

Desarrollo de un observatorio turístico digital: caso Acapulco

Material Suplementario (Anexos 1 – 15)

Ing. Cecilia Jiménez Meza

Maestría en Tecnologías de la Información

Acapulco, Gro., 2025

Contenido

ÍNDICE DE FIGURAS	2
ÍNDICE DE TABLAS	5
VIII. ANEXOS.....	6
ANEXO 1. PROCESO ETL PARA LA INTEGRACIÓN DE DATOS TURÍSTICOS EN UN <i>DATA WAREHOUSE</i>	6
ANEXO 2. CONSTRUCCIÓN DEL <i>DATA WAREHOUSE</i> TURÍSTICO Y CONEXIÓN CON APACHE SUPERSET	14
ANEXO 3. <i>SCRIPTS</i> EN PYTHON PARA CARGA DE DATASETS EN LAS TABLAS DE HECHOS Y DIMENSIONES	33
ANEXO 4. VISTAS SQL DEL <i>DATA WAREHOUSE</i> TURÍSTICO	54
ANEXO 5. GENERALES DEL CASO DE USO	63
ANEXO 6. PANTALLAS DE LA PLATAFORMA WEB	77
ANEXO 7. DIAGRAMA ENTIDAD RELACIÓN DE BODEGA_TURISTICA	83
ANEXO 8. INTEGRACIÓN DE DATOS Y VISUALIZACIÓN CON APACHE SUPERSET	84
ANEXO 9. ESPECIFICACIONES DEL ENTORNO DEL SERVIDOR DE DESARROLLO	95
ANEXO 10. MANTENIMIENTO DEL SISTEMA ANALÍTICO Y GESTIÓN DE DATOS TURÍSTICOS	97
ANEXO 11. DETALLE DE LAS PRUEBAS	104
ANEXO 12. INSTRUMENTO DE PRUEBAS DE ACEPTACIÓN DE USUARIO (UAT)	117
ANEXO 13. INFORME DE RESULTADOS: PRUEBAS DE ACEPTACIÓN DE USUARIO (UAT)	129
ANEXO 14. TRANSCRIPCIÓN DE RESPUESTAS CUALITATIVAS DE UAT (ANONIMIZADO).....	131
ANEXO 15. RESPUESTAS DETALLADAS DE EVALUADORES SELECCIONADOS	135

Índice de Figuras

FIGURA 22 SCRIPT EN PYTHON PARA UNIFICAR LOS DATASETS DE SEMARNAT Y COFEPRIS.....	8
FIGURA 23 ESTRUCTURA DE LOS DATASETS EN EL SISTEMA DE ARCHIVOS.....	13
FIGURA 24 MODELO LÓGICO DIMENSIONAL DEL DATA WAREHOUSE TURÍSTICO.	16
FIGURA 25 DESPLIEGUE DE PRUEBA EN APACHE SUPERSET DE LA VISTA _LLEGADAS_CON_FECHA.....	18
FIGURA 26 DESPLIEGUE EN APACHE SUPERSET DE LA TABLA HECHOS_OCPACION.....	19
FIGURA 27 TABLA SQL DE DIMENSIÓN DIM_TIEMPO.....	19
FIGURA 28 TABLA SQL DIM_ZONA.....	20
FIGURA 29 TABLA DIM_TURISTA	20
FIGURA 30 TABLA SQL DIM_CATEGORIA_HOTEL.....	20
FIGURA 31 TABLA SQL DIM_MOTIVO_VIAJE.	21
FIGURA 32 TABLA SQL DIM_TRANSPORTE.	21
FIGURA 33 TABLA SQL HECHOS_DEMANDA.	22
FIGURA 34 TABLA SQL HECHOS_DEMANDA.....	25
FIGURA 35 TABLA SQL HECHOS_OCPACION.	25
FIGURA 36 TABLA SQL HECHOS_GASTO.....	26
FIGURA 37 TABLA SQL HECHOS_EMPLEO.	26
FIGURA 38 TABLA SQL HECHOS_TRANSPORTE.....	26
FIGURA 39 TABLA SQL HECHOS_AMBIENTAL.	27
FIGURA 40 TABLA SQL HECHOS_OCPACION_CATEGORIA.....	27
FIGURA 41 TABLA SQL HECHOS_DEMANDA_CATEGORIA.....	28
FIGURA 42 CONSULTA A DIM_TIEMPO.....	28
FIGURA 43 SE MUESTRA LA TABLA DIM_ZONA.	29
FIGURA 44 UNIÓN DE LAS TABLAS HECHOS_DEMANDA, DIM_TIEMPO Y DIM_TURISTA.	29
FIGURA 45 INTEGRACIÓN DE HECHOS_OCPACION CON DIM_TIEMPO Y DIM_CATEGORIA_HOTEL.	30
FIGURA 46 SE MUESTRA EL NÚMERO DE REGISTROS POR TABLA DE HECHOS.	31

FIGURA 47 SE MUESTRA EL TOTAL DE LLEGADAS POR ORIGEN DEL TURISTA	32
FIGURA 48 CAPTURA DE PANTALLA DE LA VISTA VISTA_OCUACION_CON_TIEMPO	58
FIGURA 49 CAPTURA DE LA VISTA VISTA_GASTO_CON_DESCRIPCION	58
FIGURA 50 CAPTURA DE LA TABLA DIM_TRANSPORTE	58
FIGURA 51 CAPTURA DE LA VISTA_OCUACION_CATEGORIA_ANUAL	59
FIGURA 52 CAPTURA DE LA VISTA_DEMANDA_CATEGORIA_ANUAL	59
FIGURA 53 GRÁFICO GENERADO CON APACHE SUPERSET A PARTIR DE UNA VISTA SQL	60
FIGURA 54 GRÁFICO GENERADO CON APACHE SUPERSET SOBRE EVOLUCIÓN ANUAL DE LLEGADAS	60
FIGURA 55 GRÁFICO GENERADO CON APACHE SUPERSET SOBRE EVOLUCIÓN DE TURISTAS-NOCHE	61
FIGURA 56 GRÁFICO GENERADO CON APACHE SUPERSET SOBRE EVOLUCIÓN DE PERNOCTACIONES	61
FIGURA 57 EJEMPLO DE DASHBOARD EN APACHE SUPERSET UTILIZANDO VISTAS SQL	62
FIGURA 58 MAQUETA QUE MUESTRA LA SECCIÓN DE MAPAS/VUELOS.....	77
FIGURA 59 MAQUETA QUE MUESTRA LA SECCIÓN DE MAPAS/EMBARCACIONES.....	77
FIGURA 60 MAQUETA QUE MUESTRA LA SECCIÓN DE MAPAS/RADAR METEOROLÓGICO.....	78
FIGURA 61 MAQUETA QUE MUESTRA LA SECCIÓN DE MAPAS/VIENTOS	78
FIGURA 62 MAQUETA QUE MUESTRA LA SECCIÓN DE MAPAS/MAREAS	79
FIGURA 63 MAQUETA QUE MUESTRA LA SECCIÓN DE MAPAS/TEMPERATURA.....	79
FIGURA 64 MAQUETA DE LA SECCIÓN ACAPULCO ON-LIVE,	80
FIGURA 65 MAQUETA SUBMENÚ ACTIVIDAD HOTELERA	80
FIGURA 66 MAQUETA DE ACTIVIDAD HOTELERA/DISPONIBILIDAD DE CUARTOS.....	81
FIGURA 67 MAQUETA DE DISPONIBILIDAD DE CUARTOS, CON TABLAS Y GRÁFICOS	81
FIGURA 68 MAQUETA DE BIBLIOTECAS	82
FIGURA 69 TABLA SQL DIM_TIEMPO CON LA COLUMNA FECHA DE TIPO DATE	84
FIGURA 70 DASHBOARD MOSTRANDO LOS FILTROS CONFIGURADOS POR DESTINO	87
FIGURA 71 DASHBOARD CON FILTROS APLICADOS COMPARANDO ACAPULCO CON OTROS DESTINOS DE PLAYA	87
FIGURA 72 VISUALIZACIÓN DE LA VALORACIÓN A PLAYAS UTILIZANDO POWER BI.....	89
FIGURA 73 DASHBOARD DE SUPERSET EMBEBIDO EN UNA PÁGINA WEB.....	92

FIGURA 74 INTERFAZ PRINCIPAL DEL SISTEMA PHP	98
FIGURA 75 RESULTADO DEL ESCANEO Y RESUMEN DE ARCHIVOS CARGADOS	98
FIGURA 76 RESULTADO DE LA DESCARGA DEL DATASET TURÍSTICO	99
FIGURA 77 PRESENTACIÓN DEL OBSERVATORIO TURÍSTICO DE ACAPULCO DURANTE LA APLICACIÓN DE PRUEBAS DE ACEPTACIÓN (UAT).	140

Índice de Tablas

TABLA 4. FUENTES Y FORMATOS DE DATOS UTILIZADOS EN EL ANÁLISIS TURÍSTICO.....	6
TABLA 5 PROCESO DE TRANSFORMACIÓN DE DATOS.....	7
TABLA 6 RESUMEN DE LOS DATASETS PROCESADOS.....	10
TABLA 7 TABLA DE DIMENSIONES DE BODEGA_TURISTICA	14
TABLA 8 TABLAS DE HECHOS DE BODEGA_TURISTICA.....	15
TABLA 9 NOMBRES DE LOS DATASETS ORIGINALES Y LAS TABLAS EN MySQL.....	17
TABLA 10 VISTAS SQL DEL DATA WAREHOUSE TURÍSTICO	57
TABLA 11 DATASETS CON SUS TABLAS EN MySQL Y EL TOTAL DE REGISTROS AGREGADOS.....	85
TABLA 12 ESTRUCTURA DEL DATASET DE LA VALORACIÓN A PLAYAS.	88
TABLA 13 FRECUENCIAS DE PUBLICACIÓN Y ACCIONES SUGERIDAS POR DATASET	97
TABLA 14 RESPUESTAS DE EVALUADORES SELECCIONADOS.....	135

VIII. ANEXOS

Anexo 1. Proceso ETL para la integración de datos turísticos en un *Data Warehouse*

Tabla 4.

Fuentes y formatos de datos utilizados en el análisis turístico.

Fuente	Indicadores Extraídos	Formato Original	Método de Extracción
SECTUR	Ocupación hotelera, flujo turístico	XLSX	Descarga directa
INEGI	Gasto de turistas, empleo turístico	XLSX, CSV	Descarga directa
BANXICO	Tipo de cambio, ingreso de divisas	CSV	Descarga directa
OMT	Ranking mundial de turismo	XLSX	Descarga directa
COFEPRIS	Calidad del agua en playas	PDF	Conversión a tabla con OCR
SEMARNAT	Calidad del agua de mar en playas	PNG	Transcripción manual
ASA	Tráfico aéreo	XLSX	Descarga directa
Secretaría de Turismo Municipal	Ocupación hotelera por zona turística	Imágenes en redes sociales	Captura manual
SEMAR (Secretaría de Marina)	Flujo turístico en cruceros	XLSX	Descarga directa

Nota. Elaboración propia con base en fuentes oficiales. La heterogeneidad de los formatos de origen (XLSX, CSV, PDF, PNG) requirió múltiples estrategias de extracción (conversión OCR, transcripción manual, etc.) para la consolidación de los 95 *datasets* iniciales.

Tabla 5*Proceso de transformación de datos*

Categoría	Proceso	Detalles
Limpieza y normalización de datos	Eliminación de valores nulos e inconsistentes	<ul style="list-style-type: none"> Ocupación hotelera: Valores faltantes en métricas numéricas se asignaron como cero (0) si correspondían a períodos de inactividad. Calidad del agua (playas): Valores NMP/100 mL faltantes por clima adverso o cierre se mantuvieron como nulos, registrando justificación en columna “Clasificación”. Llegada de turistas y tráfico aéreo: Valores faltantes (trimestrales/mensuales) se trataron con interpolación lineal. Si no era posible, se mantenía nulo. Archivos PDF/PNG: Valores ilegibles se validaron manualmente antes de convertirse en nulos.
	Corrección de tipos de datos	<ul style="list-style-type: none"> Estandarización de formatos de fecha a DD/MM/AAAA. Codificación uniforme de variables categóricas (ej. Tipo de turista: Nacional/Extranjero).
	Conversión de coordenadas geográficas	<ul style="list-style-type: none"> Datos de latitud y longitud (SEMARNAT, COFEPRIS) se transformaron de formato grados, minutos y segundos (GMS) a formato decimal.
	Agrupación de series temporales	<ul style="list-style-type: none"> Datos con periodicidad mensual y anual fueron normalizados mediante una columna de referencia temporal.
	Unificación de indicadores similares	<ul style="list-style-type: none"> Métricas redundantes (ej. llegadas de turistas nacionales e internacionales) se consolidaron en un solo <i>dataset</i> con una columna de clasificación (Tipo_Turista).
	Conversión de datos no estructurados	<ul style="list-style-type: none"> PDF (COFEPRIS): Procesados con herramientas OCR para extraer datos tabulares. PNG (SEMARNAT): Transcritos manualmente y validados. Se usó un <i>script</i> de Python (pandas) para consolidar 27 <i>datasets</i> de calidad del agua. Datos de redes sociales (ocupación hotelera): Estructurados en formato tabular.
	Cálculo de indicadores compuestos	<ul style="list-style-type: none"> Se eliminaron métricas redundantes (Porcentaje de Ocupación, Densidad, Estadía) calculables desde datos base. Se consolidó Nacionales/Internacionales (aéreo, cruceros) en una tabla con columna Tipo_Turista. Se generó la variable Tipo_Llegada (vuelos o pasajeros) en la base de tráfico aéreo.

Figura 22

Script en Python para unificar los datasets de SEMARNAT y COFEPRIS.

```

import os
import pandas as pd

# Cambia esta ruta por la carpeta donde están tus 26 archivos Excel
carpeta_archivos = "C:/Users/Cecilia/Documents/Maestria IT
UAGro/INDICADORES/Monitoreo_a_Playas"

# Lista para almacenar los DataFrames
dataframes = []

# Recorremos cada archivo en la carpeta
for archivo in os.listdir(carpeta_archivos):
    if archivo.endswith(".xlsx") or archivo.endswith(".xls"): # Solo procesar
archivos Excel
        ruta_completa = os.path.join(carpeta_archivos, archivo)

        # Leer el archivo y agregarlo a la lista
        try:
            df = pd.read_excel(ruta_completa, engine="openpyxl") # Usa openpyxl
para .xlsx
            dataframes.append(df)
            print(f"\n✓ Archivo procesado: {archivo}")
        except Exception as e:
            print(f"\n✗ Error al leer {archivo}: {e}")

# Unir todos los archivos en uno solo
df_final = pd.concat(dataframes, ignore_index=True)

# Guardar el archivo consolidado
archivo_unificado = os.path.join(carpeta_archivos, "MonitoreoPlayasAcapulco2013-
2024.xlsx")
df_final.to_excel(archivo_unificado, index=False)

print(f"\n✓ Unificación completada. Archivo guardado en: {archivo_unificado}")

```

Nota. Script de elaboración propia para la consolidación de los datasets de monitoreo de playas.

1. Estandarización de nombres de columnas:

- Se homogenizaron los nombres de las columnas en todas las tablas según una convención estándar, eliminando espacios, caracteres especiales y asegurando que los nombres sean auto explicativos.
- Se modificaron los nombres de columnas provenientes de diferentes fuentes para que fueran consistentes en toda la estructura del almacén de datos. Ejemplo:
 - *Sitio de Muestreo → Sitio_Muestreo*

- *Fecha de inicio de muestreo* → *Inicio_Muestreo*
- *Fecha de fin de muestreo* → *Fin_Muestreo*
- *NMP/100 mL* → *NMP_x_100_mL*

2. Conversión de moneda y ajuste temporal:

- Los valores económicos provenientes de BANXICO fueron convertidos a **dólares estadounidenses (USD)** usando el tipo de cambio diario del período correspondiente.
- En *datasets* donde solo existía una columna de **Año**, se normalizó el formato de fecha asignando el primer día del año en formato **01/01/AÑO** para mantener consistencia con otras tablas del *Data Warehouse*.
- En los datos de empleo turístico trimestral (ITET), se asignó la fecha correspondiente al inicio del trimestre según la siguiente lógica:
 - Trimestre I → **01/01/AÑO**
 - Trimestre II → **01/04/AÑO**
 - Trimestre III → **01/07/AÑO**
 - Trimestre IV → **01/10/AÑO**

Estas transformaciones garantizan la coherencia de los datos y facilitan su explotación analítica en el *Data Warehouse*. La variable temporal se estandarizó para hacerla compatible con el resto del almacén de datos. Para ello se utilizó la siguiente fórmula en Excel:

```
=SI(B2="I", FECHA(A2,1,1), SI(B2="II", FECHA(A2,4,1), SI(B2="III",
FECHA(A2,7,1), SI(B2="IV", FECHA(A2,10,1))))))
```

Donde:

- **A2** contiene el **Año**
- **B2** contiene el **Trimestre en número romano**

- La función **FECHA(AÑO, MES, DÍA)** convierte los valores en una fecha estructurada, asignando el primer día del trimestre como fecha de referencia.

Esta transformación garantiza que los datos trimestrales sean compatibles con los modelos analíticos y puedan ser utilizados en consultas basadas en series temporales.

3. Carga de datos

La carga se realizó en un esquema de almacenamiento basado en un modelo dimensional, donde los datos fueron organizados en tablas de hechos y dimensiones según su granularidad. En la Tabla 6 se presenta un resumen de los *datasets* procesados y su estructura final:

Tabla 6

Resumen de los datasets procesados.

Carpeta	Archivo(s) generado(s) del .zip	Nombre de los archivos procesados	Formato	Granularidad
5_1 (Se renombra a OcupacionHotelera)	5_1	<ul style="list-style-type: none"> • CUARTOS_MENSUAL_1992_2023 • LLEGADA_TURISTAS_1992_2023 • TURISTAS_NOCHE_MENSUAL 1992_2023 	CSV	Mensual y anual. La columna “Año” se cambió por “Fecha”, normalizada como 01/01/AÑO
5_2 (Se renombra a OcupacionHotelera XCategoría)	5_2	<ul style="list-style-type: none"> • CUARTOS ANIO & CATEGORIA-1992-2023 • LLEGADA TURISTAS ANIO & CATEGORIA-1992-2023 • TURISTAS NOCHE ANIO & CATEGORIA-1992-2023 	CSV	Anual. La columna “Año” se ha cambiado por “Fecha”, normalizada al formato 01/01/AÑO.
OmtLlegadas2023	OmtLlegadas2023	<ul style="list-style-type: none"> • RANKING-MUNDIAL-LLEGADA-TURISTAS 2023 	XLSX	De tipo informativo, no normalizado al no ser integrado a la base de datos.
OmtDivisas20xx (xx año del indicador)	OmtDivisas20xx (xx año del indicador)	<ul style="list-style-type: none"> • RANKING-MUNDIAL-INGRESO-DIVISAS-2023 	CSV	De tipo informativo, no será integrado a la base de datos.
DatosAbiertos_OM_T	9_1 9_2	<ul style="list-style-type: none"> • DIVISAS-CAPTADAS-EN-MMD 	CSV	Anual (fecha normalizada a 01/01/AÑO)

Carpeta	Archivo(s) generado(s) del .zip	Nombre de los archivos procesados	Formato	Granularidad
	9_3	<ul style="list-style-type: none"> • PRINCIPALES-DESTINOS-POR-LLEGADAS-TURISTAS-2015-2023 • TURISTAS-CAPTADOS-EN MILLONES 		
DatosAbiertos_CR UCEROS	7_21	<ul style="list-style-type: none"> • CRUCEROS_PASAJEROS CSV _2016_2024 		Mensual y anual. La columna “Año” se cambió por “Fecha” con el formato 01/MES/AÑO.
OCUP_X_ZONA TURISTICA		<ul style="list-style-type: none"> • Ocupación_x_zona_tratado CSV _2016_2024 		Refleja temporada vacacional (fin de semana, fin de año, Semana Santa, verano). La columna Fecha tiene el formato DD/MM/AAAA.
Monitoreo_a_Playas		<ul style="list-style-type: none"> • MONITOREO_PLAYAS_A CSV CAPULCO_2013-2024 		Anual. Se manejan varios períodos de muestreo al año, por lo general antes de cada periodo vacacional. El formato es DD/MM/AAAA.
EVI		<ul style="list-style-type: none"> • GASTO_TURISTAS_2018_ CSV 2023 		Mensual y anual. Se maneja la columna “Fecha”, con el formato 01/MM/AAAA.
ITET	ITET	<ul style="list-style-type: none"> • Empleo_Turistico_2006- CSV 2024 		Trimestral (Fecha normalizada: 01/01 para I, 01/04 para II, 01/07 para III y 01/10 para IV)
Nota.	Datos	obtenidos	de	SECTUR (DataTur),

<https://datatur.sectur.gob.mx/SitePages/MapaDeSitio.aspx>

3.3. Próximos pasos para la carga en el *Data Warehouse*

El almacenamiento actual en CSV permite una transición eficiente hacia cualquier arquitectura de *Data Warehouse*, asegurando que los datos ya están estructurados y listos para ser integrados.

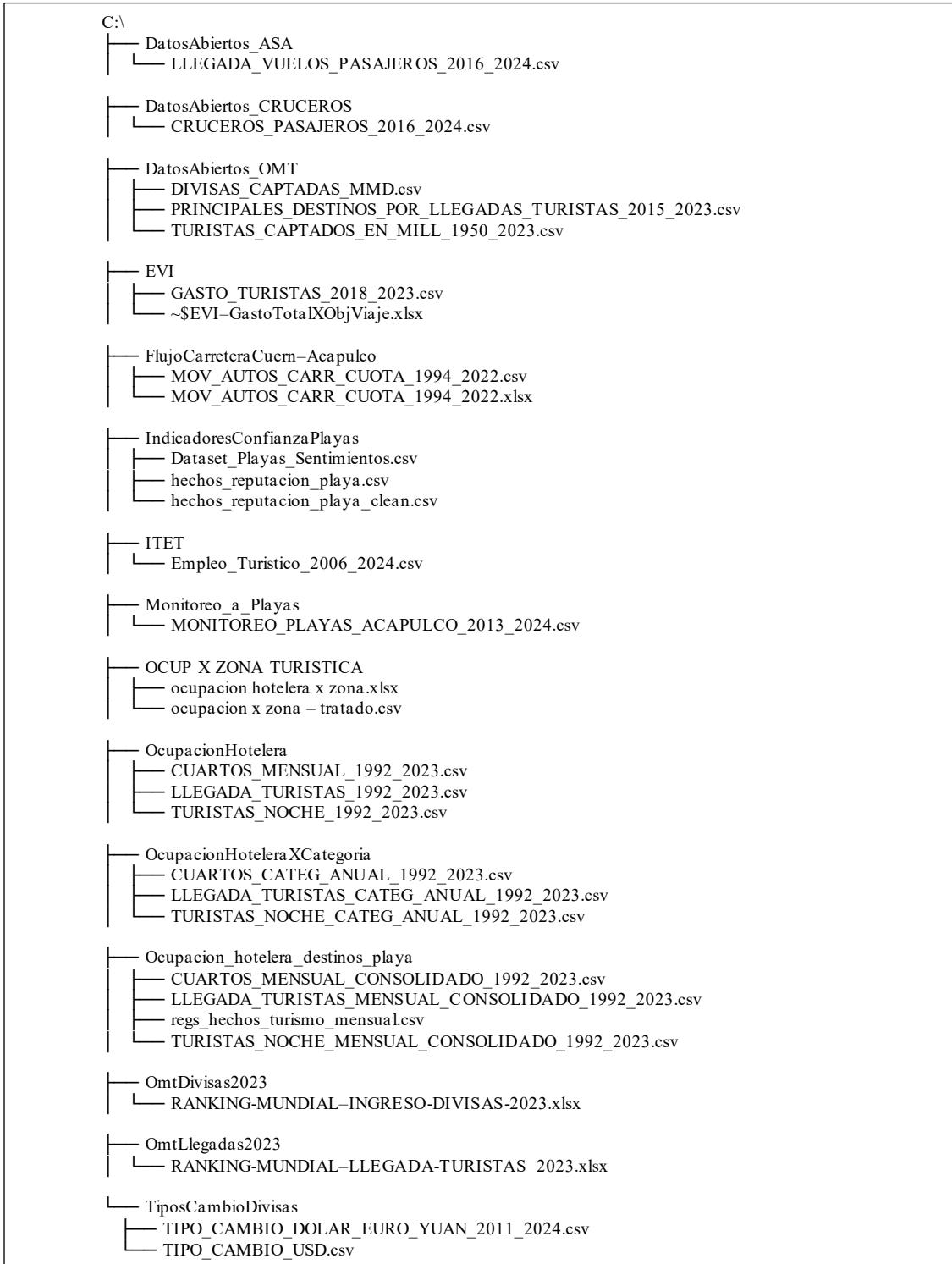
4. Estructura de almacenamiento de datos

Para garantizar la organización y accesibilidad de los datos procesados, los archivos han sido almacenados en una estructura jerárquica de carpetas. Esta organización permite una gestión eficiente y facilita la futura carga en el *Data Warehouse*.

En la Figura 23 se muestra la estructura de almacenamiento actual de los *datasets* en el sistema de archivos.

Figura 23

Estructura de los datasets en el sistema de archivos.



Nota. Elaboración propia.

Anexo 2. Construcción del *Data Warehouse* turístico y conexión con Apache Superset

1. Introducción

Se detalla la construcción del *Data Warehouse* `bodega_turistica` en MySQL, incluyendo su modelo dimensional, estrategia de carga de datos y conexión inicial a Apache Superset, para la exploración visual.

2. Construcción del *Data Warehouse*

2.1 Creación del DW

Se utilizó **MySQL Workbench** para crear la base de datos denominada `bodega_turistica`.

6.1. Las tablas de dimensiones contienen los atributos descriptivos que dan contexto a los datos numéricos (hechos), como se muestra en la Tabla 7.

Tabla 7

Tabla de dimensiones de bodega_turistica

Nombre de la Dimensión	Descripción / Atributos Principales
dim_tiempo	Incluye atributos como año, trimestre, mes, nombre del mes.
dim_zona	Zonas turísticas de Acapulco (Tradicional, Dorada, Diamante).
dim_turista	Tipo de turista (Nacionales, Extranjeros).
dim_categoria_hotel	Categorías hoteleras (1 a 5 estrellas).
dim_motivo_viaje	Motivos de viaje (Placer, Trabajo, Estudio, etc.).
dim_transporte	Tipo de transporte (Aéreo, Crucero, etc.).

6.2.Tablas de Hechos almacenan las métricas numéricas y los eventos de negocio que se quieren analizar, como se describe en la Tabla 8.

Tabla 8

Tablas de hechos de bodega_turistica.

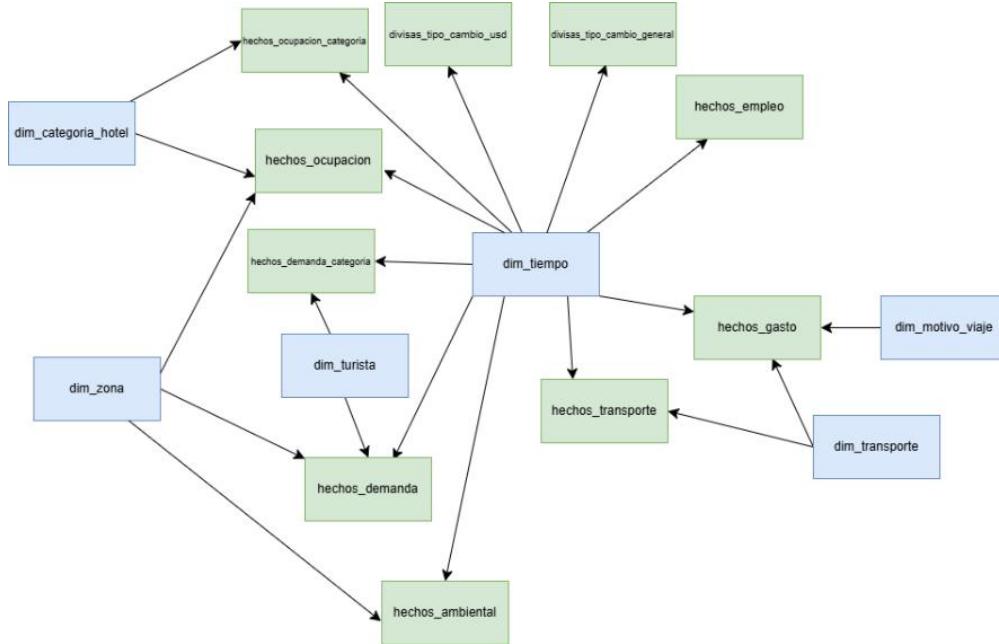
Nombre de la Tabla de Hechos	Descripción / Métricas Principales
hechos_demandas	Llegadas, turistas noche, estadía.
hechos_ocupacion	Cuartos ocupados, disponibles, porcentaje y densidad.
hechos_gasto	Gasto total por tipo de transporte y motivo de viaje.
hechos_empleo	Empleo turístico por año.
hechos_transporte	Llegadas y pasajeros por transporte.
hechos_ambiental	Calidad del agua (enterococos por playa).
hechos_ocupacion_categoria	Ocupación hotelera por categoría.
hechos_demandas_categoria	Llegadas y turistas noche por categoría.
divisas_tipo_cambio_usd	Tipo de cambio peso-dólar.
divisas_tipo_cambio_general	Tipo de cambio para dólar, euro, yuan.

2.4 Diagrama del modelo dimensional

A continuación, en la Figura 24 se presenta el modelo lógico del *Data Warehouse* turístico implementado bajo un esquema estrella, siguiendo los lineamientos de Kimball & Ross (2013). Este esquema permite organizar la información histórica de turismo en torno a hechos medibles y dimensiones descriptivas.

Figura 24

Modelo lógico dimensional del Data Warehouse turístico.



Nota. Elaboración propia con base en Kimball y Ross (2013).

3. Carga de datos

3.1 Estrategia de carga

Se utilizó **Python** con **pandas** y **SQLAlchemy** para cargar los datos desde archivos .CSV ubicados en carpetas locales del equipo. Cada *script*:

- Lee el archivo.
- Transforma los datos (fechas, códigos, IDs).
- Inserta en las tablas con validación y conversión de claves foráneas.

Se agrega el listado de *scripts* en el Anexo 3.

3.2 Datasets por tabla, en la Tabla 8 se muestra la lista de los *datasets* y su correspondiente tabla en MySQL.

Tabla 9

Nombres de los datasets originales y las tablas en MySQL.

Dataset	Tabla poblada
CUARTOS_MENSUAL_1992_2023.CSV	hechos_ocupacion
LLEGADA_TURISTAS_1992_2023.CSV	hechos_demanda
TURISTAS_NOCHE_1992_2023.CSV	hechos_demanda
ocupacion x zona - tratado.csv	hechos_ocupacion
Empleo_Turistico_2006-2024.csv	hechos_empleo
GASTO_TURISTAS_2018_2023.csv	hechos_gasto
LLEGADA_VUELOS&PASAJEROS_2016_2024.csv	hechos_transporte
CRUCEROS_PASAJEROS_2016_2024.csv	hechos_transporte
MONITOREO_PLAYAS_ACAPULCO_2013-2024.csv	hechos_ambiental
CUARTOS_CATEG_ANUAL_1992_2023.csv	hechos_ocupacion_categoria
LLEGADA_TURISTAS_CATEG_ANUAL_1992_2023.csv	hechos_demanda_categoria
TURISTAS_NOCHE_CATEG_ANUAL_1992-2023.csv	hechos_demanda_categoria
TIPO_CAMBIO_USD.csv	divisas_tipo_cambio_usd
TIPO_CAMBIO_DOLAR_EURO_YUAN_2011_2024.csv	divisas_tipo_cambio_general

Nota. La tabla hechos_ocupacion_categoria fue alimentada a partir de CUARTOS_CATEG_ANUAL_1992_2023.csv, el cual contiene información anual (mes de enero) por categoría de hotel. Esto es propio del *dataset* original, no es un error de carga. La información mensual completa está en la tabla hechos_ocupacion para análisis general.

Las tablas segmentadas por categoría (**hechos_ocupacion_categoria** y **hechos_demandas_categoria**) fueron validadas correctamente y contienen únicamente registros anuales, cuya normalización se hizo tomando como referencia el mes de enero de cada año. Por tanto, las vistas y visualizaciones derivadas de estas tablas no deben incluir la dimensión mes, ya que esta no aporta información adicional.

4. Conexión con Apache Superset

4.1 Configuración de conexión

- Se inició Apache Superset (instalado con Docker).
- Se configuró la conexión con MySQL utilizando:
 - Host: host.docker.internal
 - Puerto: 3306
 - Usuario: root
 - Base: bodega_turistica

4.2 Registro de datasets en Apache Superset

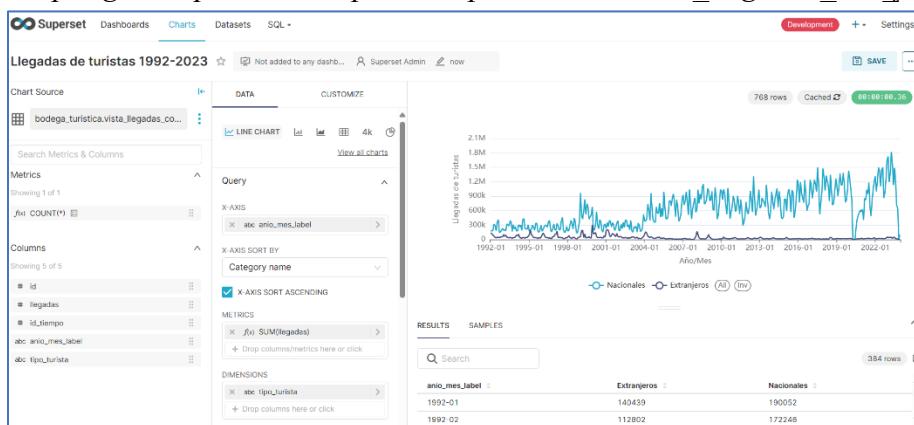
Se crearon *datasets* desde las tablas y vistas para habilitar la exploración visual.

