística de que tanto se aproxima la línea a los puntos de datos se utiliza el coeficiente de determinación  $R^2$ .



Un coeficiente  $R^2$  de 1 indica que la línea de regresión se ajusta perfectamente a los datos, en este caso también se visualiza este coeficiente en la aplicación web, como se observa en la **Figura 10** el coeficiente de esta regresión lineal entre la generación bruta del estado de Puebla y la generación eólica total en México es solo de 0.02 .

### Resultado Regresión Lineal

**R2:** 0.02

**Equation:**  $y = -0.0047 * x + 10002.84$

Figura 10: Resultado regresión lineal

## CONCLUSIONES

Se encontraron diferentes resultados de la relación entre la generación por fuentes de energías limpias y la generación bruta por entidad federativa, dentro de los resultados obtenidos se logra concluir que en comparación con la generación por fuentes de energía convencionales como la producción de energía mediante ciclo combinado y termoeléctrica se tienen mayores coeficientes de medidas estadísticas de estas fuentes de generación.

Uno de los principales obstáculos observados durante el desarrollo de esta aplicación web fue la localización de datos específicos de generación, al no contar con la generación por municipio de cada estado se tiene una ventana de error en la aplicación, esta ventana puede ser atacada mediante bases de datos más robustas y con datos más específicos.

El desarrollo del proyecto de aplicación web “Sistema de Información Energética” llevó a grandes retos en cuanto a análisis de datos, se logró construir una plataforma para la visualización de datos energéticos amigable con el usuario y como objetivo principal se logró crear un entorno web de bajo costo en donde se puede hacer un análisis sobre relación entre las diferentes fuentes de energía que se utilizan en México actualmente.

Al igual que las diferentes herramientas que existen en el mercado para el análisis y visualización de datos, existen varias utilidades y diferente adaptación del usuario con el software o aplicación, es necesario seguir desarrollando el código fuente de la aplicación web de manera abierta para que se tenga un mayor alcance y mejor funcionamiento a las necesidades del usuario.

Se propone continuar desarrollando el trabajo atacando áreas de oportunidad como lo son:

**Bases de datos:** En cuanto al desarrollo de bases de datos de energía lo ideal sería una actualización de datos automática en tiempo real para obtener la mayor exactitud posible en el entendimiento del SEN y para el desarrollo de informes actualizados. Se propone, como objetivo inmediato a mejorar las bases de datos de generación y capacidad incluyendo separaciones de datos por hora, día, mes y año, esto mejoraría el funcionamiento de la aplicación y ayudaría a desarrollar más tipos de análisis.

Protocolos de seguridad:

**Protocolos de seguridad:** Tienen como objetivo guiar y acompañar al usuario en prevenir, atender y gestionar las distintas eventualidades o crisis que pueden poner en riesgo la integridad de la red lo cual permitirá el análisis constante de la comunicación de los datos, el protocolo de seguridad ayudará a resolver de manera exitosa una crisis determinada. Se propone desarrollar protocolos de seguridad web robustos, esto brindará medidas y reglas de prevención que serán diseñadas para ofrecer una mayor protección y control sobre los diversos datos que circulan o se transfieren mediante las redes de Internet. Con el fin de tener al usuario seguro, sin preocupación y satisfecho de que la información está segura.

**Desarrollo de comunidad:** El código de fuente del proyecto desarrollado es uno de los aspectos más importantes para la continuidad del desarrollo de la aplicación web.

Se propone crear un entorno comunitario donde los grupos de programadores contribuyan, mejoren o reutilicen el código abierto esta comunidad será divulgada en diferentes redes y canales web. Por otra parte aumentará la productividad de los programadores de desarrollar u código de software que les permita combinar capacidades como editar, crear, probar y empaquetar software en una aplicación fácil de usar

## ANEXO 1: Encabezado index.html

```
1.  <!DOCTYPE html>
2.  <html>
3.  <head lang="en">
4.  <meta charset="UTF-8">
5.  <link rel="shortcut icon" href="data/json/icon.png">
6.  <title>Sistema de Información Energética</title>
7.  <!-- Incoorporación de bibliotecas -->
8.  <script src="libs/d3/d3.min.js" charset="utf-8"></script>
9.  <script src="libs/jquery/jquery-2.1.1.min.js" charset="utf-8"></script>
10. <script src="libs/bootstrap/js/bootstrap.min.js" charset="utf-8"></script>
11. <script src="libs/queue/queue.v1.min.js"></script>
12. <script
    type="text/javascript"src="https://maps.googleapis.com/maps/api/js?key=AIzaSyB_S
    fhqfQajrVyTAYJCZwsAMevuBAODcdU&callback=initMap">
13. </script>
14. <!--Estilo de tablas-->
15. <link
    rel="stylesheet"type="text/css"href="libs/bootstrap/css/bootstrap.min.css">
16. <!-- Fuentes-->
17. <link
    href='http://fonts.googleapis.com/css?family=PT+Sans:400,700'rel='stylesheet'typ
    e='text/css'>
18. <!-- Google Analytics-->
19. <script>
    (function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=r;i[r]=i[r]||function(
    ){{(i[r].q=i[r].q||[]).push(arguments)},i[r].l=1*new Date();a=s.createElement(o),
    m=s.getElementsByTagName(o)[0];a.async=1;a.src=g;m.parentNode.insertBefore(a,m)}})
    (window,document,'script','/www.google-
    analytics.com/analytics.js','ga');ga('create', 'UA-58380235-4',
    'auto');ga('send', 'pageview');
20. </script>
21. <!--Código para definir colores-->
22. <script type='text/javascript' >
    var generacionColor = "green"; <!--"#ABDB25"-->
    var chilledWaterColor = "blue";
    var steamColor = "red";
    var allColor = "purple"
23. </script>
```

```

24. <!--vis classes-->
25. <script src = "js/mapvis.js"></script>
26. <script src = "js/energyvis.js"></script>
27. <script src = "js/CleanEnergyvis.js"></script>
28. <script src = "js/rankvis.js"></script>
29. <script src = "js/explore1vis.js"></script>
30. <script src="js/toastr.js"></script>
31. <!--Estilos de tablas-->
32. <link rel="stylesheet" type="text/css" href="css/myStyle.css">
33. <link href="css/toastr.css" rel="stylesheet" type="text/css" />
34. <!--Funciones locales-->
35. <script type="text/javascript">

    function SelectTab(id)
    {
        if(id == 1)
        {
            document.getElementById('tab1').style.display = "block";
            document.getElementById('tab2').style.display = "none";
        }
        if(id == 2)
        {
            document.getElementById('tab1').style.display = "none";
            document.getElementById('tab2').style.display = "block";
        }
    }

    function tooltipText() {
        return "test";
    }

36. </script>
37. </head>

```

## ANEXO 2: Cuerpo index.html

```
1. <body>
2. <div class="container" style="text-align :center">

3. <div style="background-color: rgb(255, 255, 255); padding-bottom: 0px"
   class="navbar-fixed-top">
4. <h1>Sistema de Información Energética</h1>

5. <div class="row">
6. <p> La siguiente herramienta visualiza datos obtenidos de páginas oficiales
   mexicanas.
7. |<a href="#" id="Read_me" data-toggle="modal" data-
   target="#readMeModal">Instrucciones</a> |
8. |<a href="https://www.datos.gob.mx/busca/organization/sener" target="_blank"
   id="">Datos Abiertos SENER</a>
9. |<a href="https://sie.energia.gob.mx/bdiController.do?action=about"
   target="_blank" id="">SIE SENER</a>
10. |<a href="https://github.com/SCastillo14/Energy-
   Visualization/blob/main/readme.md" target="_blank" id="Youtube_demo">Repositorio
   Github</a>|
11. </p>
12. </div>
13. <div class="modal fade" id="readMeModal">
14. <div class="modal-dialog">
15. <div class="modal-content">
16. <div class="modal-header">
17. <button type="button" class="close" data-dismiss="modal" aria-
   hidden="true">&times;</button><h4 class="modal-title"
   id="myModalLabel">Instructions</h4>
18. </div>
19. <div class="modal-body" align="left">Usa la barra de control para filtrar
   estados y seleccionar el tipo de energía. <br> Seleccione un estado en el mapa o
   en el gráfico de clasificación. <br> Pase el cursor sobre una etiqueta o título
   de gráfico para obtener más información.
20. </div>
21. <div class="modal-footer">
22. <button type="button" class="btn btn-primary" data-
   dismiss="modal">Close</button>
23. </div>
24. </div>
25. </div>
26. </div>
```

```

27. <div class="row">
28. <form id="form_opt" class="form-inline">
29. <label id = "Show_Estate_with_data_of" class="control-label gray-tooltip">Tipo
    de información:</label>
30. <select id="Estate_opt" class="form-control selectWidth controls">
31. <option value="generacion" selected>Generación por entidad federativa</option>
32. <option value="Capacidad">Capacidad por entidad federativa</option>
33. <option value="all">Todos los datos</option>
34. </select>
35. <label id = "Select_energy_type" class="control-label gray-tooltip">Selecciona
    el tipo de energía:</label>
36. <select id="energy_opt" class="form-control selectWidth controls">
37. <option value="generacion" selected>Generación bruta por entidad
    federativa</option>
38. <option value="Capacidad">Capacidad instalada por entidad federativa</option>
39. </select>
40. <label id = "Filter_Estate_by_function" class="control-label gray-tooltip"
    >Filtrado por regiones del SEN:</label>
41. <select id="function_opt" class="form-control selectWidth controls">
42. <option value="all" selected>Todos</option>
43. <option value="Occidental">Occidental</option>
44. <option value="Baja California">Baja California</option>
45. <option value="Noroeste">Noroeste</option>
46. <option value="Baja California Sur">Baja California Sur</option>
47. <option value="">Oriental</option>
48. <option value="Norte">Norte</option>
49. <option value="Peninsular">Peninsular</option>
50. <option value="Central">Central</option>
51. </select>
52. </form>
53. </div>
54. <hr style="margin-top: 15px; margin-bottom: 0px">
55. </div>
56. <div class="row" style="margin-top: 165px">
57. <div class="col-md-8"><span id="Map" class = "gray-tooltip-right">Mapa de
    México</span></div>
58. <div class="col-md-4">
59. <form class="form-inline">
60. <span id="Annual_Energy_Ranking" class = "gray-tooltip-left">Ranking |
    </span>
61.

