

Plantilla de documentación del proceso de diseño de soluciones y proyectos

Nombre completo

Jane Andrey Méndez Blanco

IteraFlex: Diseño Ágil de Soluciones y Proyectos

El proceso IteraFlex es una herramienta para guiar el desarrollo de soluciones y proyectos y convertir nuestras ideas en realidad. IteraFlex facilita la empatía con el usuario final, el aprendizaje continuo de los errores y la resolución de problemas de forma creativa. Este método es iterativo, lo que significa que puede repetirse varias veces para encontrar la mejor solución posible.

En esta plantilla, completaremos la información necesaria para documentar el proceso de desarrollo de su proyecto mediante IteraFlex.



Paso 1: Definir el problema

En este paso se define el problema a resolver o proyecto a desarrollar de forma clara. También se identifican las limitaciones y los obstáculos a los que nos enfrentamos. Esto usualmente implica comunicarnos con empatía con las personas para las que se diseña. La idea es clarificar cuál es la necesidad o problema a resolver y abstraer lo esencial del problema a resolver para enfocarnos en ello.

Tomando en cuenta la definición anterior conteste las siguientes preguntas:

Pregunta A

¿Cuál es el objetivo del proyecto que va a realizar?

El objetivo del proyecto es analizar un conjunto de datos de clientes para detectar posibles patrones de fraude ocurridos entre los meses de noviembre y diciembre. A través del uso de Python y librerías de análisis de datos, se busca identificar cuentas que comparten información sensible como contraseñas, números telefónicos, estados de residencia y dominios de correo electrónico, con el fin de determinar comportamientos sospechosos asociados a fraude.

Pregunta B

¿Qué componentes debe tener?

El proyecto debe contar con los siguientes componentes:

- 1. Un archivo de datos en formato CSV que contenga la información de los clientes.*
- 2. Un módulo de limpieza de datos que normalice columnas y prepare la información para análisis.*
- 3. Un módulo de análisis que identifique patrones de fraude en contraseñas, teléfonos, estados, correos y fechas.*
- 4. Un resumen de hallazgos que explique los patrones detectados.*
- 5. Un dashboard que visualice los resultados de manera clara.*

Paso 2: Investigar

En esta etapa podemos hablar con distintas personas, realizar búsquedas en línea para investigar qué proyectos o soluciones ya existen, o qué tecnologías podrían adaptarse a nuestras necesidades. También es importante indagar el contexto y las implicaciones del problema.

Investigar es hacer preguntas y buscar respuestas. Muchas veces las respuestas nos llevan a otras preguntas que nos hacen pensar sobre nuestro problema desde perspectivas nuevas. Ahora investigue en relación a la aplicación que va a crear para su proyecto final y complete las siguientes preguntas:

Pregunta C

Liste al menos dos fuentes en las que indagó sobre el contexto del proyecto a realizar.

- | | |
|-----------|---|
| 1. | Documentación oficial de Pandas (pandas.pydata.org) |
|-----------|---|

2.	<i>Documentación oficial de Python (docs.python.org)</i>
3.	<i>Artículos sobre detección de fraude y análisis de datos en plataformas como Medium</i>
4.	<i>Ejemplos académicos de análisis de datos vistos en clase</i>
5.	<i>Documentación y ejemplos de visualización de datos con Plotly y Streamlit</i>

Pregunta D

Comparta al menos una solución existente que aborde su necesidad.

1.	<i>Sistemas de detección de fraude utilizados por bancos que analizan coincidencias en datos de clientes.</i>
2.	<i>Herramientas de análisis antifraude basadas en reglas (rule-based fraud detection).</i>
3.	<i>Plataformas de análisis de datos que detectan duplicados y comportamientos anómalos.</i>

Paso 3: Plantear una solución

Este es el momento de abrir la mente y generar ideas de posibles abordajes para nuestra solución o proyecto y desarrollar tantas soluciones como sea posible. Estas soluciones pueden ser simples o complejas. Recuerde que las ideas evolucionan, se mezclan unas con otras y ¡no existen malas ideas!

