

INFORME TÉCNICO DE PRÁCTICAS PRE PROFESIONALES Y PASANTÍAS

1. DATOS GENERALES

NOMBRE DEL ESTUDIANTE:			
CÉDULA	1751204072	CELULAR	0999711098
CORREO	hapauki@gmail.com		
CARRERA	Tecnología superior en Desarrollo de software		
SEMESTRE	Abril – Septiembre 2023	NIVEL	Quinto
EMPRESA	Instituto Nacional de Meteorología e Hidrología INAMHI		
ACTIVIDAD PRINCIPAL DE LA EMPRESA	Recopila, estudia, procesa, publica, y difunde la información hidrometeorológica.		
JEFE INMEDIATO	Ing. Diego Gonzales	CARGO	Coordinador del área de TICS
TELÉFONO	23971100	MAIL	servicio@inamhi.gob.ec
FECHA DE INICIO	27/06/2023	FECHA DE FINALIZACIÓN	25/08/2023
ÁREA EN LA QUE REALIZA LA PRÁCTICA	Tecnologías de la información y comunicación	NO. HORAS	246

2. DESCRIPCIÓN DE LA EMPRESA

INAMHI es el Servicio Meteorológico e Hidrológico Nacional del Ecuador creado por Ley, como una necesidad y un derecho fundamental de la comunidad, con capacidad y la obligación de suministrar información vital sobre el tiempo, el clima y los recursos hídricos del pasado, presente y futuro, que necesita conocer el país para la protección de la vida humana y los bienes materiales.

Es una Institución con representación nacional e internacional, miembro de la Organización Meteorológica Mundial, OMM, organización intergubernamental especializada de las Naciones Unidas para la Meteorología (el tiempo y el clima), la Hidrología Operativa y las ciencias conexas.

Es un organismo técnico que en el contexto nacional está adscrito al Ministerio de Ambiente y Agua; con personal técnico y profesional especializado en Meteorología e Hidrología, que contribuye al desarrollo económico y social del país.

3. OBJETIVOS DE LA PRÁCTICA

OBJETIVOS	
GENERAL	Aplicar los conocimientos adquiridos durante todo el proceso educativo y así mismo adquirir experiencia en su aplicación en un ámbito laboral.

ESPECÍFICOS	1. Adquirir la experiencia necesaria en aplicación de los conocimientos adquiridos durante el proceso de aprendizaje en la carrera correspondiente.
	2. Resolver las actividades y conflictos inherentes a las mismas.
	3. Mejorar el manejo de la información siendo parte esencial de una empresa con la automatización de ciertos procesos que así lo requieran.

4. DESCRIBIR LAS ACTIVIDADES REALIZADAS EN EL DESARROLLO DE LA PRÁCTICA

En la primera semana se solicita realizar un directorio de contactos interno que permita visualizar los registros de cada funcionario (Nombre, cédula, puesto, unidad administrativa, dirección, teléfono, extensión y correo electrónico), para lo cual se ha definido el desarrollo de este directorio con Python y el framework Django por ser lo más efectivo en relación al tiempo e incluso por la organización que este framework ofrece, al igual que las facilidades no solo en la creación de una app web, si no también a la hora de administrar los registros, usuarios, roles y permisos.

De igual forma se ha definido que se va a desarrollar esta app con VSCode y en un entorno virtual como buena práctica, así que se instalan las dependencias, extensiones y bibliotecas necesarias para desarrollar de manera correcta una app de este tipo y se crea la app, en la cual se instalan de igual forma dentro de las configuraciones de Django. Y ahora como primer paso para la creación en sí de la aplicación solicitada se analizan los campos necesarios en base al documento proporcionado por el coordinador del área (documento que se va a automatizar), se crean los modelos correspondientes y se realizan las migraciones para que se cree la tabla en la base de datos, necesaria para almacenar los datos que se ingresarán posteriormente.