```

```

37. <label id = "Sort_ranking_by" class="gray-tooltip">Filtrar por:</label>
38.   <select id="rank_opt" class="form-control selectWidth controls input-sm">
39. <option value="energyUse" selected>Uso de Energía</option>
40. <option value="EstateName">A-Z</option>
41.   </select>
42. </form>
43. </div>
44. </div>
45. <div class="row" style="margin-top: 10px">
46. <div class="col-md-8" id="mapVis"></div>
47. <div class="col-lg-4 col-md-4 content" id="rankVis"></div>
48. </div>
49. <div class="row">
50.   <nav>
51.     <ul class="pager">
52.       <span id="Energy_Use_and_Weather" class = "controls subtitle gray-
tooltip">Capacidad y Generación: </span>
53.       <li><a href="#" onClick="SelectTab(1)">Basica</a></li>
54.       <li><a href="#" onClick="SelectTab(2)">Explorativa</a></li>
55.     </ul>
56.   </nav>
57. </div>
58. <div class="row" id="tab1" style="display:block;">
59. <div class="col-md-8" id="energyVis">
60.   <span id="Monthly_Energy_Use" class = "gray-tooltip">
61.   <span >Energía Mensual - </span>
62.   <span id="EstateName" style="text-transform:capitalize">EstateName</span>
63.   <span id="EnergyType" style="text-transform:capitalize">>EnergyType</span>
64.   </span>
65. </div>
66. <div class="col-md-4" id="tempVis"></div>
67. <div class="col-md-4" id="solarVis"></div>
68. <div class="col-md-4" id="windVis"></div>
69. </div>

```



```

70. <div class="row" id="tab2" style="display:block;">
71. <div class="col-md-8" id="explore1Vis">
72.     <form class="form-inline" style="text-transform:capitalize">
73.         <span id="Monthly_energy_use_vs" class = "gray-tooltip">
74.             <span id="EstateName_explore1Vis">All</span>
75.             <span> - Energía mensual vs.</span>
76.         </span>
77. <select id="explore1Vis_x_opt" class="form-control selectWidth controls"
78. style="text-transform:capitalize">
79.     <option value ="Hidroeléctrica" >Hidroeléctrica</option>
80.     <option value ="dehumidification">Geotermica</option>
81.     <option value ="Generación eólica total" selected>Eólica total</option>
82.     <option value ="Ciclo Combinado">Ciclo combinado</option>
83.     <option value ="Fotovoltaica">Fotovoltaica</option>
84.     <option value ="wind speed">Eólica por CFE</option>
85.     <option value ="min temperature">Eólica por PIE</option>
86.     <option value ="Termoeléctrica">Termoelectrica</option>
87. </select>
88. </form>
89. </div>
90. <div><br>Resultado Regresión Lineal </div>
91. <div align = "left" id = "explore1Equation"></div>
92.     <hr>
93. <div align = "left">
    El coeficiente de determinación R2 es una medida estadística de qué tan
    bien la línea de regresión se aproxima a los puntos de datos reales. Un R2 de 1
    indica que la línea de regresión se ajusta perfectamente a los datos.<hr>
    Según las observaciones realizadas en la investigación, la generación
    eléctrica mediante energías limpias tiene una creciente relación lineal con la
    generación. La generación de energía por fuentes convencionales sigue teniendo
    una mayor relación con la generación bruta total en México. Revisa el proyecto
    desarrollado aquí
94. <a href = "https://github.com/SCastillo14/Energy-Visualization"
    target="_blank">aquí</a>
    El uso, reproducción, distribución y modificación de este código está sujeto a
    los términos y condiciones de la licencia MIT, disponible en.
95. <a href = "http://www.opensource.org/licenses/mit-license.php"
    target="_blank">aquí</a> Se siguió la plantilla de desarrollo de los autores
    Benjamin Norman | Bing Wang | Bin Yan.
96. </div>
97. </div>
98. </div>
99. </div>

```

```

100. <script>
101. $(function(){ // esta función se llama después de que el documento HTML esté
    completamente cargado
102. //tooltip
103. placeTooltip();
104. // variables que mantienen el conocimiento global de los datos
105. var defaultBuilding = "Puebla";
106. var monthlyEnergy = [];
107. var weather = [];
108. var option = {"EstateName ": defaultBuilding, "buildingEnergyType": '',
    "energyType": '', "buildingFunction": '', "sortType": ''};
109. var buildingLocation = [];
110. // llamar a esta función después de cargar, reformatear y vincular los datos a
    las variables
111. var initVis = function(){
112. // Leer valores de selección
113. d3.select("#energy_opt").property('disabled', true);
114. readOption();
115. d3.select("#EstateName").text(option.EstateName);
116. d3.select("#EnergyType").text(option.energyType);
117. d3.select("#EstateName_explore1Vis").text(option.EstateName);
118. //Creación de variable eventHandler --> DONE :)
119. var MyEventHandler = new Object();
120. //Crea instancias de todos los Objetos Vis aquí
121. var map_vis = new MapVis(d3.select("#mapVis"), monthlyEnergy, option,
    MyEventHandler);
122. var rank_vis = new RankVis(d3.select("#rankVis"), monthlyEnergy, option,
    MyEventHandler);
    var energy_vis = new EnergyVis(d3.select("#energyVis"), monthlyEnergy, option,
    MyEventHandler);
123. var temp_vis = new CleanEnergyvis(d3.select("#tempVis"), weather,
    "Hidroeléctrica", MyEventHandler);
124. var solar_vis = new CleanEnergyvis (d3.select("#solarVis"), weather, "Solar",
    MyEventHandler);
125. var wind_vis = new CleanEnergyvis (d3.select("#windVis"), weather, "Wind",
    MyEventHandler);
126. var explore1_vis = new Explore1Vis(d3.select("#explore1Vis"),
    d3.select("#explore1Equation"), monthlyEnergy, weather, option);
127. // Esconder Bloque
128. document.getElementById('tab2').style.display = "none";
129. // vincular eventHandler a Vis Objects
130. $(MyEventHandler).bind("selectionChanged", function (event, EstateName) {
131. // llamar a la función de cambio en cada vis que necesita cambiar
132. map_vis.onSelectionChange(EstateName);

```

```

133. //se añade Bin
134. option.EstateName = EstateName;
135. // evitar que la opción seleccionada no tenga el tipo de energía seleccionado
136. if(!monthlyEnergy[option.EstateName][option.energyType]) {
137. option.EstateName = defaultBuilding;}
138. energy_vis.onSelectionChange(option);
139. explore1_vis.onSelectionChange();
140. rank_vis.onSelectionChange();
141. d3.select("#EstateName").text(option.EstateName);
142. d3.select("#EstateName_explore1Vis").text(option.EstateName);
143. // Animación de Scroll
144. var top = rank_vis.getYPosition(EstateName) - 320;
145. if (top < 0) top = 0;
146. $("#rankVis").animate({ scrollTop: top} );
147. });
148. // Aquí los cambios de selección
149. d3.select("#form_opt").on("change", function() {
150. // Listas para construir las selecciones
151. if (d3.select("#Estate_opt").property('value') != "all") {
152. d3.select("#energy_opt").property('value',
153. d3.select("#Estate_opt").property('value'));
154. d3.select("#energy_opt").property('disabled', true);
155. }
156. else {d3.select("#energy_opt").property('disabled', false);}
157. readOption();
158. // evitar que la opción seleccionada no tenga el tipo de energía seleccionado
159. if(!monthlyEnergy[option.EstateName][option.energyType]) {
160. option.EstateName = defaultBuilding; toastr.options = {
161. "closeButton": false,
162. "debug": false,
163. "newestOnTop": false,
164. "progressBar": false,
165. "positionClass": "toast-top-left",
166. "preventDuplicates": false,
167. "onclick": null,
168. "showDuration": "300",
169. "hideDuration": "1000",
170. "timeOut": "3000",
171. "extendedTimeOut": "1000",
172. "showEasing": "swing",
173. "hideEasing": "linear",
174. "showMethod": "fadeIn",
175. "hideMethod": "fadeOut"
176. }

