



Facultad de
Ciencias

UNAM

Manual de Proyecto

Presenta(n):

Flores Arriola Edson Rafael - 423118018

Ortega Medina David - 319111866

Rivera Soto Aline Daniela - 320333035

Modelado y Programación

Semestre 2025-1

6 de octubre de 2024



LICENCIATURA
EN CIENCIAS DE
LA COMPUTACIÓN

Índice

1. Proyecto	1
1.1. Introducción	1
1.2. Objetivo	1
2. Lineamientos generales	2
2.1. Tratados Internacionales	2
2.2. Leyes	2
2.3. Códigos	3
2.4. Reglamentos	3
3. Generalidades del proyecto	4
3.1. ¿Cómo está formado?	4
3.2. ¿A quién está dirigido?	5
3.3. Roles y responsabilidades	5
4. Información del Software	6
4.1. Software	6
4.2. Requerimientos	6
4.3. ¿En que está enfocado?	6
4.4. Diagrama de flujo	7
4.5. Pseudocódigo	8
4.6. cache.py	8
4.7. weather_api.py	9
4.8. weatherpasa_api.py	10
4.9. city_data_utils.py	11
4.10. Página principal y secundarias	12
4.11. Imágenes y partes gráficas	14
4.12. Colores y tipografías	15
5. Procedimientos	16
5.1. Mantenimiento	16
5.2. Actualizaciones	16
Referencias	18

Capítulo 1

Proyecto

1.1. Introducción

El presente manual tiene como finalidad dar a conocer de manera detallada y sencilla la estructura de la web “Clima AICM”, para que cualquier usuario pueda sacar el máximo partido de la misma. El sitio fue diseñado para que cada quien pueda, de una forma intuitiva, realizar búsquedas eficientes mediante tickets, nombre de ciudades o IATAS del clima en distintas ciudades de México y del mundo.

1.2. Objetivo

Brindar una descripción clara y detallada sobre el funcionamiento y uso de los distintos elementos de la página web para guiar al usuario y desarrolladores en la búsqueda de información.

Capítulo 2

Lineamientos generales

2.1. Tratados Internacionales

- Tratado sobre la Promoción, protección y disfrute de los derechos humanos en internet.

2.2. Leyes

- Ley Federal de Protección de Datos en Posesión de los Particulares.(LFDPP)
- Ley General de Responsabilidades Administrativas.
- Ley Federal de Responsabilidades Administrativas de los Servidores Públicos.
- Ley de Adquisiciones, Arrendamientos y Servicios del Sector Público.
- Ley Orgánica de la Administración Pública Federal.
- Ley Federal de las Entidades Paraestatales.
- Ley General de Transparencia y Acceso a la Información Pública.
- Ley General de Archivos.

2.3. Códigos

- Código de conducta de las entidades Aeropuerto Internacional de la Ciudad de México, S.A. de C.V. (AICM) y Servicios Aeroportuarios de la Ciudad de México, S.A. de C.V. (SACM)
- Código de conducta de las y los servidores públicos.

2.4. Reglamentos

- Reglamento Interior de la Secretaría de la Función Pública.
- Reglamento de la Ley Federal de las Entidades Paraestatales.
- Reglamento de la Ley Federal de Transparencia y Acceso a la información.
- Reglamento de la Ley de Aeropuertos.

Capítulo 3

Generalidades del proyecto

3.1. ¿Cómo está formado?

El proyecto tiene 3 carpetas principales.

1. **Static**

Aquí encontraremos una carpeta con las imágenes utilizadas en la interfaz gráfica, los archivos java script encargados de hacer el multilenguaje y por último los archivos '.csv'.

2. **Backend**

- city_data_utils.py se encarga de asociar las ciudades con sus respectivos códigos IATA.
- weather_api.py y weatherpasa_api.py se encargan de hacer la llamada a las API para obtener los datos para los perfiles de pasajeros y tripulación.

3. **Templates**

Tenemos 7 plantillas HTML diferentes, las cuales desglosaremos a continuación.

■ **base.html**

Aquí está el código correspondiente a la barra de navegación y al footer.