Se crean las vistas, principalmente la vista que mostrará la tabla previamente creada con los registros que se vayan ingresando, y de igual forma se crean vistas para el manejo de estos datos desde la interfaz que se está desarrollando para un manejo más amigable para el usuario. En este caso y por la misma razón que se ha mencionado del manejo amigable para el usuario, se crea un formulario con los campos que se definieron en el análisis de los registros que se incluyeron en el modelo y se incluyen ciertas validaciones, por ejemplo para la cédula que es un campo fundamental en pasos posteriores, para correo electrónico y el teléfono.

Para poder visualizar las vistas que se han creado y el formulario se crean plantillas en html y se deben incluir las urls para acceder desde un navegador, entonces se realizan estas configuraciones dentro de la “app” destinada a

manejar estas vistas, modelos y formularios y de igual forma se configuran las urls que direccionan a esta app en la app principal.

Se realizan pruebas con resultados exitosos y se continúa con el desarrollo de esta app. Ahora el enfoque va hacia el manejo de usuarios, es decir, creación de usuarios, inicio y cierre de sesión e incluso la asignación de roles. Aunque esto tiene la facilidad que ofrece Django con sus funciones de login, registro de usuarios y manejo de roles, es necesario configurar estas funciones y al igual que en las otras funcionalidades, se necesita crear una interfaz fácil de manejar para los usuarios, entonces se crean de igual forma una página en html para el inicio de sesión y una para el registro de usuario, aunque a esta última solamente tendrá acceso el administrador por cuestiones de seguridad.

Django ofrece ciertas funciones preconfiguradas como lo es el panel de administración de Django, en el cual se pueden visualizar los usuarios que están registrados incluyendo el mismo administrador y es ahí donde se pueden otorgar permisos de administrador y el acceso a este panel de control. Pero esto debe adaptarse a los requerimientos del cliente, por lo que se realizan ciertas modificaciones en el panel de administración, por ejemplo, en el manejo de las credenciales, que se solicitó que incluyan un nombre de usuario, correo electrónico y contraseña, siendo que esta contraseña se va a encriptar por seguridad y se puede modificar a través de otro formulario incluido con Django.

En la segunda semana, para mejorar el aspecto de la seguridad, se ha creado una validación con ciertas bibliotecas que ofrecen funciones de validación, para que en este caso se controle los intentos de inicio de sesión, que se puede definir el número de intentos y el tiempo en el que se puede bloquear la ip del usuario que realice dicho número de intentos fallidos. Y ahora se procede con la parte “frontend” de la app.

Para la parte del frontend de la app se ha creado primeramente un archivo base que contiene el formato general de todas las páginas que se podrán observar, para esto se crea una página html que contiene dicho formato y se configura el uso de la misma en las páginas html creadas en pasos anteriores. Y ahora se comienza a crear los estilos para la app web, para lo cual se utiliza CSS en un solo archivo que incluirá todos los estilos de cada una de las vistas tanto para su uso en PC como en dispositivos móviles.

Una parte importante que se debe mencionar es que, al igual que para las vistas creadas desde cero, para las vistas y funciones que ofrece Django y que han sido adaptadas a los requerimientos también se deben incluir las urls en los archivos correspondientes. De igual forma, cabe mencionar que las vistas se presentan de forma diferente para administradores y usuarios comunes; para los administradores se presentan los registros con las funciones para modificar y eliminar los registros y otra para agregar nuevos registros (aquí se presenta el formulario que se había

mencionado), y, por otra parte, para los usuarios comunes se presenta simplemente la lista de registros en la tabla definida en los pasos antes mencionados.

Anteriormente se mencionó que el campo que registra la cédula del funcionario incluido en los registros de contacto sería una parte fundamental, esto es debido a que se va a incluir una función de búsqueda, así que, en esta tercera semana se implementa un cuadro en el que se ingresa la cédula del contacto que se desea obtener y se realiza la búsqueda. Ahora, al haber realizado varias pruebas, se realiza una primera presentación a algunos de los usuarios a quienes va dirigido este proyecto y se definen varios cambios necesarios, como por ejemplo la necesidad de crear un informe de los registros en formato Excel y CSV; hablando de la búsqueda de contactos se ha solicitado que no sea solamente por número de cédula, si no también por nombre, área en la que trabaja y otros campos relevantes.