4.3 Gráficas generadas

- Gráfico de línea: **Llegadas de turistas 1992–2023**
- Gráfico de línea: **% Ocupación mensual 1992-2023**
- Se construyó una vista *vista_demanda_con_tiempo* como ejemplo para mostrar *anio_mes_label* como eje X. Las gráficas se muestran de la Figura 25 a la Figura 26.

Figura 25

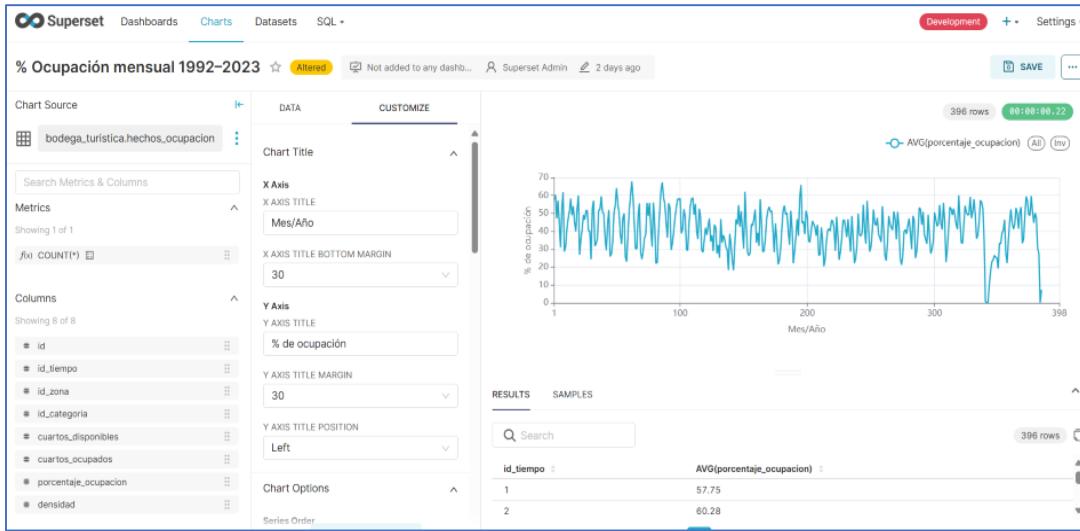
Despliegue de prueba en Apache Superset de la vista_llegadas_con_fecha.



Nota. Captura de pantalla del entorno de validación.

Figura 26

Despliegue en Apache Superset de la tabla hechos_ocupacion.



Nota. Captura de pantalla del entorno de validación.

5. Evidencia de tablas de dimensiones en MySQL Workbench

A continuación se muestran algunas tablas que se capturaron como evidencia visual de validación de la estructura del *DW*.

5.1 Tablas de dimensiones. Su estructura y contenido se detallan en la Figura 27 a la Figura 32.

Figura 27

Tabla SQL de dimensión dim_tiempo.

	id_tiempo	anio	trimestre	mes	nombre_mes
▶	1	1992	1	1	Enero
	2	1992	1	2	Febrero
	3	1992	1	3	Marzo
	4	1992	2	4	Abril
	5	1992	2	5	Mayo
*	HULL	HULL	HULL	HULL	HULL

Nota. Evidencia técnica de la implementación y validación de la estructura del *Data Warehouse* en MySQL Workbench.

Figura 28

Tabla SQL dim_zona.

	id_zona	nombre_zona
▶	1	Tradicional
	2	Dorada
	3	Diamante
*	NULL	NULL

Nota. Evidencia técnica de la implementación y validación de la estructura del Data Warehouse en MySQL Workbench.

Figura 29

Tabla dim_turista

	id_turista	tipo_turista
▶	1	Extranjeros
	2	Nacionales
*	NULL	NULL

Nota. Evidencia técnica de la implementación y validación de la estructura del Data Warehouse en MySQL Workbench.

Figura 30

Tabla SQL dim_categoria_hotel.

	id_categoria	nombre_categoria
▶	1	1 Estrella
	2	2 Estrellas
	3	3 Estrellas
	4	4 Estrellas
	5	5 Estrellas
*	NULL	NULL

Nota. Evidencia técnica de la implementación y validación de la estructura del Data Warehouse en MySQL Workbench.

Figura 31

Tabla SQL dim_motivo_viaje.

	id_motivo	descripcion
▶	1	Placer
	2	Trabajo o negocios
	3	Estudio
	4	En transito
*	5	Visita familia o amigos
	NULL	NULL

Nota. Evidencia técnica de la implementación y validación de la estructura del Data Warehouse en MySQL Workbench.

Figura 32

Tabla SQL dim_transporte.

	id_transporte	tipo_transporte
▶	1	Aéreo
	2	Crucero
	3	Vía aérea
	4	En automóvil
*	5	Peatones
	NULL	NULL

Nota. Evidencia técnica de la implementación y validación de la estructura del Data Warehouse en MySQL Workbench.

5.2 Tablas de hechos. Su estructura y contenido se muestra en la Figura 33.

Figura 33

Tabla SQL *hechos_demanda*.

	id	id_tiempo	id_zona	id_turista	llegadas	turistas_noche	estadia_promedio
▶	1	1	NULL	1	46813	285842	6.11
2	1	NULL	1	46813	NULL	NULL	
3	1	NULL	2	95026	NULL	NULL	
4	2	NULL	1	56401	NULL	NULL	
5	2	NULL	2	86123	NULL	NULL	
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Nota. Evidencia técnica de la implementación y validación de la estructura del Data Warehouse en MySQL Workbench.

Corrección de datos faltantes en *hechos_demanda* (columna *turistas_noche*)

Motivo: Se detectaron valores nulos en la columna *turistas_noche* de la tabla *hechos_demanda*, debido a una omisión en el *script* de carga original que no procesó correctamente los datos del archivo *TURISTAS_NOCHE_1992_2023.csv*.

Estrategia de corrección:

1. Se creó una tabla auxiliar (*tmp_turistas_noche_raw*) adaptada a la estructura original del archivo.
2. Se utilizó un *script* en Python (*carga_turistas_noche_raw.py*) para importar 768 registros válidos desde el archivo CSV. Este *script* **carga_turistas_noche_raw.py** se encuentra en el Anexo 3.
3. Los datos fueron transformados en una segunda tabla (*tmp_turistas_noche*) con claves referenciales (*anio*, *mes*, *id_zona*, *id_turista*) necesarias para realizar el UPDATE.

Transformaciones aplicadas:

- Conversión del campo fecha a año y mes.
- Traducción semántica de estatus a id_turista (Nacionales, Extranjeros).
- Asignación de un valor estándar de zona (id_zona = 4) para registros no georreferenciados.

Asignación de zona genérica (valor 'Sin especificar'):

Dado que el *dataset* no incluye segmentación geográfica, se agregó una nueva entrada en la tabla dim_zona con el valor 'Sin especificar'. Esta decisión permite mantener la integridad referencial sin comprometer el modelo dimensional.

Impacto positivo:

- Los datos faltantes fueron completados sin sobrescribir información válida.
- Los registros globales podrán ser analizados sin segmentación forzada.
- Se mantuvo la trazabilidad completa de las acciones aplicadas.

Tablas involucradas:

- tmp_turistas_noche_raw (carga inicial).
- tmp_turistas_noche (transformación).
- dim_zona (asignación de clave genérica).
- hechos_demandas (actualización final).

Actualización final de la columna turistas_noche en hechos_demandas

Se actualizaron los valores nulos de la columna turistas_noche en la tabla hechos_demandas, utilizando los datos corregidos y transformados contenidos en la tabla auxiliar tmp_turistas_noche, mediante las siguientes acciones:

- Se utilizó un UPDATE con JOIN entre hechos_demandas, dim_tiempo y tmp_turistas_noche, cruzando por año, mes, id_zona y id_turista.
- Se desactivó temporalmente el modo seguro de actualización (SQL_SAFE_UPDATES) para permitir ejecutar el UPDATE debido a restricciones de MySQL Workbench.

Script utilizado:

```
SET SQL_SAFE_UPDATES = 0;

UPDATE hechos_demandas hd
JOIN dim_tiempo dt ON hd.id_tiempo = dt.id_tiempo
JOIN tmp_turistas_noche tmp ON
    dt.anio = tmp.anio AND
    dt.mes = tmp.mes AND
    hd.id_zona = tmp.id_zona AND
    hd.id_turista = tmp.id_turista
SET hd.turistas_noche = tmp.turistas_noche
WHERE hd.turistas_noche IS NULL;

SET SQL_SAFE_UPDATES = 1;
```

Resultado:

La columna turistas_noche fue completada exitosamente sin pérdida de datos y respetando las claves referenciales definidas en el modelo dimensional.

Actualización de valores nulos en turistas_noche

Los valores NULL en la columna turistas_noche de la tabla hechos_demandas se debieron a que los registros originales no tenían segmentación geográfica (zona).

Acciones realizadas:

1. Se asignó una zona genérica ('Sin especificar') en la dimensión dim_zona con un id_zona = 4.
2. Se actualizó hechos_demandas para sustituir valores NULL en id_zona por id_zona = 4.
3. Se utilizó la tabla tmp_turistas_noche para cruzar los datos (año, mes, id_zona, id_turista) y actualizar los valores faltantes de turistas_noche.

Resultado:

La tabla hechos_demandas quedó completamente poblada en su campo turistas_noche, validado mediante consulta SQL que arrojó **cero registros nulos** y muestra datos consistentes a partir de 1995 en adelante, como muestra en la Figura 34. El resto de las tablas de hechos se muestran a partir de las Figura 35 a la Figura 41.

Figura 34

Tabla SQL hechos_demandas.

anio	mes	tipo_turista	turistas_noche
1995	12	Extranjeros	115373
1995	12	Extranjeros	115373
1995	12	Nacionales	410009
1995	12	Nacionales	410009
1996	1	Extranjeros	293878
1996	1	Extranjeros	293878
1996	1	Nacionales	244990
1996	1	Nacionales	244990
1996	2	Extranjeros	357435
1996	2	Extranjeros	357435

Nota. Evidencia técnica de la implementación y validación de la estructura del Data Warehouse en MySQL Workbench.

Figura 35

Tabla SQL hechos_ocupacion.

	id	id_tiempo	id_zona	id_categoria	cuartos_disponibles	cuartos_ocupados	porcentaje_ocupacion	densidad
▶	1	1	NULL	3	524892	303116	57.75	1.00
	2	1	NULL	NULL	524892	303116	57.75	NULL
	3	2	NULL	NULL	493174	297268	60.28	NULL
	4	3	NULL	NULL	516367	244139	47.28	NULL
*	5	4	NULL	NULL	503850	287919	57.14	NULL
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Nota. Evidencia técnica de la implementación y validación de la estructura del Data Warehouse en MySQL Workbench.

Figura 36

Tabla SQL hechos_gasto.

	id	id_tiempo	id_transporte	id_motivo	gasto_total
▶	1	319	1	1	46244392.00
	2	319	3	1	1061709488.00
	3	319	3	2	59643083.00
	4	319	3	3	11441188.00
*	5	319	3	4	2787829.00
	HULL	HULL	HULL	HULL	HULL

Nota. Evidencia técnica de la implementación y validación de la estructura del Data Warehouse en MySQL Workbench.

Figura 37

Tabla SQL hechos_empleo.

	id	id_tiempo	empleo_turistico
*	HULL	HULL	HULL

Nota. Evidencia técnica de la implementación y validación de la estructura del Data Warehouse en MySQL Workbench.

Figura 38

Tabla SQL hechos_transporte.

	id	id_tiempo	id_transporte	llegadas	pasajeros
▶	1	289	1	497	29068
	2	289	1	HULL	29068
	3	289	1	497	HULL
	4	290	1	HULL	29067
*	5	290	1	419	HULL
	HULL	HULL	HULL	HULL	HULL

Nota. Evidencia técnica de la implementación y validación de la estructura del Data Warehouse en MySQL Workbench.

Figura 39

Tabla SQL hechos_ambiental.

	id	id_tiempo	id_zona	nivel_enterococos
▶	1	253	1	16
	2	256	1	161
	3	256	1	118
	4	256	1	43
	5	256	1	87
	6	256	1	152
	7	256	2	21
	8	256	2	NULL
	9	256	2	NULL
	10	256	2	16
*	HULL	HULL	HULL	HULL

Nota. Evidencia técnica de la implementación y validación de la estructura del Data Warehouse en MySQL Workbench.

Figura 40

Tabla SQL hechos_ocupacion_categoria.

	id	id_tiempo	id_categoria	cuartos_disponibles	cuartos_ocupados
▶	1	1	1	482111	121030
	2	1	2	997896	286222
	3	1	3	978996	362915
	4	1	4	1573670	852826
	5	1	5	751798	489121
*	HULL	HULL	HULL	HULL	HULL

Nota. Evidencia técnica de la implementación y validación de la estructura del Data Warehouse en MySQL Workbench.

Figura 41

Tabla SQL *hechos_demandas_categoria*.

	<i>id</i>	<i>id_tiempo</i>	<i>id_categoria</i>	<i>id_turista</i>	<i>llegadas</i>	<i>turistas_noche</i>	<i>estadia_promedio</i>
▶	1	1	1	2	118956	208606	1.75
	2	1	1	1	6314	20539	3.25
	3	1	2	2	249392	576116	2.31
	4	1	2	1	9989	55664	5.57
	5	1	3	2	228874	741605	3.24
	6	1	3	1	17441	126116	7.23
	7	1	4	2	403419	1500159	3.72
	8	1	4	1	70353	494159	7.02
	9	1	5	2	218850	634055	2.90
	10	1	5	1	93331	355390	3.81
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Nota. Evidencia técnica de la implementación y validación de la estructura del Data Warehouse en MySQL Workbench.

5.3 Consulta a dimensiones. A continuación se anotan las consultas y la captura correspondiente como evidencia del resultado de estas.

```
SELECT * FROM dim_tiempo ORDER BY anio, mes LIMIT 12;
```

Muestra la correcta carga del tiempo desde 1992 (Figura 42).

Figura 42

Consulta a *dim_tiempo*.

	<i>id_tiempo</i>	<i>anio</i>	<i>trimestre</i>	<i>mes</i>	<i>nombre_mes</i>
▶	1	1992	1	1	Enero
	2	1992	1	2	Febrero
	3	1992	1	3	Marzo
	4	1992	2	4	Abril
	5	1992	2	5	Mayo
	6	1992	2	6	Junio
	7	1992	3	7	Julio
	8	1992	3	8	Agosto
	9	1992	3	9	Septiembre
	10	1992	4	10	Octubre
	11	1992	4	11	Noviembre
	12	1992	4	12	Diciembre

Nota. Evidencia técnica de la implementación y validación de la estructura del Data Warehouse en MySQL Workbench.

```
SELECT * FROM dim_zona;
```

En la Figura 43 se verifica la carga de las zonas turísticas: Tradicional, Dorada, Diamante.

Figura 43

Se muestra la tabla dim_zona.

	id_zona	nombre_zona
▶	1	Tradicional
	2	Dorada
	3	Diamante
*	HULL	HULL

Nota. Evidencia técnica de la implementación y validación de la estructura del Data Warehouse en MySQL Workbench.

5.4 Uniones entre hechos y dimensiones. Los resultados de los siguientes *scripts* se muestran en las capturas. En el caso siguiente, se valida la unión de hechos con tiempo y tipo de turista (Figura 44).

```
SELECT dt.anio, dt.mes, dtu.tipo_turista, hd.llegadas, hd.turistas_noche,
hd.estadia_promedio
FROM hechos_demandas hd
JOIN dim_tiempo dt ON dt.id_tiempo = hd.id_tiempo
JOIN dim_turista dtu ON dtu.id_turista = hd.id_turista
LIMIT 12;
```

Figura 44

Unión de las tablas hechos_demandas, dim_tiempo y dim_turista.

	anio	mes	tipo_turista	llegadas	turistas_noche	estadia_promedio
▶	1992	1	Extranjeros	46813	285842	6.11
	1992	1	Extranjeros	46813	HULL	HULL
	1992	2	Extranjeros	56401	HULL	HULL
	1992	3	Extranjeros	45653	HULL	HULL
	1992	4	Extranjeros	34372	HULL	HULL
	1992	5	Extranjeros	22531	HULL	HULL
	1992	6	Extranjeros	16501	HULL	HULL
	1992	7	Extranjeros	22624	HULL	HULL
	1992	8	Extranjeros	24737	HULL	HULL
	1992	9	Extranjeros	18117	HULL	HULL
	1992	10	Extranjeros	25586	HULL	HULL
	1992	11	Extranjeros	27651	HULL	HULL

Nota. Evidencia técnica de la implementación y validación de la estructura del Data Warehouse en MySQL Workbench.

```

SELECT
    dt.anio, dt.mes,
    COALESCE(dch.nombre_categoria, 'Sin categoría') AS nombre_categoria,
    ho.cuartos_disponibles, ho.cuartos_ocupados, ho.porcentaje_ocupacion
FROM hechos_ocupacion ho
JOIN dim_tiempo dt ON dt.id_tiempo = ho.id_tiempo
LEFT JOIN dim_categoria_hotel dch ON dch.id_categoria = ho.id_categoria
ORDER BY dt.anio, dt.mes
LIMIT 12;

```

La consulta mostrada en la Figura 45 permite validar la integración de la tabla de hechos hechos_ocupacion con sus dimensiones dim_tiempo y dim_categoria_hotel. Se observa que, aunque muchos registros no incluyen categoría, el uso de LEFT JOIN permite mostrar esos datos con la etiqueta “Sin categoría”, asegurando que se mantenga la integridad analítica del modelo dimensional.

Figura 45

Integración de hechos_ocupacion con dim_tiempo y dim_categoria_hotel.

	anio	mes	nombre_categoria	cuartos_disponibles	cuartos_ocupados	porcentaje_ocupacion
▶	1992	1	3 Estrellas	524892	303116	57.75
	1992	1	Sin categoría	524892	303116	57.75
	1992	2	Sin categoría	493174	297268	60.28
	1992	3	Sin categoría	516367	244139	47.28
	1992	4	Sin categoría	503850	287919	57.14
	1992	5	Sin categoría	520645	230850	44.34
	1992	6	Sin categoría	502230	155796	31.02
	1992	7	Sin categoría	522908	273823	52.37
	1992	8	Sin categoría	523156	323517	61.84
	1992	9	Sin categoría	504150	144397	28.64
	1992	10	Sin categoría	521389	163463	31.35
	1992	11	Sin categoría	515370	212211	41.18

Nota. Evidencia técnica de la implementación y validación de la estructura del Data Warehouse en MySQL Workbench.

5.5 Conteo de registros por tabla. Se utilizó el siguiente script SQL para contar el total de registros por tabla.

```

SELECT 'hechos_demandas' AS tabla, COUNT(*) AS registros FROM hechos_demandas
UNION
SELECT 'hechos_ocupacion', COUNT(*) FROM hechos_ocupacion
UNION
SELECT 'hechos_transporte', COUNT(*) FROM hechos_transporte
UNION
SELECT 'hechos_ambiental', COUNT(*) FROM hechos_ambiental
UNION
SELECT 'hechos_empleo', COUNT(*) FROM hechos_empleo
UNION
SELECT 'hechos_gasto', COUNT(*) FROM hechos_gasto;

```

En la Figura 46 se muestra el volumen de datos insertados por tabla

Figura 46

Se muestra el número de registros por tabla de hechos.

	tabla	registros
▶	hechos_demandas	1537
	hechos_ocupacion	643
	hechos_transporte	433
	hechos_ambiental	639
	hechos_empleo	0
	hechos_gasto	3121

Nota. Evidencia técnica de la implementación y validación de la estructura del Data Warehouse en MySQL Workbench.

5.6 Agregaciones por categoría

```

SELECT dt.anio, dtu.tipo_turista, SUM(hd.llegadas) AS total_llegadas
FROM hechos_demandas hd
JOIN dim_tiempo dt ON dt.id_tiempo = hd.id_tiempo
JOIN dim_turista dtu ON dtu.id_turista = hd.id_turista
GROUP BY dt.anio, dtu.tipo_turista
ORDER BY dt.anio;

```

La Figura 47 permite mostrar comportamiento por año y segmento de turista.

Figura 47

Se muestra el total de llegadas por origen del turista.

	anio	tipo_turista	total_llegadas
▶	1992	Extranjeros	783453
	1992	Nacionales	2831202
	1993	Extranjeros	618598
	1993	Nacionales	3184424
	1994	Extranjeros	626132
	1994	Nacionales	3234034
	1995	Extranjeros	750790
	1995	Nacionales	2812736
	1996	Extranjeros	673952
	1996	Nacionales	3154118
	1997	Extranjeros	692504
	1997	Nacionales	3026882
	1998	Extranjeros	472512
	1998	Nacionales	3324932

Nota. Evidencia técnica de la implementación y validación de la estructura del Data Warehouse en MySQL Workbench.

6. Conclusión

Hasta este punto, se ha logrado estructurar y poblar un modelo dimensional robusto en MySQL con datos turísticos de Acapulco. Se estableció una conexión funcional con Apache Superset y se realizaron las primeras visualizaciones con éxito. El siguiente paso será elaborar las vistas necesarias para habilitar el análisis completo y construir *dashboards* interactivos.

Anexo 3. *Scripts en python para carga de datasets en las tablas de hechos y dimensiones*

cargar_llegadas_turistas.py

```

import pandas as pd
from sqlalchemy import create_engine, text
import mysql.connector

# Configura tus credenciales de conexión
usuario = 'root'
contraseña = 'contraseña'
host = 'localhost'
puerto = 3306
base_datos = 'bodega_turistica'

# Ruta del archivo CSV
ruta_csv =
r'C:\Users\CECILIA\Documents\INDICADORES\OcupacionHotelera\LLEGADA_TURISTAS_1992_2023.
csv'

# Crear la conexión a MySQL con SQLAlchemy
engine =
create_engine(f"mysql+mysqlconnector://{{usuario}}:{{contraseña}}@{{host}}:{{puerto}}/{{base_da
tos}}")

# Cargar CSV
df = pd.read_csv(ruta_csv, encoding='utf-8')

# Convertir la columna de fecha al formato datetime
df['Fecha'] = pd.to_datetime(df['Fecha'], dayfirst=True)
df['anio'] = df['Fecha'].dt.year
df['mes'] = df['Fecha'].dt.month

# Renombrar columnas para conveniencia
df = df.rename(columns={'Num_turistas': 'llegadas', 'Estatus': 'tipo_turista'})

# Conexión directa para búsquedas de ID
# with engine.begin() as conn:

    # Función para obtener id_tiempo e id_turista
    def obtener_ids(anio, mes, tipo_turista):
        id_tiempo = conn.execute(
            text("SELECT id_tiempo FROM dim_tiempo WHERE anio = :anio AND mes = :mes"),
            {"anio": int(anio), "mes": int(mes)}
        ).scalar()

        id_turista = conn.execute(
            text("SELECT id_turista FROM dim_turista WHERE tipo_turista = :tipo"),
            {"tipo": tipo_turista.capitalize()}
        ).scalar()

        return id_tiempo, id_turista

    # Insertar datos uno a uno
    insertados = 0
    with engine.begin() as conn:
        for _, fila in df.iterrows():
            id_tiempo, id_turista = obtener_ids(fila['anio'], fila['mes'],
fila['tipo_turista'])

```

```

        if id_tiempo and id_turista:
            conn.execute(
                text("""
                    INSERT INTO hechos_demanda (id_tiempo, id_zona, id_turista,
llegadas, turistas_noche, estadia_promedio)
                    VALUES (:id_tiempo, NULL, :id_turista, :llegadas, NULL, NULL)
                    """
                ),
                {
                    "id_tiempo": id_tiempo,
                    "id_turista": id_turista,
                    "llegadas": int(fila['llegadas'])
                }
            )
            insertados += 1

# conn.commit()
conn.close()
print(f"Registros insertados correctamente: {insertados}")

```

cargar_ocupacion_cuartos.py

```

import pandas as pd
from sqlalchemy import create_engine, text
import mysql.connector

# Configuración de conexión
usuario = 'root'
contraseña = 'tu_contraseña'
host = 'localhost'
puerto = 3306
base_datos = 'bodega_turistica'

# Ruta del archivo CSV
ruta_csv = r'C:\Users\CECILIA\Documents\INDICADORES\OcupacionHotelera\CUARTOS_MENSUAL_1992_2023.csv'

# Crear engine SQLAlchemy
engine = create_engine(f"mysql+mysqlconnector://{{usuario}}:{{contraseña}}@{{host}}:{{puerto}}/{{base_datos}}")

# Leer el CSV
df = pd.read_csv(ruta_csv, encoding='utf-8')
df['Fecha'] = pd.to_datetime(df['Fecha'], dayfirst=True)
df['anio'] = df['Fecha'].dt.year
df['mes'] = df['Fecha'].dt.month

# Convertir nombres para consistencia
df['Estatus'] = df['Estatus'].str.strip().str.capitalize()

# Separar disponibles
disponibles_df = df[df['Estatus'] == 'Disponible'][['Fecha', 'Num_cuartos']]
disponibles_df = disponibles_df.rename(columns={'Num_cuartos': 'cuartos_disponibles'})

# Separar ocupados por periodo
ocupados_1992_1999 = df[(df['Estatus'] == 'Sin clasificar') & (df['anio'] <= 1999)]
ocupados_2000_mas = df[(df['Estatus'].isin(['Nacionales', 'Extranjeros'])) & (df['anio'] >= 2000)]

```

```

# Agrupar y sumar
ocupados_1992_1999 = ocupados_1992_1999.groupby('Fecha')['Num_cuartos'].sum().reset_index()
ocupados_2000_mas = ocupados_2000_mas.groupby('Fecha')['Num_cuartos'].sum().reset_index()
ocupados = pd.concat([ocupados_1992_1999, ocupados_2000_mas])
ocupados = ocupados.groupby('Fecha')['Num_cuartos'].sum().reset_index()
ocupados = ocupados.rename(columns={'Num_cuartos': 'cuartos_ocupados'})

# Unir con disponibles
resumen = pd.merge(ocupados, disponibles_df, on='Fecha', how='inner')
resumen['anio'] = resumen['Fecha'].dt.year
resumen['mes'] = resumen['Fecha'].dt.month
resumen['porcentaje_ocupacion'] = (resumen['cuartos_ocupados'] / resumen['cuartos_disponibles']) * 100

# Insertar en base de datos
with engine.begin() as conn:
    for _, fila in resumen.iterrows():
        id_tiempo = conn.execute(
            text("SELECT id_tiempo FROM dim_tiempo WHERE anio = :anio AND mes = :mes"),
            {"anio": int(fila['anio']), "mes": int(fila['mes'])}
        ).scalar()

        if id_tiempo:
            conn.execute(
                text("""
                    INSERT INTO hechos_ocupacion (
                        id_tiempo, id_zona, id_categoria,
                        cuartos_disponibles, cuartos_ocupados,
                        porcentaje_ocupacion, densidad
                    ) VALUES (
                        :id_tiempo, NULL, NULL,
                        :cuartos_disponibles, :cuartos_ocupados,
                        :porcentaje_ocupacion, NULL
                    )
                """),
                {
                    "id_tiempo": id_tiempo,
                    "cuartos_disponibles": int(fila['cuartos_disponibles']),
                    "cuartos_ocupados": int(fila['cuartos_ocupados']),
                    "porcentaje_ocupacion": round(fila['porcentaje_ocupacion']),
                    2)
                }
            )
print(f"Registros insertados correctamente: {len(resumen)}")

```

cargar_ocupacion_zona.py

```

import pandas as pd
from sqlalchemy import create_engine, text
import mysql.connector
import numpy as np

# Configurar conexión
usuario = 'root'
contraseña = 'tu_contraseña'
host = 'localhost'

```

```

puerto = 3306
base_datos = 'bodega_turistica'

# Ruta del archivo CSV
ruta_csv      =      r'C:\Users\CECILIA\Documents\INDICADORES\OCUP          X      ZONA
TURISTICA\ocupacion x zona - tratado.csv'

# Crear conexión con SQLAlchemy
engine           =
create_engine(f"mysql+mysqlconnector://{{usuario}}:{{contraseña}}@{{host}}:{{puerto}}/{{base_da
tos}}")

# Cargar el CSV
df = pd.read_csv(ruta_csv, encoding='utf-8')
df['Fecha'] = pd.to_datetime(df['Fecha'], dayfirst=True)
df['anio'] = df['Fecha'].dt.year
df['mes'] = df['Fecha'].dt.month

zonas = ['Diamante', 'Dorada', 'Tradicional']
columnas_habitaciones = ['Hab_diamante', 'Hab_dorada', 'Hab_tradicional']
columnas_ocupacion = ['Diamante', 'Dorada', 'Tradicional']

registros = []

# Reorganizar datos por zona
for zona, col_hab, col_pct in zip(zonas, columnas_habitaciones,
columnas_ocupacion):
    temp = df[['Fecha', 'anio', 'mes', col_hab, col_pct]].copy()
    temp = temp.rename(columns={col_hab: 'cuartos_ocupados', col_pct:
'porcentaje_ocupacion'})
    temp['zona'] = zona
    registros.append(temp)

df_zonas = pd.concat(registros)
df_zonas['cuartos_ocupados'] = pd.to_numeric(df_zonas['cuartos_ocupados'],
errors='coerce')
df_zonas['porcentaje_ocupacion'] = pd.to_numeric(df_zonas['porcentaje_ocupacion'],
errors='coerce')

# Insertar en base de datos
with engine.begin() as conn:
    insertados = 0
    for _, fila in df_zonas.iterrows():
        id_tiempo = conn.execute(
            text("SELECT id_tiempo FROM dim_tiempo WHERE anio = :anio AND mes =
:mes"),
            {"anio": int(fila['anio']), "mes": int(fila['mes'])}
        ).scalar()

        id_zona = conn.execute(
            text("SELECT id_zona FROM dim_zona WHERE nombre_zona = :zona"),
            {"zona": fila['zona']}
        ).scalar()

        if id_tiempo and id_zona:
            conn.execute(
                text("""
                    INSERT INTO hechos_ocupacion (
                        id_tiempo, id_zona, id_categoria,
                        cuartos_disponibles, cuartos_ocupados,
                        porcentaje_ocupacion, densidad
                    ) VALUES (
                        :id_tiempo, :id_zona, NULL,
                """)
            )

```

```

        NULL, :cuartos_ocupados,
        :porcentaje_ocupacion, NULL
    )
"""
),
{
    "id_tiempo": id_tiempo,
    "id_zona": id_zona,
    "cuartos_ocupados": int(fila['cuartos_ocupados']) if not
np.isnan(fila['cuartos_ocupados']) else None,
    "porcentaje_ocupacion": round(fila['porcentaje_ocupacion']),
2) if not np.isnan(fila['porcentaje_ocupacion']) else None
}
)
insertados += 1

print(f"Registros insertados correctamente: {insertados}")

```

cargar_turistas_noche.py

```

import pandas as pd
from sqlalchemy import create_engine, text
import mysql.connector

# Configuración
usuario = 'root'
contraseña = 'tu_contraseña'
host = 'localhost'
puerto = 3306
base_datos = 'bodega_turistica'

# Ruta al CSV
ruta_csv =
r'C:\Users\CECILIA\Documents\INDICADORES\OcupacionHotelera\TURISTAS_NOCHE_1992_2023.cs
v'

# Crear conexión
engine =
create_engine(f"mysql+mysqlconnector://{{usuario}}:{{contraseña}}@{{host}}:{{puerto}}/{{base_da
tos}}")

# Leer datos
df = pd.read_csv(ruta_csv, encoding='utf-8')
df['Fecha'] = pd.to_datetime(df['Fecha'], dayfirst=True)
df['anio'] = df['Fecha'].dt.year
df['mes'] = df['Fecha'].dt.month
df['Estatus'] = df['Estatus'].str.capitalize()

# Renombrar
df = df.rename(columns={'Turistas_noche': 'turistas_noche', 'Estatus': 'tipo_turista'})

# Cargar en base de datos
with engine.begin() as conn:
    insertados = 0
    actualizados = 0
    for _, fila in df.iterrows():
        id_tiempo = conn.execute(
            text("SELECT id_tiempo FROM dim_tiempo WHERE anio = :anio AND mes = :mes"),

```

```

        {"anio": int(fila['anio']), "mes": int(fila['mes'])}
    ).scalar()

    id_turista = conn.execute(
        text("SELECT id_turista FROM dim_turista WHERE tipo_turista = :tipo"),
        {"tipo": fila['tipo_turista']}
    ).scalar()

    if id_tiempo and id_turista:
        existe = conn.execute(
            text("SELECT COUNT(*) FROM hechos_demandas WHERE id_tiempo = :id_tiempo AND id_turista = :id_turista"),
            {"id_tiempo": id_tiempo, "id_turista": id_turista}
        ).scalar()

        if existe:
            conn.execute(
                text("UPDATE hechos_demandas SET turistas_noche = :tn WHERE id_tiempo = :id_tiempo AND id_turista = :id_turista"),
                {"tn": int(fila['turistas_noche']), "id_tiempo": id_tiempo,
                 "id_turista": id_turista}
            )
            actualizados += 1
        else:
            conn.execute(
                text("""
                    INSERT INTO hechos_demandas (
                        id_tiempo, id_zona, id_turista, llegadas,
                        turistas_noche, estadia_promedio
                    ) VALUES (
                        :id_tiempo, NULL, :id_turista, NULL,
                        :tn, NULL
                    )
                """),
                {"id_tiempo": id_tiempo, "id_turista": id_turista, "tn": int(fila['turistas_noche'])}
            )
            insertados += 1
    print(f"Actualizados: {actualizados} | Insertados nuevos: {insertados}")