```

```

175. toastr.info('Selected building does not have selected energy type.', 'Reset to
    default building.')
176. }
177. // pasar las opciones seleccionadas a los métodos onSelectChange para cada vis
    para actualizarlas en función de la selección
178. energy_vis.onSelectionChange(option);
179. map_vis.onSelectionChange(option.EstateName);
180. explore1_vis.onSelectionChange();
181. rank_vis.onSelectionChange();
182. if (option.EstateName != "null") {
183.     d3.select("#EstateName").text(option.EstateName);}
184.     else {d3.select("#EstateName").text("No Building Selected");}
185.     d3.select("#EnergyType").text(option.energyType);
186.     d3.select("#EstateName_explore1Vis").text(option.EstateName);
187. // desplácese hacia atrás en caso de que solo se muestre en blanco
188. var top = rank_vis.getYPosition(option.EstateName) - 320;
189. if(!top || top < 0) {top = 0}
190.     $("#rankVis").animate({ scrollTop: top} );
191. });
192. d3.select("#rank_opt").on("change", function() {
193.     readOption();
194.     rank_vis.onSelectionChange();
195.     var top = rank_vis.getYPosition(option.EstateName) - 320;
196.     if(!top || top < 0) {top = 0}
197.     $("#rankVis").animate({ scrollTop: top} );
198. });
199. // Explore 1 Vis onchange
200. d3.select("#explore1Vis").on("change", function () {
201.     explore1_vis.onSelectionChange()
202. });
203. }
204. // llame a esta función después de cargar ambos archivos; el error debe ser
    "nulo" si no hay error
205. var dataLoaded = function (error, _monthlyEnergy, _weather, _buildingLocation) {
206.     if (!error) {
207.         monthlyEnergy = _monthlyEnergy;
208.         weather = _weather;
209.         buildingLocation = _buildingLocation;
210.         initVis();
211.     }
212. };

```

```

213. var startHere = function(){
214. //TODO: aqui se cargan los datos utilizando "dataLoaded" afterwards
215. queue()
216.     .defer(d3.json, "data/json/monthlyEnergy_nameAsKey_0415.json")
217.     .defer(d3.json, "data/json/weather_0423.json")
218.     .await(function(error, file1, file2) { dataLoaded(error, file1, file2);});
219. }
220. startHere();
221. var readOption = function() {
222. // Read selection values
223. option.buildingEnergyType = d3.select("#Estate_opt").property('value');
224. option.energyType = d3.select("#energy_opt").property('value');
225. option.buildingFunction = d3.select("#function_opt").property('value');
226. option.sortType = d3.select("#rank_opt").property('value');
227. };
228. $('#readMeModal').on('show.bs.modal', function () {
229. $(this).find('.modal-body').css({
230.     width:'auto', //probablemente no se necesite
231.     height:'auto', //probablemente no se necesite
232.     'max-height':'100%'
233. });
234. $(this).find('.modal-dialog').css({
235.     width:'450px' //probablemente no se necesite
236. });
237. });
238. });
239. // ##### Funciones de ayuda #####
240. var getInnerWidth = function(element) {
241.     var style = window.getComputedStyle(element.node(), null);
242.     return parseInt(style.getPropertyValue('width'));
243. };
244. var linearRegression = function(x, y){
245.     var lr = {};
246.     var n = y.length;
247.     var sum_x = 0;
248.     var sum_y = 0;
249.     var sum_xy = 0;
250.     var sum_xx = 0;
251.     var sum_yy = 0;

```

```

252.   for (var i = 0; i < y.length; i++) {
253.       sum_x += x[i];
254.       sum_y += y[i];
255.       sum_xy += (x[i]*y[i]);
256.       sum_xx += (x[i]*x[i]);
257.       sum_yy += (y[i]*y[i]);
258.   }
259.   lr['slope'] = (n * sum_xy - sum_x * sum_y) / (n*sum_xx - sum_x * sum_x);
260.   lr['intercept'] = (sum_y - lr.slope * sum_x)/n;
261.   lr['r2'] = Math.round(Math.pow((n*sum_xy - sum_x*sum_y)/Math.sqrt((n*sum_xx -
sum_x*sum_x)*(n*sum_yy-sum_y*sum_y)),2) * 100) / 100;
262.   return lr;
263. }
264. var placeTooltip = function() {
265.   $('#Energy_Use_and_Weather').tooltip({ placement : 'top',
266.   title: "Selecciona el tipo de visualizacion Basica | Explorativa"});
267.   $('#Show_buildings_with_data_of').tooltip({ placement : 'bottom',
268.   title: "Selecciona el tipo de información que necesites."});
269.   $('#Select_energy_type').tooltip({ placement : 'bottom',
270.   title: "Al seleccionar el tipo de energía se mostrará la generación específica
del proceso seleccionado."});
271.   $('#Filter_buildings_by_function').tooltip({placement : 'bottom',
272.   title: "Al seleccionar la filtración de datos por regiones del SEN podrás
encontrar las 8 regiones interconectadas en el SEN"});
273.   $('#Sort_ranking_by').tooltip({placement : 'bottom',
274.   title: "Filtra los datos por orden de mayor o menor y orden alfabetico"});
275.   $('#Map').tooltip({ placement : 'right',
276.   title: "Coloca el mouse en los puntos marcados en cada estado para vizualizar
sus datos."});
277.   $('#Annual_Energy_Ranking').tooltip({ placement : 'left',
278.   title: "Show the ranking of EUI (Energy Use Intensity, energy use normalized by
area) of the selected energy type and building function."});
279.   $('#Monthly_Energy_Use').tooltip({placement : 'top',
280.   title: "Energía mensual del año 2021."});
281.   $('#Monthly_energy_use_vs').tooltip({placement : 'top',
282.   title:'Seleccione una variable climática para explorar la relación entre el uso
de energía y el clima. ' + 'Use el control en la parte superior para cambiar el
tipo de uso de energía.'});
283. </script>
284. </body>
285. </html>

```

## ANEXO 3: energyvis.js

```
1.  var color_energy = {"Capacidad": "blue", "steam": "red", "generacion":  
    "green", "HiDrO": "yellow"};  
2.  EnergyVis = function(_parentElement, _data, _option, _eventHandler){  
3.      this.parentElement = _parentElement;  
4.      this.data = _data;  
5.      this.eventHandler = _eventHandler;  
6.      this.displayData = [];  
7.      this.option = _option;  
8.      // se definen constantes  
9.      this.margin = {top: 20, right: 50, bottom: 20, left: 80},  
10.     //this.title = -15,  
11.     this.width = getInnerWidth(this.parentElement) - this.margin.left -  
    this.margin.right,  
12.     this.height = 350 - this.margin.top - this.margin.bottom;  
13.  
14.     this.initVis();  
15. }  
16. /*** Método que configura el SVG y las variables ***/  
17. EnergyVis.prototype.initVis = function(){  
18.     var that = this;  
19.     // construye diseño SVG  
20.     this.svg = this.parentElement.append("svg")  
21.         .attr("width", this.width + this.margin.left + this.margin.right)  
22.         .attr("height", this.height + this.margin.top + this.margin.bottom)  
23.         .append("g")  
24.         .attr("transform", "translate(" + this.margin.left + "," +  
    this.margin.top + ")");  
25.     // crea ejes y escalas  
26.     this.x = d3.scale.ordinal()  
27.         .rangeRoundBands([0, this.width], .1);  
28.     this.y = d3.scale.linear()  
29.         .range([0, this.height]);  
30.     this.color = d3.scale.category20();  
31.     this.xAxis = d3.svg.axis()  
32.         .scale(this.x)  
33.         .orient("bottom")  
34.         .tickFormat(function(d, i) { return that.displayData.time[i]; });  
35.     this.yAxis = d3.svg.axis()  
36.         .scale(this.y)  
37.         .orient("left");
```

```

38. // Agregar elementos visuales de ejes
39.     this.svg.append("g")
40.         .attr("class", "x axis")
41.         .attr("transform", "translate(0," + this.height + ")");
42.     this.svg.append("g")
43.         .attr("class", "y axis")
44.         .append("text");
45. // filtrar, agregar, modificar fecha
46.     this.wrangleData(this.option.EstateName, this.option.energyType);
47. // llamar al método de actualización
48.     this.updateVis(this.option.energyType);
49. }
50. /** la función de dibujo: debe usar la selección D3, ingresar, salir */
51. EnergyVis.prototype.updateVis = function(_option){
52.     var that = this;
53. // actualiza escalas
54.     this.x.domain(this.displayData.time.map(function(d, i) { return i; }));
55.     this.y.domain([0, 1.2 * d3.max(this.displayData.data, function(d) { return
56. d; })]).range([this.height, 0]);
57. // actualización de la gráfica
58. // Unión de datos
59.     var bar = this.svg.selectAll(".bar")
60.         .data(this.displayData.data);
61. // crear un soporte div para el tooltip
62.     var div = d3.select("body").append("div")
63.         .attr("class", "tooltip_svg_weather")
64.         .style("opacity", 0);
65. // Agregar nuevos grupos de barras, si es necesario
66.     var bar_enter = bar.enter().append("g");
67. // Agregue un rect y un texto solo para el conjunto Enter (new
68. g) bar_enter.append("rect")
69.     .attr("class", "verticalEnergyBar");
70. // Agregar atributos (posición) a todas las barras
71.     bar
72.         .attr("class", "bar");
73. // Retire las barras adicionales
74.     bar.exit()
75.         .remove();
76. // Actualice todos los rectos y textos internos (tanto actualizar como ingresar
77. conjuntos)
78.     bar.select("rect")
79.         .attr("width", that.x.rangeBand())
80.         .style("fill", function() {