Gracias a esta primera presentación se definieron ciertos requerimientos que ya se han mencionado, así que se procede con la modificación y adición de lo solicitado. Se configura la búsqueda para que pueda realizarse por todos los campos relevantes y en una primera prueba se presentan ciertas dificultades por el manejo de caracteres especiales, por lo que se realiza una normalización de los datos que ingresan a la base de datos y los que se presentan en la interfaz de usuario para que no pierdan su formato anterior pero permita un mejor manejo de los datos que incluyan caracteres especiales y no exista ningún problema en la búsqueda y filtros. Ahora se pueden realizar búsquedas correctamente y también se pueden ordenar los datos con el encabezado de cada columna de la tabla. Y por último se ha creado la función para poder crear un informe y descargarlo en los dos formatos solicitados.

Por último, se realizan pruebas exhaustivas para detectar y evitar a futuro errores que pudieran afectar de manera significativa a los usuarios y dificultar el uso de esta app. Se obtienen resultados exitosos y no se ha detectado ningún problema, al menos de primera vista, por lo que se procede con la configuración del servidor en el que se va a desplegar la aplicación para el directorio interno de la empresa.

Para el despliegue de esta app se utiliza un servidor de Ubuntu independiente, se realiza la importación del proyecto de Windows al sistema Unix (servidor de Ubuntu) y se realizan las configuraciones internas de la app para lanzarla en producción. También se ha realizado una migración de la base de datos de SQLite (predeterminada de Django) a una más robusta, en este caso se utiliza PostgreSQL y a continuación se realiza la configuración del servidor web para lo cual se utilizará Nginx como proxy inverso y Unicorn como servidor de aplicaciones y se realizan las pruebas respectivas para dejar la app funcional.

Para la configuración previa del servidor se solicitó al coordinador realizar este paso, pero hubo ciertas dificultades externas a este paso como lo son problemas técnicos varios y fallos en la red, por lo que en ese momento se solicitó

revisar un proyecto realizado por otro desarrollador, pero en este caso fue un proyecto desarrollado en PHP. Entonces se procede con la revisión exhaustiva, profundización en los conocimientos de esta herramienta y la detección de errores y funciones faltantes.

Aunque se inició con la corrección de los errores detectados y funciones faltantes, no se pudo concretar este proyecto por falta de tiempo, ya que se llegó a la conclusión de que se necesitaría al menos 6 meses más para una correcta conclusión, y para este momento ya se contaba con el servidor para la app del directorio de contactos interno de la empresa y se procedió con los pasos antes mencionados para el despliegue de la app.

5. ASPECTOS POSITIVOS Y NEGATIVOS IDENTIFICADOS EN EL DESARROLLO DE LA PRÁCTICA

ASPECTOS	DESCRIPCIÓN
POSITIVOS	<p>1. El conocimiento que se adquiere en el proceso de aprendizaje en una carrera universitaria es una buena base, pero necesita ser aplicada en un ámbito laboral (real) para entender su verdadero valor.</p> <p>2. Al aplicar estos conocimientos en un ámbito real se pueden conocer aspectos y conflictos que pueden presentarse y que no se conocen en un ámbito educativo.</p> <p>3. Los conocimientos adquiridos durante las prácticas pre profesionales tendrán un gran impacto en futuros empleos y proyectos.</p>
NEGATIVOS	<p>1. Por lo general, falta de tiempo y coordinación causados por problemas inesperados, pero igualmente inherentes del puesto de trabajo como problemas técnicos.</p> <p>2. Falta de presupuesto destinado al área de tecnología y a las herramientas necesarias para un desarrollo más eficiente de las soluciones que requiere la empresa.</p> <p>3. Falta de una documentación adecuada por parte de otros desarrolladores y por lo tanto un retraso en la resolución de conflictos en los proyectos desarrollados por los mismos.</p>

6. CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES	RECOMENDACIONES
1. Se han aplicado los conocimientos teóricos adquiridos en proceso educativo en un	1. Mostrar iniciativa y estar dispuesto a asumir responsabilidades adicionales puede destacar

entorno real. Se trabajó con tecnologías actuales y herramientas de desarrollo, lo cual permitió fortalecer habilidades técnicas.	a un pasante. Participar en proyectos más allá de las tareas asignadas demuestra dedicación y compromiso.
2. Se presentaron desafíos técnicos y situaciones inesperadas. Estas experiencias han dado oportunidad de desarrollar habilidades de resolución de problemas y adaptación, encontrando soluciones creativas y eficientes.	2. La documentación es clave para la colaboración efectiva en proyectos. Se recomienda mantener un registro detallado de los pasos y decisiones tomadas. La comunicación clara y abierta con el equipo también es esencial.
3. La industria de la tecnología está en constante evolución. Por lo que se ha comprendido la importancia del aprendizaje continuo para mantenerse actualizado en las últimas tendencias y tecnologías emergentes.	3. Aprender a gestionar el tiempo de manera eficaz es esencial para cumplir con plazos y entregas. Se recomienda desarrollar habilidades de organización y priorización.

7. ANEXOS

ANEXOS	DESCRIPCIÓN
Anexo 1	



ITSQM^{ET}
INSTITUTO TECNOLÓGICO SUPERIOR
QUITO METROPOLITANO

```

1  {% extends "base.html" %}
2
3  {% block title %}Inicio de sesión{% endblock %}
4
5  {% block content %}
6  <div class="form-container">
7    <div class="form"><h2>Inicio sesión</h2>
8    <div class="messages">
9      {% for message in messages %}
10         <div>{% if message.tags %} class="{% if message.tags %} {% endif %} {% message %}</div>
11      {% endfor %}
12    </div>
13    <div>
14      <div class="form">
15        <div class="form"><div class="form">
16          <div class="form"><div class="form">
17            <div class="form">
18              <div class="form">
19                <div class="form">
20                  <div class="form">
21                    <div class="form">
22                  <div class="form">
23                <div class="form">
24              <div class="form">
25            <div class="form">
26          <div class="form">
27        <div class="form">
28      </div>
29    </div>
30  </div>
31  </div>
32  </div>
33  </div>
34  </div>
35  </div>
36  </div>
37  </div>
38  </div>
39  </div>
40  </div>
41  </div>
42  </div>
43  </div>
44  </div>
45  </div>
46  </div>
47  </div>
48  </div>
49  </div>
50  </div>
51  </div>
52  </div>
53  </div>
54  </div>
55  </div>
56  </div>
57  </div>
58  </div>
59  </div>
60  </div>
61  </div>
62  </div>
63  </div>
64  </div>
65  </div>
66  </div>
67  </div>
68  </div>
69  </div>
70  </div>
71  </div>
72  </div>
73  </div>
74  </div>
75  </div>
76  </div>
77  </div>
78  </div>
79  </div>
80  </div>
81  </div>
82  </div>
83  </div>
84  </div>
85  </div>
86  </div>
87  </div>
88  </div>
89  </div>