```

cargar_gasto_turistas_null.py

```

import pandas as pd
from sqlalchemy import create_engine, text
import mysql.connector
import numpy as np

# Configuración
usuario = 'root'
contraseña = 'tu_contraseña'
host = 'localhost'
puerto = 3306
base_datos = 'bodega_turistica'

# Ruta al CSV
ruta_csv =
r'C:\Users\CECILIA\Documents\INDICADORES\EVI\GASTO_TURISTAS_2018_2023.csv'

```

```

# Conexión a MySQL
engine =
create_engine(f"mysql+mysqlconnector://{{usuario}}:{{contraseña}}@{{host}}:{{puerto}}/{{base_datos}}")

# Leer CSV
df = pd.read_csv(ruta_csv, encoding='utf-8')
df['Fecha'] = pd.to_datetime(df['Fecha'], dayfirst=True)
df['anio'] = df['Fecha'].dt.year
df['trimestre'] = df['Fecha'].dt.month.map({1:1, 2:1, 3:1, 4:2, 5:2, 6:2, 7:3,
8:3, 9:3, 10:4, 11:4, 12:4})

df = df.rename(columns={
    'Motivo_viaje': 'motivo',
    'Medio_transporte': 'transporte',
    'Gasto_total_USD': 'gasto'
})

df['gasto'] = pd.to_numeric(df['gasto'], errors='coerce')

with engine.begin() as conn:
    insertados = 0
    for _, fila in df.iterrows():
        id_tiempo = conn.execute(
            text("SELECT id_tiempo FROM dim_tiempo WHERE anio = :anio AND
trimestre = :trimestre"),
            {"anio": int(fila['anio']), "trimestre": int(fila['trimestre'])}
        ).scalar()

        id_motivo = conn.execute(
            text("SELECT id_motivo FROM dim_motivo_viaje WHERE descripcion =
:desc"),
            {"desc": fila['motivo']}
        ).scalar()

        id_transporte = conn.execute(
            text("SELECT id_transporte FROM dim_transporte WHERE
tipo_transporte = :tipo"),
            {"tipo": fila['transporte']}
        ).scalar()

        if id_tiempo and id_motivo and id_transporte:
            conn.execute(
                text("""
                    INSERT INTO hechos_gasto (
                        id_tiempo, id_transporte, id_motivo, gasto_total
                    ) VALUES (
                        :id_tiempo, :id_transporte, :id_motivo, :gasto
                    )
                """),
                {
                    "id_tiempo": id_tiempo,
                    "id_transporte": id_transporte,
                    "id_motivo": id_motivo,
                    "gasto": float(fila['gasto']) if not
np.isnan(fila['gasto']) else None
                }
            )
            insertados += 1

    print(f"Registros insertados correctamente (incluyendo gasto NULL):
{insertados}")

```

cargar_vuelos_pasajeros.py

```

import pandas as pd
from sqlalchemy import create_engine, text
import mysql.connector

# Configuración
usuario = 'root'
contraseña = 'tu_contraseña'
host = 'localhost'
puerto = 3306
base_datos = 'bodega_turistica'

# Ruta al CSV
ruta_csv =
r'C:\Users\CECILIA\Documents\INDICADORES\DatosAbiertos ASA\LLEGADA_VUELOS&PASAJEROS_20
16_2024.csv'

# Crear engine
engine =
create_engine(f"mysql+mysqlconnector://{{usuario}}:{{contraseña}}@{{host}}:{{puerto}}/{{base_da
tos}}")

# Leer archivo
df = pd.read_csv(ruta_csv, encoding='utf-8')
df['Fecha'] = pd.to_datetime(df['Fecha'], dayfirst=True)
df['anio'] = df['Fecha'].dt.year
df['mes'] = df['Fecha'].dt.month

# Agrupar por año, mes y tipo_llegada
agg_df = df.groupby(['anio', 'mes',
'Tipo_llegada'])['Cantidad'].sum().reset_index()

# Insertar en base de datos
with engine.begin() as conn:
    id_transporte = conn.execute(
        text("SELECT id_transporte FROM dim_transporte WHERE tipo_transporte =
'Aéreo'"))
    .scalar()

    insertados = 0
    for _, fila in agg_df.iterrows():
        id_tiempo = conn.execute(
            text("SELECT id_tiempo FROM dim_tiempo WHERE anio = :anio AND mes =
:mes"),
            {"anio": int(fila['anio']), "mes": int(fila['mes'])})
        .scalar()

        if id_tiempo and id_transporte:
            if fila['Tipo_llegada'] == 'Vuelos':
                conn.execute(
                    text("""
                        INSERT INTO hechos_transporte (
                            id_tiempo, id_transporte, llegadas, pasajeros
                        ) VALUES (:id_tiempo, :id_transporte, :valor, NULL)
                    """),
                    {"id_tiempo": id_tiempo, "id_transporte": id_transporte,
                     "valor": int(fila['Cantidad'])})
            )
            elif fila['Tipo_llegada'] == 'Pasajeros':
                conn.execute(
                    text("""

```

```

        INSERT INTO hechos_transporte (
            id_tiempo, id_transporte, llegadas, pasajeros
        ) VALUES (:id_tiempo, :id_transporte, NULL, :valor)
        """),
        {"id_tiempo": id_tiempo, "id_transporte": id_transporte,
"valor": int(fila['Cantidad'])}
    )
    insertados += 1

print(f"Registros insertados correctamente: {insertados}")

```

cargar_cruceros_pasajeros.py

```

import pandas as pd
from sqlalchemy import create_engine, text
import mysql.connector

# Configuración
usuario = 'root'
contraseña = 'tu_contraseña'
host = 'localhost'
puerto = 3306
base_datos = 'bodega_turistica'

# Ruta al CSV
ruta_csv =
r'C:\Users\CECILIA\Documents\INDICADORES\DatosAbiertos_CRUCEROS\CRUCEROS_PASAJEROS_201
6_2024.csv'

# Conexión a la base de datos
engine =
create_engine(f"mysql+mysqlconnector://{{usuario}}:{{contraseña}}@{{host}}:{{puerto}}/{{base_da
tos}}")

# Leer CSV
df = pd.read_csv(ruta_csv, encoding='utf-8')
df['Fecha'] = pd.to_datetime(df['Fecha'], dayfirst=True)
df['anio'] = df['Fecha'].dt.year
df['mes'] = df['Fecha'].dt.month

# Agrupar por año, mes, tipo de movimiento
agg_df = df.groupby(['anio', 'mes',
'Tipo_mov'])['Cantidad'].sum().reset_index()

# Insertar en base de datos
with engine.begin() as conn:
    id_transporte = conn.execute(
        text("SELECT id_transporte FROM dim_transporte WHERE tipo_transporte =
'Crucero'"))
    .scalar()

    insertados = 0
    for _, fila in agg_df.iterrows():
        id_tiempo = conn.execute(
            text("SELECT id_tiempo FROM dim_tiempo WHERE anio = :anio AND mes =
:mes"),
            {"anio": int(fila['anio']), "mes": int(fila['mes'])})
        .scalar()

        if id_tiempo and id_transporte:
            if fila['Tipo_mov'] == 'Arribo cruceros':

```

```

        conn.execute(
            text("""
                INSERT INTO hechos_transporte (
                    id_tiempo, id_transporte, llegadas, pasajeros
                ) VALUES (:id_tiempo, :id_transporte, :valor, NULL)
            """),
            {"id_tiempo": id_tiempo, "id_transporte": id_transporte,
             "valor": int(fila['Cantidad'])}
        )
    elif fila['Tipo_mov'] == 'Pasajeros':
        conn.execute(
            text("""
                INSERT INTO hechos_transporte (
                    id_tiempo, id_transporte, llegadas, pasajeros
                ) VALUES (:id_tiempo, :id_transporte, NULL, :valor)
            """),
            {"id_tiempo": id_tiempo, "id_transporte": id_transporte,
             "valor": int(fila['Cantidad'])}
        )
    insertados += 1

print(f"Registros insertados correctamente: {insertados}")

```

cargar_monitoreo_playas_con_zona.py

```

import pandas as pd
from sqlalchemy import create_engine, text
import mysql.connector
import numpy as np

# Configuración
usuario = 'root'
contraseña = 'tu_contraseña'
host = 'localhost'
puerto = 3306
base_datos = 'bodega_turistica'

ruta_csv =
r'C:\Users\CECILIA\Documents\INDICADORES\Monitoreo_a_Playas\MONITOREO_PLAYAS_ACAPULCO_2013-2024.csv'

# Diccionario playa → zona turística
zona_map = {
    'Playa Caletilla': 'Tradicional',
    'Playa Caleta': 'Tradicional',
    'Playa La Roqueta': 'Tradicional',
    'Playa Hornos': 'Tradicional',
    'Playa Tlacopanocha': 'Tradicional',
    'Playa Suave': 'Dorada',
    'Playa Carabali': 'Dorada',
    'Playa Papagayo': 'Dorada',
    'Playa El Morro': 'Dorada',
    'Playa Condesa': 'Dorada',
    'Playa Copacabana': 'Dorada',
    'Playa Icacos': 'Dorada',
    'Playa Puerto Marqués': 'Diamante',
    'Playa Majahua': 'Diamante',
    'Playa Revolcadero': 'Diamante'
}

# Conexión

```

```

engine =
create_engine(f"mysql+mysqlconnector://{{usuario}}:{{contraseña}}@{{host}}:{{puerto}}/{{base_datos}}")

df = pd.read_csv(ruta_csv, encoding='utf-8')
df['Inicio_muestreo'] = pd.to_datetime(df['Inicio_muestreo'], dayfirst=True)
df['anio'] = df['Inicio_muestreo'].dt.year
df['trimestre'] = df['Inicio_muestreo'].dt.month.map({1:1, 2:1, 3:1, 4:2, 5:2,
6:2, 7:3, 8:3, 9:3, 10:4, 11:4, 12:4})
df['zona'] = df['Playa'].map(zona_map)
df['NMP_x_100_mL'] = pd.to_numeric(df['NMP_x_100_mL'], errors='coerce')

with engine.begin() as conn:
    insertados = 0
    for _, fila in df.iterrows():
        id_tiempo = conn.execute(
            text("SELECT id_tiempo FROM dim_tiempo WHERE anio = :anio AND
trimestre = :trimestre"),
            {"anio": int(fila['anio']), "trimestre": int(fila['trimestre'])}
        ).scalar()

        id_zona = None
        if pd.notna(fila['zona']):
            id_zona = conn.execute(
                text("SELECT id_zona FROM dim_zona WHERE nombre_zona =
:nombre"),
                {"nombre": fila['zona']}
            ).scalar()

        if id_tiempo:
            conn.execute(
                text("""
                    INSERT INTO hechos_ambiental (
                        id_tiempo, id_zona, nivel_enterococos
                    ) VALUES (
                        :id_tiempo, :id_zona, :nivel
                    )
                """),
                {
                    "id_tiempo": id_tiempo,
                    "id_zona": id_zona,
                    "nivel": int(fila['NMP_x_100_mL']) if not
np.isnan(fila['NMP_x_100_mL']) else None
                }
            )
            insertados += 1

print(f"Registros insertados correctamente con zona turística: {insertados}")

```

cargar_ocupacion_categoria_fix.py

```

import pandas as pd
from sqlalchemy import create_engine, text
import mysql.connector

usuario = 'root'
contraseña = 'tu_contraseña'
host = 'localhost'
puerto = 3306
base_datos = 'bodega_turistica'

```

```

ruta_csv =
r'C:\Users\CECILIA\Documents\INDICADORES\OcupacionHoteleraXCategoría\CUARTOS_CATEG_ANUAL_1992_2023.csv'

engine =
create_engine(f"mysql+mysqlconnector://{{usuario}}:{{contraseña}}@{{host}}:{{puerto}}/{{base_datos}}")

df = pd.read_csv(ruta_csv, encoding='utf-8')
df['Fecha'] = pd.to_datetime(df['Fecha'], dayfirst=True)
df['anio'] = df['Fecha'].dt.year

df_pivot = df.pivot_table(
    index=['anio', 'Categoria'],
    columns='Estado',
    values='Cuartos_Ocupados',
    aggfunc='sum'
).reset_index()

df_pivot = df_pivot.rename(columns={
    'Categoria': 'categoria',
    'Disponibles': 'cuartos_disponibles',
    'Ocupados': 'cuartos_ocupados'
})

with engine.begin() as conn:
    insertados = 0
    for _, fila in df_pivot.iterrows():
        id_tiempo = conn.execute(
            text("SELECT id_tiempo FROM dim_tiempo WHERE anio = :anio AND mes = 1"),
            {"anio": int(fila['anio'])}
        ).scalar()

        id_categoria = conn.execute(
            text("SELECT id_categoria FROM dim_categoria_hotel WHERE nombre_categoria = :nombre"),
            {"nombre": fila['categoria']}
        ).scalar()

        if id_tiempo and id_categoria:
            conn.execute(
                text("""
                    INSERT INTO hechos_ocupacion_categoria (
                        id_tiempo, id_categoria,
                        cuartos_disponibles, cuartos_ocupados
                    ) VALUES (
                        :id_tiempo, :id_categoria,
                        :cuartos_disponibles, :cuartos_ocupados
                    )
                """),
                {
                    "id_tiempo": id_tiempo,
                    "id_categoria": id_categoria,
                    "cuartos_disponibles": int(fila['cuartos_disponibles']) if
not pd.isna(fila['cuartos_disponibles']) else None,
                    "cuartos_ocupados": int(fila['cuartos_ocupados']) if not
pd.isna(fila['cuartos_ocupados']) else None
                }
            )
            insertados += 1

```

```
    print(f"Registros insertados correctamente en hechos_ocupacion_categoria:
{insertados}")
```

cargar_llegadas_categoria.py

```
import pandas as pd
from sqlalchemy import create_engine, text
import mysql.connector

# Configuración
usuario = 'root'
contraseña = 'tu_contraseña'
host = 'localhost'
puerto = 3306
base_datos = 'bodega_turistica'

ruta_csv =
r'C:\Users\CECILIA\Documents\INDICADORES\OcupacionHoteleraXCategoría\LLEGADA_TURISTAS_
CATEG_ANUAL_1992_2023.csv'

engine =
create_engine(f"mysql+mysqlconnector://{{usuario}}:{{contraseña}}@{{host}}:{{puerto}}/{{base_da
tos}}")
df = pd.read_csv(ruta_csv, encoding='utf-8')
df['Fecha'] = pd.to_datetime(df['Fecha'], dayfirst=True)
df['anio'] = df['Fecha'].dt.year

with engine.begin() as conn:
    insertados = 0
    for _, fila in df.iterrows():
        id_tiempo = conn.execute(
            text("SELECT id_tiempo FROM dim_tiempo WHERE anio = :anio AND mes =
1"),
            {"anio": int(fila['anio'])}
        ).scalar()

        id_categoria = conn.execute(
            text("SELECT id_categoria FROM dim_categoria_hotel WHERE
nombre_categoria = :nombre"),
            {"nombre": fila['Categoria']}
        ).scalar()

        id_turista = conn.execute(
            text("SELECT id_turista FROM dim_turista WHERE tipo_turista =
:tipo"),
            {"tipo": fila['Tipo_turista']}
        ).scalar()

        if id_tiempo and id_categoria and id_turista:
            conn.execute(
                text("""
                    INSERT INTO hechos_demanda_categoria (
                        id_tiempo, id_categoria, id_turista,
                        llegadas, turistas_noche, estadia_promedio
                    ) VALUES (
                        :id_tiempo, :id_categoria, :id_turista,
                        :llegadas, NULL, NULL
                    )
                """),
                {
                    "id_tiempo": id_tiempo,
```

```

        "id_categoria": id_categoria,
        "id_turista": id_turista,
        "llegadas": int(fila['Llegadas']))
    }
)
insertados += 1

print(f"Registros insertados correctamente en hechos_demandas_categoria:
{insertados}")

```

actualizar_turistas_noche_categoria.py

```

import pandas as pd
from sqlalchemy import create_engine, text
import mysql.connector

usuario = 'root'
contraseña = 'tu_contraseña'
host = 'localhost'
puerto = 3306
base_datos = 'bodega_turistica'

ruta_csv =
r'C:\Users\CECILIA\Documents\INDICADORES\OcupacionHoteleraXCategoria\TURISTAS_NOCHE_CATEG_ANUAL_1992-2023.csv'

engine =
create_engine(f"mysql+mysqlconnector://{{usuario}}:{{contraseña}}@{{host}}:{{puerto}}/{{base_datos}}")

df = pd.read_csv(ruta_csv, encoding='utf-8')
df['Fecha'] = pd.to_datetime(df['Fecha'], dayfirst=True)
df['anio'] = df['Fecha'].dt.year

with engine.begin() as conn:
    actualizados = 0
    for _, fila in df.iterrows():
        anio = int(fila['anio'])
        categoria = fila['Categoria']
        tipo = fila['Tipo_turista']
        turistas_noche = int(fila['Turistas_noche'])

        id_tiempo = conn.execute(
            text("SELECT id_tiempo FROM dim_tiempo WHERE anio = :anio AND mes = 1"),
            {"anio": anio}
        ).scalar()

        id_categoria = conn.execute(
            text("SELECT id_categoria FROM dim_categoria_hotel WHERE nombre_categoria = :nombre"),
            {"nombre": categoria}
        ).scalar()

        id_turista = conn.execute(
            text("SELECT id_turista FROM dim_turista WHERE tipo_turista = :tipo"),
            {"tipo": tipo}
        ).scalar()

        # Obtener llegadas actuales

```

```

        if id_tiempo and id_categoria and id_turista:
            result = conn.execute(
                text("""
                    SELECT llegadas FROM hechos_demanda_categoria
                    WHERE id_tiempo = :id_tiempo AND id_categoria =
:categoria AND id_turista = :id_turista
                    """
                ),
                {
                    "id_tiempo": id_tiempo,
                    "id_categoria": categoria,
                    "id_turista": id_turista
                }
            ).fetchone()

            if result:
                llegadas = result[0]
                estadia = round(turistas_noche / llegadas, 2) if llegadas > 0
            else None

            conn.execute(
                text("""
                    UPDATE hechos_demanda_categoria
                    SET turistas_noche = :noche,
                        estadia_promedio = :estadia
                    WHERE id_tiempo = :id_tiempo AND id_categoria =
:categoria AND id_turista = :id_turista
                    """
                ),
                {
                    "noche": turistas_noche,
                    "estadia": estadia,
                    "id_tiempo": id_tiempo,
                    "id_categoria": categoria,
                    "id_turista": id_turista
                }
            )
            actualizados += 1

        print(f"Registros actualizados correctamente en hechos_demanda_categoria:
{actualizados}")
    
```

cargar_tipo_cambio_usd.py

```

import pandas as pd
from sqlalchemy import create_engine, text
import mysql.connector

usuario = 'root'
contraseña = 'tu_contraseña'
host = 'localhost'
puerto = 3306
base_datos = 'bodega_turistica'

ruta_csv =
r'C:\Users\CECILIA\Documents\INDICADORES\TiposCambioDivisas\TIPO_CAMBIO_USD.csv'

engine =
create_engine(f"mysql+mysqlconnector://{{usuario}}:{{contraseña}}@{{host}}:{{puerto}}/{{base_datos}}")
df = pd.read_csv(ruta_csv, encoding='utf-8')
df['Fecha'] = pd.to_datetime(df['Fecha'], dayfirst=True)
df['anio'] = df['Fecha'].dt.year
    
```

```

df['mes'] = df['Fecha'].dt.month

with engine.begin() as conn:
    # Asegurar que 'Dólar' existe en dim_moneda
    id_moneda = conn.execute(
        text("SELECT id_moneda FROM dim_moneda WHERE nombre_moneda = 'Dólar'"))
    .scalar()

    if not id_moneda:
        conn.execute(
            text("INSERT INTO dim_moneda (nombre_moneda) VALUES ('Dólar')"))
    id_moneda = conn.execute(
        text("SELECT id_moneda FROM dim_moneda WHERE nombre_moneda =
'Dólar'"))
    .scalar()

    insertados = 0
    for _, fila in df.iterrows():
        id_tiempo = conn.execute(
            text("SELECT id_tiempo FROM dim_tiempo WHERE anio = :anio AND mes =
:mes"),
            {"anio": int(fila['anio']), "mes": int(fila['mes'])})
        .scalar()

        if id_tiempo and id_moneda:
            conn.execute(
                text("""
                    INSERT INTO hechos_divisas (id_tiempo, id_moneda,
:tipo_cambio)
                    VALUES (:id_tiempo, :id_moneda, :tipo_cambio)
                    """),
                {
                    "id_tiempo": id_tiempo,
                    "id_moneda": id_moneda,
                    "tipo_cambio": float(fila['Tipo_Cambio'])
                })
            insertados += 1

print(f"Registros insertados correctamente para 'Dólar': {insertados}")

```

cargar_tipo_cambio_varias_divisas.py

```

import pandas as pd
from sqlalchemy import create_engine, text
import mysql.connector
import numpy as np

usuario = 'root'
contraseña = 'tu_contraseña'
host = 'localhost'
puerto = 3306
base_datos = 'bodega_turistica'

ruta_csv =
r'C:\Users\CECILIA\Documents\INDICADORES\TiposCambioDivisas\TIPO_CAMBIO_DOLAR_EURO_YUA
N_2011_2024.csv'

engine =
create_engine(f"mysql+mysqlconnector://{{usuario}}:{{contraseña}}@{{host}}:{{puerto}}/{{base_da
tos}}")
df = pd.read_csv(ruta_csv, encoding='utf-8')

```

```

df['Fecha'] = pd.to_datetime(df['Fecha'], dayfirst=True)
df['anio'] = df['Fecha'].dt.year
df['mes'] = df['Fecha'].dt.month
df['Tipo_Cambio'] = pd.to_numeric(df['Tipo_Cambio'], errors='coerce')

# Normalizar nombre de divisas
df['Divisa'] = df['Divisa'].replace({
    'Dolar USD': 'Dólar',
    'Euro': 'Euro',
    'Yuan chino': 'Yuan'
})

with engine.begin() as conn:
    insertados = 0
    for _, fila in df.iterrows():
        if pd.isna(fila['Tipo_Cambio']):
            continue

        id_tiempo = conn.execute(
            text("SELECT id_tiempo FROM dim_tiempo WHERE anio = :anio AND mes = :mes"),
            {"anio": int(fila['anio']), "mes": int(fila['mes'])}
        ).scalar()

        id_moneda = conn.execute(
            text("SELECT id_moneda FROM dim_moneda WHERE nombre_moneda = :nombre"),
            {"nombre": fila['Divisa']}
        ).scalar()

        if not id_moneda:
            conn.execute(
                text("INSERT INTO dim_moneda (nombre_moneda) VALUES (:nombre)"),
                {"nombre": fila['Divisa']}
            )
        id_moneda = conn.execute(
            text("SELECT id_moneda FROM dim_moneda WHERE nombre_moneda = :nombre"),
            {"nombre": fila['Divisa']}
        ).scalar()

        if id_tiempo and id_moneda:
            conn.execute(
                text("""
                    INSERT INTO hechos_divisas (id_tiempo, id_moneda,
                    tipo_cambio)
                    VALUES (:id_tiempo, :id_moneda, :tipo_cambio)
                """),
                {
                    "id_tiempo": id_tiempo,
                    "id_moneda": id_moneda,
                    "tipo_cambio": float(fila['Tipo_Cambio'])
                }
            )
            insertados += 1

    print(f"Registros insertados correctamente para múltiples divisas: {insertados}")

```

carga_turistas_noche_raw

```

import pandas as pd
from sqlalchemy import create_engine

# Ruta del archivo CSV
archivo_csv =
"C:/Users/CECILIA/Documents/INDICADORES/OcupacionHotelera/TURISTAS_NOCHE_1992_2023.csv"
"

# Conexión a la base de datos
engine =
create_engine("mysql+mysqlconnector://root:tu_contraseña@localhost:3306/bodega_turistica")

# Cargar CSV con pandas
df = pd.read_csv(archivo_csv)

# Renombrar columnas si es necesario
df.columns = [col.strip().lower().replace(" ", "_") for col in df.columns]
df = df.rename(columns={"estatus": "estatus", "turistas_noche": "turistas_noche", "fecha": "fecha"})

# Convertir la fecha al tipo adecuado
df["fecha"] = pd.to_datetime(df["fecha"], dayfirst=True, errors="coerce")

# Filtrar filas con fechas válidas y turistas_noche no nulos
df = df[df["fecha"].notna() & df["turistas_noche"].notna()]

# Insertar en la tabla auxiliar
df.to_sql("tmp_turistas_noche_raw", con=engine, if_exists="append",
index=False)

print(f"Registros insertados correctamente: {len(df)}")

```

insert_playa_data.py

```

import pandas as pd
from sqlalchemy import create_engine, text

# Configuración de conexión a la base de datos
usuario = "root"
contraseña = "tu_contraseña"
host = "localhost"
puerto = "3306"
db = "bodega_turistica"

# Crear motor SQLAlchemy
engine =
create_engine(f"mysql+pymysql://{{usuario}}:{{contraseña}}@{{host}}:{{puerto}}/{{db}}")

# Cargar el archivo CSV
archivo =
r"C:\\\\Users\\\\CECILIA\\\\Documents\\\\INDICADORES\\\\Monitoreo_a_Playas\\\\MONITOREO_PLAYAS_ACA
PUICO_2013-2024.csv"
df = pd.read_csv(archivo)

# Reemplazar valores vacíos con None para SQL
df = df.where(pd.notnull(df), None)

# Convertir fechas
df['Inicio_muestreo'] = pd.to_datetime(df['Inicio_muestreo'], dayfirst=True,
errors='coerce')

```

```

df['Fin_muestreo']      = pd.to_datetime(df['Fin_muestreo'], dayfirst=True,
errors='coerce')

# Insertar fila por fila
with engine.connect() as conn:
    for _, row in df.iterrows():
        insert_stmt = text("""
            INSERT INTO hechos_playas (
                playa, sitio_muestreo, latitud_norte, longitud_oeste,
                inicio_muestreo, fin_muestreo, nivel_enterococos, clasificacion
            ) VALUES (
                :playa, :sitio_muestreo, :latitud_norte, :longitud_oeste,
                :inicio_muestreo, :fin_muestreo, :nivel_enterococos,
                :clasificacion
            )
        """)
        conn.execute(insert_stmt, {
            'playa': row['Playa'],
            'sitio_muestreo': row['Sitio_muestreo'],
            'latitud_norte': row['Latitud_norte'],
            'longitud_oeste': row['Longitud_oeste'],
            'inicio_muestreo': row['Inicio_muestreo'],
            'fin_muestreo': row['Fin_muestreo'],
            'nivel_enterococos': row['NMP_x_100_mL'],
            'clasificacion': row['Clasificacion']
        })
    conn.commit()

print("Registros insertados correctamente en hechos_playas.")

```

insertar_hechos_gasto_frontera.py pendiente de actualizar

```

import pandas as pd
from sqlalchemy import create_engine, text
import math

# Conexión
engine =
create_engine("mysql+pymysql://root:tucontraseña@localhost:3306/bodega_turistica")

# Cargar CSV
archivo =
"C:/Users/CECILIA/Documents/INDICADORES/EVI/GASTO_TURISTAS_2018_2023.csv"
df = pd.read_csv(archivo)

# Limpiar y transformar
df = df.dropna(subset=["Motivo_viaje", "Tipo_turista", "Medio_transporte",
"Fecha"])
df["Fecha"] = pd.to_datetime(df["Fecha"])
df["anio"] = df["Fecha"].dt.year
df["mes"] = df["Fecha"].dt.month

def limpiar_gasto(x):
    try:
        valor = float(x)
        return valor if not math.isnan(valor) else None
    except:
        return None

df["Gasto_total_USD"] = df["Gasto_total_USD"].apply(limpiar_gasto)

```

```

# Insertar
with engine.begin() as conn:
    registros_insertados = 0
    for _, row in df.iterrows():
        gasto = row["Gasto_total_USD"]
        if gasto is None or math.isnan(gasto):
            continue # Evita insertar NaN

        id_tiempo = conn.execute(text("""
            SELECT id_tiempo FROM dim_tiempo
            WHERE anio = :anio AND mes = :mes
        """), {"anio": int(row["anio"]), "mes": int(row["mes"]))).scalar()

        id_transporte = conn.execute(text("""
            SELECT id_transporte FROM dim_transporte
            WHERE tipo_transporte = :tipo
        """), {"tipo": row["Medio_transporte"]}).scalar()

        id_motivo = conn.execute(text("""
            SELECT id_motivo FROM dim_motivo_viaje
            WHERE descripcion = :descripcion
        """), {"descripcion": row["Motivo_viaje"]}).scalar()

        id_turista = conn.execute(text("""
            SELECT id_turista_frontera FROM dim_turista_frontera
            WHERE tipo_turista = :tipo
        """), {"tipo": row["Tipo_turista"]}).scalar()

        if all([id_tiempo, id_transporte, id_motivo, id_turista]):
            conn.execute(text("""
                INSERT INTO hechos_gasto_frontera (
                    id_tiempo, id_transporte, id_motivo, id_turista_frontera,
gasto_total
                    ) VALUES (
                        :id_tiempo, :id_transporte, :id_motivo, :id_turista, :gasto
                    )
            """), {
                "id_tiempo": id_tiempo,
                "id_transporte": id_transporte,
                "id_motivo": id_motivo,
                "id_turista": id_turista,
                "gasto": gasto
            })
            registros_insertados += 1

print(f"✓ Registros insertados exitosamente: {registros_insertados}")

```

actualiza_hechos_gasto_frontera.py

```

import pandas as pd
from sqlalchemy import create_engine, text

# Conexión
engine =
create_engine("mysql+pymysql://root:root@localhost:3306/bodega_turistica")

```

```

# Ruta del archivo CSV
archivo =
"C:/Users/CECILIA/Documents/INDICADORES/EVI/GASTO_TURISTAS_2018_2023.csv"
df = pd.read_csv(archivo)

# Convertir fechas y extraer año/mes
df["Fecha"] = pd.to_datetime(df["Fecha"], errors="coerce")
df["anio"] = df["Fecha"].dt.year
df["mes"] = df["Fecha"].dt.month

# Limpieza de filas inválidas
df = df.dropna(subset=["Motivo_viaje", "Tipo_turista", "Medio_transporte",
"Fecha", "Gasto_total_USD"])

# Filtrar filas con valores no numéricos en 'Gasto_total_USD'
df = df[~df["Gasto_total_USD"].isin(["ND", ""])]]

# Eliminar espacios en blanco (opcional pero recomendable)
df["Gasto_total_USD"] = df["Gasto_total_USD"].str.strip()

with engine.begin() as conn:
    for _, row in df.iterrows():
        try:
            gasto = float(row["Gasto_total_USD"])
        except ValueError:
            continue # Omitir si no se puede convertir

        conn.execute(text("""
            UPDATE hechos_gasto_frontera
            SET anio = :anio,
                mes = :mes
            WHERE id_transporte = (
                SELECT id_transporte FROM dim_transporte WHERE tipo_transporte =
:transporte
            )
            AND id_motivo = (
                SELECT id_motivo FROM dim_motivo_viaje WHERE descripcion = :motivo
            )
            AND id_turista_frontera = (
                SELECT id_turista_frontera FROM dim_turista_frontera WHERE
tipo_turista = :turista
            )
            AND gasto_total = :gasto
            LIMIT 1
        """), {
            "anio": int(row["anio"]),
            "mes": int(row["mes"]),
            "transporte": row["Medio_transporte"],
            "motivo": row["Motivo_viaje"],
            "turista": row["Tipo_turista"],
            "gasto": gasto
        })

print("Actualización de columnas anio y mes completada.")

```

Anexo 4. Vistas SQL del *Data Warehouse* Turístico

Se documenta la construcción de vistas SQL (Vistas) dentro del *Data Warehouse* bodega_turistica. Estas vistas consolidan los datos de hechos y dimensiones para optimizar las consultas y facilitar la conexión con plataformas de visualización como Apache Superset.

a. Los objetivos de estas vistas son:

- Garantizar la relación correcta entre hechos y dimensiones.
- Proveer coherencia en etiquetas y descripciones para el usuario final.
- Preparar *datasets* eficientes para el análisis exploratorio y la creación de *dashboards*, evitando operaciones JOIN complejas en tiempo real.

b. Tabla Maestra de Vistas SQL

La Tabla 9 consolida todas las vistas desarrolladas, su propósito, las tablas que la componen y su estatus o uso principal.

c. Registro de Calidad y Modificaciones

Durante el desarrollo de las vistas, se realizaron varias intervenciones para asegurar la calidad de los datos.

● Modificaciones a vistas

- **vista_ocupacion_con_tiempo:** Se eliminaron las columnas nombre_zona (datos incompletos) y densidad_hotelera (no disponible directamente). Se propuso crear vista_indicadores_kpi_mensual para calcular KPIs derivados (densidad, estadía) y evitar sesgos en esta vista principal.

● Notas de calidad y correcciones

- **Inconsistencia en dim_transporte:** Se detectaron valores duplicados ("Aéreo", "Vía aérea").
 - **Acción:** Se unificó todo bajo el id_transporte = 1 ("Aéreo").
 - **Impacto:** Se actualizaron las claves foráneas en hechos_gasto y hechos_transporte para apuntar al id_transporte correcto.
- **Problemas Generales Detectados:**
 - Inconsistencias en formatos de fecha (Solución: cambio a DD/MM/AAAA).
 - Valores 'N/A' y 'N/D' en campos de gasto (Solución: Corregidos en preprocesamiento).
 - Duplicidad de registros en sanidad de playas (Solución: Consolidados).
- **Evidencia de validación (figuras de captura)**
 - Las vistas fueron validadas mediante consultas SELECT * directas. Las capturas de pantalla de estas validaciones (Figura 47 a la Figura 51, al final de este anexo) sirven como evidencia de la correcta creación y carga de datos.
- **Resultados: Gráficas en Apache Superset**
 - Las vistas generadas se conectaron a Apache Superset como “datasets” para la creación de dashboards. Las siguientes figuras muestran ejemplos de las visualizaciones resultantes:
 - Figura 52: Gráfico de llegadas de turistas por categoría de hotel.
 - Figura 53: Gráfico de evolución anual de llegadas.
 - Figura 54: Gráfico de evolución de turistas-noche.
 - Figura 55: Gráfico de evolución de pernoctaciones.
 - Figura 56: Ejemplo de dashboard consolidado en Superset.

d. *Scripts SQL*

Para mantener el documento principal limpio, los *scripts* CREATE OR REPLACE VIEW de las 19 vistas se almacenan en el archivo de respaldo `vistas_respaldo_utiles2.sql`.