Después de imaginar todas las posibles soluciones es tiempo de considerar las necesidades, limitaciones e investigación de los pasos anteriores y comparar las ideas, para seleccionar una solución y elaborar un plan para seguir adelante.

Al seleccionar una posible solución es útil modularizarla (plantear una solución programada que está en procedimientos que se ejecutan de manera articulada) para comprender sus partes y cómo estas se pueden ejecutar de forma articulada. A la hora de crear un plan es útil formular algoritmos o instrucciones paso a paso para completar una tarea.

En el paso 4 la idea es clarificar cuál es la necesidad o problema a resolver y abstraer lo esencial del problema a resolver para enfocarnos en ello.

Tomando en cuenta la definición anterior conteste las siguientes preguntas:

Pregunta E

Escriba sus ideas de solución (mínimo 3 ideas)

1.	<i>Analizar contraseñas repetidas como indicador de cuentas falsas o compartidas.</i>
2.	<i>Detectar números telefónicos usados por múltiples clientes.</i>
3.	<i>Identificar clientes que pertenecen al mismo estado y comparten credenciales.</i>
4.	<i>Analizar dominios de correo electrónico asociados a patrones sospechosos.</i>
5.	<i>Visualizar los resultados en un dashboard interactivo.</i>

Pregunta F

De las ideas de solución (Pregunta A), seleccione la idea que va a desarrollar

Se selecciona la solución basada en el análisis de patrones de fraude mediante coincidencias en contraseñas, teléfonos, estados y dominios de correo electrónico, utilizando Python y Pandas.

Pregunta G

Modularice y liste los procedimientos necesarios para llegar a la solución seleccionada (mínimo 3 procedimientos)

1.	<i>Cargar el archivo CSV con la información de clientes.</i>
2.	<i>Limpiar y normalizar los datos (columnas, valores nulos, formatos).</i>
3.	<i>Filtrar los registros correspondientes a noviembre y diciembre.</i>
4.	<i>Analizar patrones de coincidencia en contraseñas, teléfonos, estados y correos.</i>
5.	<i>Generar un resumen de hallazgos de fraude.</i>

Pregunta H

Desarrolle un algoritmo (en lenguaje cotidiano o lenguaje natural) que describa paso a paso las instrucciones para conseguir la solución a la idea seleccionada. (mínimo 3 pasos)

1.	<i>Leer el archivo de datos de clientes desde un archivo CSV.</i>
2.	<i>Limpiar los datos eliminando errores, caracteres innecesarios y valores nulos.</i>
3.	<i>Filtrar los registros por fechas relevantes (noviembre y diciembre).</i>
4.	<i>Analizar cada columna clave para identificar valores repetidos.</i>
5.	<i>Mostrar los resultados y un resumen de los patrones de fraude detectados.</i>

Pregunta I

Cree un programa siguiendo el algoritmo y que dé respuesta al problema planteado. Luego de crearlo y depurarlo, agregue imágenes del código del programa en este espacio:

El programa fue desarrollado en Python y dividido en módulos para facilitar su mantenimiento y comprensión.

Incluye un archivo para la limpieza de datos (datos.py) y otro para el análisis de fraude (analisis.py).

El código permite identificar contraseñas, teléfonos, estados y dominios de correo electrónico compartidos entre múltiples clientes, generando un resumen final de los hallazgos.