90  </div>
91  </div>
92  </div>
93  </div>
94  </div>
95  </div>
96  </div>
97  </div>
98  </div>
99  </div>
100 </div>

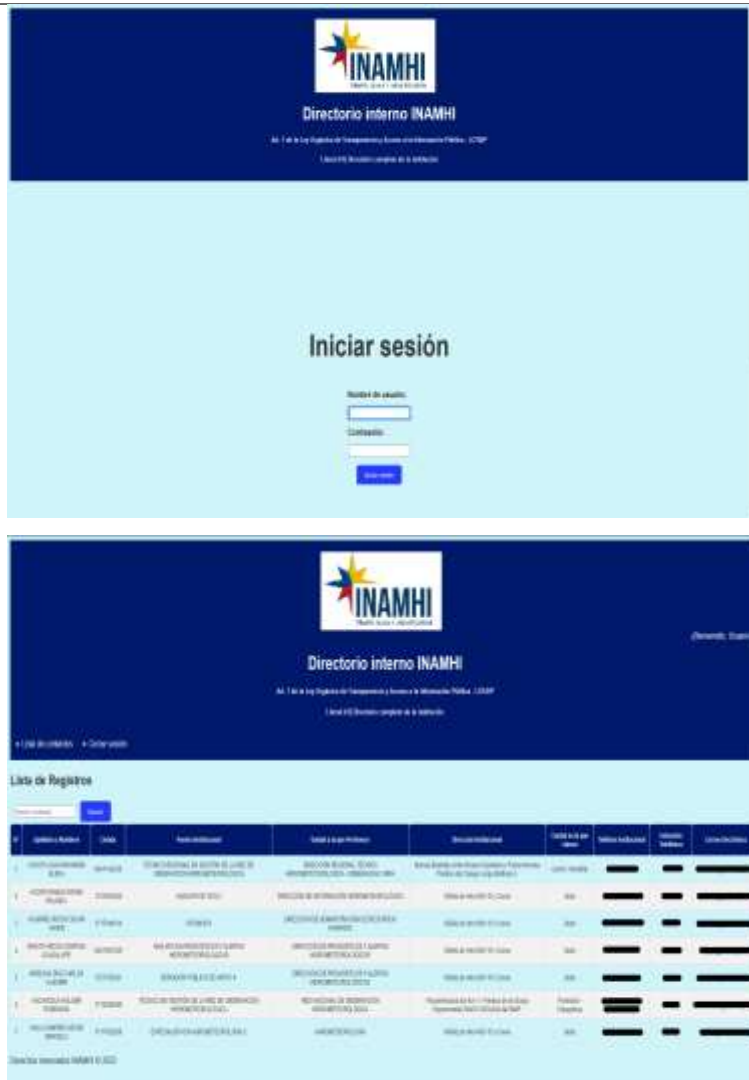
```

```

1 # Import the pandas module
2 import pandas as pd
3
4 # Create a DataFrame with 5 rows and 4 columns
5 data = {'Year': 2010, 'Country': 'USA', 'GDP': 15000, 'Population': 300000000}
6 data = data.append({'Year': 2011, 'Country': 'USA', 'GDP': 15500, 'Population': 305000000})
7 data = data.append({'Year': 2012, 'Country': 'USA', 'GDP': 16000, 'Population': 310000000})
8 data = data.append({'Year': 2013, 'Country': 'USA', 'GDP': 16500, 'Population': 315000000})
9 data = data.append({'Year': 2014, 'Country': 'USA', 'GDP': 17000, 'Population': 320000000})
10
11 # Print the DataFrame
12 print(data)
13
14 # Filter the DataFrame to only include rows where the GDP is greater than 15000
15 filtered_data = data[data['GDP'] > 15000]
16
17 # Print the filtered DataFrame
18 print(filtered_data)
19
20 # Sort the DataFrame by GDP in descending order
21 sorted_data = data.sort_values(by='GDP', ascending=False)
22
23 # Print the sorted DataFrame
24 print(sorted_data)
25
26 # Group the DataFrame by Country and calculate the mean GDP
27 grouped_data = data.groupby('Country').mean()
28
29 # Print the grouped DataFrame
30 print(grouped_data)
31
32 # Merge two DataFrames
33 data1 = {'Year': 2010, 'Country': 'USA', 'GDP': 15000, 'Population': 300000000}
34 data1 = data1.append({'Year': 2011, 'Country': 'USA', 'GDP': 15500, 'Population': 305000000})
35 data1 = data1.append({'Year': 2012, 'Country': 'USA', 'GDP': 16000, 'Population': 310000000})
36 data1 = data1.append({'Year': 2013, 'Country': 'USA', 'GDP': 16500, 'Population': 315000000})
37 data1 = data1.append({'Year': 2014, 'Country': 'USA', 'GDP': 17000, 'Population': 320000000})
38
39 data2 = {'Year': 2010, 'Country': 'China', 'GDP': 5000, 'Population': 1300000000}
40 data2 = data2.append({'Year': 2011, 'Country': 'China', 'GDP': 5500, 'Population': 1350000000})
41 data2 = data2.append({'Year': 2012, 'Country': 'China', 'GDP': 6000, 'Population': 1400000000})
42 data2 = data2.append({'Year': 2013, 'Country': 'China', 'GDP': 6500, 'Population': 1450000000})
43 data2 = data2.append({'Year': 2014, 'Country': 'China', 'GDP': 7000, 'Population': 1500000000})
44
45 # Merge the two DataFrames
46 merged_data = pd.merge(data1, data2, on='Year')
47
48 # Print the merged DataFrame
49 print(merged_data)
50
51 # Join two DataFrames
52 data1 = {'Year': 2010, 'Country': 'USA', 'GDP': 15000, 'Population': 300000000}
53 data1 = data1.append({'Year': 2011, 'Country': 'USA', 'GDP': 15500, 'Population': 305000000})
54 data1 = data1.append({'Year': 2012, 'Country': 'USA', 'GDP': 16000, 'Population': 310000000})
55 data1 = data1.append({'Year': 2013, 'Country': 'USA', 'GDP': 16500, 'Population': 315000000})
56 data1 = data1.append({'Year': 2014, 'Country': 'USA', 'GDP': 17000, 'Population': 320000000})
57
58 data2 = {'Year': 2010, 'Country': 'China', 'GDP': 5000, 'Population': 1300000000}