A modo de ejemplo, se incluye el *script* de una de las vistas modificadas:

Script de la vista 'vista_ocupacion_con_tiempo' (versión final)

```
CREATE OR REPLACE VIEW vista_ocupacion_con_tiempo AS
SELECT
    dt.anio,
    dt.mes,
    dt.nombre_mes,
    ho.cuartos_disponibles AS "Cuartos disponibles",
    ho.cuartos_ocupados AS "Cuartos ocupados",
    ho.porcentaje_ocupacion AS "Porcentaje de ocupación"
FROM hechos_ocupacion ho
JOIN dim_tiempo dt ON dt.id_tiempo = ho.id_tiempo;
```

Tabla 10*Vistas SQL del Data Warehouse turístico*

Nombre de la Vista	Descripción / Propósito Principal	Tablas de Hechos	Tablas de Dimensiones	Estatus / Uso Principal
vista_ocupacion_con_tie mpo	Muestra la evolución mensual de la ocupación (cuartos, %) sin segmentar por zona.	hechos_ocupacion	dim_tiempo	Análisis global de ocupación.
vista_gasto_con_descripc ion	Presenta el gasto turístico total desglosado por transporte y motivo de viaje, agrupado por trimestre.	hechos_gasto	dim_tiempo, dim_motivo_viaje, dim_transporte	Análisis del gasto turístico.
vista_transporte_con_tipo	Agrupa llegadas y pasajeros por tipo de transporte (aéreo o crucero) y periodo de tiempo.	hechos_transporte	dim_tiempo, dim_transporte	Dashboards de movilidad.
vista_empleo_con_tiempo	Registra el número de empleos turísticos por año y trimestre.	hechos_empleo	dim_tiempo	Series temporales de evolución laboral.
vista_ambiental_con_zon a	Muestra los niveles de enterococos (NMP/100mL) por zona turística y fecha.	hechos_ambiental	dim_tiempo, dim_zona	Análisis de sostenibilidad y salud.
vista_ocupacion_categoria_anual	Resume cuartos disponibles, ocupados y % de ocupación anual por categoría hotelera.	hechos_ocupacion_categoría	dim_tiempo, dim_categoria_hotel	Análisis de oferta hotelera segmentada.
vista_demanda_categoria _anual	Aggrega llegadas y turistas noche por año, categoría hotelera y tipo de turista.	hechos_demanda_categoría	dim_tiempo, dim_categoria_hotel, dim_turista	Base para gráficas de análisis segmentado.
vista_gasto_frontera	Presenta datos del gasto de turistas fronterizos por motivo y transporte.	hechos_gasto_frontera	dim_tiempo_gasto, dim_turista_frontera	Análisis de gasto fronterizo.
vista_gasto_turismo_cons olidado	Consolida el gasto turístico segmentado por tipo de turista, transporte y motivo.	hechos_gasto	dim_tipo_turista, dim_transporte, dim_motivo_viaje, dim_tiempo	Análisis del gasto turístico anual y por segmento.
vista_movimiento_cruc eros	Ánálisis específico del arribo de cruceros y pasajeros (filtrado id_transporte = 2).	hechos_transporte	dim_transporte, dim_tiempo	Visualización de movilidad en cruceros.
vista_playas_con_coorden adas	Consolidación sanitaria de playas, incluyendo coordenadas, período y clasificación.	hechos_sanidad_playas	(No especificado)	Preparación para visualizaciones geoespaciales.
vista_indicadores_kpi_me nsual	Calcula KPIs derivados: estadía promedio, densidad hotelera y % de ocupación por mes.	hechos_demanda, hechos_ocupacion	dim_tiempo	Dashboard de KPIs mensual.
vista_llegadas_con_nomb re_turista	Aggrega llegadas mensuales según tipo de turista (Nacional/Extranjero).	hechos_demanda	dim_turista, dim_tiempo	Análisis de llegadas por tipo de turista.
vista_movilidad_transpor te_mensual	Detalla el flujo mensual de pasajeros por tipo de transporte.	hechos_transporte	dim_transporte, dim_tiempo	Paneles de movilidad (complementaria).
vista_nivel_enterococos_c on_fecha	Evolución de la concentración bacteriana (calidad del agua) en playas.	hechos_sanidad_playas	dim_fecha	Calidad sanitaria del agua de mar.
vista_ocupacion_categoria _con_tiempo	Ocupación hotelera mensual por categoría de establecimiento.	hechos_ocupacion_categoría	dim_tiempo	Análisis de hotelería.
vista_ocupacion_resumen _anual	Consolidado anual del porcentaje de ocupación hotelera (general).	hechos_ocupacion	dim_tiempo	Indicadores anuales de % ocupación.
vista_vuelos_anual	Resumen anual de flujo de pasajeros por vía aérea.	hechos_transporte	dim_transporte, dim_tiempo	Movilidad aérea.
vista_vuelos_mensual	Movimiento mensual de pasajeros por transporte aéreo.	hechos_transporte	dim_transporte, dim_tiempo	Complemento en análisis de movilidad aérea.

Figura 48

Captura de pantalla de la vista vista_ocupacion_con_tiempo

	anio	mes	nombre_mes	Cuartos disponibles	Cuartos ocupados	Porcentaje de ocupación
▶	1992	1	Enero	524892	303116	57.75
	1992	1	Enero	524892	303116	57.75
	1992	2	Febrero	493174	297268	60.28
	1992	3	Marzo	516367	244139	47.28
	1992	4	Abril	503850	287919	57.14
	1992	5	Mayo	520645	230850	44.34
	1992	6	Junio	502230	155796	31.02
	1992	7	Julio	522908	273823	52.37
	1992	8	Agosto	523156	323517	61.84

Nota. Validación de la vista en MySQL Workbench.

Figura 49

Captura de la vista vista_gasto_con_descripcion.

	anio	trimestre	motivo_viaje	tipo_transporte	gasto_total
▶	2018	3	Placer	Aéreo	46244392.00
	2018	3	Placer	Vía aérea	1061709488.00
	2018	3	Trabajo o negocios	Vía aérea	59643083.00
	2018	3	Estudio	Vía aérea	11441188.00
	2018	3	En transito	Vía aérea	2787829.00
	2018	3	Visita familia o amigos	Vía aérea	213527626.00
	2018	3	Atencion medica	Vía aérea	NULL
	2018	3	Compras	Vía aérea	NULL
	2018	3	Otros	Vía aérea	9641692.00
	2018	3	Placer	Vía terrestre	37061291.00

Nota. Validación de la vista en MySQL Workbench.

Figura 50

Captura de la tabla dim_transporte

	id_transporte	tipo_transporte
▶	1	Aéreo
	2	Crucero
	4	En automóvil
	5	Peatones
	6	Vía terrestre
*	NULL	NULL

Nota. Validación de la vista en MySQL Workbench.

Figura 51

Captura de la vista_ocupacion_categoria_anual.

categoria_c	anio	nombre_categoria	cuartos_disponibles	cuartos_ocupados	porcentaje_ocupacion
int	1992	1 Estrella	482111	121030	25.10
varchar(5)	1992	1 Estrella	482111	121030	NULL
int	1992	1 Estrella	482111	121030	NULL
int	1992	2 Estrellas	997896	286222	NULL
int	1992	2 Estrellas	997896	286222	NULL
int	1992	2 Estrellas	997896	286222	28.68
int	1992	3 Estrellas	978996	362915	NULL
int	1992	3 Estrellas	978996	362915	37.07
int	1992	3 Estrellas	978996	362915	NULL
int	1992	4 Estrellas	1573670	852826	54.19

Nota. Validación de la vista en MySQL Workbench.

Figura 52

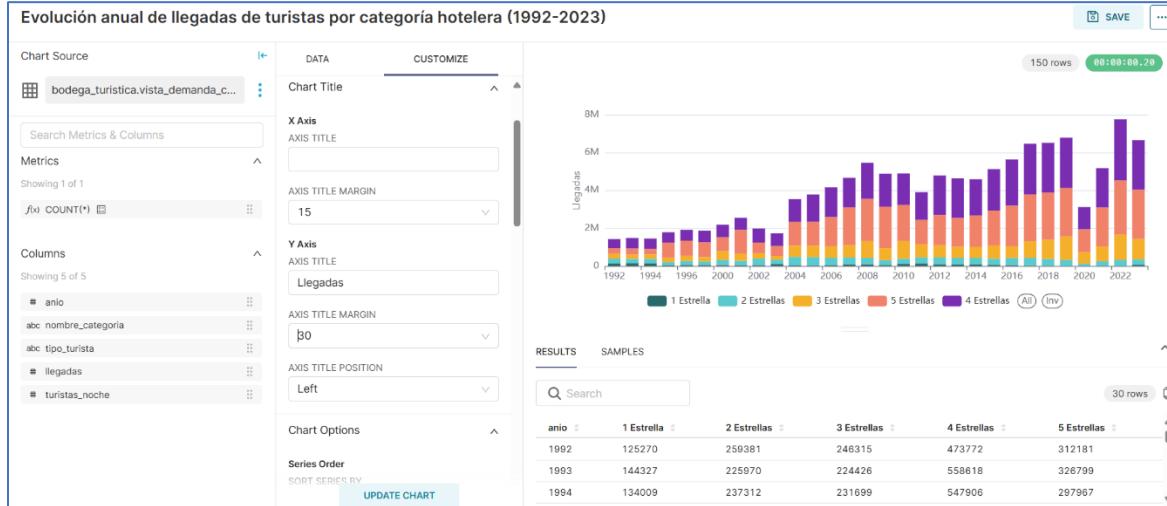
Captura de la vista_demanda_categoria_anual.

anio	nombre_categoria	tipo_turista	llegadas	turistas_noche
1997	4 Estrellas	Nacionales	541271	1351335
1997	5 Estrellas	Extranjeros	242887	1101728
1997	5 Estrellas	Nacionales	535806	1382643
2000	1 Estrella	Extranjeros	5626	9664
2000	1 Estrella	Nacionales	34131	54756
2000	2 Estrellas	Extranjeros	11457	29208
2000	2 Estrellas	Nacionales	262047	373968
2000	3 Estrellas	Extranjeros	23924	138604
2000	3 Estrellas	Nacionales	450870	839516
2000	4 Estrellas	Extranjeros	117716	502799

Nota. Validación de la vista en MySQL Workbench.

Figura 53

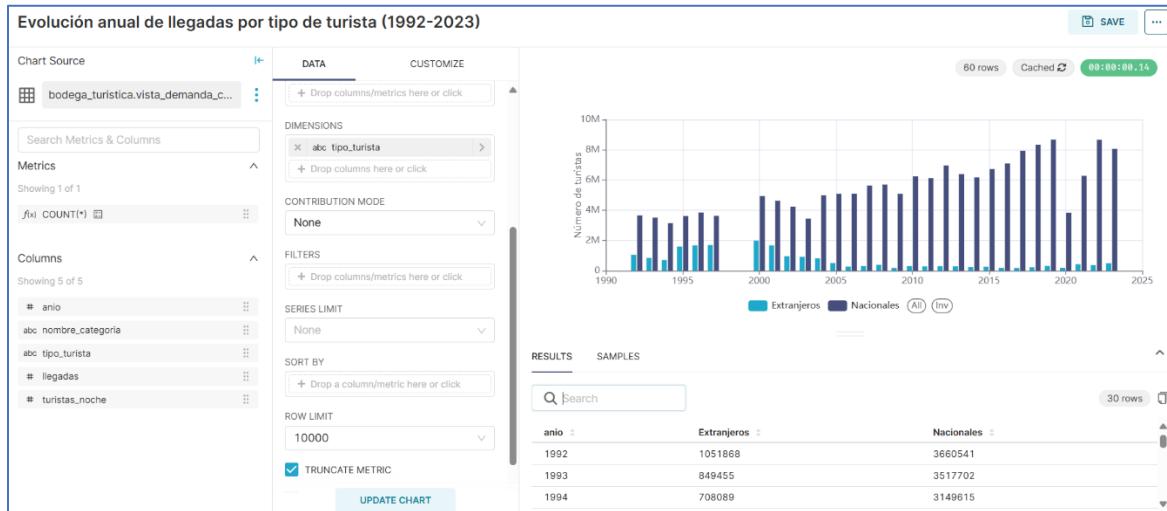
Gráfico generado con Apache Superset a partir de una vista SQL.



Nota. Se muestra el histórico de ocupación en Acapulco por categoría de hotel

Figura 54

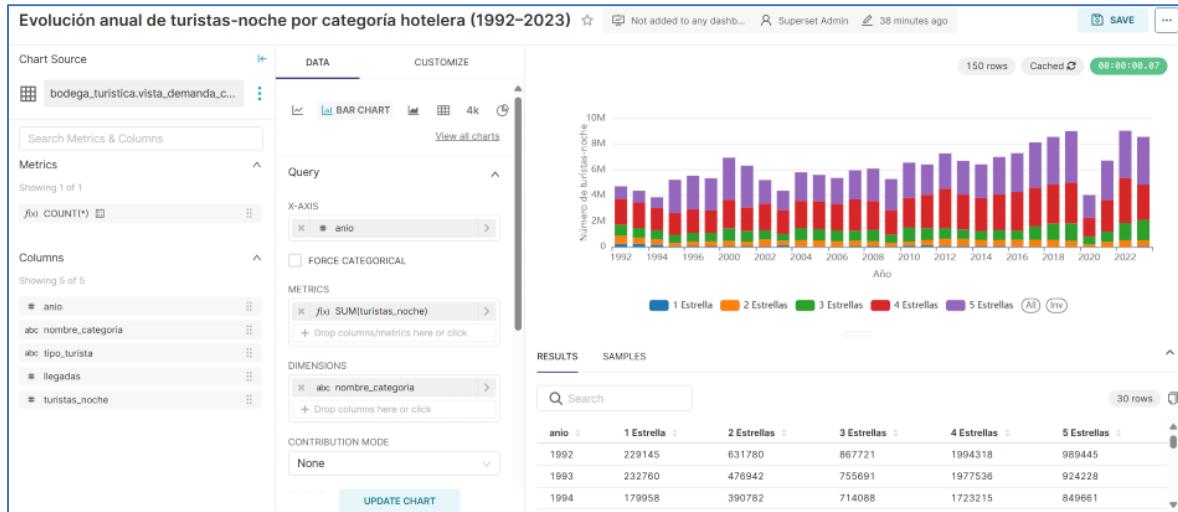
Gráfico generado con Apache Superset sobre evolución anual de llegadas.



Nota. Se muestra el histórico de ocupación en Acapulco por origen del turista.

Figura 55

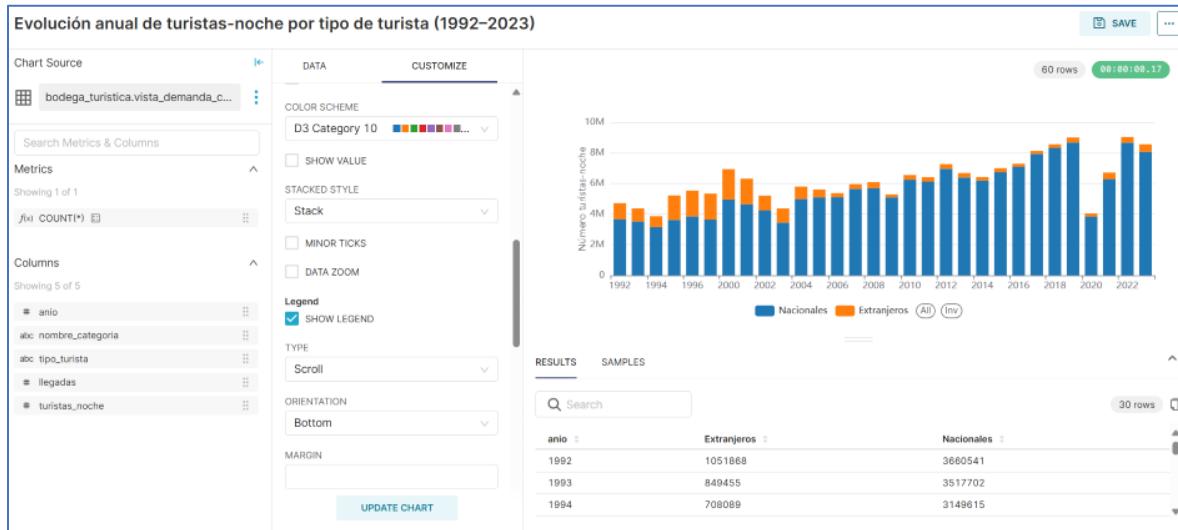
Gráfico generado con Apache Superset sobre evolución de turistas-noche.



Nota. Se muestra el histórico anual de turistas-noche en Acapulco por categoría de hotel.

Figura 56

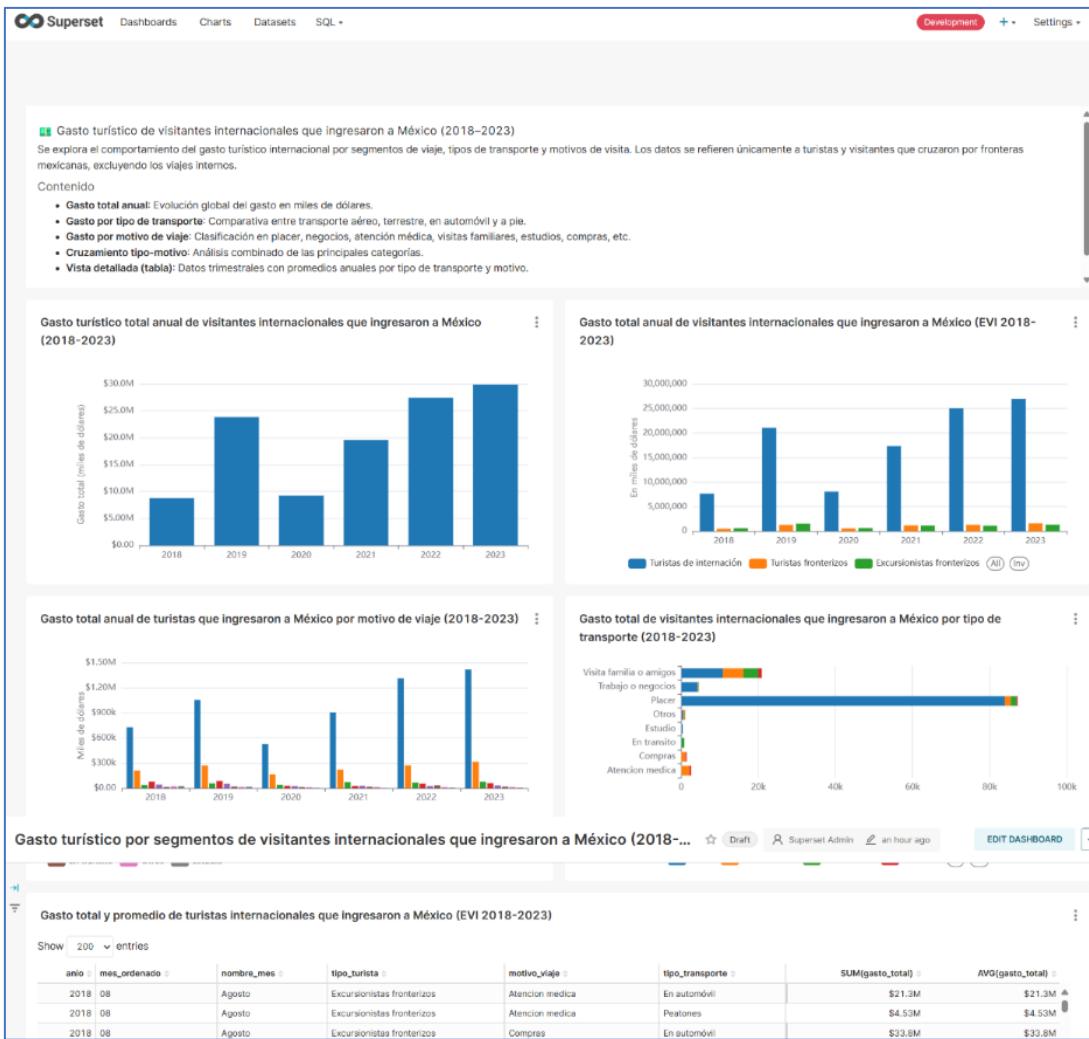
Gráfico generado con Apache Superset sobre evolución de pernoctaciones.



Nota. Se muestra la evolución anual de la ocupación hotelera en Acapulco por origen del turista.

Figura 57

Ejemplo de dashboard en Apache Superset utilizando vistas SQL.



Nota. Se agrupan varios gráficos relacionados (gasto turístico) para mostrarlos en un

dashboard.

Anexo 5. Generales del Caso de Uso

Caso de Uso “Visualizar datos turísticos”

Generales del Caso de Uso					
Nombre Caso Uso	“Visualizar datos turísticos”				
Creación	Cecilia Jiménez Meza	Fecha	05/diciembre/2024		
Ultima modificación		Fecha	19/diciembre/2024		
Objetivo					
<i>Este caso de uso permite a los usuarios visualizar datos turísticos disponibles en el Observatorio Turístico de Acapulco. Los datos pueden incluir estadísticas de visitantes, ocupación hotelera, entre otros.</i>					
Nivel del Caso de Uso	Prioridad	Complejidad			
Caso de uso de Administrador, Investigador y Público en general	ALTA	BAJA			
Actores involucrados					
Administrador, Investigador, Usuario estándar					
Evento Disparador					
Visualizar datos					
Precondiciones					
<ul style="list-style-type: none"> El sistema debe tener datos turísticos cargados y accesibles. El usuario debe tener acceso a un dispositivo con conexión a internet. El sistema debe tener datos turísticos actualizados y accesibles. El sistema debe tener la capacidad de generar gráficos y tablas interactivos. 					
Postcondiciones					
<ul style="list-style-type: none"> El usuario visualiza los datos turísticos solicitados en formato gráfico y/o tabular. El usuario puede realizar análisis básicos y exportar los datos si es necesario. 					
Referir					
Cargar tablas de datos turísticos, Descargar reportes.					

Flujo Principal

Paso	Usuario	Paso	Sistema
1	Accede a la página principal del Observatorio Turístico de Acapulco.		
		2	Muestra el menú principal con las categorías de datos turísticos disponibles.
3	Selecciona una categoría de datos turísticos (por ejemplo, ocupación hotelera).		
		4	Muestra una vista previa de los datos turísticos en formato gráfico y/o tabular.
5	Selecciona diferentes filtros y opciones de visualización para refinar los datos mostrados.		
		6	Actualiza la visualización de los datos según los filtros y opciones seleccionadas.
7	Analiza los datos y, si lo desea, exporta la información en formato CSV o PDF.		
		8	Proporciona opciones de exportación y genera el archivo en el formato solicitado.

Flujos Alternos

FA01 –“Si no hay datos disponibles para la categoría seleccionada” Ligado al paso No.4 del Flujo principal de los eventos.		
Paso	Acción	
	Usuario	Sistema
1		<i>Muestra un mensaje indicando que no hay datos disponibles para la categoría seleccionada.</i>
2	<i>Tiene la opción de seleccionar otra categoría de datos turísticos.</i>	

FA02 –“Si ocurre un error al cargar los datos.” Ligado al paso No.6 del Flujo principal de los eventos.		
Paso	Acción	
	Usuario	Sistema
1		<i>Muestra un mensaje de error indicando que ha ocurrido un problema al cargar los datos.</i>
2	<i>Tiene la opción de reintentar o contactar al administrador del sistema.</i>	

Autorización		
Nombre	Cargo	Firma

Bitácora de Cambios		
Fecha [dd/mm/aaaa]	Responsable	Descripción
19/dic/2024	Cecilia Jiménez Meza	Correcciones

Caso de Uso “Visualizar datos climáticos, de vuelos o embarcaciones”

Generales del Caso de Uso					
Nombre Caso Uso		Visualizar datos climáticos, de vuelos o embarcaciones			
Creación	Cecilia Jiménez Meza	Fecha	05/diciembre/2024		
Última modificación		Fecha	19/diciembre/2024		
Objetivo					
<i>Este caso de uso permite a los usuarios visualizar datos relacionados con el clima, vuelos y embarcaciones en el Observatorio Turístico de Acapulco. La información puede incluir pronósticos del tiempo, horarios de vuelos, estado de embarcaciones, entre otros.</i>					
Nivel del Caso de Uso	Prioridad	Complejidad			
Caso de uso de Administrador, Investigador y público en general	Media	Baja			
Actores involucrados					
Usuario Administrador, Investigador, Usuario estándar					
Evento Disparador					
Visualizar datos climáticos					
Precondiciones					
<ul style="list-style-type: none"> El sistema debe tener datos climáticos, de vuelos y embarcaciones cargados y accesibles. El usuario debe tener acceso a un dispositivo con conexión a internet. La interfaz de visualización debe ser intuitiva y fácil de usar. Los gráficos y tablas deben ser interactivos y permitir el filtrado y análisis dinámico de los datos. La información debe actualizarse en tiempo real, especialmente para datos de vuelos y embarcaciones. 					
Postcondiciones					
<ul style="list-style-type: none"> El usuario visualiza los datos solicitados en forma de mapas. 					
Referir					
Ninguno					

Flujo Principal

Paso	Usuario	Paso	Sistema
1	<i>Accede a la página principal del Observatorio Turístico de Acapulco.</i>		
		2	<i>Muestra las opciones de navegación.</i>
3	<i>Selecciona la opción “Mapas” de la barra de navegación.</i>		
		4	<i>Muestra el menú principal con las categorías de mapas disponibles: Vuelos, Embarcaciones, Radar meteorológico, Vientos, Mareas y Temperatura.</i>
5	<i>Selecciona una categoría de mapa (por ejemplo, Vientos).</i>		
		6	<i>Muestra una vista previa del mapa seleccionado.</i>
7	<i>Selecciona diferentes filtros y opciones de visualización para refinar los datos mostrados (por ejemplo, pronóstico por horas, días, etc.).</i>		
		8	<i>Actualiza la visualización de los datos según los filtros y opciones seleccionadas.</i>

9	<i>Analiza los datos climáticos y, si lo desea, puede cambiar a la visualización de datos de vuelos, embarcaciones u otra categoría de mapas.</i>		
		10	<i>Permite la visualización de datos de vuelos, embarcaciones o la categoría de mapa seleccionado.</i>

Flujos Alternos

FA01 –“Si no hay datos disponibles para la categoría seleccionada.” *Ligado al paso No.4 del Flujo principal de los eventos.*

Paso	Acción	
	Usuario	Sistema
1		<i>Muestra un mensaje indicando que no hay datos disponibles para la categoría seleccionada.</i>
2	<i>Tiene la opción de seleccionar otra categoría de datos.</i>	

FA02 –“Si ocurre un error al cargar los datos”. *Ligado al paso No.8 del Flujo principal de los eventos.*

Paso	Acción	
	Usuario	Sistema
1		<i>Muestra un mensaje de error indicando que ha ocurrido un problema al cargar los datos.</i>
2	<i>Tiene la opción de reintentar o contactar al administrador del sistema.</i>	

Autorización		
Nombre	Cargo	Firma

Bitácora de Cambios		
Fecha [dd/mm/aaaa]	Responsable	Descripción
19/dic/2024	Cecilia Jiménez Meza	Correcciones

Caso de Uso “Cargar Tesis”

Generales del Caso de Uso					
Nombre Caso Uso	Cargar tesis al repositorio del servidor				
Creación	Cecilia Jiménez Meza	Fecha	05/diciembre/2024		
Última modificación		Fecha	19/diciembre/2024		
Objetivo					
<i>Incorporar un nuevo documento de tesis al repositorio designado en el servidor para que esté disponible para consulta y descarga a través de la plataforma web del Observatorio (“Centro Documental”).</i>					
Nivel del Caso de Uso	Prioridad	Complejidad			
Caso de uso de Administrador	ALTA	MEDIA			
Actores involucrados					
Administrador					
Evento Disparador					
<i>Necesidad de añadir una nueva tesis al Centro Documental.</i>					
Precondiciones					
<ul style="list-style-type: none"> El Administrador posee las credenciales válidas de acceso al servidor SFTP donde reside la carpeta de documentos del Observatorio. El documento de tesis a cargar está en formato PDF y cumple con las restricciones de tamaño definidas (máximo 20 MB). El servidor del Observatorio está operativo y accesible para conexiones remotas seguras. La aplicación web del Observatorio está configurada para leer el contenido de la carpeta designada. 					
Postcondiciones					
<ul style="list-style-type: none"> El documento de la tesis se encuentra almacenado correctamente en la carpeta designada del servidor. El documento es posteriormente indexado o listado por la aplicación web del Observatorio y queda disponible para ser visualizado y descargado en el “Centro Documental”. Si la transferencia del archivo al servidor falla, el documento no se incorpora al repositorio y no estará disponible en la plataforma. 					
Referir					
Visualizar tesis					

Flujo Principal

Paso	Usuario	Paso	Sistema (Servidor/Aplicación web)
1	<i>El usuario Administrador establece una conexión segura (SFTP) con el servidor utilizando sus credenciales.</i>		
		2	<i>(Servidor) Autentica al Administrador y concede acceso al sistema de archivos según sus permisos.</i>
3	<i>Navega hasta la carpeta designada específicamente para los documentos del “Centro Documental” en el servidor.</i>		
4	<i>Transfiere (copia/sube) el archivo PDF de la tesis desde su equipo local a la carpeta destino en el servidor.</i>		
		5	<i>(Servidor) Guarda el archivo PDF en la carpeta designada.</i>
		6	<i>(Aplicación web) De forma automática lee el contenido de la</i>

			<i>carpeta designada para actualizar su lista interna de documentos disponibles.</i>
7	<i>(Opcional) Verifica a través de la interfaz web del Observatorio que la nueva tesis aparece listada y es descargable en el “Centro Documental”.</i>		
8	<i>Cierra la conexión segura con el servidor.</i>		

Flujos Alternos**FA01 –“Fallo de autenticación al servidor.” Ligado al paso No.2 del Flujo principal de los eventos.**

Paso	Acción	
	Usuario	Sistema
1		<i>(Servidor) Rechaza las credenciales proporcionadas.</i>
2	<i>(Administrador) Recibe mensaje de error en su cliente de conexión. Debe verificar credenciales o contactar al administrador del servidor. No puede proceder con la carga.</i>	

FA02 –“Si ocurre un error de transferencia de archivo.” Ligado al paso No.5 del Flujo principal de los eventos.

Paso	Acción	
	Usuario	Sistema
1		<i>(Servidor) No puede almacenar el archivo, mostrando el tipo de error (permisos, espacio, conexión interrumpida).</i>
2	<i>(Administrador) Recibe un mensaje de error en su cliente de transferencia (SFTP). Debe diagnosticar el problema (verificar permisos, espacio) y reintentar la transferencia. El archivo no está disponible.</i>	

FA03 –“Documento no visible.” Ligado al paso No. 6 del Flujo principal de los eventos.

Paso	Acción	
	Usuario	Sistema
1		<i>(Aplicación web) No ha ejecutado aún el proceso de escaneo de la carpeta después de la carga.</i>
2	<i>(Administrador) Recibe un mensaje de error en su cliente de transferencia (SFTP). Debe diagnosticar el problema (verificar permisos, espacio) y reintentar la transferencia. El archivo no está disponible.</i>	
Autorización		
Nombre	Cargo	Firma

Bitácora de Cambios

Fecha [dd/mm/aaaa]	Responsable	Descripción
19/12/2024	Cecilia Jiménez Meza	Correcciones

Caso de Uso “Visualizar tesis”

Generales del Caso de Uso							
Nombre Caso Uso	Visualizar tesis						
Creación	Cecilia Jiménez Meza		Fecha	05/diciembre/2024			
Última modificación			Fecha	19/diciembre/2024			
Objetivo							
<i>Este caso de uso permite a los usuarios visualizar tesis disponibles en el Observatorio Turístico de Acapulco. Los documentos están cargados previamente en una carpeta del sistema.</i>							
Nivel del Caso de Uso	Prioridad	Complejidad					
Caso de uso de Administrador, Investigador y público en general	ALTA	BAJA					
Actores involucrados							
Administrador, Investigador, Usuario estándar							
Evento Disparador							
Visualizar tesis							
Precondiciones							
<ul style="list-style-type: none"> El sistema debe tener tesis cargadas y accesibles en el Centro Documental. El usuario debe tener acceso a un dispositivo con conexión a internet. La interfaz de visualización debe ser intuitiva y fácil de usar. El sistema debe soportar la búsqueda, visualización y descarga de tesis búsqueda de palabras clave. Los documentos de tesis deben estar disponibles en un formato accesible (PDF). 							
Postcondiciones							
<ul style="list-style-type: none"> El usuario visualiza y/o descarga las tesis solicitadas en formato digital (PDF). El usuario puede realizar búsquedas de cada tesis. 							
Referir							
Cargar tesis al repositorio del servidor							

Flujo Principal

Paso	Usuario	Pa so	Sistema
1	<i>Accede al Centro Documental del Observatorio Turístico de Acapulco.</i>		
		2	<i>Muestra las opciones de la barra de navegación.</i>
3	<i>Selecciona la opción "Centro Documental".</i>		
		4	<i>Muestra una lista de tesis disponibles con títulos.</i>
5	<i>Busca una tesis por medio de una palabra clave.</i>		
		6	<i>Actualiza la lista de tesis según los criterios de búsqueda del usuario.</i>
7	<i>Selecciona una tesis de la lista.</i>		
		8	<i>Muestra los detalles de la tesis seleccionada y proporciona enlaces para descargar el documento en formato PDF.</i>

9	<i>Hace clic en el enlace para descargar la tesis en su dispositivo.</i>		
		10	<i>Descarga el documento seleccionado al dispositivo del usuario.</i>

Flujos Alternos

FA01 –“Si no hay tesis disponibles en el sistema.” Ligado al paso No.4 del Flujo principal de los eventos.