■ **inicio.html**

Esta plantilla hereda de base.html y contiene el código para escoger un perfil, además de encontrar los links para hacer los check in en distintas aerolíneas e indicaciones de como llegar a ambas terminales del AICM.

■ **tripulacion.html y pasajeros.html**

Ambas plantillas tienen un buscador , la diferencia entre ambas es la siguiente plantilla a la que te redirijen.

■ **clima.html**

Aquí aparecera la información del clima de la ciudad buscada para el perfil de la tripulación.

■ **climapa.html**

Esta plantilla hereda de 'clima.html' y da la información del clima para los pasajeros, contiene información distinta a la dada en el perfil de tripulación.

■ **index.py**

El archivo encargado de establecer rutas para las plantillas html y encargada de leer la entrada.

3.2. ¿A quién está dirigido?

La página web esta dirigida para los pasajeros y la tripulación(pilotos y sobrecargos) que viajan hacia y desde el Aeropuerto Internacional de la Ciudad de México.

3.3. Roles y responsabilidades

A continuación expondremos los roles que cada integrante del equipo tuvo en la elaboración de la página web así como sus responsabilidades.

■ Backend

Los encargados de esta parte del proyecto fueron responsables de:

- Obtener los datos del clima de la API.
- Leer y aceptar como entradas de busqueda tickets, nombres de ciudades y IATAS.
- Manejo de errores.

Los responsables de realizar las tareas anteriormente mencionadas son Flores Arriola Edson Rafael y Ortega Medina David.

- Conexión de la interfaz gráfica con todo el código.

■ Frontend

Los encargados de esta parte del proyecto fueron responsables de:

- Diseño de la interfaz de la página web.
- Incluir todos los links de referencias a páginas externas.
- Hacer la página multilenguaje.

La responsable de realizar las tareas anteriormente mencionadas es Rivera Soto Aline Daniela.

Por otro lado, el lider de este proyecto fue Ortega Medina David.

Capítulo 4

Información del Software

4.1. Software

Para obtener una mejor experiencia con la interfaz grafica es recomendable correr la aplicación desde algún navegador de un equipo de computo, con sistema:

- Windows 10 o posterior
- Mac
- Linux (Fedora, Ubuntu, etc)

4.2. Requerimientos

Para poder ejecutar la aplicación será necesario tener instalados los siguientes requisitos:

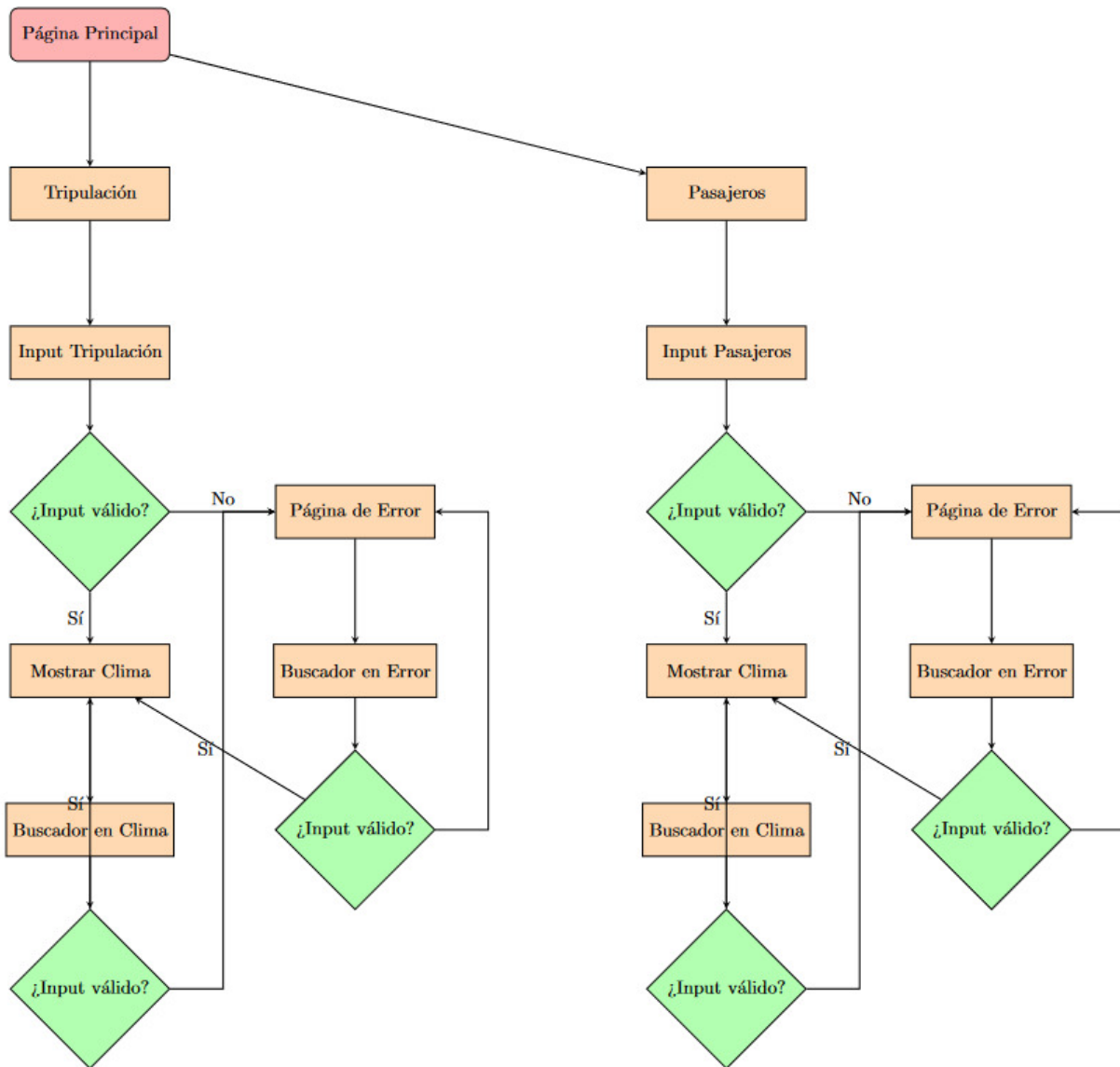
- Python 3
- pip (administrador de paquetes de Python)
- Cuenta de OpenWeather y su clave de API
- Flask

Nota: En caso de querer subir el proyecto a la web será necesario pagar las imágenes utilizadas en las misma o cambiarlas, pues estan usadas solo como sample.

4.3. ¿En que está enfocado?

El desarrollo de esta página web esta enfocado en tener un diseño interactivo y amigable que permita consultar la información actualizada del clima en tiempo real para distintas ciudades además de ser capaz de manejar errores de entrada comunes (por ejemplo, variaciones en los nombres de ciudades).

4.4. Diagrama de flujo



4.5. Pseudocódigo

Estos pseudocódigos son de las funciones principales, se encuentran en la carpeta `backend`, estos son los archivos que permiten que la aplicación funcione.

4.6. `cache.py`

```
1: Declarar diccionario global _cache
2: Declarar cache_duration = 1800 segundos (duración de 30 minutos)
3: Función get_from_cache(key):
4:   Si key está en _cache:
5:     Obtener value y last_updated de _cache[key]
6:     Si la diferencia entre el tiempo actual y last_updated es menor a cache_duration:
7:       Retornar value y last_updated
8:   Retornar None, None si la clave no se encuentra o ha expirado
9: Función set_to_cache(key, value):
10:  Asignar en _cache[key] el par (value, tiempo_actual) donde tiempo_actual es el
    tiempo presente
11:  Retornar (actualiza el caché con la clave y el valor)
12: Función clear_cache():
13:  Vaciar el diccionario _cache
```