```



```

78.         return color_energy[that.option.energyType];
79.     })
80.     .transition()
81.     .attr("x", function(d, i) { return that.x(i)})
82.     .attr("y", function(d) {return that.y(d)})
83.     .attr("height", function(d, i) {
84.         return that.height - that.y(d);
85.     })
86. // Código para que se actualicen los ejes
87.     this.svg.select(".y.axis")
88.         .call(this.yAxis)
89.         .select("text")
90.         .attr("y", -15)
91.         .attr("x", 0)
92.         .attr("dy", ".71em")
93.         .style("text-anchor", "end")
94.         .style("font-size", "12px")
95.         .text(function () { if (_option == "generacion") { return " MWh "; }
96.             if (_option == "Capacidad") { return " MWh "; }
97.             if (_option == "steam") { return "MWh"; }
98.             if (_option == "HiDrO") { return "MWh"; } } });
99.     this.svg.select(".x.axis")
100.        .call(this.xAxis);
101. }
102. /*** Es llamado por el controlador de eventos y debe crear una nueva agregación
de datos agregados mediante la función "agregar (filtro)". El filtro debe
definirse aquí***/.
103. * @param selection
104. */
105. EnergyVis.prototype.onSelectionChange = function (_option){
106. // filtrar, agregar, modificar fecha
107.     this.wrangleData(this.option.EstateName, this.option.energyType);
108. // llamar al método de actualización
109.     this.updateVis(this.option.energyType);
110. }
111. /**Método para disputar los datos. En este caso toma un objeto de opciones
112. * @param _filterFunction - una función que filtra datos o "null" si no hay
113. */
114. EnergyVis.prototype.wrangleData= function(_EstateName, _energyType){
115. // displayData debe contener los datos que se visualizan
116.     this.displayData = this.filterAndAggregate(_EstateName, _energyType);
117. }

```

```

118.  /**** La función agregada que crea los conteos para cada edad para un filtro
      dado***/.
119.  /**@param _filter - Un filtro puede ser, por ejemplo, una función que solo es
      verdadera para datos de un rango de tiempo dado * @returns {Array|*}**//
120.  EnergyVis.prototype.filterAndAggregate = function(_EstateName, _energyType){
121.  // Establezca el filtro en una función que acepte todos los elementos
122.  // SOLO si el parámetro _filter NO es nulo, use este parámetro
123.      var filter = function(){return true;}
124.      var dateFormatter = d3.time.format("%Y-%m-%d");
125.      var monthNameFormat = d3.time.format("%b");
126.      if (_EstateName != "null"){
127.          filter = _EstateName;
128.      }
129.      var filteredData;
130.      if (_EstateName != "null"){
131.          if (_energyType == "generacion"){
132.              filteredData = this.data[filter].generacion;
133.          }
134.          else if (_energyType == "Capacidad"){
135.              filteredData = this.data[filter]["Capacidad"]; // default
136.          }
137.          else if (_energyType == "steam"){
138.              filteredData = this.data[filter].steam; // default
139.          }
140.          else if (_energyType == "HiDrO"){
141.              filteredData = this.data[filter]["HiDrO"]; // default
142.          }
143.      }
144.      // una variante más moderna de esta construcción sería:
145.      // var filter = _filter || function(){return true;}
146.      var area = 0;
147.      if (_EstateName != "null"){
148.          area = this.data[this.option.EstateName]['area'];
149.      }
150.      var res = {};
151.      res.time = [];
152.      res.data = [];

```

```

153.         if (_buildingName != "null"){
154.             for (i = 0; i < filteredData.length; i++) {
155.                 if (dateFormatter.parse(filteredData[i].month) <
dateFormatter.parse("2022-06-01") && dateFormatter.parse(filteredData[i].month)
>= dateFormatter.parse("2021-01-01")) {
156.                     res.time.push(monthNameFormat(dateFormatter.parse(filteredData[i
].month))));
157.                     res.data.push(filteredData[i].consumption / area * 1000);
158.                 }
159.             }
160.         }
161.         else {
162.             res.time = ["Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug",
"Sep", "Oct", "Nov", "Dec"];
163.             res.data = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0];
164.         }
165.
166.         return res;
167.     }
168.

```

## ANEXO 4: explore1vis.js

```
1.  /* Se siguió la plantilla vis de init - wrangle - update
2.  /** @param _parentElement -- el elemento HTML o SVG (nodo D3) al que adjuntar
    la vis
3.  * @param _data -- la matriz de datos
4.  * @param _eventHandler -- el objeto de manejo de eventos para emitir datos
    (consulte la tarea 4)
5.  * @constructor
6.  */
7.  var RADIUS = 6;
8.  var DURATION = 500;
9.  var dateFormatter = d3.time.format("%Y-%m-%d");
10. var dateTimeFormatter = d3.time.format("%Y-%m-%d %H:%M");
11. var xVariable;
12. var color_energy = {"Capacidad": "blue", "steam": "red", "generacion": "green"};
13. Explore1Vis = function(_parentElement, _textElement, _data, _weather, _option){
14.     var that = this;
15.     this.parentElement = _parentElement;
16.     this.textElement = _textElement;
17.     this.data = _data;
18.     this.weather = _weather;
19.     this.displayData = [];
20.     this.option = _option;
21.     // se definen constantes
22.     this.margin = {top: 40, right: 60, bottom: 40, left: 70};
23.     this.width = getInnerWidth(this.parentElement) - this.margin.left -
    this.margin.right;
24.     this.height = 340 - this.margin.top - this.margin.bottom;
25.     this.initVis();
26. };
27. /**** Método que configura el SVG y las variables ****/
28. Explore1Vis.prototype.initVis = function(){
29.     var that = this;
30.     // construye diseño SVG
31.     this.svg = this.parentElement.append("svg")
32.         .attr("width", this.width + this.margin.left + this.margin.right)
33.         .attr("height", this.height + this.margin.top + this.margin.bottom)
34.         .append("g")
35.         .attr("transform", "translate(" + this.margin.left + "," +
    this.margin.top + ")");
36.     // crea ejes y escalas
37.     this.y = d3.scale.linear()
38.         .range([this.height, 0]);
39.     this.x = d3.scale.linear()
40.         .range([0, this.width]);
```

```

41.  this.xAxis = d3.svg.axis()
42.      .scale(this.x)
43.      .orient("bottom");
44.  this.yAxis = d3.svg.axis()
45.      .scale(this.y)
46.      .orient("left");
47.  // Agregar elementos visuales de ejes
48.  this.svg.append("g")
49.      .attr("class", "x axis")
50.      .attr("transform", "translate(0," + this.height + ")")
51.      .append("text");
52.  this.svg.append("g")
53.      .attr("class", "y axis")
54.      .append("text");
55.  this.svg.append("text")
56.      .attr("class", "R2")
57.  // filtrar, agregar, modificar datos
58.  this.wrangleData(null);
59.  // llamar al método de actualización
60.  this.updateVis();
61.  };
62.  /** Método para disputar los datos. En este caso toma un objeto de opciones
63.   * @param _filterFunction - una función que filtra datos o "null" si no hay */
64.  Explore1Vis.prototype.wrangleData= function(_filterFunction){
65.  // displayData debe contener los datos que se visualizan //this.displayData =
  this.filterAndAggregate(_filterFunction);
66.      xVariable = d3.select("#explore1Vis_x_opt").property('value');
67.      this.displayData = [];
68.  //// es posible que pueda pasar algunas opciones, si no pasa las opciones,
  establezca las opciones predeterminadas, el valor predeterminado es: var options
  = {filter: function(){return true;}}
69.  //var options = _options || {filter: function(){return true;}};
70.      var data_y = this.data[this.option.EstateName][this.option.energyType];
71.      var data_x = this.weather["monthly"];
72.      var area = this.data[this.option.EstateName]['area'];
73.      console.log(data_x)
74.      var k = 0;
75.      for (i = 0; i < data_y.length; i++) for (j = k; j < data_x.length; j++) {
76.          var y_time = dateFormatter.parse(data_y[i].month);
77.          var x_time = dateTimeFormatter.parse(data_x[j].time);
78.          if (y_time.getYear() == x_time.getYear() && y_time.getMonth() ==
  x_time.getMonth()) {