Paso	Acción	
	Usuario	Sistema
1		<i>Muestra un mensaje indicando que no hay tesis disponibles en el Centro Documental.</i>
2	<i>Tiene la opción de realizar otra búsqueda o volver a la página principal.</i>	
FA02 –“Si ocurre un error al cargar los datos” Ligado al paso No.5 del Flujo principal de los eventos.		
Paso	Acción	
	Usuario	Sistema
1		<i>Muestra un mensaje de error indicando que ha ocurrido un problema al cargar los datos.</i>
2	<i>Tiene la opción de reintentar o contactar al administrador del sistema.</i>	

Autorización		
Nombre	Cargo	Firma

Bitácora de Cambios		
Fecha [dd/mm/aaaa]	Responsable	Descripción
19/12/2024	Cecilia Jiménez Meza	Correcciones

Caso de Uso “Cargar documentos de investigación”

Generales del Caso de Uso					
Nombre Caso Uso	Cargar documentos de investigación en el repositorio del servidor				
Creación	Cecilia Jiménez Meza	Fecha	05/diciembre/2024		
Última modificación		Fecha	29/diciembre/2024		
Objetivo					
<i>Incorporar un nuevo documento de investigación al repositorio designado en el servidor para que pueda consultarse y descargarse en la plataforma web del Observatorio (“Centro Documental”).</i>					
Nivel del Caso de Uso		Prioridad	Complejidad		
Caso de uso de Administrador	ALTA		MEDIA		
Actores involucrados					
Administrador					
Evento Disparador					
<i>Necesidad de añadir un nuevo documento de investigación en el Centro Documental.</i>					
Precondiciones					
<ul style="list-style-type: none"> El administrador posee credenciales válidas de acceso al servidor. El servidor del Observatorio debe estar operativo y accesible. La aplicación web del Observatorio está configurada para leer el contenido de la carpeta designada. El documento debe ser un archivo en formato PDF con un tamaño máximo de 20 MB. 					
Postcondiciones					
<ul style="list-style-type: none"> Los documentos de investigación han sido cargados correctamente y están disponibles para visualización por todos los usuarios. Si la carga falla, los documentos no se registran en el sistema y el usuario recibe un mensaje de error. 					
Referir					
Visualizar documentos de investigación					

Flujo Principal

Paso	Usuario	Paso	Sistema (Servidor/Aplicación web)
1	<i>Accede al sistema mediante una conexión segura con el servidor utilizando sus credenciales.</i>		
		2	<i>(Servidor) Verifica las credenciales del usuario y permite el acceso.</i>
3	<i>Navega a la carpeta designada para la carga de documentos de investigación.</i>		
4	<i>Transfiere el archivo PDF del documento de investigación a la carpeta destino en el servidor.</i>		
	.	5	<i>(Servidor) Almacena físicamente el archivo PDF en la carpeta designada.</i>
		6	<i>(Aplicación web) De forma automática lee el contenido de la carpeta designada para actualizar su lista interna.</i>
7	<i>(Opcional) Verifica en la interfaz web del Observatorio que el documento aparece en la lista y es descargable en el “Centro Documental”.</i>		
8	<i>Cierra la conexión segura con el servidor.</i>		

Flujos Alternos

FA01 –“Fallo de autenticación al servidor”. Ligado al paso No.2.		
Paso	Acción	
	Usuario	Sistema
1		<i>(Servidor) Rechaza las credenciales proporcionadas.</i>
2	<i>Recibe mensaje de error en su cliente de conexión. Debe verificar credenciales o contactar al administrador del servidor. No procede con la carga.</i>	

FA02 –“Error durante la transferencia del archivo”. Ligado al paso No.5.		
Paso	Acción	
	Usuario	Sistema
1		<i>(Servidor) No puede almacenar el archivo (ej. error de permisos, disco lleno, conexión interrumpida).</i>
2	<i>Recibe un mensaje de error en su cliente de transferencia. Debe reintentar la transferencia. El archivo no está disponible.</i>	

FA03 –“Documento no visible en la web inmediatamente”. Ligado al paso No.6.		
Paso	Acción	
	Usuario	Sistema
1		<i>(Aplicación web) No ha ejecutado aún el proceso de escaneo de la carpeta después de la carga.</i>
2	<i>No ve el documento listado en el “Centro Documental”. Debe esperar el próximo ciclo de escaneo o reintentar la carga.</i>	

Autorización		
Nombre	Cargo	Firma

Bitácora de Cambios		
Fecha [dd/mm/aaaa]	Responsable	Descripción
29/dic/2024	Cecilia Jiménez Meza	Correcciones

Caso de Uso “Visualizar documentos de investigación”

Generales del Caso de Uso					
Nombre Caso Uso	<i>Visualizar documentos de investigación</i>				
Creación	Cecilia Jiménez Meza	Fecha	05/diciembre/2024		
Última modificación		Fecha	29/diciembre/2024		
Objetivo					
<i>Permitir a los usuarios buscar, visualizar y descargar los documentos de investigación disponibles en el Centro Documental.</i>					
Nivel del Caso de Uso	Prioridad	Complejidad			
Caso de uso de Administrador, Investigador y público en general	ALTA	BAJA			
Actores involucrados					
Administrador, Investigador, usuario estándar					
Evento Disparador					
<i>Visualizar documento</i>					
Precondiciones					
<ul style="list-style-type: none"> El sistema debe tener documentos de investigación cargados y accesibles en el Centro Documental. El usuario debe tener acceso a un dispositivo con conexión a internet. La interfaz de visualización debe ser intuitiva y fácil de usar. El sistema debe soportar la búsqueda, visualización y descarga de documentos por diferentes criterios (autor, tema, año). Los documentos de investigación deben estar disponibles en formatos accesibles (por ejemplo, PDF). 					
Postcondiciones					
<ul style="list-style-type: none"> El usuario visualiza y/o descarga los documentos de investigación solicitados en formato digital (por ejemplo, PDF). El usuario puede realizar búsquedas y acceder a información detallada sobre cada documento. 					
Referir					
Cargar documentos de investigación en el repositorio del servidor.					

Flujo Principal

Paso	Usuario	Paso	Sistema
1	<i>Accede a la página principal del Observatorio Turístico de Acapulco.</i>		
		2	<i>Muestra el menú principal, incluyendo una opción para el "Centro Documental".</i>
3	<i>Selecciona la opción "Centro Documental".</i>		
		4	<i>Muestra una lista de documentos de investigación disponibles con títulos y descripciones.</i>
5	<i>Busca un documento específico usando un campo de búsqueda o filtros (autor, tema, año).</i>		
		6	<i>Actualiza la lista de documentos según los criterios de búsqueda del usuario.</i>
7	<i>Selecciona un documento de la lista.</i>		
		8	<i>Muestra los detalles del documento seleccionado y proporciona enlaces para visualizar y descargar el documento en</i>

			<i>formato digital.</i>
9	<i>Hace clic en el enlace para abrir y/o descargar el documento en su dispositivo.</i>		
		10	<i>Abre el documento digital para que el usuario pueda leerlo o lo descarga al dispositivo del usuario.</i>

Flujos Alternos

FA01 –“Si no hay documentos disponibles en el sistema.” <i>Ligado al paso No.6 del Flujo principal de los eventos.</i>					
Paso	Acción				
	<table border="1"> <tr> <th>Usuario</th> <th>Sistema</th> </tr> <tr> <td>1</td> <td><i>Muestra un mensaje indicando que no hay documentos disponibles en el Centro Documental.</i></td> </tr> </table>	Usuario	Sistema	1	<i>Muestra un mensaje indicando que no hay documentos disponibles en el Centro Documental.</i>
Usuario	Sistema				
1	<i>Muestra un mensaje indicando que no hay documentos disponibles en el Centro Documental.</i>				
2	<i>Tiene la opción de realizar otra búsqueda o volver a la página principal.</i>				

FA02 –“Si ocurre un error al cargar los datos.” <i>Ligado al paso No.5 del Flujo principal de los eventos.</i>					
Paso	Acción				
	<table border="1"> <tr> <th>Usuario</th> <th>Sistema</th> </tr> <tr> <td>1</td> <td><i>Muestra un mensaje de error indicando que ha ocurrido un problema al cargar los datos.</i></td> </tr> </table>	Usuario	Sistema	1	<i>Muestra un mensaje de error indicando que ha ocurrido un problema al cargar los datos.</i>
Usuario	Sistema				
1	<i>Muestra un mensaje de error indicando que ha ocurrido un problema al cargar los datos.</i>				
2	<i>Tiene la opción de reintentar o contactar al soporte técnico.</i>				

Autorización		
Nombre	Cargo	Firma

Bitácora de Cambios		
Fecha [dd/mm/aaaa]	Responsable	Descripción
29/12/2024	Cecilia Jiménez Meza	Correcciones

Caso de Uso “Descargar reportes”

Generales del Caso de Uso					
Nombre Caso Uso		<i>Descargar reportes</i>			
Creación	Cecilia Jiménez Meza	Fecha	05/diciembre/2024		
Última modificación		Fecha	29/diciembre/2024		
Objetivo					
<i>Este caso de uso permite a los usuarios descargar reportes generados a partir de las visualizaciones de datos turísticos. Los reportes, que pueden incluir tablas y gráficos en formato PDF. Los reportes disponibles abarcan diversos aspectos del turismo, como datos turísticos por período, ocupación hotelera, PIB y gasto turístico.</i>					
Nivel del Caso de Uso		Prioridad			
Caso de uso de Administrador, Investigador, público en general		ALTA			
Actores Involucrados					
Administrador, Investigador, usuario estándar					
Evento Disparador					
<i>Descargar reporte</i>					
Precondiciones					
<ul style="list-style-type: none"> El usuario debe tener acceso a un dispositivo con conexión a internet. El sistema debe tener datos cargados y accesibles. Los análisis estadísticos deben estar previamente realizados y disponibles para generar los reportes. 					
Postcondiciones					
<ul style="list-style-type: none"> El usuario descarga el reporte solicitado en formato PDF. El usuario puede guardar o imprimir el reporte descargado. 					
Referir					
<i>Visualizar datos turísticos, Visualizar tesis.</i>					

Flujo Principal

Paso	Usuario	Paso	Sistema
1	<i>Accede a la sección de visualización de los reportes del sistema.</i>		
		2	<i>Muestra la interfaz de visualización de los reportes con tablas y/o gráficos relevantes.</i>
3	<i>Selecciona el tipo de reporte que desea descargar.</i>		
4	<i>Selecciona el período para el reporte.</i>		
		5	<i>Genera el reporte en formato PDF, el cual debe estar previamente analizado con métodos estadísticos, según el tipo de reporte y el período seleccionados.</i>
		6	<i>Genera un enlace de descarga del reporte.</i>
7	<i>Hace clic en el enlace para descargar el reporte.</i>		
		8	<i>Descarga el reporte al dispositivo del usuario.</i>

Flujos Alternos

<i>FA01 –“Si no hay datos disponibles en el período para generar el reporte.” Ligado al paso No.5 del Flujo principal de los eventos.</i>

Paso	Acción	
	Usuario	Sistema
1		<i>Muestra un mensaje indicando que no hay datos disponibles en ese período para generar el reporte.</i>
2	<i>Tiene la opción de seleccionar un período distinto, elegir otro reporte o volver a la página principal.</i>	
FA02 –“Si ocurre un error durante la generación del reporte.” Ligado al paso No.5 del Flujo principal de los eventos.		
Paso	Acción	
	Usuario	Sistema
1		Muestra un mensaje de error indicando que ha ocurrido un problema al generar el reporte.
2	<i>Tiene la opción de reintentar la descarga o contactar al administrador del sistema.</i>	

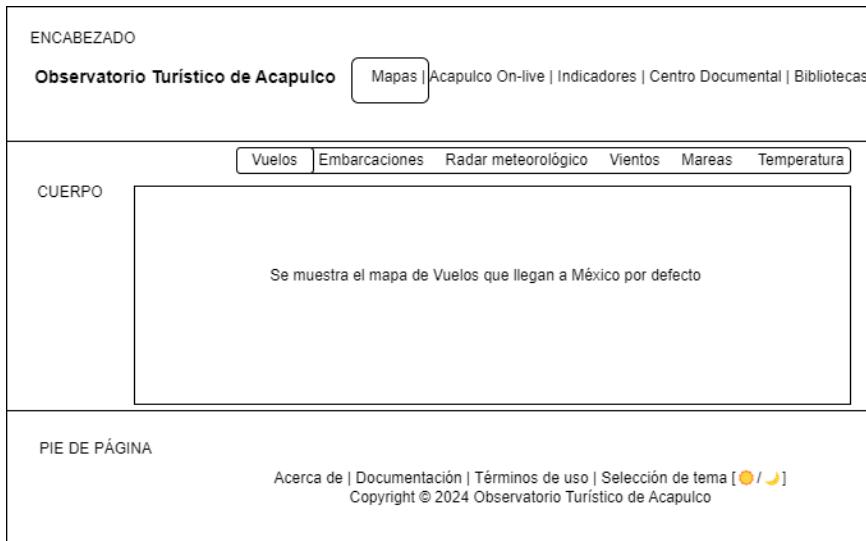
Autorización		
Nombre	Cargo	Firma

Bitácora de Cambios		
Fecha [dd/mm/aaaa]	Responsable	Descripción
29/12/2024	Cecilia Jiménez Meza	Correcciones

Anexo 6. Pantallas de la plataforma web

Figura 58

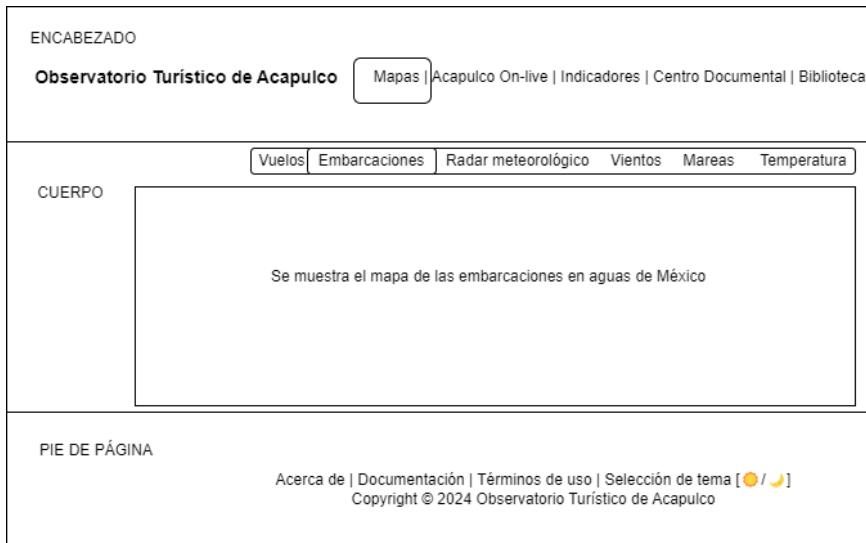
Maqueta que muestra la sección de Mapas/Vuelos.



Nota. Elaboración propia, inspirada en los principios del diseño propuestos por Pressman (2010).

Figura 59

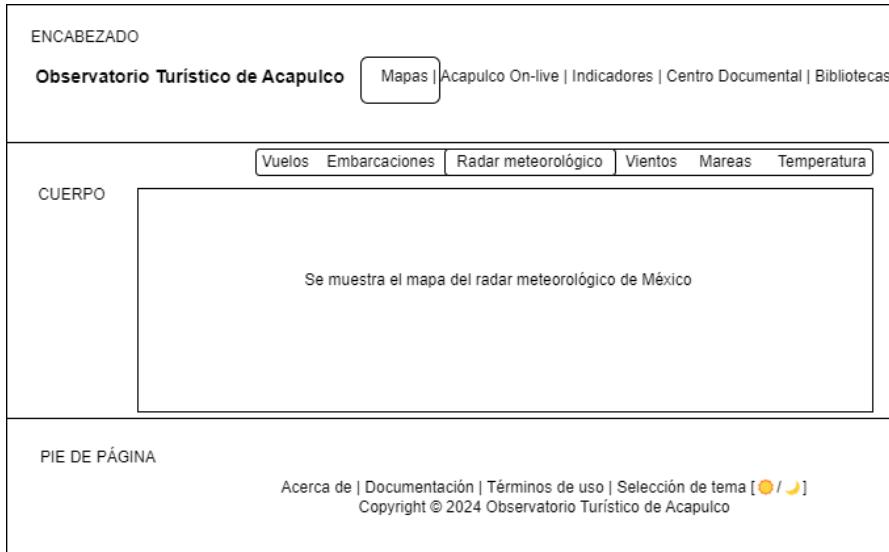
Maqueta que muestra la sección de Mapas/Embarcaciones.



Nota. Elaboración propia, inspirada en los principios del diseño propuestos por Pressman (2010).

Figura 60

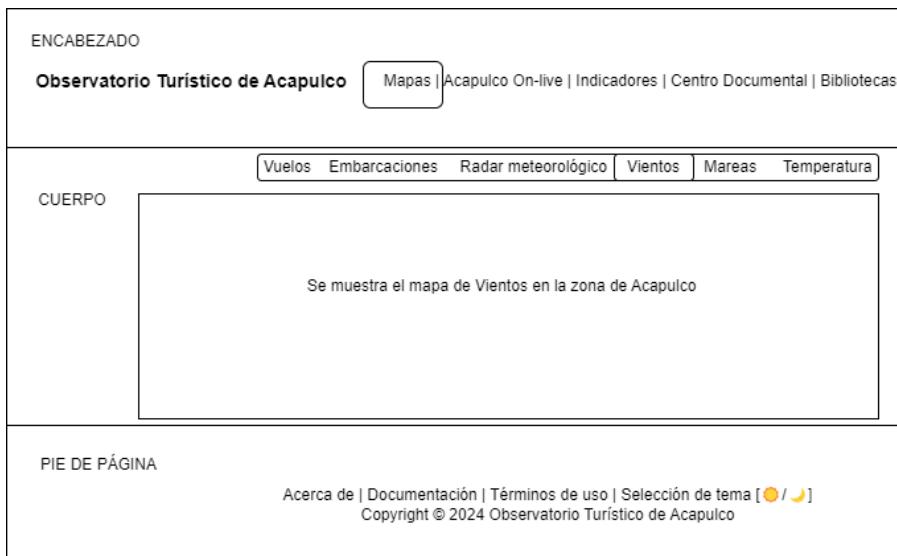
Maqueta que muestra la sección de Mapas/Radar meteorológico.



Nota. Elaboración propia, inspirada en los principios del diseño propuestos por Pressman (2010).

Figura 61

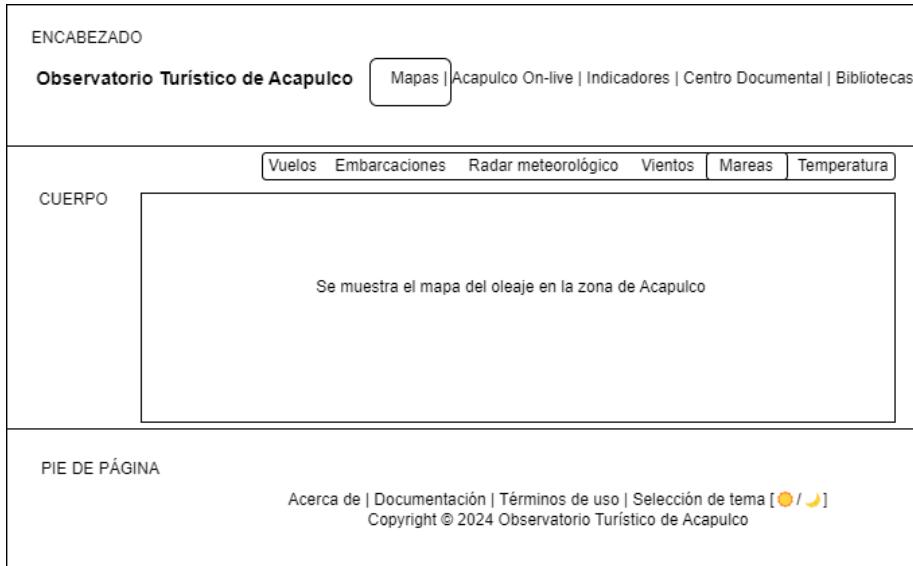
Maqueta que muestra la sección de Mapas/Vientos.



Nota. Elaboración propia, inspirada en los principios del diseño propuestos por Pressman (2010).

Figura 62

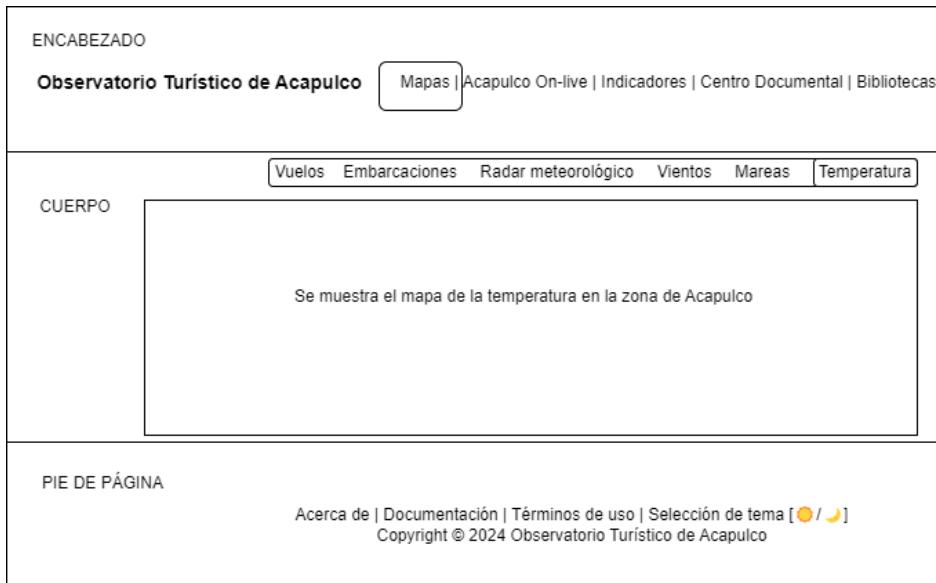
Maqueta que muestra la sección de Mapas/Mareas.



Nota. Elaboración propia, inspirada en los principios del diseño propuestos por Pressman (2010).

Figura 63

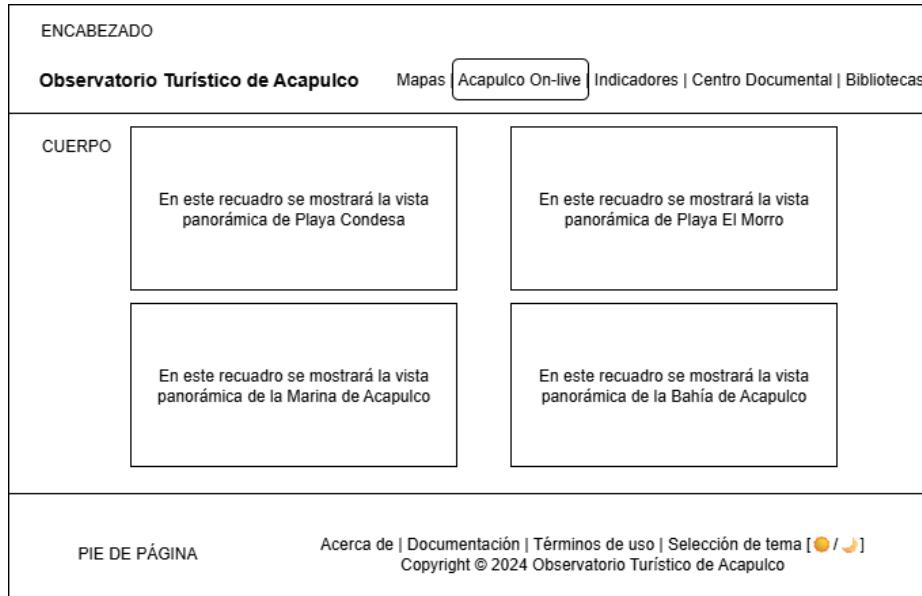
Maqueta que muestra la sección de Mapas/Temperatura.



Nota. Elaboración propia, inspirada en los principios del diseño propuestos por Pressman (2010).

Figura 64

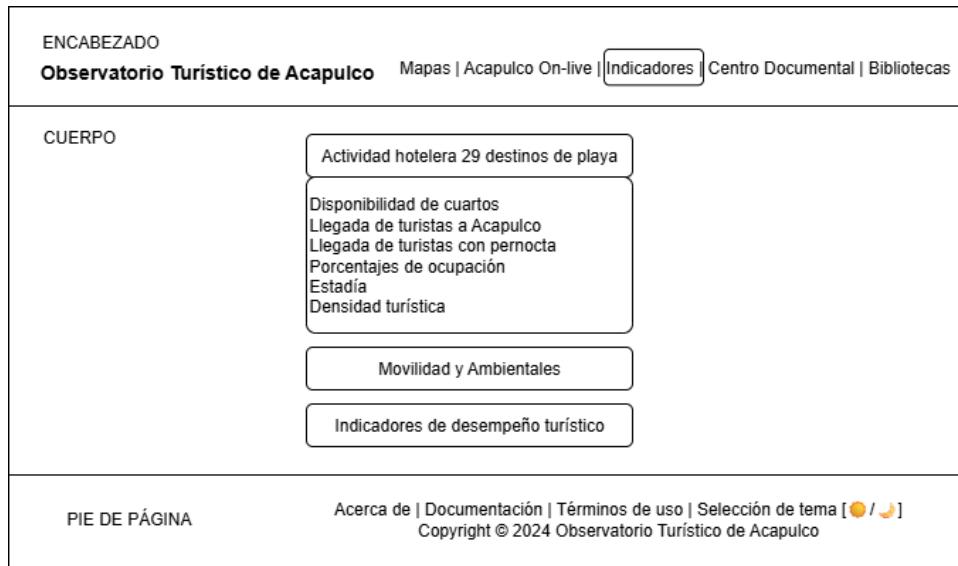
Maqueta de la sección Acapulco On-live,



Nota. Elaboración propia, inspirada en los principios del diseño propuestos por Pressman (2010).

Figura 65

Maqueta Submenú Actividad hotelera.



Nota. Elaboración propia, inspirada en los principios del diseño propuestos por Pressman (2010).

Figura 66

Maqueta de Actividad hotelera/Disponibilidad de cuartos.

ENCABEZADO
Observatorio Turístico de Acapulco Mapas | Acapulco On-live | Indicadores | Centro Documental | Bibliotecas

CUERPO

- Actividad hotelera 29 destinos de playa
- Disponibilidad de cuartos
- Llegada de turistas a Acapulco
- Llegada de turistas con pernocta
- Porcentajes de ocupación
- Estadía
- Densidad turística

- Movilidad y ambientales
- Indicadores de desempeño turístico

PIE DE PÁGINA Acerca de | Documentación | Términos de uso | Selección de tema [🌎 / 🎯]
Copyright © 2024 Observatorio Turístico de Acapulco

Nota. Elaboración propia, inspirada en los principios del diseño propuestos por Pressman (2010).

Figura 67

Maqueta de Disponibilidad de cuartos, con tablas y gráficos.

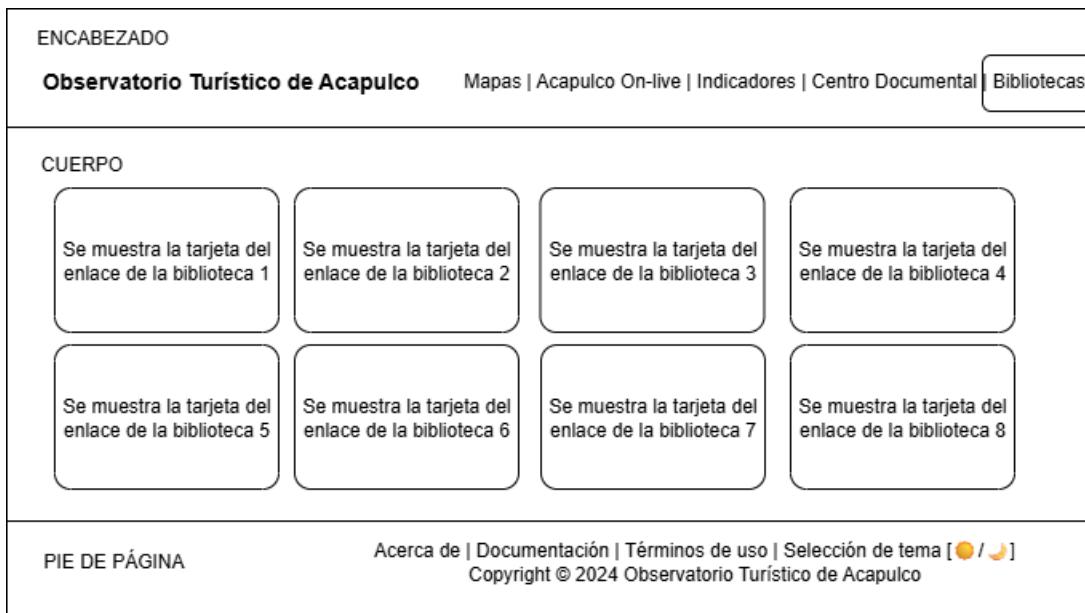
ENCABEZADO
Observatorio Turístico de Acapulco Mapas | Acapulco On-live | Indicadores | Centro Documental | Bibliotecas

CUERPO

Disponibilidad de cuartos	Selector año
Dashboard de disponibilidad de cuartos	
Se muestra una gráfica de datos relacionados al indicador seleccionado (cuartos disponibles, ocupados por nacionales, ocupados por extranjeros)	Se muestra una tabla de datos relacionados con el indicador seleccionado

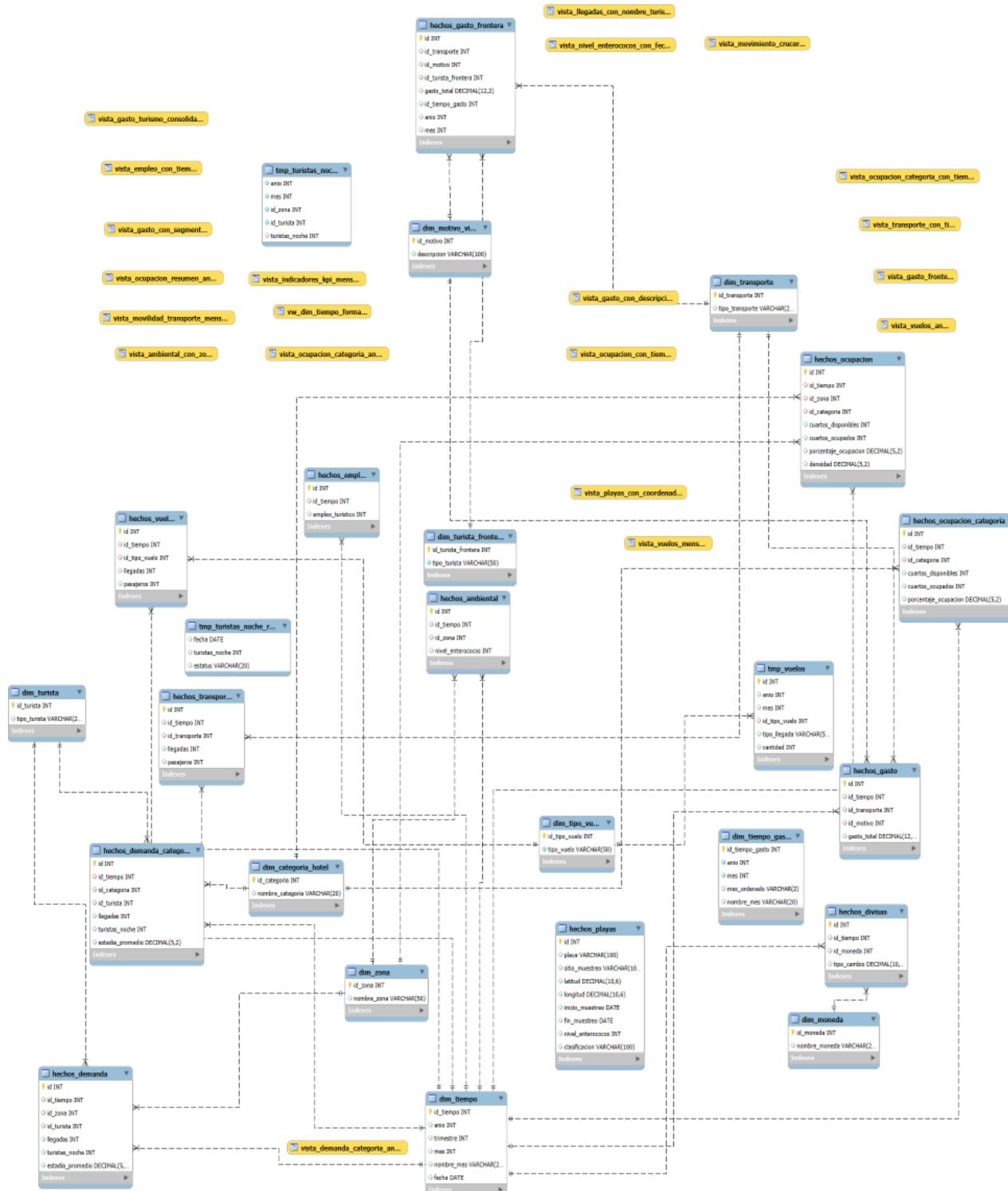
PIE DE PÁGINA Acerca de | Documentación | Términos de uso | Selección de tema [🌎 / 🎯]
Copyright © 2024 Observatorio Turístico de Acapulco

Nota. Elaboración propia, inspirada en los principios del diseño propuestos por Pressman (2010).