59 data2 = data2.append({'Year': 2011, 'Country': 'China', 'GDP': 5500, 'Population': 1350000000})
60 data2 = data2.append({'Year': 2012, 'Country': 'China', 'GDP': 6000, 'Population': 1400000000})
61 data2 = data2.append({'Year': 2013, 'Country': 'China', 'GDP': 6500, 'Population': 1450000000})
62 data2 = data2.append({'Year': 2014, 'Country': 'China', 'GDP': 7000, 'Population': 1500000000})
63
64 # Join the two DataFrames
65 joined_data = data1.join(data2, how='outer')
66
67 # Print the joined DataFrame
68 print(joined_data)
69
70 # Concatenate two DataFrames
71 data1 = {'Year': 2010, 'Country': 'USA', 'GDP': 15000, 'Population': 300000000}
72 data1 = data1.append({'Year': 2011, 'Country': 'USA', 'GDP': 15500, 'Population': 305000000})
73 data1 = data1.append({'Year': 2012, 'Country': 'USA', 'GDP': 16000, 'Population': 310000000})
74 data1 = data1.append({'Year': 2013, 'Country': 'USA', 'GDP': 16500, 'Population': 315000000})
75 data1 = data1.append({'Year': 2014, 'Country': 'USA', 'GDP': 17000, 'Population': 320000000})
76
77 data2 = {'Year': 2010, 'Country': 'China', 'GDP': 5000, 'Population': 1300000000}
78 data2 = data2.append({'Year': 2011, 'Country': 'China', 'GDP': 5500, 'Population': 1350000000})
79 data2 = data2.append({'Year': 2012, 'Country': 'China', 'GDP': 6000, 'Population': 1400000000})
80 data2 = data2.append({'Year': 2013, 'Country': 'China', 'GDP': 6500, 'Population': 1450000000})
81 data2 = data2.append({'Year': 2014, 'Country': 'China', 'GDP': 7000, 'Population': 1500000000})
82
83 # Concatenate the two DataFrames
84 concatenated_data = pd.concat([data1, data2])
85
86 # Print the concatenated DataFrame
87 print(concatenated_data)
88
89 # Reshape the DataFrame
90 data = {'Year': 2010, 'Country': 'USA', 'GDP': 15000, 'Population': 300000000}
91 data = data.append({'Year': 2011, 'Country': 'USA', 'GDP': 15500, 'Population': 305000000})
92 data = data.append({'Year': 2012, 'Country': 'USA', 'GDP': 16000, 'Population': 310000000})
93 data = data.append({'Year': 2013, 'Country': 'USA', 'GDP': 16500, 'Population': 315000000})
94 data = data.append({'Year': 2014, 'Country': 'USA', 'GDP': 17000, 'Population': 320000000})
95
96 # Reshape the DataFrame
97 reshaped_data = data.reset_index()
98
99 # Print the reshaped DataFrame
100 print(reshaped_data)
101
102 # Pivot the DataFrame
103 data = {'Year': 2010, 'Country': 'USA', 'GDP': 15000, 'Population': 300000000}
104 data = data.append({'Year': 2011, 'Country': 'USA', 'GDP': 15500, 'Population': 305000000})
105 data = data.append({'Year': 2012, 'Country': 'USA', 'GDP': 16000, 'Population': 310000000})
106 data = data.append({'Year': 2013, 'Country': 'USA', 'GDP': 16500, 'Population': 315000000})
107 data = data.append({'Year': 2014, 'Country': 'USA', 'GDP': 17000, 'Population': 320000000})
108
109 # Pivot the DataFrame
110 pivoted_data = data.pivot(index='Year', columns='Country', values='GDP')
111
112 # Print the pivoted DataFrame
113 print(pivoted_data)
114
115 # Melt the DataFrame