```

```

79.   this.displayData.push({"x": data_x[j][xVariable],
80.                           "y": data_y[i]["consumption"]/area * 1000,
81.                           "month": data_y[i].month.split("-")[0] + "-" +
data_y[i].month.split("-")[1]});
82.       k = j;
83.       continue;
84.   }
85. }
86.   console.log(this.displayData)
87. };
88. /** la función de dibujo: debe usar la selección D3, enter, exit ***/
89. Explore1Vis.prototype.updateVis = function(){
90.   // Es posible que pueda pasar algunas opciones como parameter _option, pero no
es necesario para resolver la tarea.
91.   // var options = _options || {};
92.   var that = this;
93.   // updates scales
94.   this.y.domain([d3.min(this.displayData, function(d) { return d.y; }),
95.                  d3.max(this.displayData, function(d) { return d.y; })]);
96.   this.x.domain([d3.min(this.displayData, function(d) { return d.x; }),
97.                  d3.max(this.displayData, function(d) { return d.x; })]);
98.   //this.color.domain(this.displayData.map(function(d) { return d.type }));
99.   // updates axis
100.   var unit = {"Hidroeléctrica": "(" + String.fromCharCode(176) + "MWh)",
"dehumidification": "(MWh)",
101.               "Generación eólica total": "(" + String.fromCharCode(176) + "MWh)",
"Ciclo Combinado": "(" + String.fromCharCode(176) + "MWh)",
102.               "Fotovoltaica": "(MWh)", "wind speed": "(MWh)",
103.               "generacion" : "(MWh)", "Capacidad": "(MWh)", "steam" : "(MWh)"
104.   };
105.   this.svg.select(".y.axis")
106.     .call(this.yAxis)
107.     .select("text")
108.     .attr("y", -15)
109.     .attr("x", 0)
110.     .attr("dy", ".71em")
111.     .style("text-anchor", "middle")
112.     .style("font-size", "12px")
113.     .text(this.option.energyType+ " " + unit[this.option.energyType]);
114.   this.svg.select(".x.axis")
115.     .call(this.xAxis)
116.     .select("text")
117.     .attr("y", 0)

```

```

118.         .attr("x", that.width)
119.         .attr("dy", "2.5em")
120.         .style("text-anchor", "end")
121.         .style("font-size", "12px")
122.         .text(xVariable + " " + unit[xVariable]);
123. // Actualizar gráfica
124.     var div = d3.select("body").append("div")
125.         .attr("class", "tooltip_svg_exploreVis")
126.         .style("opacity", 0);
127.     var dots = this.svg.selectAll(".dot")
128.         .data(this.displayData);
129.     dots.enter().append("g").append("circle");
130.     dots.attr("class", "dot")
131.     dots.transition()
132.         .duration(DURATION)
133.         .attr("transform", function(d,i) {
134.             return "translate(" + that.x(d.x) + "," + that.y(d.y) + ")";
135.         })
136.         .select("circle")
137.         .attr("r", RADIUS)
138.         .style("fill", function() {
139.             return color_energy[that.option.energyType];
140.         })
141.     dots.on("mouseover", function(d) {
142.         div.transition()
143.             .duration(200)
144.             .style("opacity", .9);
145.         div.html(d.month)
146.             .style("left", (d3.event.pageX + 8) + "px")
147.             .style("top", (d3.event.pageY - 28) + "px");
148.     })
149.     dots.on("mouseout", function(d) {
150.         div.transition()
151.             .duration(500)
152.             .style("opacity", 0);
153.     });
154.     dots.exit()
155.         .remove();
156.     var line = linearRegression(this.displayData.map(function(d){return d.x}),
157.                                this.displayData.map(function(d){return d.y}));
158.     var x1 = this.x.domain()[0];
159.     var y1 = x1 * line['slope'] + line['intercept'];
160.     var x2 = this.x.domain()[1];
161.     var y2 = x2 * line['slope'] + line['intercept'];
162.     var trendData = [{ 'x1':x1, 'y1':y1, 'x2':x2, 'y2':y2 }];

```

```

162.     var trendline = this.svg.selectAll(".trendline")
163.         .data(trendData);
164.     trendline.enter()
165.         .append("line")
166.         .attr("class", "trendline");
167.     trendline
168.         .transition()
169.         .duration(DURATION)
170.         .attr("x1", function(d) { return that.x(d.x1); })
171.         .attr("y1", function(d) { return that.y(d.y1); })
172.         .attr("x2", function(d) { return that.x(d.x2); })
173.         .attr("y2", function(d) { return that.y(d.y2); });
174.     // *** mostrar r-cuadrado en el gráfico
175.     this.svg.selectAll(".R2")
176.         .text("R2: " + line['r2'])
177.         .attr("x", function() {return that.width/2 - 20;})
178.         .attr("y", 0);***
179.     var equation_div = d3.select("#explore1Equation");
180.     equation_div.html("<br> <label>R2</label>: " + line['r2'] + "<br>"
181.         + "<label>Equation</label>: y = " + Math.round(line['slope'] *
10000)/10000 + " * x + " + Math.round(line['intercept']*100)/100 + "<br>"
182.         );
183.     };
184.     /*** Es llamado por el controlador de eventos y debe crear una nueva agregación
de datos agregados mediante la función "agregar (filtro)". El filtro tiene que
ser definido aquí.
185.     * @param selection **/
186.     Explore1Vis.prototype.onSelectionChange = function (){
187.         //this.wrangleData(function(d){return d.time>=selectionStart &&
d.time<=selectionEnd;});
188.         this.wrangleData(null);
189.         this.updateVis();
190.     };

```



## ANEXO 5: mapvis.js

```
1.  var circlePadding = 2;
2.  MapVis = function(_parentElement, _monthlyEnergy, _option, _eventHandler){
3.      this.parentElement = _parentElement;
4.      this.monthlyEnergy = _monthlyEnergy;
5.      this.option = _option;
6.      this.eventHandler = _eventHandler;
7.      this.margin = {top: 0, right: 0, bottom: 0, left: 0};
8.      this.height = 400 - this.margin.top - this.margin.bottom;
9.      this.width = getInnerWidth(this.parentElement) - this.margin.left -
this.margin.right;
10.     this.wrangleData();
11.     this.initVis();
12. }
13. /** Método que configura el SVG y las variables **/
14. MapVis.prototype.initVis = function() {
15.     var that = this;
16.     var Mexicomap = new google.maps.LatLng(23.634501, -102.552784);
17.     var MY_MAPTYPE_ID = 'ltc_style';
18.     var featureOpts = [
19.         {
20.             featureType: "all",
21.             elementType: "all",
22.             stylers: [
23.                 { saturation: -80 },
24.                 { lightness: 30 }
25.             ]
26.         },
27.         {
28.             featureType: "road",
29.             elementType: "geometry",
30.             stylers: [
31.                 { color: "#FDF5E6" },
32.                 { lightness: 50 },
33.                 { visibility: "simplified" }
34.             ]
35.         },
36.         {
37.             featureType: "landscape.man_made",
38.             elementType: "geometry.fill",
39.             stylers: [
40.                 { color: "#e3e2dd" }
41.             ]
42.         },
```

```

43.  {
44.      featureType: "landscape.man_made",
45.      elementType: "geometry.stroke",
46.      stylers: [
47.          { color: "#e3e2dd" }
48.      ]
49.  },
50.  {
51.      featureType: "landscape.natural",
52.      elementType: "geometry.fill",
53.      stylers: [
54.          { color: "#e3e2dd" }
55.      ]
56.  }
57.  ];
58.  var mapOptions = {
59.      zoom: 5,
60.      center: Mexicomap ,
61.      mapTypeControlOptions: {
62.          mapTypeIds: [google.maps.MapTypeId.ROADMAP, MY_MAPTYPE_ID]
63.      },
64.      mapTypeId: MY_MAPTYPE_ID,
65.      panControl: true,
66.      zoomControl: true,
67.      mapTypeControl: false,
68.      scaleControl: false,
69.      streetViewControl: false,
70.      overviewMapControl: false,
71.      scrollwheel: false,
72.      draggable: true,
73.      disableDefaultUI: true,
74.      disableDoubleClickZoom: true
75.  };
76.  var mapContainer = this.parentElement.node();
77.  this.map = new google.maps.Map(mapContainer, mapOptions);
78.  var styledMapOptions = {
79.      name: 'Harvard Map'
80.  };
81.  var ltcMapType = new google.maps.StyledMapType(featureOpts,
82.  styledMapOptions);
83.  this.map.mapTypes.set(MY_MAPTYPE_ID, ltcMapType);
84.  this.overlay = new google.maps.OverlayView();
85.  this.overlay.onAdd = function() {
86.      that.svg = d3.select(this.getPanels().overlayMouseTarget).append("div")