Figura 68*Maqueta de Bibliotecas.*

Nota. Elaboración propia, inspirada en los principios del diseño propuestos por Pressman (2010).

Anexo 7. Diagrama Entidad Relación de bodega_turistica



Anexo 8. Integración de datos y visualización con Apache Superset

Actualización de la tabla dim_tiempo

Se mejoró la compatibilidad de la tabla dim_tiempo con herramientas de análisis y visualización (como Apache Superset, Power BI), al permitir el uso de fechas reales (DATE) en lugar de representaciones numéricas. La estructura actualizada se muestra en la Figura 69.

Figura 69

Tabla SQL dim_tiempo con la columna fecha de tipo DATE.

	id_tiempo	anio	trimestre	mes	nombre_mes	fecha
▶	1	1992	1	1	Enero	1992-01-01
	2	1992	1	2	Febrero	1992-02-01
	3	1992	1	3	Marzo	1992-03-01
	4	1992	2	4	Abril	1992-04-01
	5	1992	2	5	Mayo	1992-05-01
	6	1992	2	6	Junio	1992-06-01
	7	1992	3	7	Julio	1992-07-01
	8	1992	3	8	Agosto	1992-08-01
	9	1992	3	9	Septiembre	1992-09-01
	10	1992	4	10	Octubre	1992-10-01
	11	1992	4	11	Noviembre	1992-11-01
	12	1992	4	12	Diciembre	1992-12-01

Nota. Captura de pantalla de la tabla dim_tiempo en MySQL Workbench después de la actualización para incluir el campo de tipo DATE.



Creación de datasets consolidados de ocupación hotelera mensual

Se unificaron los datos de capacidad y ocupación hotelera por destino turístico en tres *datasets* estructurados para análisis comparativo, que incluyen a Acapulco y otros destinos de playa relevantes.

Carga Final de Datos Turísticos Mensuales

Se completó la inserción de los *datasets* históricos mensuales consolidados (1992–2023) en la base de datos bodega_turistica. Cabe señalar que, en una primera etapa, los registros se

limitaban únicamente Acapulco. Sin embargo, para asegurar una validación comparativa robusta y permitir análisis entre los destinos, se ampliaron los datos incluyendo 28 destinos de playa adicionales, todos con estructura homogénea y valores compatibles. Los *datasets* agregados se muestran en la Tabla 11.

Tabla 11

Datasets con sus tablas en MySQL y el total de registros agregados.

Dataset origen	Tabla destino en MySQL	Registros insertados
CUARTOS_MENSUAL.CSV	cuartos_mensual	31,668
LLEGADA_TURISTAS_MENSUAL.CSV	llegada_turistas_mensual	22,272
TURISTAS_NOCHE_MENSUAL.CSV	turistas_noche_mensual	22,272

Ajustes técnicos realizados

- **Encabezados corregidos** a minúsculas para compatibilidad con Python y MySQL.
- **Valores negativos conservados**, ajustando tipos de dato:
 - num_cuartos: INT
 - num_turistas: INT
- **Tipo de fecha validado**: DATE, con formato %d/%m/%Y.
- **NULLs tratados explícitamente**, respetando campos vacíos en las fuentes originales.
- **Carga realizada mediante Python (pandas + mysql.connector)** para mayor control y velocidad. Estos *scripts* se presentan al final de este reporte.

Tratamiento de valores inconsistentes

Con base en la revisión de las *Notas aclaratorias para centros turísticos* incluidas en el documento oficial de SECTUR titulado "**Ocupación hotelera en centros turísticos seleccionados**", se identificó que múltiples destinos presentan inconsistencias temporales,

actualizaciones asimétricas de oferta, cambios en clasificación hotelera y ausencia de datos para ciertas categorías o meses. Estas inconsistencias se traducen en valores nulos, negativos o atípicos en variables como cuartos disponibles, turistas noche o estadía, los cuales no reflejan condiciones reales del sector turístico, sino vacíos de reporte o ajustes administrativos.

Por tal motivo, se adoptó un enfoque conservador, excluyendo del análisis dichos valores para asegurar la consistencia metodológica, la comparabilidad temporal y la validez analítica de los indicadores derivados (como porcentaje de ocupación, estadía y densidad). Esta decisión se aplicó exclusivamente a los destinos de playa incluidos en el *dataset* actual, con el fin de evitar generalizaciones indebidas y mantener el enfoque temático del observatorio.

Visualización en Superset

Se implementaron visualizaciones preliminares en Superset, tomando como base las tablas cuartos_mensual, llegada_turistas_mensual y turistas_noche_mensual.

Dado el volumen de registros y la complejidad de los indicadores, la visualización directa de todas las observaciones resultaba saturada. Para facilitar el análisis, se diseñó un *dashboard* con filtros dinámicos incorporados. Por defecto, se configuró la vista para mostrar el año más reciente disponible, enfocado exclusivamente en el destino Acapulco, presentando sus valores correspondientes para cada uno de los tres indicadores. El *dashboard* resultante con filtros dinámicos quedó como en la Figura 70.

Figura 1

Dashboard mostrando los filtros configurados por destino.

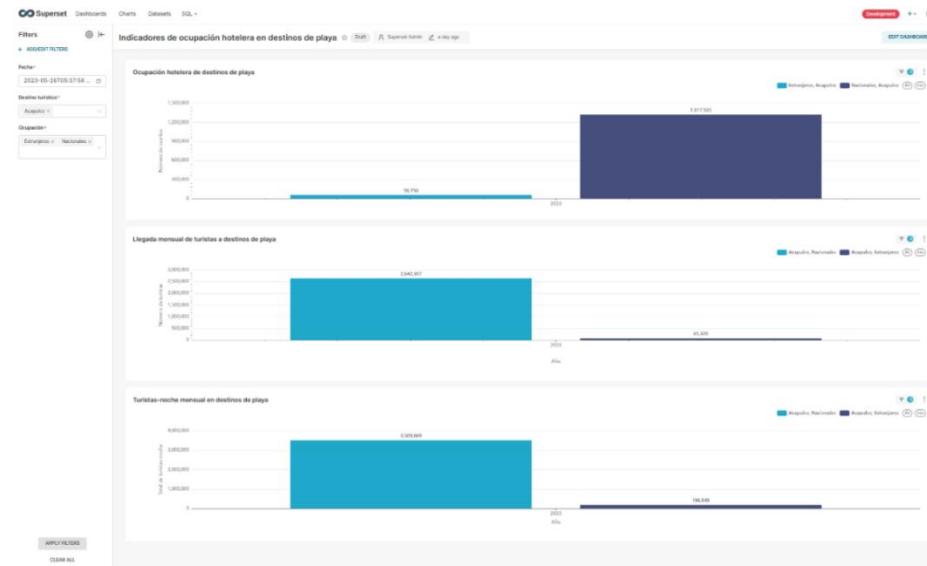


Nota. Captura del dashboard de indicadores en el entorno de validación (Apache Superset) mostrando la configuración de filtros por defecto.

Con los filtros para comparar a Acapulco con otros destinos de playa, se tiene la Figura 71.

Figura 2

Dashboard con filtros aplicados comparando Acapulco con otros destinos de playa.



Nota. Captura del dashboard creado en el entorno de desarrollo con Apache Superset.

Dataset de Reputación de playas en Google Maps

Se construyó un *dataset* consolidado para evaluar la reputación digital de las playas de Acapulco, a partir de reseñas extraídas de Google Maps.

1. Estructura del dataset

Para mitigar el sesgo asociado a un bajo volumen de evaluaciones, se aplicaron ajustes estadísticos correctivos. Se integró un análisis de sentimientos sobre los textos de las reseñas, con el propósito de estimar una puntuación ajustada que reflejara de forma más precisa la percepción real de cada playa. La estructura final se muestra en la Tabla 12.

Tabla 12

Estructura del dataset de la valoración a playas.

Columna	Descripción
cantidad_resenas	Número de reseñas en Google Maps en la fecha de consulta
cantidad_resenas_sentimiento	Número de reseñas con análisis de sentimientos.
fecha_consulta	Fecha de corte o actualización de los datos extraídos
nombre_playa	Nombre de la playa según Google Maps
promedio_bayesiano	Puntuación obtenida con ajuste bayesiano para equilibrar a playas con pocas reseñas y puntuación alta queden sobre otras con muchas reseñas y menor puntuación
puntuacion	Puntuación promedio (escala 1–5) de Google Maps
puntuacion_ajustada	Puntuación corregida con método bayesiano

2. Metodología de cálculo

- **Promedio Bayesiano** aplicado con un umbral mínimo de confianza de $m=100m = 100m=100$ para evitar distorsiones en la puntuación.

- Se identificaron playas con penalización alta como casos de reputación inflada por bajo volumen.

3. Uso del Dataset

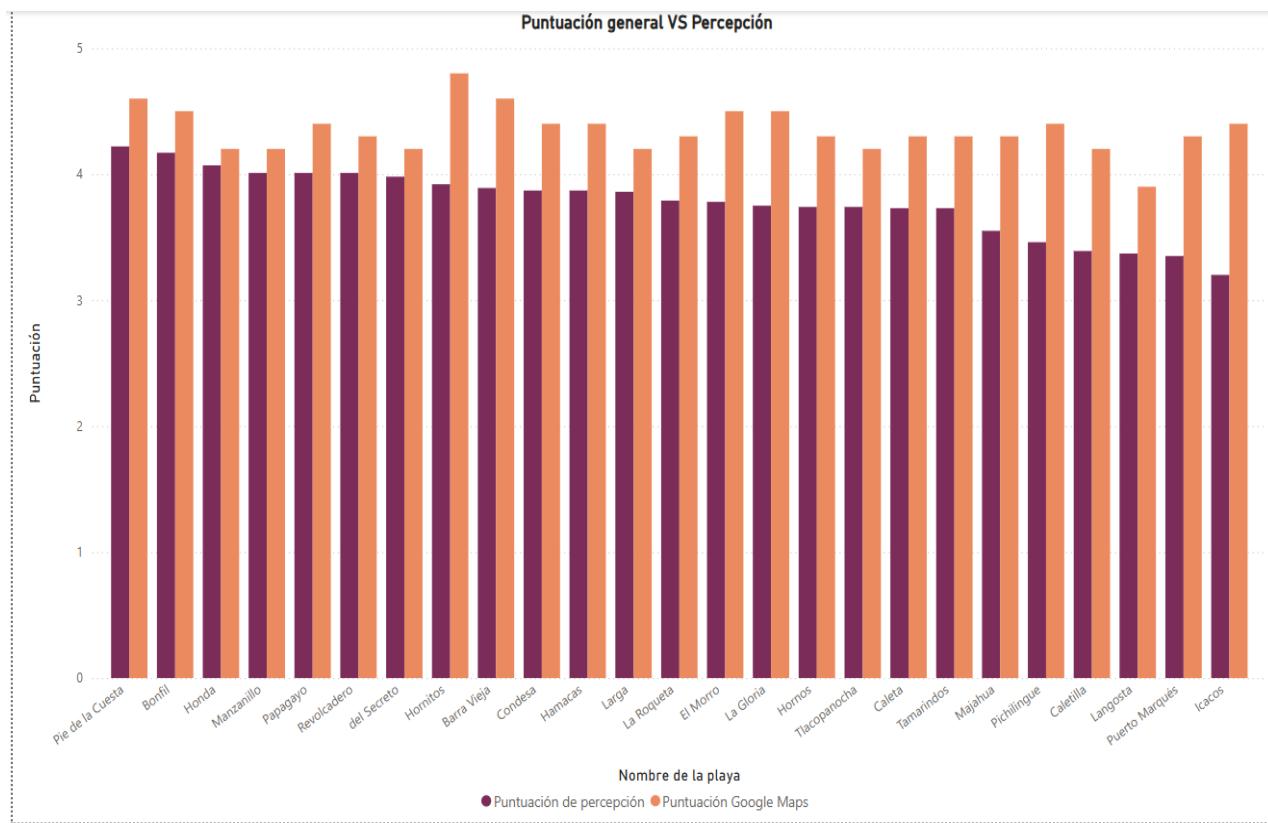
- Visualización en Power BI con segmentación por volumen, confianza o reputación consolidada.
- Identificación de playas con reputación sólida vs. reputación volátil.

4. Visualización en Power BI.

La prueba realizada en Power BI se muestra en la Figura 72.

Figura 3

Visualización de la valoración a playas utilizando Power BI.



Nota. Prueba de concepto hecha con Power BI para la validación del modelo.

Una vez que se tienen las gráficas y los *dashboards*, se procedió a configurar Superset para embeber estos trabajos.

Configuración de Apache Superset para embebido de *dashboards*

1. Contexto General

Se habilitó la visualización embebida de un *dashboard* temático sobre la “Valoración de Playas de Acapulco” a través de una página web externa. Para ello, se requirió configurar Apache Superset de forma segura, considerando las restricciones del entorno de desarrollo local (Docker, localhost) y evitando el uso de consola avanzada.

2. Ubicación de la Configuración

- **Archivo editado:** superset_config.py
- **Ruta relativa dentro del proyecto:** docker/pythonpath_dev/superset_config.py
- **Origen del contenedor:** Docker Compose
- **Modo de inicio:** vía navegador (<http://localhost:8088>)

3. Configuraciones Realizadas

Se añadieron al final del archivo superset_config.py las siguientes instrucciones para permitir el uso de Superset en un *iframe*:

```
# Configuraciones para permitir embeber dashboards
ENABLE_CORS = True
TALISMAN_ENABLED = False
```

Estas líneas permiten que Superset:

- Sea accesible desde páginas externas vía *iframe*.
- Evite la restricción X-Frame-Options: SAMEORIGIN, que bloqueaba la carga embebida.

4. Validación

Tras reiniciar el contenedor desde Docker Desktop (stop → start), se comprobó:

- Correcta carga del *dashboard* embebido en una página HTML.

- Visualización completa del contenido gráfico del *dashboard*.

5. Observaciones de Seguridad

Durante la prueba, se detectó que el usuario embebido tenía acceso al menú de navegación completo de Superset, incluyendo:

- Lista de *dashboards*.
- Lista de *datasets*.
- Acceso parcial a otras funcionalidades.

6. Medidas de Seguridad Planeadas

Se trabajó en el **ajuste de permisos del rol Public** para restringir el acceso únicamente a:

- Visualización del *dashboard* embebido.
- Lectura del *dataset* relacionado.
- Visualización del gráfico correspondiente.

Se eliminarán permisos como *can list*, *can edit*, *can explore*, entre otros, con el objetivo de mantener la integridad y privacidad del entorno analítico.

7. Evidencia del dashboard embebido

A continuación se presenta una captura de pantalla (**Figura 78**) como evidencia de la correcta integración del *dashboard* en una página web externa. En ella puede observarse:

- El título (“Valoración de Playas de Acapulco”).
- La gráfica comparativa de puntuaciones.
- La barra superior del navegador con la URL local (localhost) donde se aloja la página embebida.
- El código HTML para probar que el *dashboard* está correctamente embebido es el siguiente:

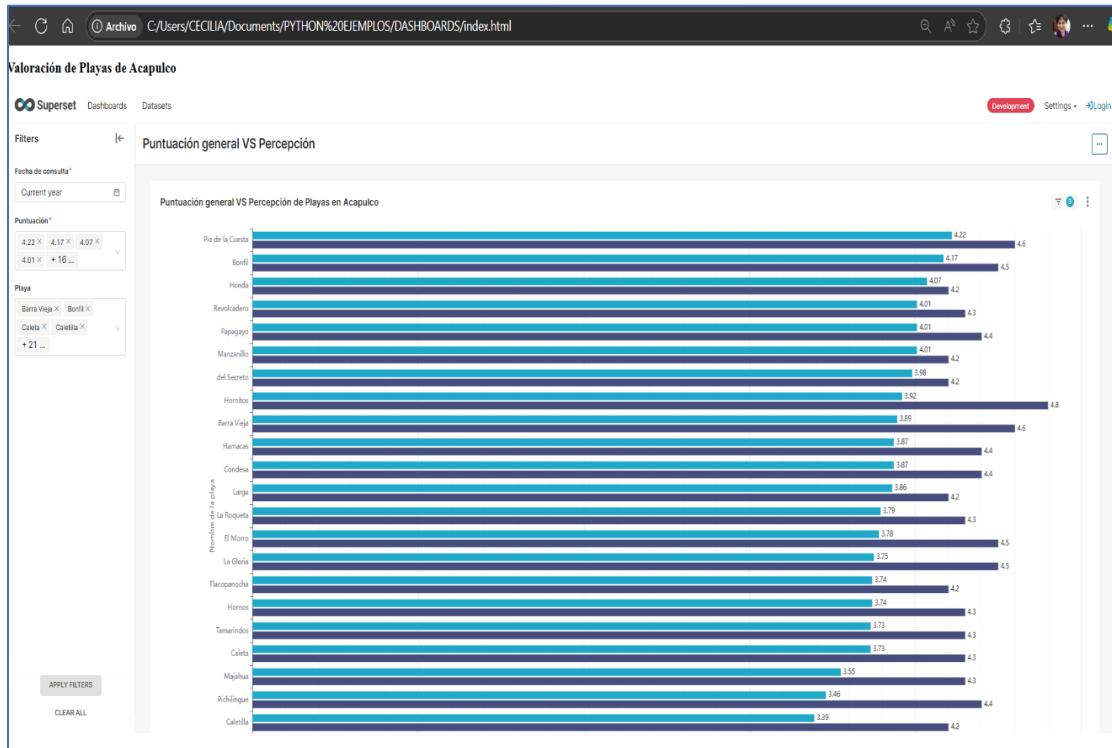
Archivo index.html de prueba

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Valoración de Playas de Acapulco</title>
</head>
<body>
    <h2>Valoración de Playas de Acapulco</h2>

    <iframe
        src="http://localhost:8088/superset/dashboard/p/5Mw8zJL8kbm/"
        width="100%"
        height="1000px"
        frameborder="0"
        allowfullscreen>
    </iframe>
</body>
</html>
```

Figura 73

Dashboard de Superset embebido en una página web.



Nota. Captura del *dashboard* embebido en una página HTML local, mostrando la URL en la barra superior del navegador y la visualización desplegada correctamente.

Scripts de Python utilizados

carga_cuartos_mensual.py

```
# Script para carga de datos del archivo CUARTOS_MENSUAL.CSV
import pandas as pd
import mysql.connector

# Configuración de conexión
config = {
    'host': '127.0.0.1',
    'port': 3306,
    'user': 'root',
    'password': 'TU CONTRASEÑA',
    'database': 'bodega_turistica'
}

# Ruta al archivo corregido
archivo_cuartos = "C:/Users/CECILIA/Desktop/CUARTOS_MENSUAL.CSV"
columnas = ["fecha", "destino", "num_cuartos", "estatus"]

# Función para cargar cuartos
def cargar_cuartos(ruta_csv, tabla, columnas):
    df = pd.read_csv(ruta_csv, encoding="latin1")
    df.columns = df.columns.str.strip().str.lower()
    df["fecha"] = pd.to_datetime(df["fecha"], format="%d/%m/%Y").dt.strftime("%Y-%m-%d")
    df = df.fillna(value=pd.NA)

    placeholders = ','.join(['%s'] * len(columnas))
    query = f"INSERT INTO {tabla} ({', '.join(columnas)}) VALUES ({placeholders})"
    valores = [tuple(None if pd.isna(x) else x for x in fila) for fila in df.itertuples(index=False, name=None)]

    conn = mysql.connector.connect(**config)
    cursor = conn.cursor()
    cursor.executemany(query, valores)
    conn.commit()
    print(f"{cursor.rowcount} filas insertadas en {tabla}.")
    cursor.close()
    conn.close()

# Ejecutar la carga
cargar_cuartos(archivo_cuartos, "cuartos_mensual", columnas)
```

carga_llegada_turistas_noche.py

```
# Carga de los datasets LLEGADA_TURISTAS_MENSUAL.CSV y
# TURISTAS_NOCHE_MENSUAL.CSV a tablas en la BD bodega_turistica
import pandas as pd
import mysql.connector

# Configuración de conexión
config = {
    'host': '127.0.0.1',
    'port': 3306,
    'user': 'root',
    'password': 'root',
    'database': 'bodega_turistica'
}
```

```

# Nuevas rutas de los archivos corregidos
rutas = {
    "llegada_turistas_mensual": "C:/Users/CECILIA/Desktop/LLEGADA_TURISTAS_MENSUAL.csv",
    "turistas_noche_mensual": "C:/Users/CECILIA/Desktop/TURISTAS_NOCHE_MENSUAL.csv"
}

# Columnas esperadas (ahora en minúsculas)
columnas_por_tabla = {
    "llegada_turistas_mensual": ["fecha", "destino", "num_turistas",
"estatus"],
    "turistas_noche_mensual": ["fecha", "destino", "num_turistas",
"estatus"]
}

# Función de carga
def cargar_csv(tabla, ruta_csv, columnas):
    df = pd.read_csv(ruta_csv, encoding="latin1")
    df.columns = df.columns.str.strip().str.lower()
    df["fecha"] = pd.to_datetime(df["fecha"],
format="%d/%m/%Y").dt.strftime("%Y-%m-%d")
    df = df.fillna(value=pd.NA)

    placeholders = ','.join(['%s'] * len(columnas))
    query = f"INSERT INTO {tabla} ({', '.join(columnas)}) VALUES ({placeholders})"
    valores = [tuple(None if pd.isna(x) else x for x in fila) for fila in
df.itertuples(index=False, name=None)]

    conn = mysql.connector.connect(**config)
    cursor = conn.cursor()
    cursor.executemany(query, valores)
    conn.commit()
    print(f"{cursor.rowcount} filas insertadas en {tabla}.")
    cursor.close()
    conn.close()

# Ejecutar cargas
for tabla, ruta in rutas.items():
    cargar_csv(tabla, ruta, columnas_por_tabla[tabla])

```

Anexo 9. Especificaciones del entorno del servidor de desarrollo

A continuación, se detallan las especificaciones técnicas y la arquitectura de software del servidor auto alojado (Orange Pi 5) que se configuró para el desarrollo, la integración y las pruebas funcionales del prototipo del Observatorio Turístico.

A. Especificaciones del hardware

- **Modelo:** Orange Pi 5
- **Procesador:** Rockchip RK3588 (8 núcleos ARM Cortex-A76 y 4x Cortex-A55)
- **RAM:** 8 GB tipo LPDDR4X
- **Almacenamiento:** El componente crítico es el uso de una unidad de estado sólido (SSD) NVMe conectada al puerto M.2 PCIe 3.0 . Esto es fundamental para el rendimiento de la base de datos y las operaciones de I/O de Apache Superset, superando significativamente las limitaciones de las tarjetas microSD
- **Eficiencia:** Consumo energético bajo, inferior a 15W bajo carga máxima, permitiendo una operación 24/7 con costo mínimo .

B. Software base (host)

- **Sistema operativo:** Linux Ubuntu Server versión ARM (Glass, 2020).
- **Orquestación:** Toda la arquitectura de servicios se gestiona mediante Docker y Docker Compose, permitiendo un despliegue aislado, consistente y replicable.
- **Firewall (Host):** Se utiliza UFW (Uncomplicated Firewall) con una política de "denegar por defecto", permitiendo únicamente el tráfico esencial (SSH, HTTP/S y el puerto de administración del proxy).

C. Arquitectura de servicios por contenedores

La arquitectura completa se define en un archivo `docker-compose.yml` que gestiona los siguientes servicios:

1. Proxy Inverso (Nginx-proxy-manager):

- Actúa como el único punto de entrada a los servicios dentro de la red local (intranet), gestionado por una interfaz gráfica.
- Su arquitectura permite la gestión centralizada de los servicios y la eventual configuración de certificados SSL, asegurando un modelo replicable para un futuro despliegue en producción.

2. Base de Datos (mysql:8.0):

- Almacena el *Data Warehouse* (`bodega_turistica`).
- Utiliza un volumen nombrado de Docker (ej. `mysql_data`) para garantizar la persistencia de los datos.

3. Plataforma BI (apache/superset:latest):

- Herramienta de análisis y visualización.
- Utiliza un volumen nombrado (ej. `superset_data`) para persistir su configuración y metadatos.
- (Nota: La configuración interna de Superset para el embebido, como `ENABLE_CORS` y `TALISMAN_ENABLED`, ya está detallada en el Anexo 8).

4. Redes de Docker:

- Se definen redes virtuales aisladas (`proxy-net` e `internal-net`) para que los servicios se comuniquen de forma segura, exponiendo solo el proxy a la red local del entorno de desarrollo.

Anexo 10. Mantenimiento del sistema analítico y gestión de datos turísticos

Frecuencia de publicación y mantenimiento de datasets.

En la Tabla 13 se resume las frecuencias observadas en la publicación de los principales *datasets* turísticos consultados, así como así como las acciones sugeridas para el monitoreo y actualización.

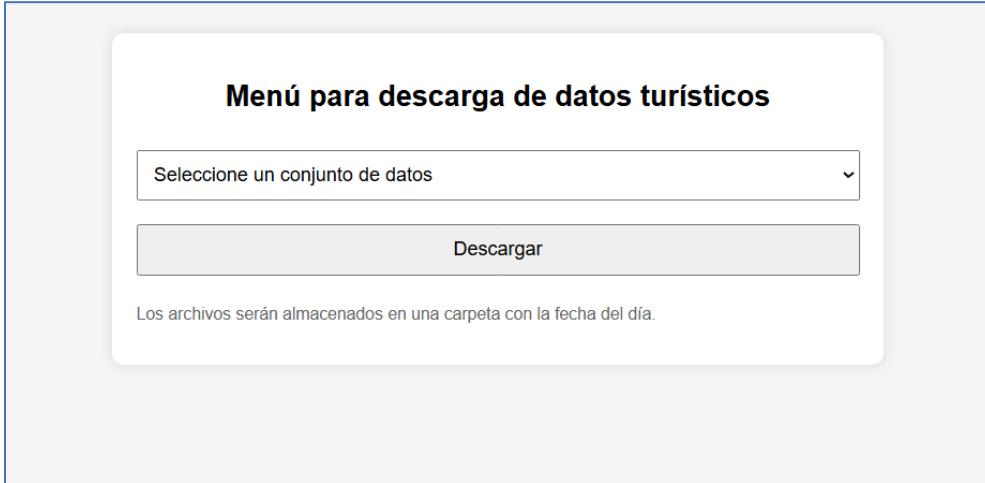
Tabla 13

Frecuencias de publicación y acciones sugeridas por dataset.

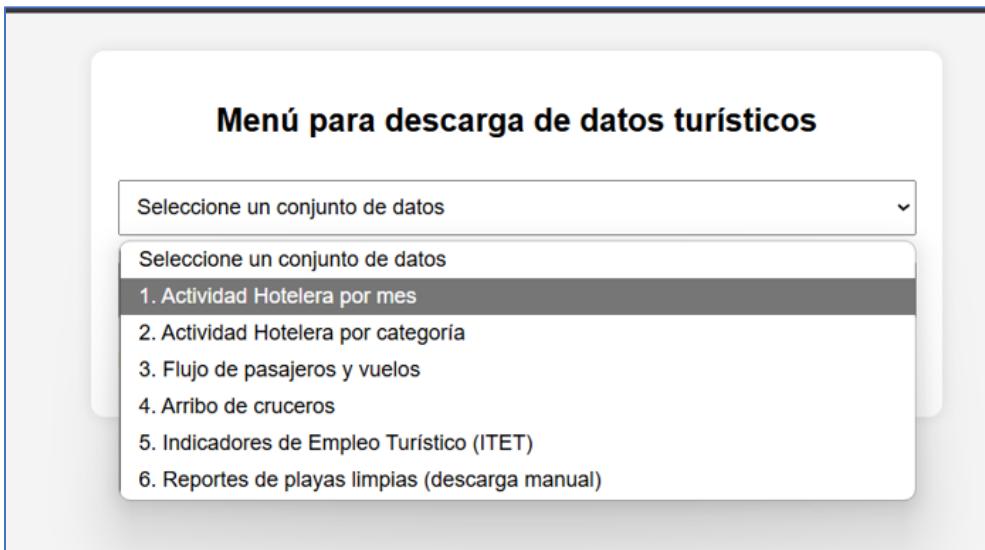
Dataset del sistema del Observatorio Turístico	Frecuencia de publicación en la página oficial	Acción recomendada
cuartos_mensual	Irregular (último dato: diciembre 2023)	Revisión mensual de fuente oficial (SECTUR), descargar en cuanto haya actualización.
llegada_turistas_mensual	Irregular	Revisión mensual de fuente oficial (SECTUR), descargar en cuanto haya actualización.
turistas_noche_mensual	Irregular	Revisión mensual de fuente oficial (SECTUR), descargar en cuanto haya actualización.
<i>Dataset</i> de reputación (Google Maps)	Trimestral (propuesta interna)	Consulta programada cada tres meses para detectar cambios significativos.
dim_tiempo	Anual (enero)	Ampliación de registros al inicio de cada año.

Sistema de descarga de datasets turísticos.

Se desarrolló un sistema en PHP que permite organizar la descarga de archivos desde fuentes oficiales. Este sistema se aloja en entorno local (XAMPP) y automatiza parte del proceso de gestión documental. De la Figura 74 a la Figura 76 se muestra el proceso para descarga de los *datasets* de las fuentes oficiales originales.

Figura 74*Interfaz principal del sistema PHP*

Nota. Captura de pantalla de la interfaz inicial de la herramienta auxiliar de descarga de datasets, mostrando el estado del sistema antes de la selección del usuario.

Figura 75*Resultado del escaneo y resumen de archivos cargados*

Nota. Captura de pantalla que muestra el menú desplegable de la herramienta, listando los conjuntos de datos preconfigurados y disponibles para la descarga.

Figura 76

Resultado de la descarga del dataset turístico.



Nota. El archivo o archivos descargados se guardarán en una carpeta con el nombre datos_turisticos seguido de la fecha de descarga, con lo que se tiene control sobre las versiones de estos archivos.

La ruta de acceso para los archivos descargados en este caso es la siguiente:

"C:\xampp\htdocs\Descarga_archivos\datos_turisticos_13-06-2025\5_1.zip"

Código en PHP utilizado para la descarga de datasets turísticos

```
<?php
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $urls = [
        "1" => "https://datatur.sectur.gob.mx/Documentos%20Publicaciones/5_1.zip",
        "2" => "https://datatur.sectur.gob.mx/Documentos%20Publicaciones/5_2.zip",
        "3" =>
        "https://datatur.sectur.gob.mx/Documentos%20compartidos/DatosAbiertos_ASA.zip",
        "4" =>
        "https://datatur.sectur.gob.mx/Documentos%20compartidos/DatosAbiertos_CRUCEROS.zip"
    ,
        "5" =>
        "https://datatur.sectur.gob.mx/SiteAssets/SitePages/ResultadosITET/ITET.zip",
        "6" => "https://apps1.semarnat.gob.mx:8443/dgeia/gob-
mx/playas/destinos/acapulco.html"
    ];
}

$opcion = $_POST['opcion'] ?? '';
$fecha = date('d-m-Y');
$carpeta = "datos_turisticos_" . $fecha;
```

```

$resultado = "";

if (!array_key_exists($opcion, $urls)) {
    $resultado = "Opcion no válida.";
} else {
    if (!file_exists($carpeta)) {
        mkdir($carpeta);
    }

    if ($opcion === "6") {
        $resultado = "Descarga manual requerida: <a href='" . $urls[$opcion] .
" target='_blank'>Abrir enlace</a>. Puedes guardar manualmente el archivo en la
carpeta '$carpeta'.";
    } else {
        $url = $urls[$opcion];
        $nombreArchivo = basename($url);
        $rutaCompleta = $carpeta . '/' . $nombreArchivo;

        if (!file_exists($rutaCompleta)) {
            $contenido = file_get_contents($url);
            file_put_contents($rutaCompleta, $contenido);
            $resultado = "Archivo '$nombreArchivo' descargado en la carpeta
'$carpeta'.";
        } else {
            $resultado = "El archivo '$nombreArchivo' ya existe en la carpeta
'$carpeta'.";
        }
    }
}
?>
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Descarga de Datos Turísticos</title>
    <style>
        body { font-family: Arial, sans-serif; background-color: #f4f4f4; padding:
20px; }
        .container { max-width: 600px; margin: auto; background: #fff; padding:
20px; border-radius: 10px; box-shadow: 0 0 10px rgba(0,0,0,0.1); }
        h2 { text-align: center; }
        select, button { width: 100%; padding: 10px; margin: 10px 0; font-size:
16px; }
        .note { font-size: 14px; color: #666; }
        .resultado { margin-top: 10px; background: #e0ffe0; padding: 10px; border-
radius: 5px; }
    </style>
</head>
<body>
    <div class="container">
        <h2>Menú para descarga de datos turísticos</h2>
        <form method="POST">
            <select name="opcion">
                <option value="">Seleccione un conjunto de datos</option>
                <option value="1">1. Actividad Hotelera por mes</option>
                <option value="2">2. Actividad Hotelera por categoría</option>
                <option value="3">3. Flujo de pasajeros y vuelos</option>
                <option value="4">4. Arribo de cruceros</option>
                <option value="5">5. Indicadores de Empleo Turístico
(ITET)</option>
                <option value="6">6. Reportes de playas limpias (descarga
manual)</option>
            </select>
        </form>
    </div>
</body>
</html>

```

```

        </select>
        <button type="submit">Descargar</button>
    </form>
    <?php if (!empty($resultado)): ?>
    <div class="resultado"><?php echo $resultado; ?></div>
    <?php endif; ?>
    <p class="note">Los archivos serán almacenados en una carpeta con la fecha
del día.</p>
    </div>
</body>
</html>
```

Integración de nuevos datasets.

Se integraron tres nuevos *datasets* orientados al análisis de la movilidad turística hacia Acapulco. De cruceros y vuelos ya se habían construido con anterioridad, solamente se incluyó la columna `fecha` de tipo `DATE` para poder hacer gráficas con temporalidad. Los *scripts* para crearlos son los siguientes:

Vista: vista_movimiento_cruceros_con_fecha

```

CREATE OR REPLACE VIEW vista_movimiento_cruceros_con_fecha AS
SELECT
    dt.fecha,
    ht.id_transporte,
    dt.anio,
    dt.mes,
    dt.nombre_mes,
    SUM(COALESCE(ht.llegadas, 0)) AS llegadas,
    SUM(COALESCE(ht.pasajeros, 0)) AS pasajeros
FROM hechos_transporte ht
JOIN dim_transporte dtp ON ht.id_transporte = dtp.id_transporte
JOIN dim_tiempo dt ON ht.id_tiempo = dt.id_tiempo
WHERE ht.id_transporte = 2
GROUP BY dt.fecha, ht.id_transporte, dt.anio, dt.mes, dt.nombre_mes;
```

Vista: vista_vuelos_mensual_con_fecha

```

CREATE OR REPLACE VIEW vista_vuelos_mensual_con_fecha AS
SELECT
    dt.fecha,
    dt.anio,
    dt.mes,
    CONCAT(LPAD(dt.mes, 2, '0'), ' - ', dt.nombre_mes) AS nombre_mes_ordenado,
    tv.tipo_vuelo,
    SUM(hv.llegadas) AS total_llegadas,
    SUM(hv.pasajeros) AS total_pasajeros,
    ROUND(AVG(CASE WHEN hv.llegadas > 0 THEN hv.pasajeros / hv.llegadas ELSE NULL
END), 0) AS promedio_pasajeros
```

```

FROM hechos_vuelos hv
JOIN dim_tiempo dt ON hv.id_tiempo = dt.id_tiempo
JOIN dim_tipo_vuelo tv ON hv.id_tipo_vuelo = tv.id_tipo_vuelo
GROUP BY dt.fecha, dt.anio, dt.mes, nombre_mes_ordenado, tv.tipo_vuelo;

```

Tabla *staging* para la creación de las tablas de hechos y dimensiones de la movilidad por carretera.

Se almacenaron en esta tabla los datos directamente desde el *dataset* original normalizado MOV_AUTOS_CARR CUOTA_1994_2022.CSV, utilizando la herramienta de importación disponible en Workbench. Este es el *script* de creación de esa tabla.