Datos – Etapa 1

```

1 import pandas as pd
2
3 def cargar_y_limpiar(ruta_archivo):
4     """
5         Lee el archivo CSV, limpia la data y devuelve un DataFrame listo para análisis.
6         """
7     try:
8         print("👉 Cargando archivo...")
9
10        # Detecta automáticamente CSV o Excel
11        if ruta_archivo.endswith(".csv"):
12            df = pd.read_csv(ruta_archivo, sep=";") # 🤝 CORRECCIÓN IMPORTANTE
13        else:
14            df = pd.read_excel(ruta_archivo)
15
16        print("✓ Archivo cargado correctamente.")
17
18        # -----
19        # 1. LIMPIEZA GENERAL
20
21        # Normalizar nombres de columnas
22        df.columns = (
23            df.columns
24            .str.strip()
25            .str.lower()
26            .str.replace(" ", "_")
27            .str.replace("-", "_")
28        )
29
30        # Quitar duplicados
31        df = df.drop_duplicates()
32
33        # Manejo de valores nulos
34        df = df.fillna({
35            "email": "unknown",
36            "phones": "unknown",
37            "password": "unknown",
38            "state": "unknown"
39        })
40
41        # Convertir columna de fecha a datetime
42        if "joined" in df.columns:
43            df["joined"] = pd.to_datetime(df["joined"], errors="coerce")
44
45        # Filtrar solo noviembre-diciembre (fraude reportado)
46        if "joined" in df.columns:
47            df = df[
48                (df["joined"].dt.month == 11) |
49                (df["joined"].dt.month == 12)
50            ]
51
52        print(f"📝 Registros filtrados (nov-dic): {len(df)}")
53
54
55        # 2. LIMPIEZA DE NOMBRES
56
57        columnas_nombres = [col for col in df.columns if "name" in col or col == "customer"]
58
59        for col in columnas_nombres:
60            df[col] = (
61                df[col]
62                .astype(str)
63                .str.replace("[^a-zA-ZáéíóúñÁÉÍÓÚÑ]", "", regex=True) # quitar ? ! % * etc
64                .str.replace("\s+", " ", regex=True)
65                .str.strip()
66                .str.title()
67            )
68
69
70        # 3. LIMPIEZA ESPECÍFICA PARA DETECCIÓN DE FRAUDE
71
72        if "phones" in df.columns:
73            df["phones"] = df["phones"].astype(str).str.replace(r"\D", "", regex=True)
74
75        if "password" in df.columns:
76            df["password"] = df["password"].astype(str).str.lower().str.strip()
77
78        if "email" in df.columns:
79            df["email"] = df["email"].astype(str).str.lower().str.strip()
80
81        if "state" in df.columns:
82            df["state"] = df["state"].astype(str).str.lower().str.strip()
83
84        print("📌 Limpieza completada.")
85
86        return df
87
88    except Exception as e:
89        print(f"🔴 Error al limpiar los datos: {e}")
90        return pd.DataFrame()
91
92
93    if __name__ == "__main__":
94        ruta = "data_clientes.csv"
95        df_limpio = cargar_y_limpiar(ruta)
96
97        print("\n--- Vista previa de datos limpios ---")
98        print(df_limpio.head())