```

```

86.     .style("position","absolute");
87.     that.createNodes();
88.     that.overlay.draw = function() {
89.         that.projection = this.getProjection();
90.         that.updateVis('Puebla');
91.     };
92. };
93. this.overlay.setMap(this.map);
94. }
95. MapVis.prototype.wrangleData= function(){
96.     var that = this;
97.     monthlyEnergy = this.monthlyEnergy;
98.     that.displayData = [];
99.     for(var building in that.monthlyEnergy){
100.         var totalgeneracion = 0;
101.         var totalChilledWater = 0;
102.         var totalSteam = 0;
103.         if(monthlyEnergy[building].hasOwnProperty('generacion')) {
104.             monthlyEnergy[building].generacion.forEach(function (d) {
105.                 totalgeneracion += d['consumption']
106.             })
107.             totalgeneracion = Math.round(totalgeneracion)
108.         }
109.         if(monthlyEnergy[building].hasOwnProperty('Capacidad')){
110.             monthlyEnergy[building]['Capacidad'].forEach(function (d){
111.                 totalChilledWater += d['consumption']
112.             })
113.             totalChilledWater = Math.round(totalChilledWater)
114.         }
115.         if(monthlyEnergy[building].hasOwnProperty('steam')) {
116.             monthlyEnergy[building]['steam'].forEach(function (d) {
117.                 totalSteam += d['consumption']
118.             })
119.             totalSteam = Math.round(totalSteam)
120.         }
121.         that.displayData.push({
122.             name:building,
123.             longitude:monthlyEnergy[building]['longitude'],
124.             latitude:monthlyEnergy[building]['latitude'],
125.             totalgeneracion:totalgeneracion,
126.             totalChilledWater:totalChilledWater,
127.             totalSteam:totalSteam,
128.             area:monthlyEnergy[building]['area'],
129.             buildingFunction:monthlyEnergy[building]['function']

```

```

130. })
131.   }
132.   this.buildingLocation = that.buildingLocation;
133. }
134. MapVis.prototype.updateVis = function(_buildingName) {
135.   var that = this;
136.   var selBuildingName = _buildingName;
137.   var nodes = d3.selectAll(".node");
138.   var circles = nodes.select("circle");
139.   var function_opt = d3.select("#function_opt").property('value')
140.   nodes
141.     .each(transform)
142.     .style("visibility", "visible")
143.     .style("width", function(d) {return (that.areaScale(d.area) +
circlePadding) * 2 + "px";})
144.     .style("height", function(d) {return (that.areaScale(d.area) +
circlePadding) * 2 + "px";})
145.     .on('mouseover', function(){d3.select(this).style('cursor',
'pointer');});
146.   nodes.select("circle")
147.     .transition() // esto es imprescindible
148.     .attr("r", function (d) {return that.areaScale(d.area) })
149.     .attr("cx", function (d) {return that.areaScale(d.area) +
circlePadding})
150.     .attr("cy", function (d) {return that.areaScale(d.area) +
circlePadding})
151.     .style("opacity", 0.7);
152.   // mostrar u ocultar estados según el tipo de datos del edificio
153.   switch(d3.select("#Estate_opt").property('value')){
154.     case 'Capacidad':
155.       circles.style("fill", chilledWaterColor).style("stroke-width",
"0px")
156.       nodes.filter(function (d){return d['totalChilledWater'] ==
0}).style("visibility", "hidden")
157.       break
158.     case 'steam':
159.       circles.style("fill", steamColor).style("stroke-width", "0px");
160.       nodes.filter(function (d){return d['totalSteam'] ==
0}).style("visibility", "hidden")
161.       break
162.     case 'generacion':
163.       circles.style("fill", generacionColor).style("stroke-width", "0px");
164.       nodes.filter(function (d){return d['totalgeneracion'] ==
0}).style("visibility", "hidden")

```

```

165.         break
166.     case 'all':
167.         circles.style("fill", allColor).style("stroke-width", "0px");
168.         nodes.filter(function (d){return d['totalgeneracion'] ==
169. 0}).style("visibility", "hidden")
170.         nodes.filter(function (d){return d['totalSteam'] ==
171. 0}).style("visibility", "hidden")
172.         nodes.filter(function (d){return d['totalChilledWater'] ==
173. 0}).style("visibility", "hidden")
174.         break
175.     default:
176.         circles.style("fill", "#CF6E00").style("stroke-width", "0px");
177.         nodes.filter(function (d){return d}).style("visibility", "visible")
178.     }
179.     var selectedNode;
180.     selectedNode = nodes.filter(function (d) {
181.         return d.name == selBuildingName
182.     });
183.     selectedNode
184.         .each(transformSelectedSvg)
185.         .style("width", function(d) {return (20 + circlePadding) * 2 + "px";})
186.         .style("height", function(d) {return (20 + circlePadding) * 2 + "px";})
187.         .style("opacity", 1);
188.     //var electricColor = "#ABDB25"
189.     selectedNode.select("circle")
190.         .style("fill", function(d) {
191.             var color;
192.             switch(d3.select("#Estate_opt").property('value')){
193.                 case 'Capacidad':
194.                     color = chilledWaterColor;
195.                     break;
196.                 case 'steam':
197.                     color = steamColor;
198.                     break;
199.                 case 'generacion':
200.                     color = generacionColor;
201.                     break;
202.                 case 'all':
203.                     color = allColor;
204.                     break;
205.                 default:
206.                     color = allColor;
207.             }
208.             return color;

```

```

206.         })
207.         //.style("stroke", "#666666")
208.         //.style("stroke-width", "1px")
209.         .attr("cx", function (d) {return 20 + circlePadding})
210.         .attr("cy", function (d) {return 20 + circlePadding})
211.         .each(pulse);
212.     function pulse(d) {
213.         var circle = d3.select(this);
214.         (function repeat() {
215.             circle = circle.transition()
216.                 .duration(1000)
217.                 .attr("r", 15)
218.                 .transition()
219.                 .duration(1000)
220.                 .attr("r", that.areaScale(d.area))
221.                 .ease('sine')
222.                 .each("end", function() { if (this.__transition__.count < 2)
repeat(); }); //important
223.             })();
224.         }
225.         //mostrar u ocultar estados según el selector de funciones de construcción
226.         if(function_opt == 'all'){
227.             return;
228.         }
229.         else {
230.             nodes.filter(function (d){return d.buildingFunction !==
function_opt}).style("visibility", "hidden")
231.         }
232.         function transform(d) {
233.             var p = new google.maps.LatLng(d.latitude, d.longitude);
234.             p = that.projection.fromLatLngToDivPixel(p);
235.             return d3.select(this)
236.                 .style("left", p.x - that.areaScale(d.area) - circlePadding + "px")
237.                 .style("top", p.y - that.areaScale(d.area) - circlePadding + "px");
238.         }
239.         function transformSelectedSvg(d) {
240.             var p = new google.maps.LatLng(d.latitude, d.longitude);
241.             p = that.projection.fromLatLngToDivPixel(p);
242.             return d3.select(this)
243.                 .style("left", p.x - 20 - circlePadding + "px")
244.                 .style("top", p.y - 20 - circlePadding + "px");
245.         }
246.     }