```

CREATE TABLE staging_autos_carretera (
    Anio INT,
    Fecha VARCHAR(10),
    Tipo_auto VARCHAR(50),
    Cantidad INT
);

```

Tabla de hechos para la movilidad por carretera:

```

CREATE TABLE hechos_autos_carretera (
    id INT AUTO_INCREMENT PRIMARY KEY,
    id_tiempo INT NOT NULL,
    tipo_auto VARCHAR(50),
    cantidad INT,
    FOREIGN KEY (id_tiempo) REFERENCES dim_tiempo(id_tiempo)
);

```

Carga final: se cargan los datos de la tabla de *staging* a la tabla de hechos, corrigiendo entradas no legibles de palabras con acentos. Posteriormente, se crea la vista correspondiente de estos datos.

```

INSERT INTO hechos_autos_carretera (id_tiempo, tipo_auto, cantidad)
SELECT
    dt.id_tiempo,
    CASE
        WHEN s.Tipo_auto LIKE '%tom%v%l%' THEN 'Automóviles'
        WHEN s.Tipo_auto LIKE '%tros%veh%culos' THEN 'Otros vehículos'
        ELSE s.Tipo_auto
    END,
    s.Cantidad
FROM staging_autos_carretera s
JOIN dim_tiempo dt ON STR_TO_DATE(s.Fecha, '%d/%m/%Y') = dt.fecha;

```

Vista agregada mensual:

```

CREATE OR REPLACE VIEW vista_autos_carretera_cuernavaca_acapulco AS
SELECT
    dt.fecha,
    dt.anio,
    dt.mes,
    dt.nombre_mes,
    ha.tipo_auto,
    SUM(ha.cantidad) AS total_vehiculos
FROM hechos_autos_carretera ha
JOIN dim_tiempo dt ON ha.id_tiempo = dt.id_tiempo
GROUP BY dt.fecha, dt.anio, dt.mes, dt.nombre_mes, ha.tipo_auto;

```

Resumen de procesos técnicos de preparación y respaldo de datos

- Todos los datos fueron estructurados en formato mensual para mantener coherencia temporal.
- La columna de fecha se generó combinando el año y mes como el primer día del mes, mediante la tabla dim_tiempo.
- Se eliminaron o corrigieron registros con errores de codificación, como acentos mal interpretados.
- Se excluyeron valores nulos o no registrados, conforme al criterio metodológico definido por las fuentes oficiales (DATATUR, SCT).
- Las vistas resultantes fueron configuradas para ser utilizadas directamente en Apache Superset, habilitando filtros temporales y segmentación analítica.
- Se realizó un respaldo completo del *Data Warehouse* en MySQL, así como de los *datasets* originales ya normalizados.
 - Se respaldaron también los *dashboards* y gráficas desarrolladas en Superset para garantizar su preservación e integridad.

Anexo 11. Detalle de las pruebas

Se presenta de forma íntegra *scripts*, procedimientos y salidas obtenidas durante la ejecución de las pruebas de unidad y de integración aplicadas al Observatorio Turístico digital de Acapulco. El objetivo es proporcionar evidencia técnica reproducible de los procesos de verificación descritos en las secciones 4.5, 4.6 y 5.9 del presente documento.

Se incluyen el código fuente, definiciones de casos de prueba, comandos utilizados y resultados textuales tal como fueron generados por las herramientas de validación y los *scripts* desarrollados, preservando el formato y contenido originales para garantizar su autenticidad y permitir la replicación del proceso por terceros.

Descripción y *scripts* de las Pruebas de Unidad aplicadas al Observatorio Turístico digital de Acapulco.

1. Checker

Fuente: src/Checker.php

```
<?php
declare(strict_types=1);

class Checker
{
    public static function hasErrors(string $report): bool
    {
        foreach (explode("\n", $report) as $line) {
            if (strpos(trim($line), 'ERROR:') === 0) {
                return true;
            }
        }
        return false;
    }
}
```

Prueba: tests/CheckerTest.php

```
final class CheckerTest extends TestCase
{
    public function testHasErrorsDetectsErrorLines(): void
    {
        $in = "OK\nERROR: Fallo\nOK";
        $this->assertTrue(Checker::hasErrors($in));
    }
    public function testHasErrorsReturnsFalseWhenNoErrors(): void
    {
```

```

        $in = "OK\nOK";
        $this->assertFalse(Checker::hasErrors($in));
    }
}

```

Ejecución y salida:

```
phpunit tests/CheckerTest.php
.. 2/2 (100%) OK (2 tests, 2 assertions)
```

2. DataTransformer

Fuente: src/DataTransformer.php

```

<?php
declare(strict_types=1);

class DataTransformer
{
    public static function toDataset(array $rows): array
    {
        return $rows;
    }
}

```

Prueba: tests/DataTransformerTest.php

```

final class DataTransformerTest extends TestCase
{
    public function testToDatasetReturnsSameArray(): void
    {
        $rows = [['id'=>1, 'name'=>'A'], ['id'=>2, 'name'=>'B']];
        $this->assertSame($rows, DataTransformer::toDataset($rows));
    }
    public function testToDatasetEmptyArray(): void
    {
        $this->assertCount(0, DataTransformer::toDataset([]));
    }
}

```

Ejecución y salida:

```
phpunit tests/DataTransformerTest.php
.. 2/2 (100%) OK (2 tests, 3 assertions)
```

3. EmbedBuilder

Fuente: src/EmbedBuilder.php

```

<?php
declare(strict_types=1);

```

```

class EmbedBuilder
{
    public static function iframeUrl(string $host, string $id): string
    {
        $host = rtrim($host, '/');
        return "https://{$host}/superset/dashboard/{$id}/";
    }
}

```

Prueba: tests/EmbedBuilderTest.php

```

final class EmbedBuilderTest extends TestCase
{
    public function testIframeUrlBasic(): void
    {
        $this->assertSame(
            'https://example.com/superset/dashboard/abc123/',
            EmbedBuilder::iframeUrl('example.com', 'abc123')
        );
    }
    public function testIframeUrlWithTrailingSlashOnHost(): void
    {
        $this->assertSame(
            'https://example.com/superset/dashboard/42/',
            EmbedBuilder::iframeUrl('example.com/', '42')
        );
    }
}

```

Ejecución y salida:

```

phpunit tests/EmbedBuilderTest.php
.. 2/2 (100%) OK (2 tests, 2 assertions)

```

4. Utils

Fuente: src/Utils.php

```

<?php
declare(strict_types=1);

class Utils
{
    public static function timestampedName(string $base): string
    {
        $ts = date('Ymd_Hi');
        return "{$base}_[{$ts}]";
    }
}

```

Prueba: tests/UtilsTest.php

```

final class UtilsTest extends TestCase

```

```

{
    public function testTimestampedNameFormat(): void
    {
        $res = Utils::timestampedName('miarchivo');
        $this->assertMatchesRegularExpression(
            '/^miarchivo_[0-9]{8}_[0-9]{4}$/',
            $res
        );
    }
}

```

Ejecución y salida:

```

PHPUnit tests/UtilsTest.php

. 1/1 (100%) OK (1 test, 2 assertions)

```

5. Ejecución de la Suite Completa

Con phpunit.xml configurado, basta:

```

cd /var/www/html
phpunit --verbose

```

Salida consolidada (TestDox):

```

Checker
✓ Has errors detects error lines
✓ Has errors returns false when no errors

Data Transformer
✓ To dataset returns same array
✓ To dataset empty array

Embed Builder
✓ Iframe url basic
✓ Iframe url with trailing slash on host

Utils
✓ Timestamped name format

Time: 00:00.009, Memory: 24.89 MB
OK (7 tests, 9 assertions)

```

Pruebas de Integración.

1. Script *check_dashboards.sh* para generar las pruebas de integración

```

#!/bin/bash
#
# check_dashboards.sh - Verifica que todas las páginas del Observatorio
# respondan con HTTP 200 OK y genera un informe de texto plano en /home/orangepi

### 1. Verificar usuario
echo "Usuario actual: $(whoami)"
echo

### 2. Variables de configuración
BASE_URL="http://localhost"
REPORT_DIR="/home/orangepi"
DATESTAMP=$(date +%Y%m%d_%H%M)
REPORT_FILE="$REPORT_DIR/informe_pruebas_$DATESTAMP.txt"

mkdir -p "$REPORT_DIR"

### 3. Lista de URLs a verificar
URLS=(
    "$BASE_URL/index.php"
    "$BASE_URL/1a.php"
    "$BASE_URL/2.php"
    "$BASE_URL/3.php"
    "$BASE_URL/3a.php"
    "$BASE_URL/3b.php"
    "$BASE_URL/4.php"
    "$BASE_URL/5.php"
    "$BASE_URL/5a.php"
    "$BASE_URL/6.php"
    "$BASE_URL/7.php"
    "$BASE_URL/8.php"
    "$BASE_URL/9.php"
    "$BASE_URL/10.php"
    "$BASE_URL/11.php"
    "$BASE_URL/12.php"
    "$BASE_URL/12a.php"
    "$BASE_URL/13.php"
)
)

### 4. Inicio del informe
echo "Pruebas automáticas - $DATESTAMP" > "$REPORT_FILE"
echo "===== >> "$REPORT_FILE"
errors=0

### 5. Bucle de comprobación
for url in "${URLS[@]}"; do
    status=$(curl -s -o /dev/null -w "%{http_code}" "$url")
    if [ "$status" -ne 200 ]; then
        echo "ERROR: $url → HTTP $status" >> "$REPORT_FILE"
        ((errors++))
    else
        echo "OK:      $url → HTTP 200" >> "$REPORT_FILE"
    fi
done

echo "----- >> "$REPORT_FILE"

```

```
## 6. Resultado global
if [ $errors -eq 0 ]; then
    echo "RESULTADO: 0 errores detectados." >> "$REPORT_FILE"
    exit 0
else
    echo "RESULTADO: $errors error(es) detectado(s)." >> "$REPORT_FILE"
    exit 1
fi
```

Ejecución manual del script

Lanzamiento con:

```
./check_dashboards.sh
```

Verificación en consola de:

- Usuario actual (orangeipi o root).
- Línea por línea: OK: <URL> → HTTP 200.

Generación y revisión del informe

El *script* crea un archivo /home/orangeipi/informe_pruebas_YYYYMMDD_HHMM.txt.

Se revisó con:

```
cat /home/orangeipi/informe_pruebas_*.txt
```

Confirmación de “**RESULTADO: 0 errores detectados.**”

2. Descripción del script *integración_completa.sh*

El *script* automatiza las siguientes fases:

- Definición de rutas y variables (DOCROOT, BASE_URL, WORKDIR, etc.).
- Limpieza de archivos temporales.
- Recolección de enlaces a páginas PHP y HTML desde header.php, Mapas.php y estadisticas.php.
- Filtrado y ordenación de enlaces únicos.

- Verificación de respuesta HTTP 200 para cada recurso.
- Comprobación de que cada página (excepto el propio index.php) incluya un href="index.php".
- Generación de un informe en texto plano con fecha y hora.

Procedimiento

- Ejecución del *script* mediante ./integracion_completa.sh.
- Salida completa capturada en
`/home/orangepi/reports/informe_integracion_YYYYMMDD_HHMM.txt`
- La salida del archivo del reporte muestra todas las rutas de navegación de las diferentes páginas que conforman el sitio web del Observatorio objeto de este trabajo.

Script integración_completa.sh

```
### 1. Verificar usuario
echo "Usuario actual: $(whoami)"
echo

### 2. Variables de configuración
BASE_URL="http://localhost"
REPORT_DIR="/home/orangepi"
DATESTAMP=$(date +%Y%m%d_%H%M)
REPORT_FILE="$REPORT_DIR/informe_pruebas_$DATESTAMP.txt"

mkdir -p "$REPORT_DIR"

### 3. Lista de URLs a verificar
URLS=(
    "$BASE_URL/index.php"
    "$BASE_URL/1a.php"
    "$BASE_URL/2.php"
    "$BASE_URL/3.php"
    "$BASE_URL/3a.php"
    "$BASE_URL/3b.php"
    "$BASE_URL/4.php"
    "$BASE_URL/5.php"
    "$BASE_URL/5a.php"
    "$BASE_URL/6.php"
    "$BASE_URL/7.php"
    "$BASE_URL/8.php"
    "$BASE_URL/9.php"
    "$BASE_URL/10.php"
    "$BASE_URL/11.php"
    "$BASE_URL/12.php"
    "$BASE_URL/12a.php"
    "$BASE_URL/13.php"
)
```

```

#### 4. Inicio del informe
echo "Pruebas automáticas - $DATESTAMP" > "$REPORT_FILE"
echo "======" >> "$REPORT_FILE"
errors=0

#### 5. Bucle de comprobación
for url in "${URLS[@]}"; do
    status=$(curl -s -o /dev/null -w "%{http_code}" "$url")
    if [ "$status" -ne 200 ]; then
        echo "ERROR: $url → HTTP $status" >> "$REPORT_FILE"
        ((errors++))
    else
        echo "OK:      $url → HTTP 200" >> "$REPORT_FILE"
    fi
done

echo "-----" >> "$REPORT_FILE"

#### 6. Resultado global
if [ $errors -eq 0 ]; then
    echo "RESULTADO: 0 errores detectados." >> "$REPORT_FILE"
    exit 0
else
    echo "RESULTADO: $errors error(es) detectado(s)." >> "$REPORT_FILE"
    exit 1
fi
orangepi@orangepi5plus:~$ ^C
orangepi@orangepi5plus:~$ cat integracion_completa.sh
#!/bin/bash
# integracion_completa.sh - Verificación integral de frontend y enlaces

# 1. Variables de configuración
DOCROOT="/var/www/html"                      # Ruta base de tus archivos PHP/HTML
BASE_URL="http://192.168.1.87"                 # IP o dominio del servidor
WORKDIR="/home/orangepi"                      # Directorio de scripts y resultados
TMP_LINKS="$WORKDIR/tmp_links.txt"             # Archivo temporal de enlaces
LINKS_FILE="$WORKDIR/links.txt"                # Archivo final de enlaces únicos
REPORT_DIR="$WORKDIR/reports"                  # Carpeta para informes
REPORT_FILE="$REPORT_DIR/informe_integracion_$(date +%Y%m%d_%H%M).txt"

# 2. Preparar entorno
mkdir -p "$WORKDIR" "$REPORT_DIR"
rm -f "$TMP_LINKS" "$LINKS_FILE"

echo "Informe de integración - $(date)" | tee "$REPORT_FILE"
echo "======" | tee -a "$REPORT_FILE"

# 3. Recolectar rutas base y enlaces anidados
printf "index.php\nheader.php\nfooter.php\n" >> "$TMP_LINKS"
grep -oE 'href="[^"]+\.\php"' "$DOCROOT/header.php" | cut -d '"' -f2 >> "$TMP_LINKS"
grep -oE 'href="[^"]+\.\html"' "$DOCROOT/Mapas.php" | cut -d '"' -f2 >> "$TMP_LINKS"
grep -oE 'href="[^"]+\.\php"' "$DOCROOT/estadisticas.php" | cut -d '"' -f2 >> "$TMP_LINKS"

# 4. Filtrar y ordenar enlaces únicos
echo "\nEnlaces detectados:" | tee -a "$REPORT_FILE"
sort -u "$TMP_LINKS" | tee "$LINKS_FILE" | tee -a "$REPORT_FILE"
echo "-----" | tee -a "$REPORT_FILE"

# 5. Verificar HTTP 200

```

```

echo "Verificando HTTP 200 para cada recurso:" | tee -a "$REPORT_FILE"
while IFS= read -r page; do
    code=$(curl -s -o /dev/null -w "%{http_code}" "$BASE_URL/$page")
    if [ "$code" -eq 200 ]; then
        echo "OK: $page → HTTP 200" | tee -a "$REPORT_FILE"
    else
        echo "ERROR: $page → HTTP $code" | tee -a "$REPORT_FILE"
    fi
done < "$LINKS_FILE"
echo "-----" | tee -a "$REPORT_FILE"

# 6. Verificar retorno a index.php desde páginas con header
echo "Verificando enlace de retorno a index.php:" | tee -a "$REPORT_FILE"
while IFS= read -r page; do
    # Omitir comprobación para index.php, header.php y footer.php
    if [[ "$page" == "index.php" || "$page" == "header.php" || "$page" == "footer.php" ]]; then
        continue
    fi
    content=$(curl -s "$BASE_URL/$page")
    if echo "$content" | grep -q 'href="index.php"'; then
        echo "Retorno OK: $page incluye enlace a index.php" | tee -a "$REPORT_FILE"
    else
        echo "Retorno FALLO: $page NO incluye enlace a index.php" | tee -a "$REPORT_FILE"
    fi
done < "$LINKS_FILE"
echo "-----" | tee -a "$REPORT_FILE"

# 7. Verificar navegación entre módulos (saltos sin pasar por index)
echo "Verificando enlaces a otros módulos desde cada página:" | tee -a "$REPORT_FILE"
# Extraer módulos principales del header
mapfile -t modules < <(grep -oE 'href="[^"]+\.\php"' "$DOCROOT/header.php" \
    | cut -d '"' -f2 \
    | grep -Ev '^(index|header|footer)\.\php$')

while IFS= read -r page; do
    # Solo para páginas PHP que usan header (todos excepto header/footer)
    if [[ "$page" == "header.php" || "$page" == "footer.php" ]]; then
        continue
    fi
    content=$(curl -s "$BASE_URL/$page")
    for mod in "${modules[@]}"; do
        if [[ "$mod" != "$page" ]]; then
            if echo "$content" | grep -q "href=\"$mod\""; then
                echo "OK: $page → enlace a módulo $mod" | tee -a "$REPORT_FILE"
            else
                echo "ERROR: $page → NO enlace a módulo $mod" | tee -a "$REPORT_FILE"
            fi
        fi
    done
done < "$LINKS_FILE"
echo "-----" | tee -a "$REPORT_FILE"

# 8. Verificar que header.php retorna a index.php
echo "Verificando que header.php incluye enlace a index.php:" | tee -a "$REPORT_FILE"
content_header=$(curl -s "$BASE_URL/header.php")
if echo "$content_header" | grep -q 'href="index.php"'; then
    echo "Header OK: header.php incluye enlace a index.php" | tee -a "$REPORT_FILE"
else

```

```

echo "Header FALLO: header.php NO incluye enlace a index.php" | tee -a
"$REPORT_FILE"
fi
echo "===== | tee -a "$REPORT_FILE"
echo "Informe generado en: $REPORT_FILE" | tee -a "$REPORT_FILE"

# 9. Enviar informe por correo (opcional)
# mail -s "Informe Integración Completa - $(date +%Y%m%d)" -a "$REPORT_FILE"
cjimenezmeza08@gmail.com < /dev/null

```

Resultado de la prueba de integración del frontend:

```

Informe de integración - mar 05 ago 2025 22:04:22 CST
=====
\nEnlaces detectados:
10.php
11.php
12a.php
12.php
13.php
1a.php
2.php
3a.php
3b.php
3.php
4.php
5a.php
5.php
6.php
7.php
8.php
9.php
archivos.php
Biblioteca2.php
camaras1.php
estadisticas.php
footer.php
header.php
index.php
Mapas.php
-----
Verificando HTTP 200 para cada recurso:
OK: 10.php → HTTP 200
OK: 11.php → HTTP 200
OK: 12a.php → HTTP 200
OK: 12.php → HTTP 200
OK: 13.php → HTTP 200
OK: 1a.php → HTTP 200
OK: 2.php → HTTP 200
OK: 3a.php → HTTP 200
OK: 3b.php → HTTP 200
OK: 3.php → HTTP 200
OK: 4.php → HTTP 200
OK: 5a.php → HTTP 200
OK: 5.php → HTTP 200
OK: 6.php → HTTP 200
OK: 7.php → HTTP 200
OK: 8.php → HTTP 200
OK: 9.php → HTTP 200
OK: archivos.php → HTTP 200
OK: Biblioteca2.php → HTTP 200
OK: camaras1.php → HTTP 200
OK: estadisticas.php → HTTP 200

```

```
OK: footer.php → HTTP 200
OK: header.php → HTTP 200
OK: index.php → HTTP 200
OK: Mapas.php → HTTP 200
-----
Verificando enlace de retorno a index.php:
Retorno OK: 10.php incluye enlace a index.php
Retorno OK: 11.php incluye enlace a index.php
Retorno OK: 12a.php incluye enlace a index.php
Retorno OK: 12.php incluye enlace a index.php
Retorno OK: 13.php incluye enlace a index.php
Retorno OK: 1a.php incluye enlace a index.php
Retorno OK: 2.php incluye enlace a index.php
Retorno OK: 3a.php incluye enlace a index.php
Retorno OK: 3b.php incluye enlace a index.php
Retorno OK: 3.php incluye enlace a index.php
Retorno OK: 4.php incluye enlace a index.php
Retorno OK: 5a.php incluye enlace a index.php
Retorno OK: 5.php incluye enlace a index.php
Retorno OK: 6.php incluye enlace a index.php
Retorno OK: 7.php incluye enlace a index.php
Retorno OK: 8.php incluye enlace a index.php
Retorno OK: 9.php incluye enlace a index.php
Retorno OK: archivos.php incluye enlace a index.php
Retorno OK: Biblioteca2.php incluye enlace a index.php
Retorno OK: camaras1.php incluye enlace a index.php
Retorno OK: estadisticas.php incluye enlace a index.php
Retorno OK: Mapas.php incluye enlace a index.php
-----
Verificando enlaces a otros módulos desde cada página:
OK: 10.php → enlace a módulo Mapas.php
OK: 10.php → enlace a módulo estadisticas.php
OK: 10.php → enlace a módulo camaras1.php
OK: 10.php → enlace a módulo archivos.php
OK: 10.php → enlace a módulo Biblioteca2.php
OK: 11.php → enlace a módulo Mapas.php
OK: 11.php → enlace a módulo estadisticas.php
OK: 11.php → enlace a módulo camaras1.php
OK: 11.php → enlace a módulo archivos.php
OK: 11.php → enlace a módulo Biblioteca2.php
OK: 12a.php → enlace a módulo Mapas.php
OK: 12a.php → enlace a módulo estadisticas.php
OK: 12a.php → enlace a módulo camaras1.php
OK: 12a.php → enlace a módulo archivos.php
OK: 12a.php → enlace a módulo Biblioteca2.php
OK: 12.php → enlace a módulo Mapas.php
OK: 12.php → enlace a módulo estadisticas.php
OK: 12.php → enlace a módulo camaras1.php
OK: 12.php → enlace a módulo archivos.php
OK: 12.php → enlace a módulo Biblioteca2.php
OK: 13.php → enlace a módulo Mapas.php
OK: 13.php → enlace a módulo estadisticas.php
OK: 13.php → enlace a módulo camaras1.php
OK: 13.php → enlace a módulo archivos.php
OK: 13.php → enlace a módulo Biblioteca2.php
OK: 1a.php → enlace a módulo Mapas.php
OK: 1a.php → enlace a módulo estadisticas.php
OK: 1a.php → enlace a módulo camaras1.php
OK: 1a.php → enlace a módulo archivos.php
OK: 1a.php → enlace a módulo Biblioteca2.php
OK: 2.php → enlace a módulo Mapas.php
OK: 2.php → enlace a módulo estadisticas.php
OK: 2.php → enlace a módulo camaras1.php
```

OK: 2.php → enlace a módulo archivos.php
OK: 2.php → enlace a módulo Biblioteca2.php
OK: 3a.php → enlace a módulo Mapas.php
OK: 3a.php → enlace a módulo estadisticas.php
OK: 3a.php → enlace a módulo camaras1.php
OK: 3a.php → enlace a módulo archivos.php
OK: 3a.php → enlace a módulo Biblioteca2.php
OK: 3b.php → enlace a módulo Mapas.php
OK: 3b.php → enlace a módulo estadisticas.php
OK: 3b.php → enlace a módulo camaras1.php
OK: 3b.php → enlace a módulo archivos.php
OK: 3b.php → enlace a módulo Biblioteca2.php
OK: 3.php → enlace a módulo Mapas.php
OK: 3.php → enlace a módulo estadisticas.php
OK: 3.php → enlace a módulo camaras1.php
OK: 3.php → enlace a módulo archivos.php
OK: 3.php → enlace a módulo Biblioteca2.php
OK: 4.php → enlace a módulo Mapas.php
OK: 4.php → enlace a módulo estadisticas.php
OK: 4.php → enlace a módulo camaras1.php
OK: 4.php → enlace a módulo archivos.php
OK: 4.php → enlace a módulo Biblioteca2.php
OK: 5a.php → enlace a módulo Mapas.php
OK: 5a.php → enlace a módulo estadisticas.php
OK: 5a.php → enlace a módulo camaras1.php
OK: 5a.php → enlace a módulo archivos.php
OK: 5a.php → enlace a módulo Biblioteca2.php
OK: 5.php → enlace a módulo Mapas.php
OK: 5.php → enlace a módulo estadisticas.php
OK: 5.php → enlace a módulo camaras1.php
OK: 5.php → enlace a módulo archivos.php
OK: 5.php → enlace a módulo Biblioteca2.php
OK: 6.php → enlace a módulo Mapas.php
OK: 6.php → enlace a módulo estadisticas.php
OK: 6.php → enlace a módulo camaras1.php
OK: 6.php → enlace a módulo archivos.php
OK: 6.php → enlace a módulo Biblioteca2.php
OK: 7.php → enlace a módulo Mapas.php
OK: 7.php → enlace a módulo estadisticas.php
OK: 7.php → enlace a módulo camaras1.php
OK: 7.php → enlace a módulo archivos.php
OK: 7.php → enlace a módulo Biblioteca2.php
OK: 8.php → enlace a módulo Mapas.php
OK: 8.php → enlace a módulo estadisticas.php
OK: 8.php → enlace a módulo camaras1.php
OK: 8.php → enlace a módulo archivos.php
OK: 8.php → enlace a módulo Biblioteca2.php
OK: 9.php → enlace a módulo Mapas.php
OK: 9.php → enlace a módulo estadisticas.php
OK: 9.php → enlace a módulo camaras1.php
OK: 9.php → enlace a módulo archivos.php
OK: 9.php → enlace a módulo Biblioteca2.php
OK: archivos.php → enlace a módulo Mapas.php
OK: archivos.php → enlace a módulo estadisticas.php
OK: archivos.php → enlace a módulo camaras1.php
OK: archivos.php → enlace a módulo Biblioteca2.php
OK: Biblioteca2.php → enlace a módulo Mapas.php
OK: Biblioteca2.php → enlace a módulo estadisticas.php
OK: Biblioteca2.php → enlace a módulo camaras1.php
OK: Biblioteca2.php → enlace a módulo archivos.php
OK: camaras1.php → enlace a módulo Mapas.php
OK: camaras1.php → enlace a módulo estadisticas.php
OK: camaras1.php → enlace a módulo archivos.php