4.7. weather_api.py

```
1: Importar módulos necesarios:
2:   os, requests, dotenv, datetime
3:   Funciones de caché: get_from_cache, set_to_cache
4: Cargar las variables de entorno:
5:   Obtener la clave de API de OpenWeather desde un archivo de entorno
6: Función principal: obtener_clima(city_or_iata_code)
7: if city_or_iata_code es None then
8:   Retornar "Ciudad no encontrada"
9: end if
10: Verificar caché:
11:   Buscar si los datos de la ciudad están almacenados en caché
12: if datos en caché existen then
13:   Retornar datos en caché
14: end if
15: Hacer solicitud a OpenWeather:
16:   Construir URL usando el nombre de la ciudad o código IATA y la clave de API
17:   Realizar la solicitud a OpenWeather y obtener respuesta en formato JSON
18: Procesar la respuesta:
19: if la ciudad fue encontrada en la respuesta de la API then
20:   Extraer datos principales: temperatura, presión, humedad
21:   Convertir las temperaturas de Kelvin a Celsius
22:   Extraer descripción del clima e ícono
23:   Extraer información adicional (viento, visibilidad, nubosidad, lluvia)
24:   Obtener la hora de amanecer y atardecer en formato legible
25:   Obtener coordenadas geográficas (latitud y longitud)
26: else
27:   Retornar "Ciudad no encontrada"
28: end if
29: Estructurar los datos del clima:
30:   Crear un diccionario con los datos procesados (temperatura, sensación térmica,
    presión, humedad, descripción del clima, viento, visibilidad, etc.)
31: Guardar datos en caché:
32:   Almacenar los datos en caché usando set_to_cache
33: Retornar los datos del clima procesados
```

4.8. weatherpasa_api.py

```
1: Importar módulos necesarios:
2:   os, requests, dotenv, datetime
3:   Funciones de caché: get_from_cache, set_to_cache
4: Cargar las variables de entorno:
5:   Obtener la clave de API de OpenWeather desde un archivo de entorno
6: Función principal: obtener_clima_pasajeros(city_or_iata_code)
7: if city_or_iata_code es None then
8:   Retornar "Ciudad no encontrada"
9: end if
10: Verificar caché:
11:   Buscar si los datos de la ciudad están almacenados en caché
12: if datos en caché existen then
13:   Retornar datos en caché
14: end if
15: Hacer solicitud a OpenWeather:
16:   Construir URL usando el nombre de la ciudad o código IATA y la clave de API
17:   Realizar la solicitud a OpenWeather y obtener respuesta en formato JSON
18: Procesar la respuesta:
19: if la ciudad fue encontrada en la respuesta de la API then
20:   Extraer datos principales: temperatura, presión, humedad
21:   Convertir las temperaturas de Kelvin a Celsius
22:   Extraer descripción del clima e ícono
23:   Extraer información adicional (viento, visibilidad, nubosidad, lluvia)
24:   Obtener la hora de amanecer y atardecer en formato legible
25:   Obtener coordenadas geográficas (latitud y longitud)
26: else
27:   Retornar "Ciudad no encontrada"
28: end if
29: Estructurar los datos del clima:
30:   Crear un diccionario con los datos procesados:
31:   temperature, feels_like, temp_min, temp_max, pressure, humidity, description,
   icon_url, wind_speed, wind_deg, visibility, cloudiness, sunrise, sunset, country,
   longitude, latitude, rain_1h
32: Guardar datos en caché:
33:   Almacenar los datos en caché usando set_to_cache
34: Retornar los datos del clima procesados
```

4.9. city_data_utils.py

```
1: Inicializar diccionario city_synonyms con sinónimos de ciudades
2: Función remove_accents(input_str):
3:     Convertir input_str a la forma NFKD
4:     Retornar input_str sin caracteres combinados (acentos)
5: Función load_iata_data(file_path):
6:     Leer archivo CSV con pandas en la ruta file_path
7:     Crear diccionario iata_to_city que mapea códigos IATA a ciudades
8:     Crear conjunto valid_cities con las ciudades únicas
9:     Retornar iata_to_city y valid_cities
10: Función normalize_city_name(user_input):
11:     Convertir user_input a minúsculas y remover acentos
12:     Para cada city_name y synonyms en city_synonyms:
13:         Normalizar cada sinónimo en minúsculas y sin acentos
14:         Si user_input coincide con un sinónimo normalizado:
15:             Retornar city_name
16:     Retornar user_input si no se encuentra un sinónimo
17: Función get_closest_city_name(user_input, city_list):
18:     Normalizar el nombre de la ciudad user_input usando normalize_city_name
19:     Buscar el match más cercano en city_list usando process.extractOne con un
    umbral de 80
20:     Si se encuentra un match cercano:
21:         Retornar el nombre de la ciudad más cercana
22:     Retornar user_input si no hay coincidencias cercanas
23: Función map_iata_to_city(iata_code, iata_to_city):
24:     Retornar el nombre de la ciudad asociada al código IATA dado por iata_code, si
    existe
25: Función hex_to_iata(hex_string, iata_to_city):
26:     Intentar convertir la cadena hexadecimal hex_string en un código IATA
27:     Si la conversión es exitosa:
28:         Retornar el código IATA
29:     Si no:
30:         Retornar None
```