```

1936

Análisis – Etapa 2



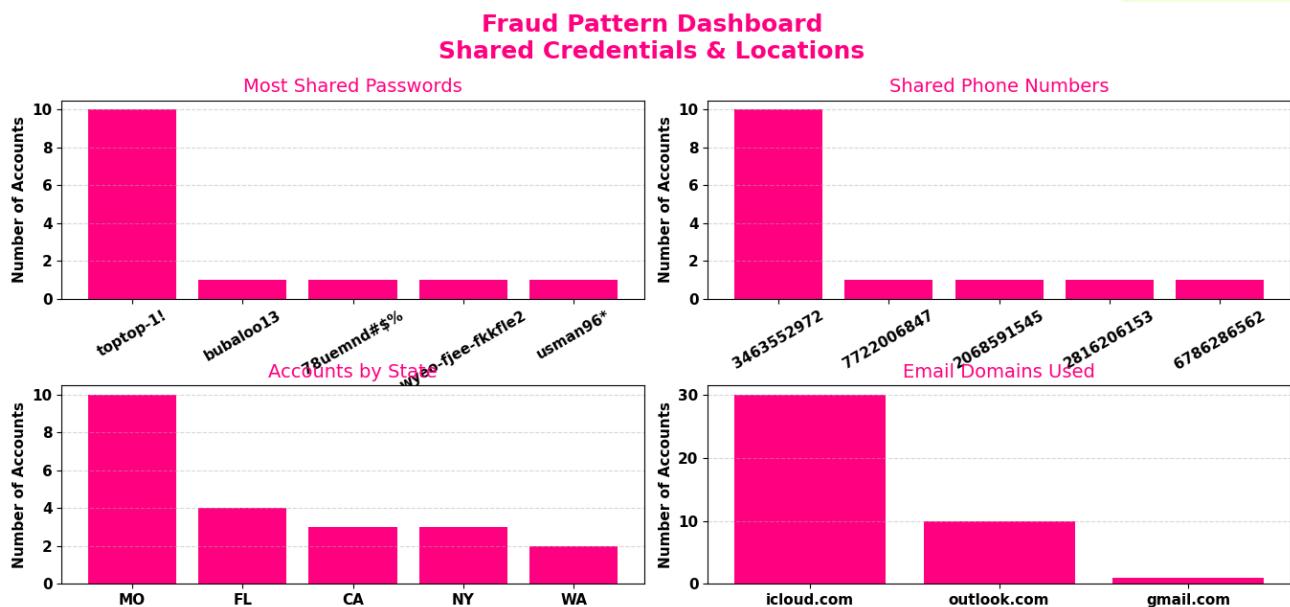
```

● ● ●
1 import pandas as pd
2 from collections import Counter
3
4 # 1. Analizar contraseñas repetidas
5 def analizar_passwords(df):
6     if "password" not in df.columns:
7         return {"error": "No existe columna 'password'."}
8
9     rep = df["password"].value_counts()
10
11     return {
12         "top_passwords": rep.head(10).to_dict(),
13         "passwords_unicas": len(df["password"].unique()),
14         "passwords_repetidas": rep[rep > 1].to_dict(),
15     }
16
17 # 2. Analizar correos sospechosos
18 def analizar_emails(df):
19     if "email" not in df.columns:
20         return {"error": "No existe columna 'email'."}
21
22     df["dominio"] = df["email"].astype(str).str.split("@").str[-1]
23
24     dominios_comunes = df["dominio"].value_counts().head(10).to_dict()
25     correos_repetidos = df["email"].value_counts()
26     correos_dup = correos_repetidos[correos_repetidos > 1].to_dict()
27
28     return {
29         "top_dominios": dominios_comunes,
30         "emails_repetidos": correos_dup
31     }
32
33 # 3. Analizar teléfonos duplicados
34 def analizar_phones(df):
35     if "phones" not in df.columns:
36         return {"error": "No existe columna 'phones'."}
37
38     phones_clean = df["phones"].astype(str)
39     conteo = phones_clean.value_counts()
40     duplicados = conteo[conteo > 1].to_dict()
41
42     return {
43         "telefonos_repetidos": duplicados,
44         "total_unicos": len(phones_clean.unique())
45     }
46
47 # 4. Actividad por estado
48 def analizar_estados(df):
49     if "state" not in df.columns:
50         return {"error": "No existe columna 'state'."}
51
52     actividad = df["state"].value_counts().to_dict()
53
54     return {
55         "actividad_por_estado": actividad
56     }
57
58 # 5. Actividad por fecha
59 def analizar_fechas(df):
60     if "joined" not in df.columns:
61         return {"error": "No existe columna 'joined'."}
62
63     df["fecha"] = df["joined"].dt.date
64     by_day = df.groupby("fecha").size().to_dict()
65
66     return {
67         "actividad_por_dia": by_day
68     }
69
70 # 6. Función principal
71 def analisis_fraude(df):
72     print("\n--- 🚩 ANALISIS DE PATRONES DE FRAUDE ---")
73
74     resultados = [
75         "passwords": analizar_passwords(df),
76         "emails": analizar_emails(df),
77         "phones": analizar_phones(df),
78         "estados": analizar_estados(df),
79         "fechas": analizar_fechas(df)
80     ]
81
82     print("\nAnalisis completado.")
83     return resultados
84
85 # 7. Ejecución directa
86 if __name__ == "__main__":
87     from datos import cargar_y_limpiar
88
89     df = cargar_y_limpiar("data_clientes.csv")
90     resultados = analisis_fraude(df)
91
92     print("\n--- 📊 RESULTADOS ---")
93     for k, v in resultados.items():
94         print(f"\n{k} ({k.upper()})")
95         print(v)
96
97     # 8. Resumen final
98
99     result_passwords = analizar_passwords(df)
100    result_emails = analizar_emails(df)
101    result_phones = analizar_phones(df)
102    result_estados = analizar_estados(df)
103    result_fechas = analizar_fechas(df)
104
105    print("\n--- 📈 RESUMEN FINAL DE PATRONES DE FRAUDE ---\n")
106
107    # PASSWORDS
108    top_pass = result_passwords.get("passwords_repetidas")
109    if top_pass:
110        pass_repetido = list(top_pass.keys())[0]
111        pass_count = top_pass[pass_repetido]
112        print(f"\n⚠️ La contraseña más repetida es '{pass_repetido}', usada por {pass_count} clientes.")
113    else:
114        print("\n⚠️ No se detectaron contraseñas repetidas.")
115
116    # EMAIL DOMAINS
117    top_domains = result_emails.get("top_dominios")
118    if top_domains:
119        dominio_comun = list(top_domains.keys())[0]
120        dominio_count = top_domains[dominio_comun]
121        print(f"\n⚠️ El dominio de email más común es '{dominio_comun}' con {dominio_count} registros.")
122
123    # PHONES
124    top_phones = result_phones.get("telefonos_repetidos")
125    if top_phones:
126        tel_repetido = list(top_phones.keys())[0]
127        tel_count = top_phones[tel_repetido]
128        print(f"\n⚠️ El teléfono más repetido es '{tel_repetido}', usado por {tel_count} clientes.")
129    else:
130        print("\n⚠️ No se detectaron teléfonos repetidos.")
131
132    # STATES
133    estados = result_estados.get("actividad_por_estado")
134    if estados:
135        estado_top = max(estados, key=estados.get)
136        print(f"\n⚠️ El estado con mayor actividad es '{estado_top.upper()}' con {estados[estado_top]} registros.")
137
138    # DATES
139    fechas = result_fechas.get("actividad_por_dia")
140    if fechas:
141        dias = len(fechas)
142        print(f"\n⚠️ La actividad abarca {dias} días consecutivos, sin días vacíos.")
143
144    # RELACIÓN: mismo estado + misma contraseña + teléfono -> dominio Outlook
145    df["dominio"] = df["email"].str.split("@").str[-1]
146
147    grupo_patron = df.groupby(["state", "password", "phones"])
148    coincidencias = grupo_patron.filter(lambda x: len(x) > 1)
149
150    if not coincidencias.empty:
151        dominios_en_patron = coincidencias["dominio"].unique().tolist()
152        print(f"\n⚠️ Sin embargo, los clientes que comparten estado, contraseña y teléfono utilizan dominios: {dominios_en_patron}")
153        if "outlook.com" in dominios_en_patron:
154            print(f"\n⚠️ Se confirma que el patrón está asociado principalmente a dominio 'outlook.com'.")
155    else:
156        print("\n⚠️ No se encontraron grupos con coincidencias para estado + contraseña + teléfono.")