```
OK: camaras1.php → enlace a módulo Biblioteca2.php
OK: estadisticas.php → enlace a módulo Mapas.php
OK: estadisticas.php → enlace a módulo camaras1.php
OK: estadisticas.php → enlace a módulo archivos.php
OK: estadisticas.php → enlace a módulo Biblioteca2.php
OK: index.php → enlace a módulo Mapas.php
OK: index.php → enlace a módulo estadisticas.php
OK: index.php → enlace a módulo camaras1.php
OK: index.php → enlace a módulo archivos.php
OK: index.php → enlace a módulo Biblioteca2.php
OK: Mapas.php → enlace a módulo estadisticas.php
OK: Mapas.php → enlace a módulo camaras1.php
OK: Mapas.php → enlace a módulo archivos.php
OK: Mapas.php → enlace a módulo Biblioteca2.php
=====
Verificando que header.php incluye enlace a index.php:
Header OK: header.php incluye enlace a index.php
=====
Informe generado en:
/home/orangeipi/reports/informe_integracion_20250805_2204.txt
```

Anexo 12. Instrumento de Pruebas de Aceptación de Usuario (UAT)

Evaluación de Aceptación de Usuario (UAT): Observatorio Turístico Digital de Acapulco

¡Gracias por su participación! Este formulario es la fase final para validar el prototipo del Observatorio Turístico Digital de Acapulco. Su retroalimentación es fundamental para asegurar que el sistema sea una herramienta útil para la comunidad académica.

Instrucciones generales:

- Utilice el siguiente enlace del sitio web para realizar la prueba: [Observatorio Turístico de Acapulco](#)
- Realice las pruebas desde un dispositivo de escritorio o portátil con una conexión a internet estable y un navegador web actualizado (se recomienda Edge, Google Chrome o Mozilla Firefox).
- Para cada prueba, por favor, siga los "Pasos de Ejecución" indicados.
- Compare lo que observe en el sitio web con el "Resultado esperado".
- Seleccione "Pasa" si el sistema funciona como se describe, o "Falla" si encuentra cualquier problema o desviación.
- Si una prueba falla, es muy importante que describa el problema en la sección de "Comentarios" y, si es posible, adjunte una captura de pantalla como evidencia.

El formulario está dividido en secciones que corresponden a los distintos módulos del sitio web.
¡Comencemos!

* Indica que la pregunta es obligatoria

1. **Nombre del evaluador:** *

2. **Rol** (Marque la opción que describa su ocupación o actividad principal) *

Marca solo un óvalo.



Investigador/Profesor de la Facultad de Turismo (UAGro)



Estudiante de maestría/doctorado de la Facultad de Turismo (UAGro)



Catedrático experto en proyectos de software

Módulo de Analítica (Indicadores y Dashboards)

ID del Caso: UAT-IND-001

Se evaluará la funcionalidad central de visualización de datos, asegurando que los dashboards interactivos presenten la información correctamente y respondan a la interacción del usuario.

Pasos de la ejecución:

1. Navegar a la sección "Indicadores".
2. Seleccionar la subsección "Indicadores de empleo, alojamiento y gasto turístico"

3. Seleccionar el indicador "Ocupación hotelera en destinos de playa".
4. Observar el dashboard que se carga por defecto.

Resultado Esperado: Se carga un dashboard con al menos una tabla y gráficos mostrando indicadores de "Cuartos Disponibles" y/o "Cuartos Ocupados". La interfaz es clara y los datos corresponden a un año seleccionado del periodo histórico 1992-2023.

3. Veredicto (UAT-IND_001) *

Marca solo un óvalo.

- Pasa
 Falla

4. Resultado Real y Comentarios (UAT-IND-001) *

5. Adjuntar evidencia (si aplica)

Archivos enviados:

ID del Caso: UAT-IND-002

Validar la funcionalidad del filtro de tiempo (año) en un dashboard de series temporales. * El dashboard de Ocupación hotelera en destinos de playa (UAT-IND-001) está visible.

Pasos:

1. Localizar el control de filtro de fecha o año en el dashboard.
2. Seleccionar un año específico del cual se sabe que existen datos (ej. 2019).
3. Aplicar el filtro.

Resultado esperado: Todos los gráficos y tablas en el dashboard se actualizan dinámicamente para mostrar únicamente los datos correspondientes al año 2019. El eje de tiempo del gráfico debe reflejar el rango seleccionado. La respuesta del sistema debe ser rápida.

6. Veredicto (UAT-IND_002): *

Marca solo un óvalo.

- Pasa
 Falla

7. Resultado Real y Comentarios (UAT-IND-002) *

8. Adjuntar evidencia (si aplica).

Archivos enviados:

ID del Caso: UAT-IND-003

Validar la correcta visualización del dashboard de reputación digital de playas.

- El usuario ha navegado a la sección de **Indicadores**, subsección **Entorno y Conectividad**, seleccionando la opción **Valoración de playas por reseñas**.

Pasos:

1. Observar el dashboard cargado.
2. Identificar el gráfico de barras que compara las playas.

Se muestra un dashboard con un gráfico que presenta una puntuación para cada playa (ej., "Puntuación" o "Percepción del turista"). Al pasar el cursor sobre una barra, se debe mostrar información adicional como el número de reseñas.

9. **Veredicto (UAT-IND_003): ***

Marca solo un óvalo.

- Pasa
 Falla

10. Resultado Real y Comentarios (UAT-IND-003) *

11. Adjuntar evidencia (si aplica).

Archivos enviados:

Módulo de Inteligencia Geoespacial (Mapas)

Se validarán las funcionalidades de visualización de datos geográficos y en tiempo real.

ID del Caso: UAT-MAP-001

El usuario se encuentra en la página de visualización de la Valoración de playas, o bien, se encuentra en la página principal. Pasos:

1. Navegar al menú "Mapas".
2. La página "Vuelos" se carga por defecto.
3. Observar el mapa cargado.
4. Intentar hacer zoom y arrastrar el mapa.

Resultado esperado: Se carga una interfaz de mapa que ocupa el área principal de la pantalla. El mapa debe mostrar una capa de datos relacionada con el tráfico aéreo (ej. iconos de aviones). Las funciones de zoom (rueda del ratón o botones +/-) y arrastrar con el ratón deben funcionar correctamente. Al hacer click en alguno de los iconos de aviones mostrará información del vuelo.

12. **Veredicto (UAT-MAP-001) ***

Marca solo un óvalo.

- Pasa
 Falla

13. Resultado Real y Comentarios (UAT-MAP-001) *

14. Adjuntar evidencia (si aplica).

Archivos enviados:

ID del Caso: UAT-MAP-002

- Se verificará la carga y funcionalidad del mapa de Vista Satelital.

El usuario se encuentra dentro de la sección **Mapas**.

1. En el menú Mapas, seleccionar el ícono "Vista Satelital".

Resultado esperado. Se carga un mapa que muestra una capa de datos meteorológicos (ej., manchas que representan precipitación). La capa de datos debe ser legible y superponerse correctamente sobre el mapa base.

15. **Veredicto (UAT-MAP-002): ***

Marca solo un óvalo.

- Pasa
 Falla

16. Resultado Real y Comentarios (UAT-MAP-002) *

17. Adjuntar evidencia (si aplica).

Archivos enviados:

ID del Caso: UAT-MAP-003

- Se verificará la carga y funcionalidad del mapa de Embarcaciones.

El usuario se encuentra dentro de la sección **Mapas**.

1. En el menú Mapas, seleccionar el ícono "Embarcaciones".

Resultado esperado. Se carga un mapa interactivo que muestra puntos en aguas nacionales correspondientes a embarcaciones. Al apuntar a cualquiera de ellos debe desplegar el nombre de la embarcación. Al hacer click en el punto o figura, debe mostrar información como nombre, nacionalidad, tipo de embarcación, foto (si está disponible) entre otros datos.

18. Veredicto (UAT-MAP-003): *

Marca solo un óvalo.



Pasa



Falla

19. Resultado Real y Comentarios (UAT-MAP-003) *

20. Adjuntar evidencia (si aplica).

Archivos enviados:

ID del Caso: UAT-MAP-004

- Se verificará la carga y funcionalidad de los mapas de Viento, Oleaje y Temperatura.

El usuario se encuentra dentro de la sección **Mapas**.

1. En el menú Mapas, seleccionar alguno de los iconos de "Viento", "Oleaje" o "Temperatura".

Resultado esperado. Se carga un mapa interactivo que muestra secciones de colores correspondientes a vientos, temperatura u oleaje, según el mapa que se haya seleccionado. Las funciones de zoom (rueda del ratón o botones +/-) y arrastrar con el ratón deben funcionar correctamente. Al hacer click en un sitio en el mapa (hacer click en Acapulco), se abre un recuadro de la información meteorológica correspondiente.

21. Veredicto (UAT-MAP-004): *

Marca solo un óvalo.

- Pasa
 Falla

22. Resultado Real y Comentarios (UAT-MAP-004) *

23. Adjuntar evidencia (si aplica).

Archivos enviados:

Módulo de Monitoreo en Tiempo Real: "Acapulco On-live"

Esta prueba valida la integración de transmisiones en vivo.

ID del

Caso: UAT-LIV-001 Objetivo. Verificar que el módulo de cámaras en vivo se carga e intenta iniciar la transmisión.

* El usuario se encuentra en la página Mapas o en la página principal.

Navegar a la sección "Acapulco On-live". **Resultado esperado:** La página se carga mostrando uno o varios reproductores de video incrustados. Los reproductores deben mostrar un estado de carga, que al hacer click sobre alguno inicia la transmisión de video desde las cámaras web configuradas.

24. Veredicto (UAT-LIV-001) *

Marca solo un óvalo.

- Pasa
 Falla

25. Resultado Real y Comentarios (UAT-LIV-001) *

26. Adjuntar evidencia (si aplica).

Archivos enviados:

Módulo de Gestión del Conocimiento (Centro Documental)

Se validarán las funcionalidades del repositorio de documentos del sistema.

ID del Caso: UAT-DOC-001

Existen documentos previamente cargados en el sistema. El evaluador puede buscar un documento por medio de una barra de búsqueda. Pasos:

1. Navegar a la sección "Centro Documental".
2. Utilizar la barra de búsqueda para introducir una palabra clave del título de un documento (Ej. "covid" o "Acapulco").
3. La lista de documentos se ajusta mostrando los que coinciden con la palabra clave.

Resultado esperado: La lista de documentos se filtra para mostrar únicamente aquellos que coinciden con el criterio de búsqueda.

27. Veredicto (UAT-DOC-001) *

Marca solo un óvalo.



Pasa



Falla

28. Resultado Real y Comentarios (UAT-DOC-001) *

29. Adjuntar evidencia (si aplica).

Archivos enviados:

ID del Caso: UAT-DOC-002

Verificar la visualización y descarga de un documento (tesis o investigación).

- El usuario ha localizado un documento mediante búsqueda o navegación (UAT-DOC-001).

Pasos:

1. Hacer click en el botón "Descargar" asociado al documento.
2. Si se abre en el navegador, verificar que el contenido es legible.
3. Utilizar la función de descarga del navegador o de la interfaz para guardar el archivo localmente.

Resultado esperado: El documento en formato PDF se abre correctamente en una nueva pestaña del navegador o se inicia su descarga. El archivo descargado debe poder abrirse con un lector de PDF estándar y su contenido debe ser íntegro.

30. Veredicto (UAT-DOC-002): *

Marca solo un óvalo.

Pasa

Falla

31. Resultado Real y Comentarios (UAT-DOC-001) *

32. Adjuntar evidencia (si aplica).

Archivos enviados:

Módulo de acceso al conocimiento (Bibliotecas)

Se evaluará la funcionalidad del portal de acceso a bibliotecas especializadas y recursos externos, que busca centralizar fuentes de información relevantes para la investigación turística.

ID del Caso: UAT-BIB-001

Verificar la correcta carga y funcionalidad de la biblioteca incrustada por defecto y la navegación entre las distintas opciones.

- El usuario se encuentra en la página principal del observatorio.

Pasos:

1. Navegar al menú "Bibliotecas".
2. Observar el contenido. Por defecto, debe cargarse la biblioteca AEST.
3. Interactuar con el sitio incrustado (ej. usar su barra de búsqueda o hacer click en un enlace interno).
4. Seleccionar otra biblioteca de la lista desplegable (ej. Dialnet) y verificar que el contenido incrustado se actualice.

Resultado esperado: La página de AEST se carga correctamente dentro del marco del sitio. La interacción con el contenido incrustado es fluida. Al seleccionar otra opción, el marco se actualiza y muestra la nueva biblioteca seleccionada.

33. Veredicto (UAT-BIB-001) *

Marca solo un óvalo.

- Pasa
 Falla

34. Resultado real y comentarios (UAT-BIB-001) *

35. Adjuntar evidencia (si aplica)

Archivos enviados:

ID del Caso: UAT-BIB-002

Validar que los enlaces a sitios externos funcionen correctamente.

- El usuario se encuentra en la sección "Bibliotecas".

Pasos:

1. Localizar la sección de enlaces a sitios externos, al final de la página (ej. Datatur, El Periplo Sustentable).
2. Hacer click en uno de los enlaces.

Resultado esperado: Se abre una nueva pestaña en el navegador cargando correctamente la página web externa correspondiente. El enlace no debe estar roto (no debe mostrar un error 404 o similar).

36. Veredicto (UAT-BIB-002) *

Marca solo un óvalo.

- Pasa
 Falla

37. Resultado real y comentarios (UAT-BIB-002) *

38. Adjuntar evidencia (si aplica)

Archivos enviados:

Requisitos No Funcionales (Calidad y Experiencia de Usuario)

Finalmente, se evaluarán atributos de calidad del sistema como la usabilidad, compatibilidad entre navegadores y rendimiento general.

Prueba UAT-NFR-001 (Usabilidad y Navegación) ID del Caso: UAT-NFR-001

Objetivo: Evaluar la intuición y consistencia del flujo de navegación principal. **Pasos de Ejecución:**

1. Desde la página de inicio, navegue al indicador "Llegada de turistas a destinos de playa".
2. Desde allí, utilice el menú principal para navegar al mapa de "Vuelos".
3. Finalmente, regrese a la página de inicio usando el nombre del sitio.

Resultado Esperado: La navegación es fluida e intuitiva. El menú principal es claro y siempre visible. El evaluador no se siente perdido en ningún momento.

39. Veredicto (UAT-NFR-001) *

Marca solo un óvalo.



Pasa



Falla

40. Comentarios sobre la navegación (UAT-NFR-001)

(Describir cualquier punto de confusión en la navegación.)

Prueba de Compatibilidad en Navegadores (opcional)

El objetivo de esta prueba es verificar que el sitio web opera correctamente en los navegadores más comunes.

Nota importante sobre la vista en móviles: Es normal y esperado que, en pantallas pequeñas, como las de los teléfonos, los dashboards requieran que se desplace la pantalla horizontalmente para ver toda la información. Esto **no se considera una falla** y permite que las gráficas se vean completas.

Por favor, realice esta prueba **únicamente en los navegadores que tenga instalados y disponibles**. Si solo puede probar en un navegador de escritorio, es suficiente y muy valioso. Si además tiene la oportunidad de probarlo en su móvil, la retroalimentación será aún más completa.

Para cada funcionalidad (fila), marque la casilla correspondiente a cada navegador (columna) en el que la prueba haya sido exitosa (Pasa). Si una prueba falla, deje la casilla sin marcar y anótelos en la pregunta de comentarios a continuación.

41. Matriz de pruebas de compatibilidad

Selecciona todas las opciones que correspondan.

	Google Chrome (escritorio)	Mozilla Firefox (escritorio)	Edge (escritorio)	Navegador móvil (Safari/Chrome)
Visualizar dashboards	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Filtro de dashboard (botón del	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Visualizar mapa de Vuelos	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Descargar documento PDF	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

42. Comentarios sobre fallas de compatibilidad

Rendimiento del sistema

ID del Caso: **UAT-NFR-002**

Objetivo: Evaluar el tiempo de carga inicial del dashboard principal de indicadores. **Pasos de Ejecución:**

1. Limpie la memoria caché de su navegador.
2. Navegue a la sección "Indicadores" y mida el tiempo que tarda la página en cargarse por completo y ser interactiva.
3. Seleccione el menú "Entorno y conectividad turística". Enseguida, seleccione "Flujo de cruceros y pasajeros en Acapulco".

Resultado Esperado: La página y todos sus componentes visuales (gráficos, tablas, filtros) deben cargarse y estar listos para la interacción en un tiempo razonable (ej., menos de 8 segundos). No debe haber un retraso perceptible entre la carga de la página y la aparición de los datos en los gráficos.

43. Veredicto (UAT-NFR-002) *

Marca solo un óvalo. Pasa (el rendimiento fue aceptable) Falla (el rendimiento no fue aceptable)

44. ¿Cómo percibió el tiempo de carga del dashboard? *

Marca solo un óvalo. Muy rápido (carga casi instantánea) Rápido (carga en pocos segundos, dentro de lo esperado) Lento (se notó una demora considerable) Muy lento (la carga fue frustrante o excesiva)

45. Comentarios sobre el Rendimiento (UAT-NFR-002)

*Anote el tiempo de carga aproximado y describa cualquier lentitud observada.***Evaluación Final y Aprobación**

Esta sección final está destinada a recoger sus impresiones cualitativas sobre el sistema en su conjunto y su veredicto.

46. ¿Cuáles considera que son las mayores fortalezas del sistema desde su * perspectiva?

47. ¿Cuáles son las áreas más significativas para la mejora en términos de * funcionalidad o usabilidad?

48. Veredicto Final del Proyecto. *

Con base en los resultados de las pruebas, emita su veredicto final sobre la aceptación del sistema.

Marca solo un óvalo. Aprobado: El sistema cumple con los requisitos esenciales y es aceptado. Aprobado con condiciones: El sistema es aceptado, condicionado a la resolución de los hallazgos documentados. Rechazado: El sistema presenta fallas críticas que impiden su aceptación.

¡Muchas gracias por su participación!

Anexo 13. Informe de resultados: Pruebas de Aceptación de Usuario (UAT)

INFORME DE RESULTADOS: PRUEBAS DE ACEPTACIÓN DE USUARIO (UAT)

Maestrante: Cecilia Jiménez Meza **Matrícula:** 23500742 **CVU:** 1294829

Directora de tesis: Dra. Petra Baldivia Noyola

Maestría en Tecnologías de la Información

Universidad Autónoma de Guerrero

Periodo: 22 de septiembre – 02 de octubre de 2025

1. Resumen general y Metodología

Objetivo: Validar formalmente el Observatorio Turístico Digital de Acapulco mediante la evaluación estructurada por parte de usuarios finales y expertos.

Periodo de pruebas: Se llevaron a cabo del 22 de septiembre al 2 de octubre de 2025, posterior a la fase de retroalimentación preliminar.

Participantes: Un total de 18 evaluadores participaron en el proceso. El grupo participante estuvo compuesto por:

- 2 Catedráticos expertos en desarrollo de sistemas.
- 3 Académicos e investigadores de la Facultad de Turismo.
- 13 estudiantes del Doctorado en Gestión Sustentable del Turismo.

Instrumento: Se utilizó un cuestionario estandarizado con 17 casos de prueba para evaluar todos los módulos del sistema. A la mayoría de los participantes se les proporcionó una demostración guiada (“prueba asistida”) antes de que realizaran la evaluación de forma independiente.

2. Resultados cuantitativos

Veredictos de pruebas individuales: No se registró ningún veredicto de “Falla” en la totalidad de los casos de prueba ejecutados por los 18 evaluadores.

Veredicto del proyecto: De los 18 evaluadores, el veredicto sobre la aceptación del sistema fue el siguiente:

- Aprobado: 17 evaluadores.
- Aprobado con condiciones: 1 evaluador.
- Rechazado: 0 evaluadores.

3. Análisis de observaciones relevantes durante la prueba

Observación 1: Monitoreo en Tiempo Real (UAT-LIV-001)

- **Hecho:** Un evaluador reportó que una de las cámaras web no se encontraba disponible durante su prueba.
- **Contexto:** Se verificó que el módulo operó correctamente. La interrupción se debió a un evento transitorio en la fuente de la transmisión externa, no a un defecto del software.

Observación 2: Rendimiento del sistema (UAT-NFR-002)

- **Hecho:** Un evaluador reportó un tiempo de carga de 30 segundos, mientras que los 16 restantes lo percibieron como "Rápido" o "Muy rápido".
- **Contexto:** El tiempo de carga extendido se atribuyó a una conexión de red local del evaluador, caracterizada como lenta y débil.

Observación 3: Funcionalidad de Mapas (UAT-MAP-001)

- **Hecho:** Un evaluador comentó sobre la necesidad de "precisar el origen y destino de los vuelos".
- **Contexto:** La funcionalidad para ver dichos detalles está implementada y se activa al hacer clic en el ícono del avión. El comentario se atribuye al proceso de familiarización del usuario con la interfaz en su primer uso.

4. Sugerencias post-prueba para futuras actualizaciones

Adicionalmente a los resultados del cuestionario, se recopilaron las siguientes sugerencias cualitativas en conversaciones con los evaluadores una vez finalizada la prueba. Estas no se consideran fallas, sino oportunidades de refinamiento para futuras versiones del sistema:

- **Refinamiento de Contenido:** Se identificó un área de oportunidad para mejorar la redacción en uno de los dashboards de indicadores, específicamente corrigiendo el uso de la frase "en base a".
- **Ajuste de Interfaz de Usuario:** Se recibió retroalimentación sobre un elemento decorativo (la franja con burbujas que muestra el título de cada sección). Un académico sugirió que podría eliminarse para maximizar el espacio útil, mientras que otro evaluador propuso, como alternativa, reducir su ancho.

5. Conclusión

El proceso de Pruebas de Aceptación de Usuario (UAT), ejecutado por un panel de 18 evaluadores, validó formalmente el prototipo del Observatorio Turístico Digital de Acapulco. El sistema aprobó la totalidad de los casos de prueba definidos. Las observaciones cualitativas registradas fueron contextualizadas y no corresponden a defectos del sistema, sino a oportunidades de mejora. Con base en esta evidencia, el prototipo se considera funcionalmente completo y aceptado por sus usuarios objetivo.

Anexo 14. Transcripción de respuestas cualitativas de UAT (Anonimizado)

Informe - Transcripción de respuestas cualitativas (anonimizado) – Cuestionario de Pruebas de Aceptación

Introducción

La siguiente tabla presenta la transcripción completa de la retroalimentación cualitativa recopilada durante las Pruebas de Aceptación de Usuario (UAT). Para proteger la privacidad de los participantes, se ha utilizado un sistema de codificación.

Nota metodológica: Se incluyen todas las respuestas donde el campo de comentario no fue dejado en blanco por el evaluador.

Sistema de Codificación:

- **EXP-S:** Catedrático experto en desarrollo de sistemas.
- **INV-T:** Académico e investigador de la Facultad de Turismo.
- **EST-P:** Estudiante del Doctorado en Gestión Sustentable del Turismo.

1. Módulo de Analítica (Indicadores y Dashboards)

ID del Caso	Código del Evaluador	Comentarios
UAT-IND-001	EXP-S-01	El dashboard funciona adecuadamente. presenta una tabla y gráficos mostrando los indicadores de "Cuartos Disponibles" y/o "Cuartos Ocupados". La interfaz es clara y los datos corresponden a un año seleccionado del periodo histórico 1992-2023.
	INV-T-01	Carga un tablero de gráficos que corresponde con la opción que seleccioné.
	INV-T-02	Me parece apto para el análisis e interpretación de datos estadísticos sobre la actividad turística de todo el planeta.
	EST-P-01	Funcional
	EST-P-02	Carga con facilidad y muestra los datos
	EST-P-03	Funcionamiento correcto del sitio
	EST-P-04	La información es clara y precisa
	EST-P-05	fácil de acceder
	EST-P-06	El Observatorio Turístico es una buena herramienta para conocer y actualizar los datos de diferentes destinos.
	EST-P-07	Me parecen buenas las gráficas buena información.
	EST-P-08	Se manejan resultados precisos y confiables
	EST-P-09	La información es clara.
	EST-P-11	Tabla y gráfico de la información solicitada
	EST-P-12	Funciona correctamente.
	EXP-S-02	Los resultados salieron igual a lo que se esperaba

ID del Caso	Código del Evaluador	Comentarios
	EST-P-13	Buena respuesta.
UAT-IND-002	EXP-S-01	Todos lo referente a los gráficos y tablas en el dashboard se actualizan dinámicamente y muestran únicamente los datos correspondientes al año 2019
	INV-T-01	Se muestran los datos según el año seleccionado.
	INV-T-02	Los resultados esperados son factibles.
	EST-P-01	Funcional.
	EST-P-02	Se hizo la actualización correspondiente.
	EST-P-04	Información valiosa.
	EST-P-07	Si me muestra la información del año. Sin problema.
UAT-IND-003	EXP-S-01	Se muestra un dashboard con un gráfico que presenta una puntuación para cada playa, y al pasar el cursor sobre una barra, se muestra información adicional como el número de reseñas.
	INV-T-01	Interesante el resultado al ver la calificación de las playas.
	INV-T-02	Se observa en el rango que hay una valoración alta hecha por Google, sin embargo la percepción del turista muestra un nivel más bajo, de acuerdo con la consulta llevada a cabo.
	EST-P-02	El gráfico es claro y muestra las puntuaciones para cada playa.

2. Módulo de Inteligencia Geoespacial (Mapas)

ID del Caso	Código del Evaluador	Comentarios
UAT-MAP-001	EXP-S-01	Se carga un mapa que muestra una capa de datos meteorológicos. La capa es legible y superponerse correctamente sobre el mapa base.
	INV-T-01	Se muestra correctamente
	INV-T-02	Muy ilustrativo el aparato de los mapas
	EST-P-11	La interfaz funciona correctamente, falta precisar el origen y destino de los vuelos
UAT-MAP-002	INV-T-01	También se muestra de forma correcta.
	EST-P-03	Funciona correctamente en tiempo real
	EST-P-11	Vista satelital correcta y en movimiento
UAT-MAP-003	INV-T-01	Todos los mapas cargaron de forma correcta.
	EST-P-11	Respuesta rápida de puntos de ubicación y embarcaciones

ID del Caso	Código del Evaluador	Comentarios
UAT-MAP-004	INV-T-02	Versión muy ilustrativa.
	EST-P-11	Excelente calidad de imagen satelital en movimiento para cualquiera de las 3 opciones

3. Módulo de Monitoreo en Tiempo Real: "Acapulco On-live"

ID del Caso	Código del Evaluador	Comentarios
UAT-LIV-001	EST-P-07	3/4 cámaras funcionan muy bien, la 4º cámara no está disponible
	INV-T-02	Una estrategia muy viable para observar Acapulco, con relación a la seguridad y otros aspectos.
	EST-P-08	Nos arrojó información precisa

4. Módulos de Conocimiento (Centro Documental y Bibliotecas)

ID del Caso	Código del Evaluador	Comentarios
UAT-DOC-001	INV-T-01	Escribió una palabra y seleccionó correctamente los archivos por el filtro de esa palabra.
UAT-DOC-002	INV-T-01	Se muestra el documento descargado.
UAT-BIB-001	INV-T-01	Las páginas cambian según se seleccionen.
UAT-BIB-002	INV-T-01	Se abre el enlace seleccionado.

5. Comentarios Generales sobre la Experiencia de Usuario

Fortalezas del Sistema:

- **EXP-S-01:** Rápida y presenta la información de manera muy precisa.
- **INV-T-01:** Presenta los datos de manera organizada, es rápida.
- **INV-T-02:** Todos los aspectos llenaron mis expectativas, estar en la globalización del turismo...
- **EST-P-01:** Amigable

- **EST-P-02:** La facilidad con la que se puede navegar en el sitio
- **EST-P-03:** diseño y funcionamiento
- **EST-P-04:** La disponibilidad de la información y su valía para el usuario
- **EST-P-05:** la información depositada
- **EST-P-11:** La recopilación de datos de fuentes fidedignas y la agilidad con la que opera la interfaz
- **EXP-S-02:** Es un sistema sencillo y tiene la información pertinente.

Áreas de Mejora Sugeridas:

- **INV-T-01:** Todo está dentro de lo esperado. En el futuro se espera que se integren nuevos datos.
- **EST-P-01:** Las estadísticas
- **EST-P-02:** Creo que está en mejora continua el sitio
- **EST-P-03:** enriquecimiento del contenido con trabajos de los estudiantes de doctorado
- **EST-P-04:** Incrementar opciones de acceso a bibliotecas digitales
- **EST-P-05:** actualización de los datos
- **EST-P-11:** Los gráficos y la información satelital
- **EXP-S-02:** Ninguna

Anexo 15. Respuestas detalladas de evaluadores seleccionados

Se presentan las respuestas completas (veredicto y comentarios) de un evaluador representativo de cada perfil (experto en software, investigador de Turismo, estudiante de doctorado) a todas las preguntas del cuestionario UAT, como evidencia detallada solicitada, en la Tabla 14. El cuestionario completo se encuentra en el Anexo 12 y la compilación de todos los comentarios cualitativos en el Anexo 14. Por último, se agrega evidencia fotográfica de cuando el instrumento de pruebas de aceptación fue aplicado en Turismo.

Tabla 14

Respuestas de evaluadores seleccionados

Pregunta	Catedrático experto en proyectos de software	Investigador/Profesor de la Facultad de Turismo (UAGro)	Estudiante de maestría/doctorado de la Facultad de Turismo (UAGro)
Veredicto (UAT-IND_001) y comentario	Pasa El dashboard funciona adecuadamente, presenta una tabla y gráficos mostrando los indicadores de "Cuartos Disponibles" y/o "Cuartos Ocupados". La interfaz es clara y los datos corresponden a un año seleccionado del periodo histórico 1992-2023.	Pasa Me parece apto para el análisis e interpretación de datos estadísticos sobre la actividad turística de todo el planeta.	Pasa El Observatorio Turístico es una buena herramienta para conocer y actualizar los datos de diferentes destinos,
Veredicto (UAT-IND_002) y comentario	Pasa Todos lo referente a los gráficos y tablas en el dashboard se actualizan dinámicamente y muestran únicamente los datos correspondientes al año 2019. El eje de tiempo del gráfico refleja el rango seleccionado, de manera rápida.	Pasa Los resultados esperados son factibles	Pasa Se encuentran datos muy rápidamente
Veredicto (UAT-IND_003) y comentario:	Pasa Se muestra un dashboard con un gráfico que presenta una puntuación para cada playa, y al pasar el cursor sobre una barra, se muestra información adicional como el número de reseñas.	Pasa Se observa en el rango que hay una valoración alta hecha por Google, sin embargo la percepción del turista muestra un nivel más bajo, de acuerdo a la consulta llevada a cabo.	Pasa Importante conocer la valoración que se hace por los turistas.

Pregunta	Catedrático experto en proyectos de software	Investigador/Profesor de la Facultad de Turismo (UAGro)	Estudiante de maestría/doctrado de la Facultad de Turismo (UAGro)
Veredicto (UAT-MAP-001) y comentario:	<p>Pasa</p> <p>Se carga una interfaz de mapa con un despliegue adecuado. El mapa muestra una capa de datos relacionada con el tráfico aéreo. Las funciones de zoom (rueda del ratón o botones +/-) y arrastrar con el ratón funcionan correctamente, y al hacer click en alguno de los iconos de aviones se muestra información del vuelo.</p>	<p>Pasa</p> <p>Muy ilustrativo el apartado de los mapas</p>	<p>Pasa</p> <p>Se visualiza los vuelos que se encuentran vigentes</p>
Veredicto (UAT-MAP-002) y comentario:	<p>Pasa</p> <p>Se carga un mapa que muestra una capa de datos meteorológicos. La capa es legible y superponerse correctamente sobre el mapa base.</p>	<p>Pasa</p> <p>Mapas muy ilustrativos y didácticos</p>	<p>Pasa</p> <p>Buena vista satelital</p>
Veredicto (UAT-MAP-003) y comentario:	<p>Pasa</p> <p>Se carga un mapa interactivo que muestra puntos en aguas nacionales correspondientes a embarcaciones. Al apuntar a cualquiera de ellos se despliega el nombre de la embarcación. Al hacer click en el punto o figura, se muestra información como</p>	<p>Pasa</p> <p>Se cumplió el resultado esperado de los mapas.</p>	<p>Pasa</p> <p>Se conoce donde llegan las embarcaciones</p>

Pregunta	Catedrático experto en proyectos de software	Investigador/Profesor de la Facultad de Turismo (UAGro)	Estudiante de maestría/doctrado de la Facultad de Turismo (UAGro)
	nombre, nacionalidad, tipo de embarcación, foto, entre otros datos.		
Veredicto (UAT-MAP-004): y comentario	<p>Pasa</p> <p>Se carga un mapa interactivo que muestra secciones de colores correspondientes a vientos, temperatura u oleaje, según el mapa que se haya seleccionado. Las funciones de zoom (rueda del ratón o botones +/-) y arrastrar con el ratón funcionan correctamente y al hacer click en un sitio en el mapa (Acapulco), se abre un recuadro de la información meteorológica correspondiente.</p>	<p>Pasa</p> <p>Versión muy ilustrativa.</p>	<p>Pasa</p> <p>Se obtiene la información metereológica</p>
Veredicto (UAT-LIV-001)	<p>Pasa</p> <p>carga mostrando uno o varios reproductores de video incrustados. Los reproductores muestran un estado de carga, que al hacer clic sobre alguno inicia la transmisión de video desde las cámaras web configuradas.</p>	<p>Pasa</p> <p>Una estrategia muy viable para observar Acapulco, con relación a la seguridad y otros aspectos.</p>	<p>Pasa</p> <p>Las cámaras web muestran en tiempo real el estado cómo se encuentra Acapulco en diferentes playas.</p>
Veredicto (UAT-DOC-001) y comentario	Pasa	Pasa	Pasa

Pregunta	Catedrático experto en proyectos de software	Investigador/Profesor de la Facultad de Turismo (UAGro)	Estudiante de maestría/doctarado de la Facultad de Turismo (UAGro)
	La lista de documentos se filtra para mostrar únicamente aquellos que coinciden con el criterio de búsqueda.	Alternativa muy viable con relación a la documentación turística, no solo de Acapulco, sino de todo el mundo, magnifico banco de datos.	Si se localizan los documentos de búsqueda
Veredicto (UAT-DOC-002) y comentario:	Pasa El documento en formato PDF se abre correctamente en una nueva pestaña del navegador o se inicia su descarga. El archivo descargado se abre con un lector de PDF estándar y su contenido debe ser íntegro.	Pasa El observatorio responde positivamente a las necesidades de información y conocimiento.	Pasa Sin problema se pueden ver los sitios
Veredicto (UAT-BIB-001) y comentario	Pasa La página de Aiest se carga correctamente dentro del marco del sitio y la interacción con el contenido incrustado es fluida. Al seleccionar otra opción, el marco se actualiza y muestra la nueva biblioteca seleccionada.	Pasa Viable como banco de información para maestrandos y doctorandos	Pasa Sin problema
Veredicto (UAT-BIB-002) y comentarios	Pasa Se abre una nueva pestaña en el navegador y carga correctamente la página web externa correspondiente. El	Pasa Resultados altamente esperados	Pasa Se pueden localizar publicaciones

Pregunta	Catedrático experto en proyectos de software	Investigador/Profesor de la Facultad de Turismo (UAGro)	Estudiante de maestría/doctorado de la Facultad de Turismo (UAGro)
	enlace no está roto y no se presenta ningún error.		
Veredicto (UAT-NFR-001) (Describir cualquier punto de confusión en la navegación.)	Pasa La navegación es fluida e intuitiva. El menú principal es claro y siempre visible.	Pasa En general es una alternativa como observatorio, viable en todos los aspectos.	Pasa Si se localiza rápidamente
Fallos de compatibilidad	No hubo fallas, me parece correcto y perfecto.	No tuve problema	
Veredicto (UAT-NFR-002)	Pasa (el rendimiento fue aceptable)	Pasa (el rendimiento fue aceptable)	Pasa (el rendimiento fue aceptable)
¿Cómo percibió el tiempo de carga del dashboard?	Muy rápido (carga casi instantánea)	Muy rápido (carga casi instantánea)	Muy rápido (carga casi instantánea)
Rendimiento (UAT-NFR-002) Anote el tiempo de carga aproximado y describa cualquier lentitud observada.	La plataforma funciona correctamente y es muy rápida.	Exacto el tiempo	Fue con buen tiempo
¿Cuáles considera que son las mayores fortalezas del sistema desde su perspectiva?	Rápida y presenta la información de manera muy precisa.	Todos los aspectos llenaron mis expectativas, estar en la globalización del turismo, la información y el conocimiento.	El conocimiento de los datos dentro del Observatorio Turístico
¿Cuáles son las áreas más significativas para la mejora en términos	La plataforma funciona a la perfección.	Solo una redacción equivocada.	Ocupación hotelera y derrama económica

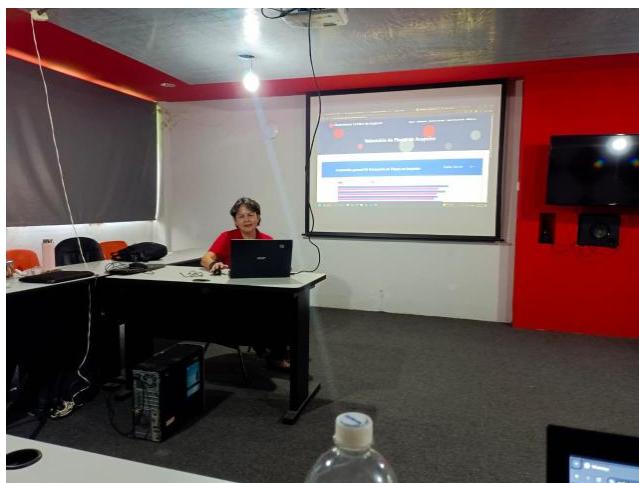
Pregunta	Catedrático experto en proyectos de software	Investigador/Profesor de la Facultad de Turismo (UAGro)	Estudiante de maestría/doctrado de la Facultad de Turismo (UAGro)
de funcionalidad o usabilidad?			
Veredicto. Con base en los resultados de las pruebas, emita su veredicto sobre la aceptación del sistema.	Aprobado: El sistema cumple con los requisitos esenciales y es aceptado.	Aprobado: El sistema cumple con los requisitos esenciales y es aceptado.	Aprobado: El sistema cumple con los requisitos esenciales y es aceptado.

Nota. Elaboración propia.

Se presenta además evidencia fotográfica en la Figura 77 sobre la aplicación de las pruebas de aceptación por usuarios.

Figura 77

Presentación del Observatorio turístico de Acapulco durante la aplicación de pruebas de aceptación (UAT).



Nota. Evidencia fotográfica de la sesión de aplicación de Pruebas de Aceptación de Usuario (UAT) con el panel de evaluadores de la Facultad de Turismo.