116 data = {'Year': 2010, 'Country': 'USA', 'GDP': 15000, 'Population': 300000000}
117 data = data.append({'Year': 2011, 'Country': 'USA', 'GDP': 15500, 'Population': 305000000})
118 data = data.append({'Year': 2012, 'Country': 'USA', 'GDP': 16000, 'Population': 310000000})
119 data = data.append({'Year': 2013, 'Country': 'USA', 'GDP': 16500, 'Population': 315000000})
120 data = data.append({'Year': 2014, 'Country': 'USA', 'GDP': 17000, 'Population': 320000000})
121
122 # Melt the DataFrame
123 melted_data = data.melt(id_vars='Year', value_vars=['Country'], var_name='Country', value_name='GDP')
124
125 # Print the melted DataFrame
126 print(melted_data)
127
128 # Stack the DataFrame
129 data = {'Year': 2010, 'Country': 'USA', 'GDP': 15000, 'Population': 300000000}
130 data = data.append({'Year': 2011, 'Country': 'USA', 'GDP': 15500, 'Population': 305000000})
131 data = data.append({'Year': 2012, 'Country': 'USA', 'GDP': 16000, 'Population': 310000000})
132 data = data.append({'Year': 2013, 'Country': 'USA', 'GDP': 16500, 'Population': 315000000})
133 data = data.append({'Year': 2014, 'Country': 'USA', 'GDP': 17000, 'Population': 320000000})
134
135 # Stack the DataFrame
136 stacked_data = data.stack()
137
138 # Print the stacked DataFrame
139 print(stacked_data)
140
141 # Unstack the DataFrame
142 data = {'Year': 2010, 'Country': 'USA', 'GDP': 15000, 'Population': 300000000}
143 data = data.append({'Year': 2011, 'Country': 'USA', 'GDP': 15500, 'Population': 305000000})
144 data = data.append({'Year': 2012, 'Country': 'USA', 'GDP': 16000, 'Population': 310000000})
145 data = data.append({'Year': 2013, 'Country': 'USA', 'GDP': 16500, 'Population': 315000000})
146 data = data.append({'Year': 2014, 'Country': 'USA', 'GDP': 17000, 'Population': 320000000})
147
148 # Unstack the DataFrame
149 unstacked_data = data.unstack()
150
151 # Print the unstacked DataFrame
152 print(unstacked_data)
153
154 # Sort the DataFrame by multiple columns
155 data = {'Year': 2010, 'Country': 'USA', 'GDP': 15000, 'Population': 300000000}
156 data = data.append({'Year': 2011, 'Country': 'USA', 'GDP': 15500, 'Population': 305000000})
157 data = data.append({'Year': 2012, 'Country': 'USA', 'GDP': 16000, 'Population': 310000000})
158 data = data.append({'Year': 2013, 'Country': 'USA', 'GDP': 16500, 'Population': 315000000})
159 data = data.append({'Year': 2014, 'Country': 'USA', 'GDP': 17000, 'Population': 320000000})
160
161 # Sort the DataFrame by multiple columns
162 sorted_data = data.sort_values(by=['GDP', 'Population'], ascending=False)
163
164 # Print the sorted DataFrame
165 print(sorted_data)
166
167 # Filter the DataFrame by multiple conditions
168 data = {'Year': 2010, 'Country': 'USA', 'GDP': 15000, 'Population': 300000000}
169 data = data.append({'Year': 2011, 'Country': 'USA', 'GDP': 15500, 'Population': 305000000})
170 data = data.append({'Year': 2012, 'Country': 'USA', 'GDP': 16000, 'Population': 310000000})
171 data = data.append({'Year': 2013, 'Country': 'USA', 'GDP': 16500, 'Population': 3
```