```



Dashboard – Etapa 3



Rúbrica de Evaluación (25% de la nota del curso)

Avance del Proyecto 5%

Evaluación propuesta del Proyecto Final

Criterio	Excelente (3 puntos)	Bueno (2 puntos)	Insuficiente (1 punto)	No Presentado (0 puntos)	Puntos
Claridad y Objetivos	Explica claramente y con detalle el problema a resolver y los objetivos del proyecto son específicos y alcanzables.	Presenta el problema y los objetivos, pero carecen de claridad o no están completamente alineados.	Los objetivos son vagos o incompletos y no se relacionan adecuadamente con el problema a resolver.	No se presenta una descripción del problema ni los objetivos.	
Diseño y Metodología	Detalla las herramientas y técnicas de Python a emplear y justifica su uso para resolver el problema.	Enumera las herramientas y técnicas de Python, pero la explicación de su uso es incompleta o superficial.	Las herramientas o técnicas no están especificadas correctamente o no se relacionan con el problema.	No se describe herramientas ni técnicas de Python.	
Cronograma y Resultados	Presenta un cronograma detallado con hitos clave y resultados esperados bien	Incluye un cronograma y resultados esperados, pero no son suficientemente	El cronograma y los resultados esperados son poco claros, incompletos o no están alineados con los objetivos.	No se presenta cronograma ni resultados esperados.	