4.10. Página principal y secundarias

■ Home-Página Principal

Una vez que el usuario ingresa a la página web, lo primero que observa es la página principal del sitio. Esta será descrita a continuación:

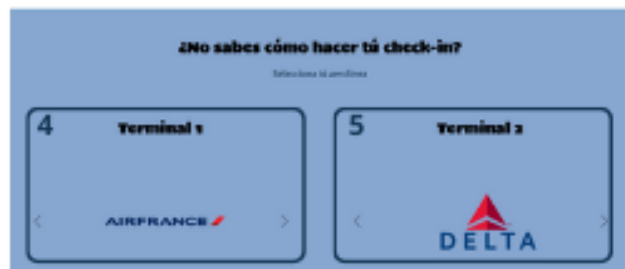


1. Logo del AICM

2. Icono para seleccionar el idioma de la página web

3. Imágenes para seleccionar el perfil de la sesión. A la derecha el perfil de los pasajeros y a la izquierda el de la tripulación.

4. Carrusel donde puedes encontrar el link para hacer el check-in en todas las aerolíneas que parten de la Terminal 1.
5. Carrusel donde puedes encontrar el link para hacer el check-in en todas las aerolíneas que parten de la Terminal 2.



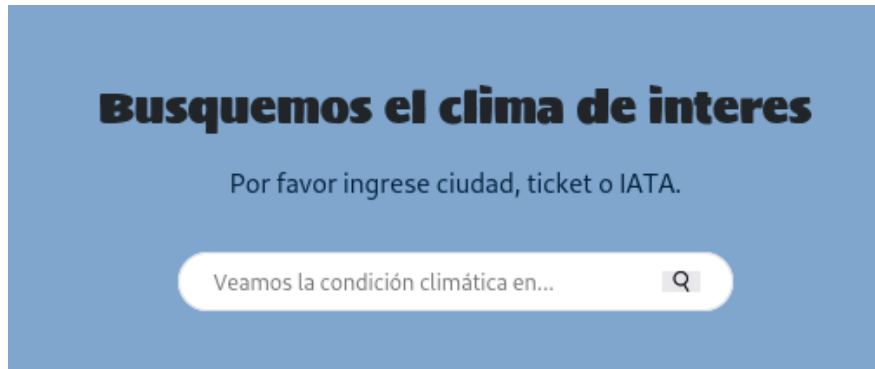
6. Se puede obtener el link para las instrucciones de Google Maps de como llegar a la Terminal 1 y 2.

7. Tenemos 5 iconos que nos redirigirán a las redes sociales del AICM entre ellas X(antes Twitter), Instagram, Facebook, Página web del AICM y el correo electrónico.



■ Buscador Pasajeros/Tripulación

Una vez que el usuario escoge un perfil es redirijido a esta plantilla en donde ingresará el nombre de la ciudad, código IATA o número de ticket que se quiera buscar.



Busquemos el clima de interes

Por favor ingrese ciudad, ticket o IATA.

Veamos la condición climática en...

■ Impresión del clima

Una vez realizada la busqueda el usuario estará en esta plantilla donde se imprimiran los datos del clima del lugar buscado, los datos que se arrojaran dependeran del perfil seleccionado.