Anexo 2



Directorio interno INAMHI
Ministerio del Ambiente y Agua
Instituto Nacional de Meteorología e Hidrología

Iniciar sesión

Nombre de usuario:
Contraseña:

Directorio interno INAMHI
Ministerio del Ambiente y Agua
Instituto Nacional de Meteorología e Hidrología

Lista de Registros

#	Nombre y Apellido	Código	Resolución	Código y tipo de documento	Resolución	Resolución	Fecha de emisión	Fecha de vigencia	Fecha de caducidad
1	ALFONSO GARCÍA GARCÍA	00000001	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	2014-05-14	2014-05-14	2014-05-14
2	ALFONSO GARCÍA GARCÍA	00000002	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	2014-05-14	2014-05-14	2014-05-14
3	ALFONSO GARCÍA GARCÍA	00000003	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	2014-05-14	2014-05-14	2014-05-14
4	ALFONSO GARCÍA GARCÍA	00000004	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	2014-05-14	2014-05-14	2014-05-14
5	ALFONSO GARCÍA GARCÍA	00000005	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	2014-05-14	2014-05-14	2014-05-14
6	ALFONSO GARCÍA GARCÍA	00000006	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	2014-05-14	2014-05-14	2014-05-14
7	ALFONSO GARCÍA GARCÍA	00000007	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	Resolución de 2014 de 14 de mayo de 2014	2014-05-14	2014-05-14	2014-05-14

Última actualización: 2014-05-14

[Crear nuevo registro](#)

```
root@ubuntu01:~# ssh root@ubuntu02 -t bash
Using username "root".
Warning: Permanently added 'ubuntu02' (SSH) host key.
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-76-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/support

System information as of Fri Aug 25 00:07:10 PM UTC 2023

System load: 0.0                               Processes:           230
Usage of /:   7.6% of 87.97GB                   Data tagged with:   1
Memory usage: 4%                                IP net addresses for eth0: 10.0.2.15
Swap usage:   0%

== There are 3 corbie processes.

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
just raised the bar for easy, skillset and secure K8s cluster deployment.

https://ubuntu.com/engage/secure-kubernetes-at-the-edge

Suspended Security Maintenance for Applications is not enabled.
A update can be applied immediately.
To see these additional updates run: apt list --upgradeable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run sudo pro status

Last login: Wed Aug 23 20:30:47 2023 from 10.0.2.15
root@ubuntu02:~# cat /etc/passwd | grep ubuntu02 | wc -l
1
root@ubuntu02:~# cat /etc/passwd | grep ubuntu02 | head -n 1
ubuntu02:x:1000:1000::/home/ubuntu02:/bin/bash
```





Firma del Estudiante

CC 1751204072