	definidos y relevantes al problema.	detallados o relevantes.			
Nota		Total, 9pts	Puntos oct.:	Calificación	



Proyecto 20%

Evaluación del Proyecto Final

Criterio	Excelente (5 puntos)	Bueno (3 puntos)	Insuficiente (1 punto)	No Presentado (0 puntos)	Puntaje Obtenido
Definición del problema y datos	Problema y datos claramente definidos, con relevancia explicada en un contexto práctico o teórico.	Problema y datos parcialmente definidos o relevancia no completamente clara.	Problema y datos vagos o sin definir, contexto ausente o no relevante.	No presenta definición del problema y datos	
Carga y Lectura de Archivos	Lectura y carga de archivos implementada correctamente, manejando errores y formatos diversos.	Lectura y carga implementada parcialmente, con errores menores o soporte limitado.	Lectura y carga de archivos incompleta o no implementada.	No realiza Carga y Lectura de Archivos	
Análisis Completo de Datos	Análisis completo, correcto y con resultados relevantes, incluyendo visualizaciones claras.	Análisis parcial y correcto, pero sin profundidad o visualizaciones limitadas.	Análisis incompleto, incorrecto o con resultados irrelevantes.	No presenta análisis Completo de Datos	
Metodología IteraFlex	Sigue todos los pasos de IteraFlex con claridad, asegurando un diseño funcional del proyecto.	Sigue parcialmente los pasos de IteraFlex. Implementación con deficiencias.	No sigue la metodología IteraFlex o la implementación es incorrecta.	No desarrolla metodología iteraflex	
Uso avanzado de variables	Emplea variables adecuadamente, optimizando su uso en el código.	Emplea variables parcialmente, con algunos errores o uso limitado.	No emplea variables adecuadamente o su uso es incorrecto.	No implementa uso avanzado de variables	
Uso de listas y diccionarios	Uso completo y eficaz de listas y diccionarios para resolver problemas.	Uso parcial o limitado de listas y diccionarios.	No emplea listas ni diccionarios o su uso es incorrecto.	No utiliza listas y diccionarios	
Conversión de datos	Realiza conversiones correctamente, sin errores.	Realiza conversiones parcialmente, con algunos errores menores.	Conversiones no realizadas o incorrectas.	No realiza conversión de datos	
Estructuras de control	Uso completo y correcto de ciclos y condicionales para resolver problemas.	Uso parcial o limitado de estructuras de control.	No utiliza ciclos ni condicionales o su implementación es incorrecta.	No implementa estructuras de control	

Criterio	Excelente (5 puntos)	Bueno (3 puntos)	Insuficiente (1 punto)	No Presentado (0 puntos)	Puntaje Obtenido
Análisis estadístico simple	Implementado correctamente, con resultados claros y relevantes.	Parcialmente implementado, resultados limitados o con algunos errores menores.	No implementado o resultados incorrectos.	No realiza análisis estadístico simple	
Interacción avanzada con usuario	Interacción significativa, con mensajes claros y funcionalidad completa.	Interacción parcial o funcionalidad básica.	Sin interacción significativa con el usuario.	No implementa interacción avanzada con usuario	
Comentarios en el código	Código bien documentado, con comentarios explicativos y claros.	Comentarios parciales, poco claros o ausentes en partes importantes del código.	Sin comentarios o documentación en el código.	No realiza comentarios en el código	
Exposición del Proyecto	Expone de manera clara, estructurada y siguiendo un flujo lógico. Demuestra dominio del tema y responde preguntas con seguridad.	Exposición con dificultades para seguir el flujo lógico. Responde preguntas con vacilación.	Exposición desorganizada o confusa. No demuestra dominio ni responde preguntas.	No realiza exposición del Proyecto	
Nota		Total, 60pts	Puntos oct.:	Calificación:	