4.11. Imágenes y partes gráficas

■ Imágenes

A continuación tendremos los links donde se pueden encontrar todas las imágenes utilizadas:

- Imagen de tripulación
- Imagen de pasajeros
- Los logos de las aerolíneas fueron todos sacados de Wikipedia.
- Logo del AICM

■ Iconos

Los iconos con colores varios fueron sacados de Bootstrap.

Los demás iconos fueron sacados de Flaticon

4.12. Colores y tipografías

A continuación pondremos los códigos HEX o RGB de los colores utilizados en la interfaz de la web.

- **Background body:** rgb(128, 166, 206)
- **Background nav-bar:** rgb(128, 166, 206)
- **Background sección clara:** #88a7d0
- **Background hover sección llegar:** rgb(221, 229, 182)
- **Border sección llegar:** #8080804d
- **Iconos del clima:** #00203b
- **Background iconos redes sociales:** rgb(240, 234, 210)
- **Background hover iconos redes sociales:** rgb(173, 193, 120)
- **Sección oscura:** #133855
- **Hover idiomas:** rgb(228, 228, 228)
- **Letras idiomas:** rgb(85, 85, 85)
- **Letras idiomas hover:** rgb(219, 219, 219)
- **Box-shadow buscador-input:** #20212447
- **Border-color buscador-input:** #dfe1e500

Ahora expondremos las fuentes utilizadas.

- **Sección-título:** “ Sigmar “
Se puede obtener en: Google Fonts-Sigmar
- **Seccion-datos:** “ Anton “
Se puede obtener en: Google Fonts-Anton
- **Datos del clima:** “ Ubuntu “
Se puede obtener en: Google Fonts-Ubuntu

Capítulo 5

Procedimientos

5.1. Mantenimiento

- **Actualización de dependencias:** Mantén las versiones de Flask, requests, pandas, y otras dependencias actualizadas para asegurar estabilidad y seguridad.
- **Mejoras en el manejo de errores:**
 - Implementar monitoreo en tiempo real con herramientas como *Sentry*.
 - Registrar logs detallados de errores para facilitar la depuración.
- **Pruebas y validación:**
 - Ampliar pruebas unitarias y de integración para cubrir más casos de uso.
 - Implementar pruebas automatizadas en CI/CD (p.ej., GitHub Actions).
- **Optimización de rendimiento:**
 - Implementar caching para evitar solicitudes repetidas a la API.
 - Manejar el rate limiting de OpenWeatherMap para evitar exceder los límites de solicitudes.
- **Mejora de seguridad:**
 - Proteger la clave API de OpenWeatherMap, evitando su exposición pública.

5.2. Actualizaciones

A continuación mencionaremos los posibles cambios y mejoras que se realizarán en futuras actualizaciones de la página web “Clima AICM”.

Para el backend

- Integración de mapas interactivos.
- Pronóstico extendido y comparaciones.
- Recomendaciones basadas en el clima.
- Personalización de datos arrojados por la búsqueda.

Para el frontend

- Iconos del clima dinámicos.
- Background dinámico acorde al clima del lugar.
- Más idiomas disponibles.
- Introducción de avisos emergentes sobre situaciones de urgencia en el AICM.
- Introducción de anuncios de comercios ubicados en el AICM.

Referencias

- [1] A. General, "Promoción, protección y disfrute de los derechos humanos en internet," Junio 2016. [Online]. Available: https://ap.ohchr.org/documents/S/HRC/d_res_dec/A_HRC_32_L20.pdf
- [2] INAI, "Normativa y legislación en pdp – marco internacional de competencias de protección de datos personales para estudiantesl," Sin fecha. [Online]. Available: https://micrositios.inai.org.mx/marcocompetencias/?page_id=370
- [3] Juárez, "Código de conducta de las y los servidores públicos," Sin fecha. [Online]. Available: <http://www.gob.mx/becasbenitojuarez/documentos/codigo-de-conducta-de-las-y-los-servidores-publicos-25274>
- [4] G. de México, "Manual de procedimientos del Órgano interno de control en aeropuerto internacional de la ciudad de México," 2018. [Online]. Available: No.deRegistro:OIC-PR-001/2018VersiÃşn/AÃşo:1.0/2018