```

```

247. MapVis.prototype.createNodes = function() {
248.
249.     var that = this;
250.     this.areas = this.displayData.map(function (d) {return d.area})
251.     var areaMax = d3.max(this.areas)
252.     var areaMin = d3.min(this.areas)
253.     this.areaScale = d3.scale.linear()
254.         .domain([areaMin, areaMax])
255.         .range([3,8])
256.     // crear un soporte div para la información sobre herramientas que muestra el
        nombre del estado
257.     this.thousandNumFormat = d3.format(",d")
258.     var div = d3.select("body").append("div")
259.         .attr("class", "tooltip_svg")
260.         .style("opacity", 0);
261.     var node = this.svg.selectAll(".node")
262.         .data(that.displayData)
263.         .enter()
264.         .append("svg:svg")
265.         .style("position", "absolute")
266.         .attr("class", "node")
267.     node.append("svg:circle")
268.         .attr("fill", "#A4B161")
269.         .attr("fill-opacity", 1)// cambio de 0.6 a 0.9
270.         .on("click", function (d){
271.             clickedBuilding(d.name)
272.         })
273.         .on("mouseover", function(d) {
274.             div.transition()
275.                 .duration(200)
276.                 .style("opacity", .9);
277.             div.html("Estado: " + d.name
278.                 + "<br>" + "Area: " + that.thousandNumFormat(d.area) + " m2"
279.                 + "<br>" + "Generación Total: " +
that.thousandNumFormat(d.totalgeneracion) + " MWh"
280.                 + "<br>" + "Capacidad Total: " +
that.thousandNumFormat(d.totalChilledWater) + " MW"
281.             )
282.                 .style("left", (d3.event.pageX + 8) + "px")
283.                 .style("top", (d3.event.pageY - 28) + "px");
284.         })

```

```

285.         .on("mouseout", function(d) {
286.             div.transition()
287.                 .duration(500)
288.                 .style("opacity", 0);
289.         });
290.
291.         function clickedBuilding(buildingName) {
292.
293.             $(that.eventHandler).trigger("selectionChanged", buildingName)
294.
295.         }
296.     }
297.
298.     MapVis.prototype.onSelectionChange= function (_buildingName){
299.         this.updateVis(_buildingName)
300.     }

```



## ANEXO 6: rankvis.js

```
1.  var barHeight = 15;
2.  var DURATION = 500;
3.  var dateFormatter = d3.time.format("%Y-%m-%d");
4.  var color_energy = {"Capacidad": "blue", "steam": "red", "generacion": "green"};
5.  var dateEnd = "2022-06-01";
6.  var dateStart = "2020-01-01";
7.  RankVis = function(_parentElement, _data, _option, _eventHandler){
8.      this.parentElement = _parentElement;
9.      this.data = _data;
10.     this.option = _option;
11.     this.eventHandler = _eventHandler;
12.     this.displayData = [];
13.     // defines constants
14.     this.margin = {top: 0, right: 50, bottom: 0, left: 50};
15.     this.width = getInnerWidth(this.parentElement) - this.margin.left -
this.margin.right;
16.     this.height = 151 * barHeight - this.margin.top - this.margin.bottom;
17.
18.     this.initVis();
19. };
20. /** Método que configura el SVG y las variables */
21. RankVis.prototype.initVis = function(){
22.     var that = this;
23.     // constructs SVG layout
24.     this.canvas = this.parentElement.append("svg")
25.         .attr("width", this.width + this.margin.left + this.margin.right)
26.         .attr("height", this.height + this.margin.top + this.margin.bottom)
27.         .attr("class", "canvas");
28.     this.svg = this.canvas
29.         .append("g")
30.         .attr("transform", "translate(" + this.margin.left + ", " +
this.margin.top + ")");
31.     // crea ejes y escalas
32.     this.x = d3.scale.linear()
33.         .range([0, that.width]);
34.     this.y = d3.scale.ordinal();
35.     // Agregar elementos visuales de ejes
36.     this.svg.append("g")
37.         .attr("class", "y axis");
38.     // filtrar, agregar, modificar datos
39.     this.wrangleData(this.option);
40.     // llamar al método de actualización
41.     this.updateVis();
42. };
pág. 80
```

```

42.  /*** la función de dibujo: debe usar la selección D3, ingresar, salir */
43.  RankVis.prototype.updateVis = function(){
44.      var that = this;
45.      d3.selectAll(".canvas")
46.          .attr("height", that.displayData.length*barHeight + that.margin.top +
that.margin.bottom);
47.      // updates scales
48.      this.x.domain([0, d3.max(this.displayData, function(d) { return d.energyUse;
}]]);
49.      this.y.rangeRoundBands([0, barHeight * that.displayData.length], 0.2)
50.          .domain(this.displayData.map(function(d) { return d.EstateName; }));
51.      // gráfico de actualizaciones
52.      // Unir Datos
53.      var bar = this.svg.selectAll(".bar")
54.          .data(this.displayData, function(d){return d.EstateName;});
55.      // Agregar nuevos grupos de barras, si es necesario
56.      var bar_enter = bar.enter().append("g")
57.          .attr("class", "bar")
58.          .on("click", function (d){
59.              clickedBuilding(d.EstateName)
60.          });
61.      // Agregue un rect y un texto solo para el conjunto Enter (nueva
g) bar_enter.append("rect");
62.      bar_enter.append("g")
63.          .attr("class", "rank")
64.          .append("text");
65.      bar_enter.append("g")
66.          .attr("class", "label")
67.          .append("text")
68.      bar.select(".rank").select("text")
69.          .text(function(d){return d.rank})
70.          .attr("x", -5)
71.          .attr("y", that.y.rangeBand() / 2)
72.          .attr("dy", "0.85em")
73.          .style("font-size", "10px")
74.          .style("text-anchor", "end");
75.      bar.select(".label").select("text")
76.          .text(function(d){return d.EstateName})
77.          .attr("x", 5)
78.          .attr("y", that.y.rangeBand() / 2)
79.          .attr("dy", "0.85em")
80.          .on('mouseover', function(){d3.select(this).style('cursor',
'pointer');})
81.          .style("text-anchor-y", "start");

```



```

82.     bar.select("rect")
83.         .attr("x", 0)
84.         .attr("y", that.y.rangeBand() / 2)
85.         .attr("height", that.y.rangeBand())
86.         .on('mouseover', function(){d3.select(this).style('cursor',
'pointer');})
87.         .style("fill", function(d) {
88.             if(d.EstateName == that.option.EstateName) {
89.                 return color_energy[that.option.energyType];}
90.             else {
91.                 return "gray";
92.             }
93.         })
94.         .style("opacity",0.7)
95.         .transition()
96.         .attr("width", function(d, i) {
97.             return that.x(d.energyUse);
98.         })
99.     bar.transition()
100.        .duration(DURATION)
101.        .attr("transform", function(d,i) {
102.            return "translate(0," + (that.y(d.EstateName)-
that.y(that.displayData[0].EstateName)) +")";});
103.    bar.exit()
104.        .remove();
105.    function clickedBuilding(EstateName) {
106.        $(that.eventHandler).trigger("selectionChanged", EstateName)
107.    }
108. };
109. /** Es llamado por el controlador de eventos y debe crear nuevos datos
agregados, la agregación se realiza mediante la función "agregar (filtro)". El
filtro tiene que ser definido aquí.
110.  * @param selection**/
111. RankVis.prototype.onSelectionChange = function (){
112.     this.wrangleData(null);
113.     this.updateVis();
114.     if(this.displayData.length == 0) {
115.         toastr.options = {
116.             "closeButton": false,
117.             "debug": false,
118.             "newestOnTop": false,
119.             "progressBar": false,
120.             "positionClass": "toast-top-right",
121.             "preventDuplicates": false,

```

```

123.         "onclick": null,
124.         "showDuration": "300",
125.         "hideDuration": "1000",
126.         "timeOut": "3000",
127.         "extendedTimeOut": "1000",
128.         "showEasing": "swing",
129.         "hideEasing": "linear",
130.         "showMethod": "fadeIn",
131.         "hideMethod": "fadeOut"
132.     }
133.     toastr.info('Selected building function does not have selected energy
type.', 'Try a different building function/energy type.')
134. }
135. };
136. /*** Método para disputar los datos. En este caso toma un objeto de opciones
137.  * @param _filterFunction - una función que filtra datos o "null" si no hay **/
138. RankVis.prototype.wrangleData= function(_filter){
139. // displayData debe contener los datos que se visualizan
140.     this.displayData = this.filterAndAggregate(_filter);
141.
142. };
143. /*** La función agregada que crea los conteos para cada edad para un filtro dado.
144.  * @param _filter - Un filtro puede ser, por ejemplo, una función que solo es
verdadera para datos de un rango de tiempo dado.
145.  * @returns {Array|*}
146.  */
147. RankVis.prototype.filterAndAggregate = function(_filter){
148.     var that = this;
149. // Establezca el filtro en una función que acepte todos los elementos // SOLO si
el parámetro _filter NO es nulo, use este parámetro
150.     var filter = function(){return true;};
151. if (_filter != null){
152.     filter = _filter;
153. }
154.     var res_org = [];
155.     for (var key in this.data) {
156.         var building = this.data[key];
157. // importante verificar que esta es propiedad propia de los objetos, no del
prototipo heredado
158.         if(building.hasOwnProperty(this.option.energyType)){
159.             var temp = 0;

```

```

160.         for (var i = 0; i < building[this.option.energyType].length; i++) {
161.             var month =
dateFormatter.parse(building[this.option.energyType][i].month);
162.             if ( month < dateFormatter.parse(dateEnd) && month >=
dateFormatter.parse(dateStart)) {
163.                 temp = temp +
building[this.option.energyType][i]['consumption']; }
164.             }
165.             temp = temp / building['area'];
166.             var all = false;
167.             if (building.hasOwnProperty('generacion') &&
building.hasOwnProperty('Capacidad') && building.hasOwnProperty('steam'))
168.                 all = true;
169.             res_org.push({EstateName: key, energyUse: temp, rank: 0, function:
building['function'], 'all': all});}
170.         }
171.         var res_filtered;
172.         if (this.option.buildingFunction != "all") {
173.             res_filtered = res_org.filter(function (d) {
174.                 return d.function == that.option.buildingFunction; });
175.             } else {res_filtered = res_org; }
176.             if (this.option.buildingEnergyType == 'all') res_filtered =
res_filtered.filter(function (d) {return d.all == true;})
177.             var res_ranked = res_filtered.sort(function(a,b) {return
d3.descending(a.energyUse, b.energyUse);});
178.             res_ranked.map(function(d,i) {d.rank = i + 1; });
179.             res_ranked.sort(function(a,b) {
180.                 if(that.option.sortType == "EstateName")
181.                     return d3.ascending(a[that.option.sortType],
b[that.option.sortType]);
182.                 else
183.                     return d3.descending(a[that.option.sortType],
b[that.option.sortType]);
184.             });
185.             return res_ranked;
186.         };
187. RankVis.prototype.getYPosition = function (EstateName){
188.     var that = this;
189.     if (that.displayData[0])
190.     return (that.y(EstateName)-that.y(that.displayData[0].EstateName));
191.     else
192.         return 0;
193. };

```

## Referencias

- [1] J. C. Vesga Ferreira, G. Granados Acuña, and J. E. Sierra Carrillo, "Optimization of a multi-service network over a PLC channel under MmQoS," *Ingeniería y Desarrollo*, vol. 33, no. 2, pp. 260–280, Jul. 2015, doi: 10.14482/inde.33.2.6368.
- [2] Gobierno de la Republica, "Programa Sectorial de Energía 2013–2018," México, Dec. 2013.
- [3] Diario Oficial de la Federación, "Ley para el Aprovechamiento de Energías Renovables y el Financiamiento de la Transición Energética," Mexico, Nov. 2008.
- [4] E. Nacional and C. Climático, "Gobierno de la República." [Online]. Available: [www.semarnat.gob.mx](http://www.semarnat.gob.mx)
- [5] C. de Diputados, D. H. Congreso De, L. A. Unión, and N. Ley, "LEY PARA EL APROVECHAMIENTO DE ENERGÍAS RENOVABLES Y EL FINANCIAMIENTO DE LA TRANSICIÓN ENERGÉTICA."
- [6] O. de Buen, M. Ente, and S. C. México, "Energy Management in Mexico: experiences, lessons and outlook."
- [7] Leonardo Beltrán Rodríguez *et al.*, "Reporte de Avance de Energías Limpias 2015", Accessed: Jul. 03, 2022. [Online]. Available: [https://www.gob.mx/cms/uploads/attachment/file/118995/Informe\\_Renovables\\_2015\\_2.pdf](https://www.gob.mx/cms/uploads/attachment/file/118995/Informe_Renovables_2015_2.pdf)
- [8] P. J. Coldwell *et al.*, "Reporte de Avance de Energías Limpias 2016." Accessed: Jun. 03, 2022. [Online]. Available: [https://www.gob.mx/cms/uploads/attachment/file/232624/Informe\\_Renovables\\_2016\\_12062017.pdf](https://www.gob.mx/cms/uploads/attachment/file/232624/Informe_Renovables_2016_12062017.pdf)
- [9] P. J. Coldwell *et al.*, "Reporte de Avance de Energías Limpias 2017." Accessed: Jun. 03, 2022. [Online]. Available: [https://www.gob.mx/cms/uploads/attachment/file/354379/Reporte\\_de\\_Avance\\_de\\_Energ\\_as\\_Limpas\\_Cierre\\_2017.pdf](https://www.gob.mx/cms/uploads/attachment/file/354379/Reporte_de_Avance_de_Energ_as_Limpas_Cierre_2017.pdf)
- [10] M. Amin, "National Infrastructures as Complex Interactive Networks," Jul. 2000, pp. 263–286.
- [11] Y. P. Cai, G. H. Huang, Q. G. Lin, X. H. Nie, and Q. Tan, "An optimization-model-based interactive decision support system for regional energy management systems planning under uncertainty," *Expert Syst Appl*, vol. 36, no. 2 PART 2, pp. 3470–3482, 2009, doi: 10.1016/j.eswa.2008.02.036.
- [12] N. Motegi, M. A. Piette, S. Kinney, and K. Herter, "Web-based energy information systems for energy management and demand response in commercial buildings," *Lawrence Berkeley National Laboratory*, 2003.

- [13] W. Price, R. Hart, E. Water, and E. Board, "Bulls-Eye Commissioning: Using Interval Data as a Diagnostic Tool."
- [14] S. J. Fowler, "Production-Ready Microservices: Building Standardized Systems Across an Engineering Organization."
- [15] "Uso de la integración continua - Azure DevOps." Sep. 2022. [Online]. Available: <https://docs.microsoft.com/es-es/devops/develop/what-is-continuous-integration>
- [16] "¿De qué forma las Acciones de GitHub automatizan las tareas de desarrollo? - Learn." [Online]. Available: <https://docs.microsoft.com/es-es/learn/modules/github-actions-automate-tasks/2-github-actions-automate-development-tasks>
- [17] M. J. T. Otto, "About." Jan. 2011. [Online]. Available: <https://getbootstrap.com/docs/5.3/about/overview/>
- [18] J. Rodriguez, "¿Qué es Bootstrap?" Jan. 2021. [Online]. Available: <https://codenotch.com/blog/que-es-bootstrap/>
- [19] M. Bostock, "D3.js - Data-Driven Documents." [Online]. Available: <https://d3js.org/>
- [20] J. S. F.- js.foundation, "jQuery." [Online]. Available: <https://jquery.com/>
- [21] "Queue (Java Platform SE 8 )." Jan. 2022. [Online]. Available: <https://docs.oracle.com/javase/8/docs/api/java/util/Queue.html>
- [22] "Queue (Java Platform SE 8 )." Jan. 2022. [Online]. Available: <https://docs.oracle.com/javase/8/docs/api/java/util/Queue.html>
- [23] "Queue (Java Platform SE 8 )." Jan. 2022. [Online]. Available: <https://docs.oracle.com/javase/8/docs/api/java/util/Queue.html>
- [24] "HTML: HyperText Markup Language | MDN." Jan. 2022. [Online]. Available: <https://developer.mozilla.org/en-US/docs/Web/HTML>
- [25]: "The Document Metadata (Header) element - HTML: HyperText Markup Language | MDN." Jan. 2022. [Online]. Available: <https://developer.mozilla.org/en-US/docs/Web/HTML/Element/head>
- [26] google, "Google Analytics." [Online]. Available: <https://analytics.google.com/analytics/web/provision/#/provision>
- [27] "Functions - JavaScript | MDN." Jan. 2022. [Online]. Available: <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Functions>
- [28] Microsoft, "How to Implement In-Place Tooltips." Jan. 2020. [Online]. Available: <https://learn.microsoft.com/en-us/windows/win32/controls/implement-in-place-tooltips>
- [29] "EventHandler (Java Platform SE 7 )." Jan. 2020. [Online]. Available: <https://docs.oracle.com/javase/7/docs/api/java/beans/EventHandler.html>



- [30] family=Dale given-i=K., *Data Visualization with Python and JavaScript: Scrape, Clean, Explore & Transform Your Data*. O'Reilly Media, 2016.
- [31] "App Service Pricing." [Online]. Available: <https://azure.microsoft.com/en-us/pricing/details/app-service/windows/>
- [32] "Recientes | Maps JavaScript API | ." [Online]. Available: <https://developers.google.com/maps/documentation/javascript/overview?hl=es-419>

## Tabla de figuras

*Figura 1: Arquitectura Aplicación Web (imagen de autoría propia).*

*Figura 2: Diagrama de Arquitectura propuesto.*

*Figura 3: Opciones principales en aplicación web.*

*Figura 4: Parte central visualización de mapa de México.*

*Figura 5: Parte inferior, visualización de datos en gráficas.*

*Figura 6: Estructura de repositorio.*

*Figura 7: Planes de Hospedaje.*

*Figura 8: Formato de texto tipo .json*

*Figura 9: Resultado de regresión lineal.*

